AutoEpsDBSCAN : DBSCAN with Eps Automatic for Large Dataset

Manisha Naik Gaonkar & Kedar Sawant

Goa College of Engineering, Computer Department, Ponda-Goa, Goa College of Engineering,
Computer Department, Ponda-Goa.
E-mail : manisha@gec.ac.in, Kedarsawant22@yahoo.com

***Abstract* -Emergence of modern techniques for scientific data collection has resulted in large scale accumulation of data pertaining to diverse fields. Conventional database querying methods are inadequate to extract useful information from huge data banks. Cluster analysis is a primary method for database mining [8]. It is either used as a stand-alone tool to get insight into the distribution of a data set or as a pre-processing step for other algorithms operating on the detected clusters. Almost all of the well-known clustering algorithms require input parameters which are hard to determine but have a significant influence on the clustering result. Furthermore, for many real-data sets there does not even exist a global parameter setting for which the result of the clustering algorithm describes the intrinsic clustering structure accurately [1], [2]. DBSCAN (Density Based Spatial Clustering of Application with Noise) [1] is a base algorithm for density based clustering techniques. This paper gives a survey of density based clustering algorithms with the proposed enhanced algorithm that automatically selects the input parameters along with its implementation and comparison with the existing DBSCAN algorithm. The experimental results shows that the proposed algorithm can detect the clusters of varied density with different shapes and sizes from large amount of data which contains noise and outliers, requires only one input parameters and gives better output then the DBSCAN algorithm.**

Keywords- Clustering Algorithms, Data mining, DBSCAN, Density, Eps, Minpts, and VDBSCAN.

I. INTRODUCTION

Most organizations have accumulated a great deal of data, but what they really want is information. The newest, hottest technology to address these concerns is data mining [8]. Data mining uses sophisticated statistical analysis and modeling techniques to uncover patterns and relationships hidden in organizational databases. Data mining finds these patterns and relationships by building models [9]. The clustering algorithms currently in popular use was driven by biologists, before being promptly taken up by statisticians. The algorithms were based on the distance-space, where the similarity between two objects was quantified by the distance between them as calculated by some distance-metric [8], [9].

In this paper we proposed a clustering algorithm based on the knowledge acquired from the data set, and apply the main idea of density based clustering algorithm DBSCAN. The proposed algorithm requires one input parameter, discovers arbitrary size and shaped clusters, is efficient even for large data sets. So the objective is to enhance the existing algorithm called DBSCAN [1] such that it will detect the cluster automatically by explicitly finding the input parameters and finding clusters with varying density. The basic idea is that, before adopting traditional DBSCAN algorithm, some methods are used to select several values of parameter Eps for different densities according to a k-dist plot. With different values of Eps, it is possible to find out clusters with varied densities simultaneity [3], [6]. For each value of Eps, DBSCAN algorithm is adopted in order to make sure that all the clusters with respect to corresponding density are clustered. And for the next process, the points that have been clustered are ignored, which avoids marking both denser areas and sparser ones as one cluster.

The paper is organized as follows. Related work on density-based clustering along with the past and the present work is briefly discussed in Section 2. In Section 3, the problems related to existing approaches of density based clustering along with the main motivation are presented. In Section 4, the proposed methods to find input parameters required for the proposed algorithm along with the algorithm itself is presented. Section 5 shows experimental results of proposed algorithm along with its comparison with DBSCAN. Section 6 concludes the paper with a summary and some directions for future research.

II. RELATED WORK

There are many clustering algorithms proposed, these 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 [7].

OPTICS [2] algorithm is an enhancement of DBSCAN. Rather than producing the data set clustering explicitly, OPTICS computes an augmented cluster ordering for automatic and interactive cluster analysis. This ordering represents the density-based clustering structure of the data. It contains information that is equivalent to density-based clustering obtained from a wide range of parameter settings. The cluster ordering can be used to extract basic clustering information as well as provide the intrinsic clustering structure. To construct the different clustering simultaneously, the objects should be processed in a specific order. This order selects an object that is density-reachable with respect to the lowest Eps value so that clusters with higher density will be finished first. However, the algorithm presents a new drawback. It only generates the clusters whose local-density exceeds some threshold instead of similar local-density clusters and does not produce clusters of a data set explicitly and it requires lot of user interaction to accept different Eps values [2].

The density-based algorithm WaveCluster [11] applies wavelet transformation to the feature space. The algorithm is able to detect clusters of arbitrary shape at different scales. Input parameters include the number of grid cells for each dimension, the wavelet to use, and the number of applications of the wavelet transformation. Using multi-resolution property of wavelet transforms, it can effectively identify arbitrary shape clusters at different degrees of accuracy. It considers the multidimensional spatial data as a multidimensional signal and it apply signal processing techniques – wavelet transforms to convert the spatial data into the frequency domain [11]. The SNN [4] algorithm, as DBSCAN, is a density-based clustering algorithm. The main difference is that it defines the similarity between points by looking at the number of nearest neighbors that two points share. Using this similarity measure in the SNN algorithm, the density is defined as the sum of the similarities of the nearest neighbors of a point. Points with high density become *core points*, while points with low density represent *noise points*. All remainder points that are strongly similar to a specific core points will represent a new clusters. However, It needs three inputs parameters [4].

KDDClus [6] 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. However it requires the value of K to be inserted by the user and different set of Eps values obtain from KD-tree data structure[6].

The DBSCAN algorithm is not capable of finding out meaningful clusters with varied densities. VDBSCAN [3] algorithm overcomes this shortcoming by detecting clusters with varied density as well as helping in selecting several values of input parameter Eps for different densities. The basic idea is that, before adopting traditional DBSCAN algorithm, some methods are used to select several values of parameter Eps for different densities according to a k-dist plot. With different values of Eps, it is possible to find out clusters with varied densities simultaneously. For each value of Eps, DBSCAN algorithm is adopted in order to make sure that all the clusters with respect to corresponding density are clustered. And for the next process, the points that have been clustered are ignored, which avoids marking both denser areas and sparser ones as one cluster. The k-dists are computed for all the data points for some k inserted by the user, sorted in ascending order, and then plotted using the sorted values; as a result, a sharp change is expected to see. So the user will enter different values of Eps based on this sharp change in the graph. VDBSCAN has the same time complexity as DBSCAN and can identify clusters with different density but requires user to enter the value of k and different Eps values based on the k-dist plot. Also the behavior of parameter k in k-dist plot depends on the dataset [3].

III. MOTIVATIONAL FACTORS

DBSCAN is a famous density-based clustering method, which can discover the clusters with arbitrary shapes and does not need to know the number of clusters initially in its algorithm [1], [7]. However, it needs to know two parameters: Eps and MinPts and the value of parameter Eps is important for DBSCAN algorithm, but the calculation of Eps is time-consuming. Due to a single global parameter Eps, it is impossible to detect some clusters using one global-MinPts. It does not perform well on multi-density data sets. In the multi-density data set, DBSCAN may merge between different clusters and may also neglect other clusters that assign them as noise. In DBSCAN, the user can specify the values of parameters Eps, but it is difficult [1], [7], [9].

An important property of many real world data sets is that their intrinsic cluster structures are unable to be characterized by global density parameters. As a result, very different local densities may be needed to reveal clusters in different regions of the data space. For example, in the data set depicted in Fig. 1, it is impossible to detect the clusters A, B, C1, C2, and C3 simultaneously using one global density parameter. A global density-based decomposition would be needed for the clusters A, B, and C, or C1, C2, and C3 [2]. So this is the main motivation behind coming up with the proposed algorithm over the existing DBSCAN algorithm.

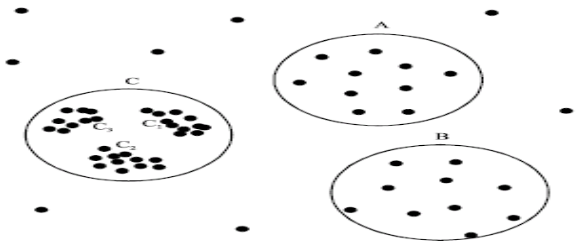


Fig. 1 Density varied datapoints

IV. PROPOSED METHODS

A. *Method for determining Different Eps' values*

To determine different range of Eps values automatically to identify the number of clusters of different densities including noise, we need to first draw a k-dist graph for all the points, for a given k which will b entered by the user. Initially we compute the average of the distances of every point to all k of its nearest neighbors [3], [6]. This is unlike VDBSCAN, where only the kth nearest neighbor is considered during the distance computation. The use of the K-dist plot structure enables efficient computation of k-nearest neighbors of a point, particularly for large data. The averaging allows a smoothing of the curve towards noise removal, for subsequent easier automated detection of density thresholds. We plot these averaged k-distances in an ascending order, to help identify noise with relative ease. We know that 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 [3], [6].

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 points. Any value less than this density threshold Eps estimate can efficiently cluster patterns whose average k distances is lower than that, implying patterns or points belonging to a certain density. Analogously all knees in the smoothed graph can collectively estimate a set of Eps's for identifying all the clusters having different density distributions [3], [6]. The knee regions are detected by clustering the sorted k-dist plot. In short to find all possible Eps values we will have to calculate the slopes at regular interval and then find the difference between slopes values at the same regular interval. By setting certain threshold value we can get different Eps values automatically based on this threshold value while discarding those with higher thresholds.

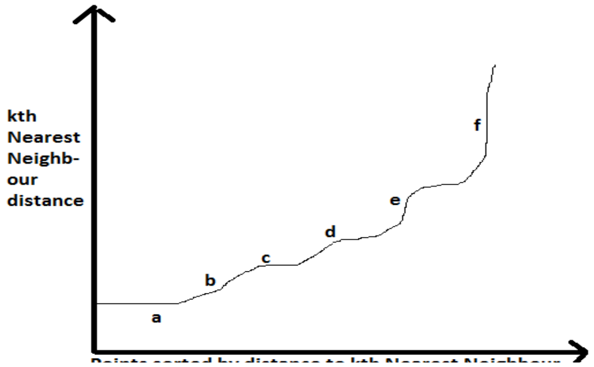


Fig. 2 Average K-dist sorted plot

The shape of the sorted k-distance plot and hence, the effectiveness of the proposed heuristic 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 [3], [6].

For datasets with widely varied density, there will be some variation, depending on the density of the cluster and the random distribution of points, but for points of the same density level, the range of variation will not be huge while a sharp change is expected to see between two density levels. Thus there will be several smooth curves connected by greatly variation ones [3]. If there are n (natural number n>1) different smooth curves in the k-dist plot, the dataset has n density levels. A dataset is of varied-density if it has several density levels and of n varied-density if it has n density levels. Specially, a dataset is of single-density if its density does not vary widely, or there is only one smooth curve in its k-dist plot. Figure 2 shows plotted k-dist graph for a given dataset with k value which will be specified by the user [3].

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 2 are such lines, which can be called level-turning lines. Line b connects line a and c, and line d connects c and e, while a, c and e stand for different density levels. Note that line f shows the k-dists of outliers and is not a level-turning line for it does not connect two smooth lines [3]. For different density levels D_i, select suitable Eps. For example, in Figure 2, 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 Eps_1, and then take line c and d as a sub-k-dist plot for Eps_2, e and f for Eps_3 finally.

After determining the optimal number of different Eps values we need to start forming clusters starting from the lowest Eps value in the sorted k-dist graph, by sequentially execute DBSCAN for each of the Eps estimated considered in ascending order. The first estimate obviously corresponds to the denser cluster. Tagging the patterns in the already detected clusters as “visited”, we proceed towards larger values of k-distance while allowing DBSCAN to work on the noise list obtain after applying the lowest value of Eps. In this manner we are able to effectively determine all clusters in a multi-density framework, in a decreasing order of density, with noise being modeled as the sparsest region [3], [6].

So we need to adopt DBSCAN algorithm for each Eps_i. Before adopt DBSCAN for Eps_i+1, mark points in clusters corresponding with Eps_i as C_i. Marked points will not be processed by DBSCAN again. Non-marked points after all the Eps_i process are recognized as outliers. And all the C_i are displayed as the results [3], [6].

B. Method for determining Minpts value

After determining the different Eps values, how to estimate the value of the MinPts is our urgent task. So firstly, the number of data objects in Eps neighborhood of every point in dataset is calculated one by one. And then mathematic expectation of all these data objects is calculated, which is the value of MinPts.

$$\text{Minpts} = 1/n \sum_{i=1}^{n} Pi$$

Where p_i is the number of points in Eps neighborhood of point i. So for each different value of Eps we will get corresponding Minpts value.

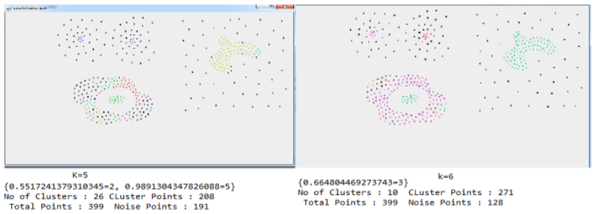
C. Proposed Algorithm- DBSCAN with Eps automatic

```
EpsDBSCAN (D, k)
{ For each point P in dataset D
{  N1 [] = getNeighbors (P, k)
For each point P’ in N1
{  Distance = Σ getDist (P, P’) }
AvgDist [] = Distance / k }
Array [] = Sort (AvgDist [])
Plot (Array [] on x-axis, AvgDist on y-axis)
For (int i=0; i<= D.size; i + 4)
{ SlopeValue [] = getSlope (array [i], array [i+4]) }
For(int i=0; i<= SlopeValue[].size; i++)
{ SlopeDiff=SlopeValue[i] – SlopeValue [i+1]
If ((SlopeDiff >= 10% * SlopeValue [i] || SlopeDiff<=20% * SlopeValue [i] ) && MinPts[i] >1)
{ Eps [] = SlopeValue [i]; }
For each Eps [i]
{ For each point P in dataset D
{ N2 [] = getNeighbors (P, Eps [i]) }
n = D.size()
$P_i$ [] = Σ N2
Minpts [i] = 1/n $\sum_{i=1}^{n} Pi$
} }
For each Eps [i] and MinPts [i]
{ ClusterPointList = DBSCAN (D, Eps [i], MinPts [i])
Mark clustered points as $C_{i.}$ }
NoiseList = D – ClusterPointList
DBSCAN (NoiseList, Eps [i+1], MinPts [i+1])  }
DBSCAN (D, Eps[i], MinPts[i])
{   C = 0
  for each unvisited point P in dataset D
    mark P as visited
    N = getNeighbors (P, Eps)
    if sizeof(N) < MinPts
      mark P as NOISE
     NoiseList [] = P
    else
      C = next cluster
expandCluster(P, N, C, Eps, MinPts)
Return NoiseList [] }
 expandCluster(P, N, C, Eps, MinPts)
{   add P to cluster C
  for each point P' in N
    if P' is not visited
      mark P' as visited
      N' = getNeighbors(P', Eps)
      if sizeof(N') >= MinPts
        N = N joined with N'
    if P' is not yet member of any cluster
      add P' to cluster C }
getSlope(Point m , Point n)
{ return (n.getY() - m.getY())/(n.getX() - m.getX()); }
```

V. EXPERIMENTAL RESULTS

A. *Experimental Setup*

Experiments were carried out on Intel core i3 2.40GHz processor with 2GB RAM, 64 bit windows 7 operating system. Programs were implemented in java on Eclipse-SDK-3.7-win32. Algorithm was applied on Compound dataset obtain from research department of The University of EDINBURGH Schools of informatics specially provided for testing new algorithms.



```
K=5
{0.5517241379310345=2, 0.9891304347826088=5}
No of Clusters : 26 CLuster Points : 208
 Total Points : 399  Noise Points : 191
```

```
                    k=6
{0.664804469273743=3}
No of Clusters : 10  CLuster Points : 271
 Total Points : 399  Noise Points : 128
```

Fig. 3 Clustered output of proposed algorithm on different k input on compound dataset

B. Comparison

DBSCAN and the proposed algorithm were applied on range of dataset to see how far the proposed algorithm works better than the existing DBSCAN algorithm and the experimental results shows that proposed algorithm works better than DBSCAN and produces correct output in less time. Below table shows the comparison between them on different set of input values on the compound dataset.

TABLE I
DBSCAN ALGORITHM ON DIFFERENT INPUTS ON COMPOUND DATASET

Eps	**MinPoints**	**Total Points**	**Clustered Points**	**Noise Points**	**No. of Clusters**
0.5	4	399	74	325	12
0.7	4	399	283	116	11
1	4	399	302	97	5
0.5	3	399	118	281	17
0.66	3	399	271	128	10
0.7	3	399	281	118	5

TABLE III
PROPOSED ALGORITHM ON DIFFERENT INPUTS ON COMPOUND DATASET

K	Total Points	Clustered Points	Noise Points	No. of Clusters
5	399	208	191	26
6	399	271	128	10
For Any other value of K, it gives no output				

Tabular content shows that DBSCAN algorithm requires more amount of initialization and more number of trial and error method to be applied to get the correct output but on the other hand proposed algorithms gives output in less time and with less number of guesses. It also requires user to vary only one K parameter in comparison to DBSCAN's two parameter. Experiments were carried out on other well known datasets also and the final outcome shows that proposed algorithm outplays DBSCAN in all respects.

VI. CONCLUSION

Among all clustering methods, density-based clustering algorithm is one of powerful tools for discovering arbitrary-shaped clusters in large spatial databases. In this paper, we presented the literature work in the field of density based clustering along with my proposed methods and algorithm to enhance the DBSCAN algorithm. Usually, user does not have enough information to determine the input parameters. Therefore, obtaining reasonable clustering results requires testing large different initializations. Minimizing input parameters will certainly be useful in reducing the errors introduced by human interference. DBSCAN algorithm requires two input parameters called Eps and MinPts, hence leading to above mention shortcomings which were tried to overcome by proposing the algorithm presented in this paper. So through this paper we have presented methods to select the range of Eps and MinPts value automatically and algorithms with single parameter automatic to find density varied clusters. Experimental results shows that proposed algorithm gives better output then the existing DBSCAN algorithm but still needs user to enter the value of K, so the future scope will be to find this value of K internally thus making the entire process automatic.

VII. ACKNOWLEDGMENT

This work was performed as part of a ME thesis in Enhancing DBSCAN Clustering of Large Dataset. We want to acknowledge the contribution of our colleagues from Goa Engineering College for all the support that was provided.

VIII. 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] M. Ankerst, M. Breunig, H.P. Kriegel, and J. Sandler. "OPTICS: Ordering Points to to Identify the Clustering Structure"; proceedings of the Int. Conf on Management of Data, pp. 49-60, 1999.

[3] zeng Liu, Dong Zhou, Naijun Wu, "Varied Density Based Spatial Clustering of Application with Noise", in proceedings of IEEE Conference ICSSSM 2007 pg 528-531.

[4] Adriano Moreira, Maribel Y. Santos and Sofia Carneiro, "Density-based clustering algorithms – DBSCAN and SNN", 2005.

[5] Hongfang Zhou, Peng Wang, Hongyan Li, "Research on Adaptive Parameters Determination in DBSCAN Algorithm", Journal of Information & Computational Science 9: 7 (2012).

[6] Sushmita Mitra1 and Jay Nandy "KDDClus: A Simple Method for Multi-Density Clustering", 2010.

[7] M.Parimala, Daphne Lopaz, N.C. Senthilkumar, "Survey on Density based Clustering Algorithm for mining large spatial databases", IJAST 2011.

[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] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. "A density based algorithm for discovering clusters in large spatial data sets with noise"; in 2nd International Conference on Knowledge Discovery and Data Mining, pp. 226–231,1996.

[11] Sheikholeslami G., Chatterjee S., Zhang A.: "WaveCluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases", Proc. 24th Int. Conf. on Very Large DataBases, New York, NY, 1998, pp. 428 - 439.

[12] Hattori K., Torii Y.: "Effective algorithms for the nearest neighbor method in the clustering problem", Pattern Recognition, 1993, Vol. 26, No. 5, pp. 741-746.

13. Mariam Rehman, Syed Atif Mehdi, "Comparison of Density-based Clustering Algorithms", 2006