A NEW DENSITY BASED SAMPLING TO ENHANCE DBSCAN CLUSTERING ALGORITHM

Safaa O. Al-mamory[1*], ***Israa S. Kamil***[2]

[1]University of Information Technology and Communications, College of Business Informatics, Baghdad, 10067, Iraq
[2] Al-Mustaqbal University College, Babylon, 51002, Iraq

Email: salmamory@uoitc.edu.iq[1*](corresponding author), IsraaSaleh@mustaqbal-college.edu.iq[2]

DOI: https://doi.org/10.22452/mjcs.vol32no4.5

ABSTRACT

DBSCAN is one of the efficient density-based clustering algorithms. It is characterized by its ability to discover clusters with different shapes and sizes, and to separate noise and outliers. However, when the dataset contain different densities, DBSCAN clustering will be inefficient. In this paper, we propose an approach to enable DBSCAN to cluster dataset having different densities by preprocess the dataset to make it with one density level. This system composed of four stages: firstly, a new approach to separate dataset based on density is presented. Secondly, a new density biased sampling technique is proposed. Thirdly, the resulted sparse data from the last two stages is clustered with DBSCAN. Finally, the remaining data from sampling will be clustered with KNN. The experimental results on synthetic and real datasets on average show that the clustering of the proposed algorithm is better than that of DBSCAN by more than 7% and retains time complexity of DBSCAN.

Keywords: database, clustering, DBSCAN, sampling.

1.0 INTRODUCTION

The process of collecting patterns in which the patterns in one group have the maximum similarity between each other and minimum similarity to patterns in other groups is called *clustering* [1]. Many of clustering algorithms were designed in different fields for different applications, but there is no algorithm convenient for all kinds of data. In general, there are six types of clustering methods such as partitioning, hierarchical, density-based, grid-based, model-based, and constraints-based techniques [2].

From different point of view, different types of clusters are available according to Tan et al. [3]. These clusters' types are well-separated, center-based, contiguity-based, conceptual, and density-based clusters. Density-based clustering forms the clusters of densely connected objects separated by low-density region, which characterized by discovering the clusters with arbitrary shapes and robust to the existence of noise. OPTICS algorithm [4] is one of density-based clustering methods that expands DBSCAN (Density Based Spatial Clustering of Applications with Noise) to obtain ordering of a cluster. This ordering has been obtained from settings for a wide range of parameters, while DENCLUE [5] (DENsity-based CLUstEring) depends on a type of mathematical functions which is called influence function to find the clusters. These influence functions implement the impact of a data point on its neighborhood.

DBSCAN [6] is one of the well-known density-based clustering methods. It is able to find clusters of various shapes and sizes. However, it fails to find clusters of different densities, this failure resulted from being used a global density threshold on all the data points. A new sampling algorithm to enhance the performance of DBSCAN when the data have density variation is presented in this paper. This algorithm assumes that every dataset (having varying densities) can be divided into two layers. These layers are dense layer and sparse layer as can be seen in Fig. 1. It should be noted that resulted number of layers is two whatever the number of density levels in the original dataset will be. The main idea is to extract the sparse layer, which has one density level in which DBSCAN can work properly. Hereafter, the dense layer could be clustered by KNN algorithm depending on core points resulting from DBSCAN. The proposed method assumes that the dataset has at least two density levels. To enhance readability of the paper, we will refer to DBSCAN with preprocessed data as DP-DBSCAN.

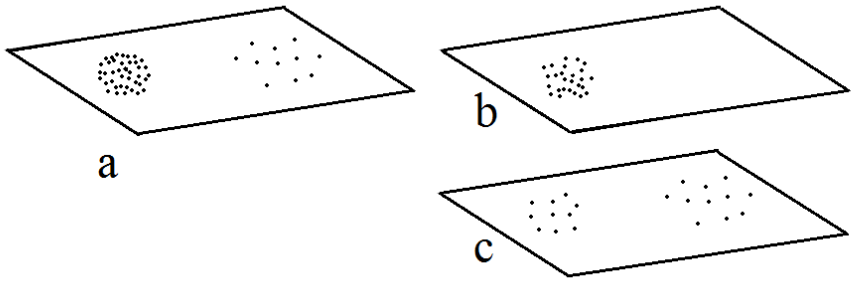


Fig. 1. Main idea of data preprocessing technique where (b) and (c) forming (a). (a) Original data. (b) Dense layer. (c) Sparse layer.

The remaining of the paper is organised as follows. Section 2 presents an overview on DBSCAN and related enhancements. The proposed algorithm will completely implemented and illustrated in Section 3. Experimental results on synthetic data and real world data were applied and tested in Section 4. Section 5 concludes this paper.

2.0 RELATED WORKS

2.1 Overview on DBSCAN

DBSCAN clustering algorithm produces a partitional clusters. It discovers high density regions which are seperated from each other by low density regions [3]. The areas with high density are expanded into clusters. A cluster (from DBSCAN algorithm point of view) is the maximal set of density-connected points [7]. The pseudo code presented in Algorithm 1 [3] illustrates the work of DBSCAN.

```
Algorithm1.  DBSCAN(points, Eps, Minpts)
Input : database D, Eps, Minpts
Output: N clusters, noise points.
1. Begin
2. Classify each point in D as core (CO), border (BR), or noise points (NO) using Minpts.
3. Eliminate NOs.
4. Make a connection between all COs which are within Eps of each other.
5. Put each set of linked COs into a distinct cluster.
6. Assign each BR to one of the clusters of its associated COs.
7. End
```

DBSCAN depends on its work on classifying data points into three categories, depending on neighboring density of any points. These categories include core points, border points, and noise points. The core points are put in the same cluster if they are close enough (within Eps distance of each other). Similarly, the border and core points are put in the same cluster if they are close enough. Noise points are eliminated.

In more details, A clusters is created if there are a sets of at least Minpts objects in a dense region with an ε (Epsilon); Minpts and ε are defined by the user. DBSCAN starts with an arbitrary starting point. The ε-neighborhood of this point is retrieved and a cluster is created if it contains sufficient number of points. Otherwise, the point is considered as noise.
Moreover, DBSCAN has several advantages like it is robust to outliers, there is no need to specify the number of clusters in data, and it is insensitive to points ordering. However, it suffers from few limitations such as the quality of DBSCAN depends on used distance measure, and it cannot cluster data well with varying densities.
The DBSCAN time complexity is $O(n^2)$, where n is the number of data points in the dataset. If spatial index structure which contains all objects is constructed beforehand, thenceforth for each data point, it is possible to compute ε-neighbor in $O(\log n)$time. Therefore, the time complexity of DBSCAN will be $O(n \log n)$.

2.2 Related Enhancements

In view of the fact that DBSCAN does not perform well when data contain various densities urged many researchers to propose multiple techniques to handle that problem in recent years. When using spatial index together with grid technique the result is GMDBSCAN [8] (Multi-Density DBSCAN Cluster Based on Grid). This method uses space-dividing technique and consider each grid as a separate part and then it estimates independent *Minpts* for every grid (part) based on its density, after that it applies multiple DBSCAN on each grid and finally, it uses distance-based method to improve boundaries.

Other researchers [9] presumed that a spatial dataset contains various point processes and clusters with difference in densities belong to various point processes. This is done by proposing new algorithm that is based on reversible jump Markov Chain Monte Carlo (MCMC) [10] strategy. The first step in this strategy, it maps each point in the data to its kth nearest neighbor, then they determine a classification threshold by a reversible jump MCMC strategy and finally, it form clusters by gathering the points whose kth nearest neighbors lie into a particular bin defined by the thresholds. In the form of constraints, Ruiz et al. [11] presented a density-based semi supervised clustering method, where they use DBSCAN to produce temporary clusters, which then combined by applying two types of constrains : Must-link constraint and cannot-link constraint.

Some approaches head toward some of data structures to handle the problem, such as using two rounds of minimum spanning trees in two phases to detect separated clusters. These separated clusters are recognized whether it is separated by distance or separated by density in the first phase. In the next phase, it performs partitioning on the sub groups resulted from the first phase called touching clusters by comparing cuts in the two rounds of minimum spanning trees [12].

Hua et al. [13] presented a new technique where firstly, the space of data is divided into a number of grids. Secondly, the space of data is divided again to get smaller partitions. This division of the space is done according to the one-dimensional or two-dimensional characteristics of the density distribution of the grid; finally, for each partition it applies an improved DBSCAN with different parameters to cluster these partitions respectively.

Deng et. Al [14] presents spatial clustering based on Delaunay triangulation. It uses a new technique to cut the edges of Delaunay triangulation into two levels: Global level and local level, they are used to find clusters spatially, where in the global effect, the clustering is applied on the features that are well separated and then after the global effect is removed, the local effect is considered.

By partitioning data into several density levels, Xiong et al. [15] have developed a new method to analyze the characteristics of these density levels, and determining Eps and Minpts differently to each density level and ultimately, applying DBSCAN multiple times, one time for each level of density.

According to Liu et al. [16], the exploitation of the spatial proximity and attributes similarity in spatial clustering will improve this clustering to detect clusters in the spatial domain. This detection is achieved by employing a modified density-based clustering method; they conclude that the objects that belong to same cluster detected by their method have a proximity in the spatial domain and similarity in an attribute domain.
In the same context, Ren et al. [17] proposed a new method called DBCAMM (density based clustering algorithm with Mahalanobis metric) that developed DBSCAN. Firstly, by replacing Euclidian distance by Mahalanobies distance metric, this metric is associated with the distribution of the data and secondly, by introducing a method to combine sub-clusters by using the information of the density of the sub-cluster.
GRPDBSCAN (Grid-based DBSCAN algorithm with referential parameters) is another solution to the problem of different densities, where the merging of multi-density based clustering and grid partition technique and the automatic generation of *Eps* and *Minpts* is performed to enhance DBSCAN [18].

Zhang et al. [19] introduced a new technique depends on four concepts: Contribution, grid technique, migration-coefficient, and tree index structure, to optimize the performance of DBSCAN to be able to discover clusters with different densities. This optimization carried out by, firstly, using grid technique to reduce the time where the algorithm will be efficient for large databases. Secondly, the optimization of the clustering's results is fulfilled by expressing the density of the grid based on the concept of contribution. Thirdly, the improving of the clustering quality is done by focusing on boundary points using migration coefficient. In M-DBSCAN (Multi density-DBSCAN) [20], neighbors did not find with a constant radius ϵ, instead the determination of neighboring radius is performed based on the data distribution around the core using standard deviation and mean values. To get the clustering results, M-DBSCAN is applied on a set of core-mini clusters where each core-mini cluster represents a virtual point lies in the center of that cluster. In M-DBSCAN, local density cluster is used as an alternative to ϵ value of DBSCAN. In this algorithm, by adding core-mini clusters that have similar mean values with a little difference determined by the standard deviation of the core, the clusters been extended.

Pathway et. al [21] proposed a parallel DBSCAN algorithm employing graph theory concepts. To form clusters, a bottom-up tree based approach is used. To break the data access order and to efficiently perform the merging process, a disjoint-set data structure is exploited. To compute local clusters in parallel, each core process is first run a sequential DBSCAN algorithm on its local points. The final clusters are obtained by merging local clusters.

Zhang et. al [22] proposed Linear DBSCAN algorithm based on LSH (Locality-Sensitive Hashing) for the purpose of devising main memory algorithm for nearest search. The advantage of using LSH is that it reduces the time complexity and the scale of data. This algorithm has time complexity O(NlogN) based on R-Tree algorithm which is better as compared to traditional DBSCAN algorithm having time complexity $O(N^2)$. Wrong points can be eliminated and approximate nearest neighbor points can be obtained. In this technique, two parts are considered. In the first part, LSH index is built and in the second part clustering is done using DBSCAN algorithm on the basis of LSH retrieval index.

Incremental DBSCAN clustering algorithm is used to handle dynamic databases. It has the ability for changing the radius threshold value dynamically. The algorithm restricts the number of the final clusters and reads the original dataset only once. At the same time, this algorithm introduces the frequency information of the attribute values. This algorithm appropriates for categorical data. The algorithm can not only overcome the impact of the inadequate of the memory when clustering the large-scale data set, but also accurately reflect the characteristics of the dataset [23].
Campello et.al [24] presented a theoretically and practically improved density-based, hierarchical clustering method (HDBSCAN), by constructing a simplified tree of important clusters and providing a clustering hierarchy. Obtaining a *flat* partition consisting of only the most important clusters is done by using a cluster stability measure, formalizing a problem of maximizing the overall stability of chosen clusters, and formulating an algorithm calculating an optimal solution to the problem.

EL-Sonbaty et.al [25] enhanced DBSCAN by partitioning the dataset to simplify the process of searching to get the neighborhoods within the specified eps threshold on the entire partition only. Consequently, this partitioning reduces the number of dataset scans as it limits the search space for core objects to the partitions rather than the whole dataset. These partitions are produced using CLARANS and then DBSCAN is used to cluster these partitions and produce dense regions for each partition.

LDBSCAN improved DBSCAN by relying on a local-density-based notion of clusters. It takes the advantage of the LOF (local outlier factor) to distinguish the noise. To find a cluster, LDBSCAN starts with a random point and retrieves the local density-reachable points from that point with respect to LOFUB, pct, and MinPts. This procedure produces a cluster when this point is a core point; otherwise, new point from the dataset will be checked [26].

3.0 DP-DBSCAN

The proposed algorithm consists of four main steps (as illustrated in Fig. 2). This section presents these steps in more details. These steps are (in order):

- Density based on recognition, which separates dataset into two regions (SD and DD) according to their densities.
- Density-biased without replacement sampling is performed on dense region (DD). The resulted sample of this step has the same density level of SD. The remaining data will be DD'.
- Do Clustering by DBSCAN step on SD and SS.

- Do clustering by KNN on DD' to nearest core points resulting from DBSCAN.

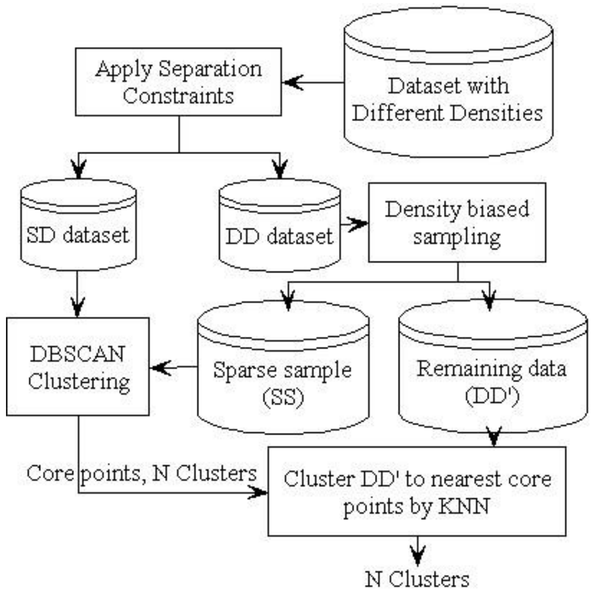


Fig. 2: Block Diagram of the proposed system.

In this paper, the data are separated into two regions. This does not mean that the data do not have more than two levels, but in this method, any data are separated into two regions even though the data consists of more levels of density. The other density levels (medium density levels) are separated to sparse and dense. Note that the proportion of points in sparse region is not equal to the proportion of points in dense region.

The main contribution of this work is the first two steps. It should be noted that the resulted number of clusters from the last two steps are N clusters. This means that the last step is just assigning every point in dense data to the nearest cluster depending on the resulted core points from the third step.

3.1 Density-Based Recognition

There are many approaches used to find spatial proximity relationships among objects of data, some of which depend on KNN to find these relations and the other depend on kind of topological relations such as Delaunay Triangulation [16]. Firstly, the distance matrix is computed for the data and then we compute a sparse matrix from the distance matrix. The sparse matrix is computed by taking *k nearest neighbors* for each object in the data and the values except this *k nearest neighbors* become zero.

The purpose of the sparse matrix is to compute the constraints that are used to recognize data in the next paragraph. To recognize data, we propose a new technique depends on applying two constraints; the first one is the Global Mean of Distances constraint, given a dataset *D*, the Global Mean of Distances is defined by Definition 1.

Definition 1: *Global Mean of Distances (GMD)* is the mean value of the distances among data objects in sparse matrix, and is computed by Eq. 1.

$$\text{GMD}=\frac{1}{n\times m}\sum_{i=1}^{n}\sum_{j=1}^{m}\boldsymbol{dist}(\boldsymbol{p}_i, \ \boldsymbol{p}_j) \tag{1}$$

Where *n* and *m* represent the dimensions of the sparse matrix and $\boldsymbol{dist}(\boldsymbol{p}_i, \ \boldsymbol{p}_j) \neq$ zero. The second constraint that is used in this stage is Local Mean of Distances, which appears in Definition 2.

Definition 2: *Local Mean of Distances (LMD)* is the mean value of k-nearest neighbors for a given object *p* and is computed by Eq. 2.

$$LM\boldsymbol{D}(\boldsymbol{p}_i)=\frac{1}{k}\sum_{j=1}^{k}\boldsymbol{dist}(\boldsymbol{p}_i\,,\,\boldsymbol{p}_j) \tag{2}$$

Where k is the number of nearest neighbors. To recognize data, we will compare the *Local Mean of Distances* of a given object *p* with the *Global Mean of Distances* of a dataset *D*, then classify the object either to be in dense region or in sparse region and that comparison is done by Definitions 3 and 4.

Definition 3: *Dense Data* is any data point $p \in \boldsymbol{D}$ that has Local Mean of Distances smaller than or equal to the Global Mean of Distances (i.e. LMD (p) ≤ GMD (D)).

Definition 4: *Sparse Data* is any data point $p \in \boldsymbol{D}$ that has Local Mean of Distances greater than the Global Mean of Distances (i.e. LMD (p) > GMD (D)).

By applying these constraints, data points will be recognized as two regions: Dense Data (DD) and Sparse Data (SD). The separation of the dataset depends on *local mean of Distances* and *global mean of Distances*. Furthermore, the parameter that affects the results of this method is the number of nearest neighbors, which is provided by the user. DD will be further processed in the next stage.

Figure 3 presents an example that illustrates this method, if we consider fish dataset, when computing GMD of this dataset its value is 0.0663. The points with red color are selected as samples to compute LMD of them and comparing its value with GMD to classify them, the labels of numbers in Fig. 3 are used as references to points in Table 1 where each label corresponds to a point in the table. As can be seen, the points with the labels (1,2,3) are classified as DD because each of them has LMD smaller than GMD, while the points with the labels(4,5,6) are classified as SD because each of them has LMD greater than GMD as that explained in Table 1.

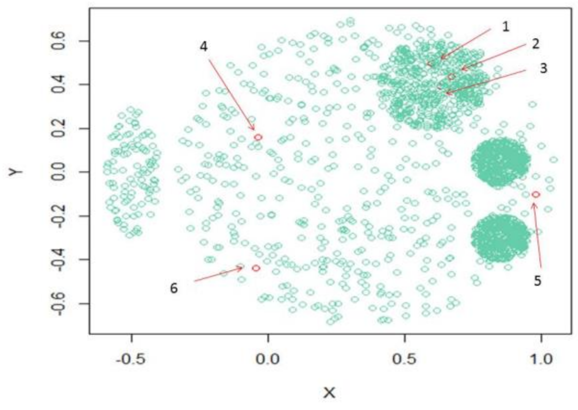


Fig. 3: Illustration of recognition method. The time complexity of this method is O ($\boldsymbol{n}^2$) because of the using of distance matrix and sparse matrix where each of them consists of $\boldsymbol{N} \times \boldsymbol{N}$ elements.

3.2 Density Biased Without Replacement Sampling

Many density-biased sampling had been proposed to address traditional sampling limitations [27]. Several density biased sampling algorithms are designed for important purposes; where oversampling dense regions and undersampling light regions are done because uniform sampling misses the rare samples [28]. In this paper, we propose a new density biased without replacement sampling approach. This approach carried out by taking sample without replacement from DD resulting from the previous stage to get a new data that have density distribution similar to that of SD. The main goal of this step is to get data having one density level; the main advantage of this approach is that it does not need to determine sample size because sample size is determined according to the density of the data. This approach could be done as illustrated in Algorithm 2.

Table 1. Local means of distances for some points.

Point label	**LMD value**	**Label of separation**
1	0.0503	DD
2	0.0621	DD
3	0.0489	DD
4	0.1750	SD
5	0.2269	SD
6	0.1586	SD

The denser the region is the lesser the value of GMD (). In other words, GMD (SD) is greater than GMD (DD) because DD points are denser than SD; the average distance between neighbors are higher in sparse region comparing with dense region. This justifies the stop condition in Line 4 of Algorithm 2. Lines 4-8 balance the density of different regions by removing points (i.e. sampling without replacement) from dense regions until GMD (SS) ≥ GMD (SD). It should be noted that modifying distance matrix does not need full recalculation as it appears in the algorithm. In each row, we need to keep indications to the k nearest neighbors. Calculation to Row_i is done if the selected sample in Line 5 represents one of k nearest neighbors to Row_i; otherwise, there is no need to do any recalculations.

```
Algorithm 2: Density biased without replacement sampling algorithm
Input: The set of dense points DD and GMD (SD)
Output: Sparse points (SS), DD'
1. Begin
2.     SS=DD
3.     Compute GMD (SS)
4.     While (GMD (SS) < GMD (SD))
5.        Get a random sample without replacement p from SS
6.        For each Row to which sample p is a KNN
7.              Recomputed KNN ($Row_i$) and LMD ($Row_i$)
8.        Recompute GMD (SS) after recalculating LMD ∀ Row(s) in Line 7
9.     END While
10.    DD'=DD−SS
11     Return (SS, DD')
12. END
```

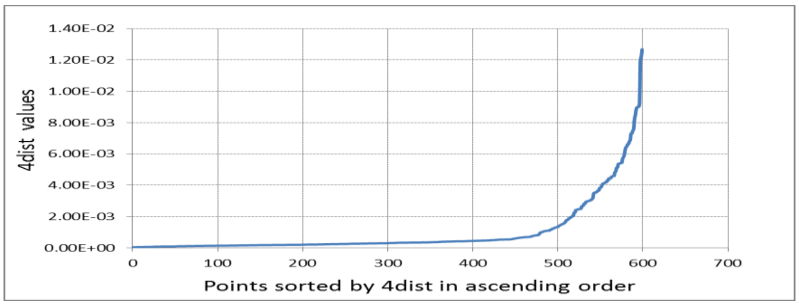
The time complexity of re-computing the sparse matrix is $O(N \times k)$ where N is the number of points and k is the number of nearest neighbors.

3.3 Applying DBSCAN on SD and SS

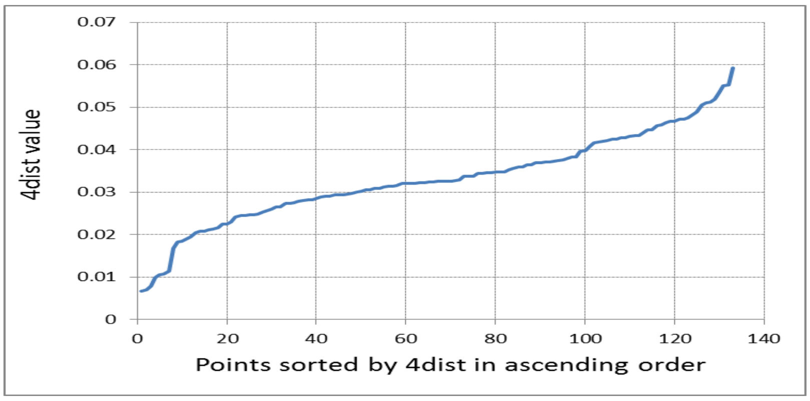
In this stage, DBSCAN will be applied on SD and SS that are resulted from the two previous stages. SD and SS have one density level where they do not contain different densities so that DBSCAN will produce good clustering with this data. To apply DBSCAN on this data, we need to determine its parameters: *Eps* and *Minpts*. The determination of these parameters is a difficult process because this algorithm is very sensitive to the setting of these parameters where any change may lead to different clustering. K-dist plot method is used to determine these parameters by computing the distance between objects and taking K-nearest neighbors for each object. Then, do sorting these distances by ascending order and plotting them. It should be noted that K-dist plot method is subjective so the knee in the curve will be the guide to the best value of *Eps*. *Minpts* value will correspond to the value of *K*.

The difference between selecting the parameters in DBSCAN and selecting them in DP-DBSCAN is that the whole data is used in DBSCAN while only sparse data (SD and SS) are used in DP-DBSCAN and consequently the curve in k-dist plot will be different. Fig. 4. Illustrate the difference between k-dist plot for the whole data and k-dist plot for sparse data for Squares dataset. As can be seen, the value of Eps in each k-dist plot is different according to the position of sharp change in the curve.

3.4 Applying KNN on DD' and Core Points



(a)



(b)

Fig. 4: K-dist plot of Squares dataset (a) for all data points (b) for sparse data points only.

In the previous stage, the result of DBSCAN is a set of core points, border points, and noise points. After applying sampling in Subsection 3.2, set of dense points (DD') were remaining. To cluster these points, KNN is used. In KNN, the distance between each point in DD' and all core points will be computed, then by taking k-nearest neighbors for each dense point and selecting the nearest neighbor with the major cluster ID, the dense point will be assigned to the cluster of that major cluster ID. Algorithm 3 presents the modified KNN clustering algorithm. The logic behind using KNN is to give the dense points remaining from sampling to the clusters that are resulted from DBSCAN to get complete clusters for the whole dataset points. After clustering all the dense points, we will get the final number of clusters for the proposed algorithm. Equation in Line 5 [3] classifies each instance according to its nearest core neighbors, where v is the class label, y_i is the class label for one of the nearest core neighbors, and $I(.)$ is an indicator function that returns 1 if its parameter is true and 0 otherwise.

```
Algorithm 3: Modified KNN Classification algorithm
Input: DD' and Core points
Output: Classification of DD'
1. Begin
2.     CP=set of core points resulting from DBSCAN
3.     Foreach point ($P_i$) $\in$ $DD'$
4.         $C$ = (k-nearest neighbors ($P_i$)) $\in$ $CP$
5.         $y' = \underset{v}{argmax} \sum_{(xi,yi)\in C} I(v = yi)$
6.     EndFor
7. EN
```

4.0 EXPERIMENTAL RESULTS

In this section, several two-dimensional synthetic datasets and three real world datasets are used to confirm the effectiveness of the proposed algorithm. These synthetic datasets contain Squares dataset, Gaussian dataset, X dataset, and Fish dataset (as can be seen in Fig. 5) that are generated by R package [29] software, Jain dataset is gained from http://cs.joensuu.fi/sipu/datasets/jain.txt while real world datasets containing Blood Transfusion Service Center, Breast Cancer Wisconsin, and Vertebral Column are gained from UCI [30]. All of these datasets have varying density property. The performance quality of the resulting clustering is validated by using some of the most popular external measures [20], which are Accuracy [17], Entropy [3], Purity [3], and F-Measure [31].

Data recognition depends on the selected number of nearest neighbors that is used to build sparse matrix and consequently effects on the values of GMD and LMD. In other words, to recognize data with highest accuracy, number of nearest neighbors should be selected carefully. This is because there is a variance in density. Table 2 illustrates the accuracy of synthetic data recognition where the data is recognized into two groups of density. The accuracy is computed by using class labels, which indicates whether the point is dense or sparse before performing the recognition method, then comparing this label with the resulting label from recognition method by using Accuracy measure as in Equation 3. As can be noted from Table 2, the method of recognition could recognize data with high accuracy for all of the synthetic datasets and this accuracy could be increased or decreased, depending on the value of k.

$$Accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ predictions} \tag{3}$$

Data recognition technique could work on dataset with multi levels of densities. In order to further prove the efficiency of the method of data recognition, experiments carried out on data with more than two levels of density (data have regions with high density, medium density, and low density). The results show that the accuracy of the recognition is excellent. The results of recognition is shown in Fig. 6(a).

To get the best results for DBSCAN algorithm, K-dist plot is applied to configure the parameters of DBSCAN, where K-dist plot is plotted for sparse data only (SD+SS). After trying many values from these curves, the best values of *Eps* are 0.09, 0.08,0.08,(0.08-0.09), and 0.05 for Jain dataset, Squares dataset, Gaussian dataset, X dataset, and Fish dataset respectively when K value equals to 4, so the value of Minpts is 4. The DBSCAN parameters are optimized for each dataset separately and for both DBSCAN and the proposed.
Figure 6 depicts all the stages of DP-DBSCAN for Fish dataset ranging from the stage of recognizing data to the final clustering process. Fig. 6(a) shows the recognition stage in which the algorithm recognizes the dense region from sparse region. Fig. 6(b) presents the dataset after doing the biased sampling. Then, the results of applying DBSCAN algorithm on the dataset appearing in Fig. 6(b) can be seen in Fig. 6(c). The last stage of DP-DBSCAN is to cluster the remaining points to get Fig. 6(d). To compare the results of clustering visually, the clustering result of original DBSCAN on Fish dataset is depicted in Fig. 6(e). From Fig. 6, the results of DP-DBSCAN outperform the results of DBSCAN algorithm.

Comparison by using performance metrics is a basic step. In order to compare the results of the clustering by using different metrics, confusion matrices of the clusterings had been used. Vertebral column dataset contains two classes and DP-DBSCAN produces two clusters, for Blood transfusion service center dataset, which contains two classes, DP-DBSCAN gives two clusters comparing with DBSCAN, which gives ten clusters. Breast Cancer Wisconsin dataset consists of two classes and both of DP-DBSCAN and DBSCAN give two clusters but with different distribution of objects, which makes the accuracy of the proposed algorithm higher than the accuracy of DBSCAN. The most associative metric with clustering process is Accuracy but the other metrics are also used. Table 3 shows the results of the Purity values by the proposed algorithm are 1, 0.9980, 0.9972, 0.7142, 0.7290, and 0.7666 for Squares, Gaussian, Jain, Fish, Vertebral Column, and Blood Transfusion service center respectively which are better than Purity values by DBSCAN where the purity value that is closer to 1 is the better. The best Entropy values by the proposed algorithm are 0, 0.0188, 0.6195, 0.7717, and 0.3058 for Squares, Gaussian, Fish, Vertebral Column, and Breast Cancer Wisconsin respectively which are better than Entropy values for the mentioned datasets using DBSCAN where the Entropy closer to zero is the best as that presented in Table 3. The results show that the values of F-Measure for the proposed algorithm are better than that of DBSCAN for most of the datasets. The F-measure for the real datasets represented by Breast Cancer Wisconsin dataset and Blood Transfusion service center dataset has better results than that of DBSCAN where the higher the value of F-Measure is the best.

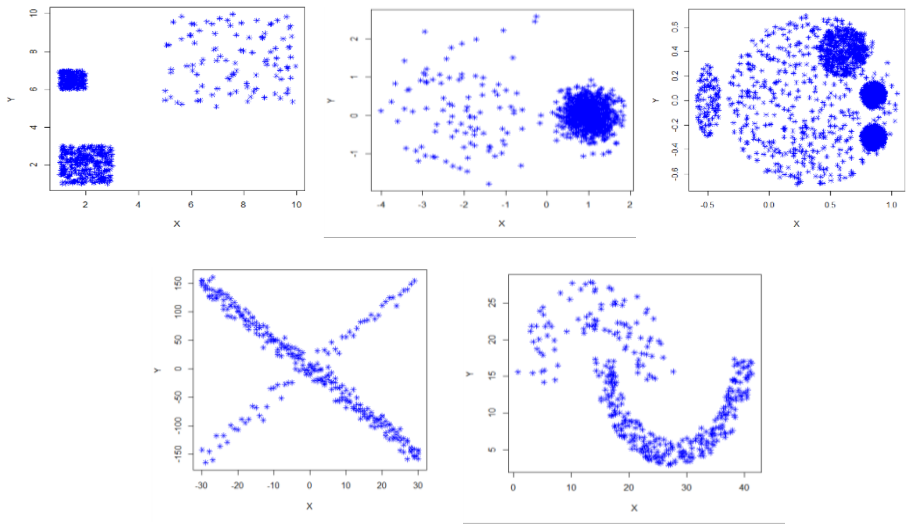


Fig. 5: Synthesized dataset used in this paper: (a) Squares dataset (b) Gaussian dataset (c) Fish dataset (d) X dataset (e) Jain dataset.

Table 2. Accuracy of data separation process on synthetic dataset

Dataset	Size	Sparse data (SS+SD)	Dense data (DD')	Accuracy
Squares	600	144(24.0%)	456(76.0%)	98%
Gaussian	1100	208(18.9%)	892(81.1%)	91.9%
Fish	2100	758(36.1%)	1342(63.9%)	94.1%
X	305	119(39.0%)	186(61.0%)	92.1%
Jain	373	179(48.0%)	194(52.0%)	96.8%

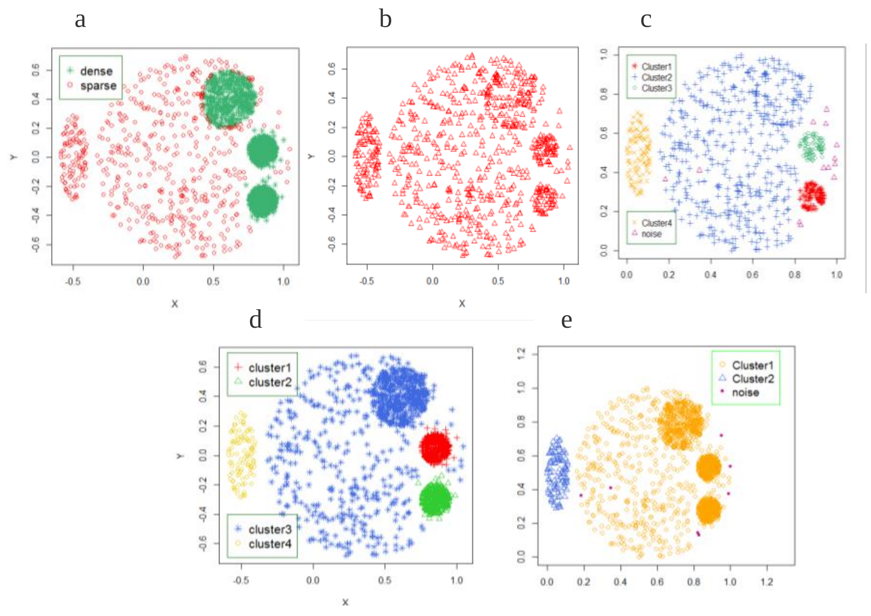


Fig. 6: Results of Fish dataset (a) data recognition (stage 1) (b) SS (stage 2) (c) DBSCAN on SD(stage 3) (d) KNN(stage 4)(e) Original DBSCAN on original

A set of comparisons have been done between the proposed algorithm and three of traditional algorithms. The comparison is done based on the choosing of algorithms according to their ability to deal with data with different densities or not. The algorithms that suffer from the problem of different densities represented by k-means and fuzzy c-means and that they have the ability to deal with different densities, which are represented by EM (Expectation Maximization) algorithm. The comparison between these algorithms is carried out by applying them on the synthetic and real datasets to confirm the efficiency of the proposed algorithm compared with the traditional algorithms. On average, the best accuracy was for the proposed algorithm (85.1%) followed by K-means (83.4%), EM (81.7%), DBSCAN (77.6%), and then FCM (66%).

Furthermore, we present a comparison to show the extent of achieved enhancement by the proposed algorithm. To make the comparison, we use two popular algorithms enhanced DBSCAN to a certain extent. These algorithms are HDBSCAN [24] and OPTICS [4] clustering algorithms. In this comparison, we use converged parameters for both used algorithms (whether eps value or Minpts value according to number of parameters that are used by each algorithm where HDBSCAN uses only Minpts to cluster the dataset). According to the illustrated results in Table 4, the proposed algorithm is better than HDBSCAN and OPTICS in two out of five datasets (i.e. Gaussian and Jain datasets) from aspect of accuracy and outperforms HDBSCAN in one dataset (i.e. Squares dataset). However, HDBSCAN and OPTICS algorithm was the best in one dataset for each (i.e. fish and X datasets respectively). From these results, it can be concluded that the proposed algorithm is competitive to the existing solutions.

Table 3. Clustering evaluation for DBSCAN and the Proposed Algorithms

Dataset	DBSCAN algorithm			Proposed algorithm		
	Entropy	Purity	F-Measure	Entropy	Purity	F-Measure
Squares	0	0.687	0.990	0	1	1
Gaussian	0.019	1.169	0.998	0.019	0.998	0.999
Fish	1.898	0.286	0.718	0.620	0.714	0.881
X	0.498	0.885	0.874	0.662	0.800	0.608
Jain	0	0.995	0.996	0.022	0.997	0.945
Vertebral Column	0.779	0.529	0.837	0.778	0.729	0.760
Breast Cancer	0.717	0.660	0.877	0.306	0.326	0.946
Blood Transfusion service center	0.708	0.710	0.796	0.784	0.767	0.885

Table 4. Comparison results with different variations of DBSCAN Algorithm

Datasets	Proposed Algorithm	HDBSCAN	OPTICS
Squares	100	98.99	100
Fish	76.19	93.99	93.47
X data	63.27	53.97	79.93
Gaussian	99.27	96.45	96.90
Jain	99.73	90.32	87.36

In order to determine the time complexity of the proposed algorithm, we need to measure time complexity of its components using asymptotic notation. Data recognition and sampling together take $\boldsymbol{O(n^2)}$ where *n* represents the number of points in the dataset. DBSCAN also takes $\boldsymbol{O(n^2)}$. The time to do clustering with KNN could be neglected because the number of points to be clustered (compared with total number of points) is too small. Then the time complexity of the proposed algorithm is the addition of the time complexity of recognition and sampling, and the time complexity of DBSCAN, which is still $\boldsymbol{O(n^2)}$. Therefore, we kept time complexity of DBSCAN with the new proposed algorithm without any increasing.

5.0 CONCLUSIONS

A new data sampling technique to enhance DBSCAN is presented in this paper by extracting a sample (which has one density level) from a dataset having different density levels to be suitable for DBSCAN clustering algorithm. The data with the proposed sampling algorithm is recognized as regions based on its density. Firstly, the data are perfectly recognized into two density levels; hereafter, the proposed sampling technique is employed to mitigate the density of dense region and get data with only one density level (sparse data). The sampling results are very effective to produce sparse data from dense data. The experiments performed on synthetic and real world data show that the proposed algorithm enhanced the performance of DBSCAN clustering algorithm on data with different density levels for large extent.

6.0 REFERENCES

[1] Jain, A.K., Murty, M.N. and Flynn, P.J," Data Clustering: A Review" *ACM Computing Surveys*, vol. 31, no. 3, 1999, pp. 264–323.

[2] Miller, H.J, "Geographic Data Mining and Knowledge Discovery". The Handbook of Geographic Information Science, 2008, pp. 651.

[3] Tan P.-N., Steinbach M. and Kumar V., "Introduction to Data Mining", Pearson Education, 2006.

[4] Ankerst, M., Breunig, M.M., Kriegel,H.P. and Sander, J, "OPTICS : Ordering Points To Identify the Clustering Structure", *Proceedings of the ACM SIGMOD International Conference on Management of Data*, vol. 28, no. 2, 1999, pp. 49-60.

[5] Hinneburg, A. and Keim, D, "A General Approach to Clustering in Large Databases with Noise", *Knowledge and Information Systems*, vol. 5, no. 4, 2003, pp. 387-415.

[6] Ester, M., Kriegel, HP, Sander, J. and Xu, X., "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", *Second International Conference on Knowledge Discovery and Data Mining*, 1996, pp.226–231.

[7] Han, J., Kamber, M. and Pei, J. "Data Mining: Concepts and Techniques", ed. 2nd, the Morgan Kaufmann Series in Data Management Systems, 2005.

[8] Xiaoyun, C, Yufang, M, Yan, Z and Ping, W., " GMDBSCAN: Multi-Density DBSCAN Cluster Based on Grid", *IEEE International Conference on E-Business Engineering*, ICEBE'08 - Workshops: AiR'08, EM2I'08, SOAIC'08, SOKM'08, BIMA'08, DKEEE'08, 2008, pp. 780–783 .

[9] Pei, T., Jasra, A., Hand, D.J., Zhu, A.X. and Zhou, C., " DECODE: A New Method for Discovering Clusters of Different Densities in Spatial Data", *Data Mining and Knowledge Discovery*. Vol. 18, no. 3, 2009, pp. 337-369.

[10] Christophe A., Nando D. F., Arnaud D., and Michael I. J.,"An Introduction to MCMC for Machine Learning", *Machine Learning*, vol. 50, 2003, pp. 5–43.

[11] Ruiz, C., Spiliopoulou, M. and Menasalvas, E., " Density-Based Semi-Supervised Clustering", *Data Mining and Knowledge Discovery*, vol. 21, no. 3, 2010, pp. 345–370 .

[12] Zhong, C., Miao, D. and Wang, R., "A Graph-Theoretical Clustering Method Based on Two Rounds of Minimum Spanning Trees", *Pattern Recognition*, vol. 43, no.3, 2010, pp. 752–766.

[13] Hua, Zand Zhenxing, W., "Clustering Algorithm Based on Characteristics of Density Distribution", second *International Conference on Advanced Computer Control* (ICACC), vol. 2, 2010, pp. 431–435 .

[14] Deng, M., Liu, Q., Cheng, T. and Shi, Y., "An Adaptive Spatial Clustering Algorithm Based on Delaunay Triangulation", Computers, *Environment and Urban Systems*, vol. 35, no. 4, 2011, pp. 320–332.

[15] Xiong, Z. and Chen, R., "Multi-Density DBSCAN Algorithm Based on Density Levels Partitioning", *Journal of Information and Computational Science*, vol. 10, no. 9, 2012, pp. 2739-2749.

[16] Liu, Q., Deng, M., Shi, Y. and Wang, J., "A Density-Based Spatial Clustering Algorithm Considering Both Spatial Proximity and Attribute Similarity", *Computers and Geosciences*, vol. 46, 2012, pp. 296–309.

[17] Ren, Y., Liu, X. and Liu, W., "DBCAMM: A Novel Density Based Clustering Algorithm via Using the Mahalanobis Metric", *Applied Soft Computing Journal*, vol. 12, no. 5, 2012, pp. 1542–1554 .

[18] Darong, H. and Peng, W., "Grid-Based DBSCAN Algorithm with Referential Parameters", *International conference on Applied Physics and Industrial Engineering*, *Physics Procedia*, vol.24, 2012, pp.1166–1170 .

[19] Zhang, L., Xu, Z. and Si, F., "GCMDDBSCAN: Multi-Density DBSCAN Based on Grid and Contribution", 2013 *IEEE 11th International Conference on Dependable*, *Autonomic and Secure Computing*, 2013, pp. 502–507 .

[20] Amini, A., Saboohi, H., Herawan, T. and Wah, T.Y., " MuDi-Stream: A Multi Density Clustering Algorithm for Evolving Data Stream", *Journal of Network and Computer Applications*, 2014, pp. 1–16.

[21] Patwary Md., Palsetia, D., Agarwal, A., Liao,W.A., Manne, F., Choudhary, A., "A New Scalable Parallel DBSCAN Algorithm Using the Disjoint-Set Data Structure", *In Proceeding SC '12 Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, IEEE 2012*.

[22] Wu, Y. Jou, J. Zhang, X., "A Linear Dbscan Algorithm Based On Lsh", Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, Hong Kong, 2007, pp. 19-22.

[23] Xiaoke Su, Yang Lan, Renxia Wan, and Yuming, "A Fast Incremental Clustering Algorithm", *international Symposium on Information Processing (ISIP'09), Huangshan, P.R.China, August*-21-23, 2009, pp. 175-178.

[24] Campello R.J.G.B., Moulavi D., Sander J., "Density-Based Clustering Based on Hierarchical Density Estimates". In: Pei J., Tseng V.S., Cao L., Motoda H., Xu G. (eds) Advances in Knowledge Discovery and Data Mining. PAKDD 2013. Lecture Notes in Computer Science, vol. 7819, 2013, Springer, Berlin, Heidelberg.

[25] Yasser, EL-Sonbaty, M.A.Ismail, Mohamed. Farouk, "An Efficient Density Based Clustering Algorithm for Large Databases", *proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI*), 2004.

[26] Lian Duan, Deyi Xiong, Jun Lee, Feng Guo, "A Local Density Based Spatial Clustering Algorithm with Noise", *IEEE International Conference on Systems, Man, and Cybernetics, October 8-11*, 2006.

[27] Nanopoulos, A., Theodoridis,Y., and Manolopoulos,Y., " Indexed-based density biased sampling for clustering applications", *Data and Knowledge Engineering*, vol. 57, no. 1, 2006, pp. 37-63.

[28] Palmer, C.R and Faloutsos, C., "Density Biased Sampling", *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data* - SIGMOD '00, 2000, pp. 82–92

[29] R Core Team, " R: A language and environment for statistical Computing". R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/, 2015.

[30] Lichman, M., UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science, 2013.

[31] Rijsbergen C. J. Van, "Information Retrieval", ed. 2nd, Butterworth-Heinemann, 1979.