Interdisciplinary Journal of Information, Knowledge, and Management

An Official Publication of the Informing Science Institute InformingScience.org

IJIKM.org

Volume 15, 2020

IDCUP ALGORITHM TO CLASSIFYING ARBITRARY SHAPES AND DENSITIES FOR CENTER-BASED CLUSTERING PERFORMANCE ANALYSIS

Saud Altaf*	Engineering, Computer and Mathematical Sciences, Auckland University of Technology, New Zealand	saltaf@aut.ac.nz
Muhammad Waseem Soomro	School of Professional Engineering, Manukau Institute of Technology, New Zealand	mwaseem@manukau.ac.nz
Laila Kazmi	Shaheed Zulfikar Ali Bhutto Institute of Science and Technology, Islamabad Pakistan	1773132@szabist-isb.pk

* Corresponding author

ABSTRACT

Aim/Purpose	The clustering techniques are normally considered to determine the significant and meaningful subclasses purposed in datasets. It is an unsupervised type of Machine Learning (ML) where the objective is to form groups from objects based on their similarity and used to determine the implicit relationships between the different features of the data. Cluster Analysis is considered a significant problem area in data exploration when dealing with arbitrary shape problems in different datasets. Clustering on large data sets has the following challenges: (1) clusters with arbitrary shapes; (2) less knowledge discovery process to decide the possible input features; (3) scalability for large data sizes. Density-based clustering has been known as a dominant method for determining the arbitrary-shape clusters.
Background	Existing density-based clustering methods commonly cited in the literature have been examined in terms of their behavior with data sets that contain nested clusters of varying density. The existing methods are not enough or ideal for such data sets, because they typically partition the data into clusters that cannot be nested.

Accepting Editor Harry T Fulgencio | Received: December 3, 2019 | Revised: April 6, 2020 | Accepted: April 22, 2020.

Cite as: Altaf, S., Soomro, M. W., & Kazmi, L. (2020). IDCUP algorithm to classifying arbitrary shapes and densities for center-based clustering performance analysis. *Interdisciplinary Journal of Information, Knowledge, and Management*, *15,* 91-108. https://doi.org/10.28945/4541

Methodology	A density-based approach on traditional center-based clustering is introduced that assigns a weight to each cluster. The weights are then utilized in calculating the distances from data vectors to centroids by multiplying the distance by the centroid weight.
Contribution	In this paper, we have examined different density-based clustering methods for data sets with nested clusters of varying density. Two such data sets were used to evaluate some of the commonly cited algorithms found in the literature. Nested clusters were found to be challenging for the existing algorithms. In utmost cases, the targeted algorithms either did not detect the largest clusters or simply divided large clusters into non-overlapping regions. But, it may be possible to detect all clusters by doing multiple runs of the algorithm with different inputs and then combining the results. This work considered three challenges of clustering methods.
Findings	As a result, a center with a low weight will attract objects from further away than a centroid with higher weight. This allows dense clusters inside larger clusters to be recognized. The methods are tested experimentally using the K-means, DBSCAN, TURN*, and IDCUP algorithms. The experimental results with different data sets showed that IDCUP is more robust and produces better clusters than DBSCAN, TURN*, and K-means. Finally, we compare K-means, DBSCAN, TURN*, and to deal with arbitrary shapes problems at different datasets. IDCUP shows better scalability compared to TURN*.
Future Research	As future recommendations of this research, we are concerned with the exploration of further available challenges of the knowledge discovery process in clustering along with complex data sets with more time. A hybrid approach based on density-based and model-based clustering algorithms needs to compare to achieve maximum performance accuracy and avoid the arbitrary shapes related problems including optimization. It is anticipated that the comparable kind of the future suggested process will attain improved performance with analogous precision in identification of clustering shapes.
Keywords	clustering, density-based, large data sets, parameter, IDCUP, arbitrary shapes

INTRODUCTION

With the rapid evolution of great volumes of data accumulated in various application areas, such as Geographical Information Systems (GIS), remote sensing, medical applications, satellite image analysis, and micro array gene expression data, efficient clustering methods are in great demand. Clustering is a discovery procedure in data that coverts it into groups of partitions such that comparing data points inside a cluster reveals much similarity between each other, but is very disparate when compared to points in other cluster groups (Wei & Zhu, 2014).

Density-based clustering has been known as a dominant method for determining the arbitrary-shape clusters. Briefly, density-based clustering approach groups the data samples into a set of associated dense factors that separated by low density data sets. Density-based clustering applies a local cluster criterion and are very prevalent for mining clusters with arbitrary shape. However, the other two challenges still remain in most existing clustering algorithms (Pai & Chang, 2013). The goal of this paper is to explore an Iterative Density-Based Clustering Method (IDCUP) to meet the current challenges. We meet the second challenge by reducing input parameters in density-based partitioning algorithm. We solve the third challenge by means of pruning and hashing techniques. Most clustering algorithms are designed for clusters with similar shapes, similar sizes, or similar densities. However, a data set may consist of clusters with different shapes, sizes, or densities.

Clustering is a process of unsupervised ML where identifies the group objects based on their similarity. Organizing the objects into groups allows us to draw conclusions based on the groups that were discovered (Agarwal & Mehta, 2019). Clustering is particularly useful for exploratory data analysis and in machine learning contexts, including data mining and image segmentation. In such contexts, it is often important to make as few assumptions about the data as possible. This is where cluster analysis as a form of unsupervised machine learning is particularly appropriate. In addition to suggesting new hypotheses and testing existing ones, clustering maybe used to make predictions based on groups. The authors (Zhang & Yuan, 2018) illustrate this with an example of patients with infectious diseases, and their reaction to drug therapy. First, all patients are clustered based on how they react to different medications. For a new patient, we can then determine the best medication by finding the closest cluster. Several existing methods can be used to attempt to solve such data sets (Fahy et al., 2019). The problem is that while they are typically good at solving clusters of arbitrary sizes and shapes, they struggle with nested clusters. Instead, they usually require the groups to be at least somewhat separated from each other (Wang, et al., 2018).

The contribution of our method is: (1) the algorithm divides the data set into clusters based on different densities; (2) it automatically determines the optimal neighborhood radius for each density group; and (3) pruning techniques and harsh functions are used to make the algorithm more efficient and scalable. Experiments show that IDCUP is more robust than DBSCAN (Pietrzykowski, 2017), TURN* (Yang et al., 2018), and K-means (Lammersen et al., 2014) with better clustering results. IDCUP is much more efficient and scalable than other test algorithms.

This paper is organized six sections as follows. Section (2) provides an overview of the state-of-art related work that leads to motivation and some observation in section (3). We briefly state the enhanced DBSCAN and IDCUP algorithms in section (4). Experimental results based on different discussed algorithms and conclusion of this paper is present in following sections.

LITERATURE REVIEW

Generally, different clustering methods can be grouped in numerous ways (Agarwal & Mehta, 2019). According the cluster structure, clustering can be subdivided into two types of data clustering. Hierarchical clustering is a layered classification of data sets partitions, while, the partitioning clustering belong to single data sets partition.

DATA PARTITIONING TECHNIQUES

Data partitioning clustering approaches are normally producing a series of partition of the data sets to retrieve the regular groups that existing in the data. Partitioning clustering techniques can be further characterized into distance-based clustering and density-based clustering according to the definition of similarity measure. This partitioning technique divides the data sets into k number of subsets, in which the similar data points are closer to each other in the same cluster than the data points in preceding clusters. The greatest conventional distance-based partitioning approaches are *k-medoids* and k-means, where every clustering group has a significant gravity central point. The time complexity of K-means method is $O(n)$ as every iteration is $O(n)$ and just a fixed series of targeted iterations is selected and processed further (Guo & Zhang, 2014).

However, there are several problems with distance-based partitioning methods: (1) k is the input parameter and needs to be predetermined; (2) the methods are only appropriate for clusters with spherical outlines; (3) they are not appropriate for those clusters which are very diverse in scope; and (4) they are not robust to the selection of the preliminary partition and may congregate to a local minimum of the criterion function (f) value if the initial partition is not appropriately chosen (Wen et al., 2017).

In density-based clustering, targeted clusters are opaque and compressed boundaries of points in the data area that are split by manifest noise and low density areas (Agarwal & Mehta, 2019). In this

method, clusters are observed as a related dense part of data sets, which propagates in several directions according to density. Density-based clustering can characteristically identify and select those clusters in which arbitrary type shapes exists without pre-determined groups of clusters. However, density-based clustering is very penetrating to density-related constraints. The most typical density-based clustering algorithm is DBSCAN (Zhu et al., 2014) in which each cluster is a highest predictable connected facts. The facts are associated each other when they are reachable. DBSCAN is very sensitive to input features, the core radius *(r)* and a minimum number of neighbors *(k)* within the optimal core neighborhood. Different values of r and k lead to different clustering results.

HIERARCHICAL CLUSTERING

Different clustering algorithms are efficiently presented in (Mohebi et al., 2015) to show the importance of the clustering method when arbitrary shapes are uncertain in terms of their behavior with data sets. Another paper (Ram et al., 2010) presented clustering algorithms from a categorized disintegration of a given data set. Its decomposition is characterized by a number of dendrograms that collectively rupture *X* into reduced subclasses until an individual subset contains an individual single point. Hierarchy clustering consists of every stages or level of a tree that signifies a clustering of value *X*. Hierarchical methods are suppler than partitioning algorithms. It doesn't need input attributes from users. However, the computational densities of hierarchical clustering algorithms are complex compared with other partitioning methods. An efficient method is presented by (Heredia & Mor, 2019) to enhance the DPC technique and its procedures. A clustering problem was discussed when clusters came close to each other or even merged; it showed some delay to become part of each other which degraded the performance efficiency of the clustering algorithm. To analyze the data, they proposed collective clustering for all available datasets. The spatial-based density partitioning method was used to test datasets. Finally, they used "solve-a-puzzle" approach to achieve the required accuracy rate for determining the arbitrary-shape clusters.

PARAMETER REDUCTION

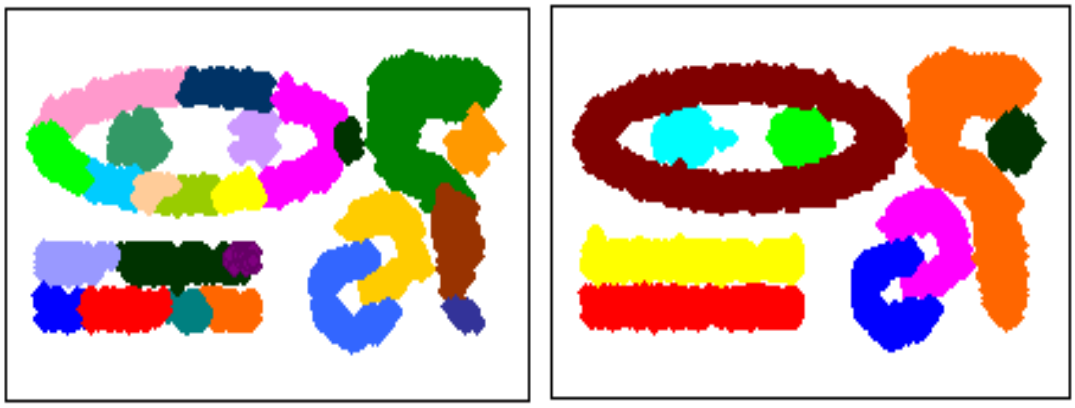
Parameter reduction has been a big challenge in clustering area. There have been many efforts to make clustering process parameter-free. Basically there are two primary approaches: (1) hierarchical clustering and (2) parameter reduction for partitioning clustering. As early as 1987, Dubes explored to decide the group of targeted clusters for the k-means method by Monte Carlo experiments (Ding et al., 2015). Given a data set, the algorithm first creates sequences of partitions and then compares adjacent partitions in terms of internal indices, such as Davies and Bouldin index and a new modification of the Hubert Γ statistics (MH) (Hou et al., 2013). In case of MH, if there is a significant difference between two adjacent partitions, then one of the two partitions with much higher value of MH will be the optimal clustering result. This method is very computationally expensive, and not suitable for big data sets.

TURN* (Zhu et al., 2014) is one of the most recent research methods for input parameter reduction. TURN* focuses on reducing the input parameter, the optimal core radius, of a density-based partitioning method. It primarily reduces the core radius so diminutively that each data point turns out to be a noise, which forms the first sequence of clusters (Zhu et al., 2014). Then the core radius is doubled to get the second sequence of clusters. The process continues doubling the core radius and generates more sequences of clusters until the number of clusters and other validation criteria stabilize, which is called a turn point (Yen et al., 2017).

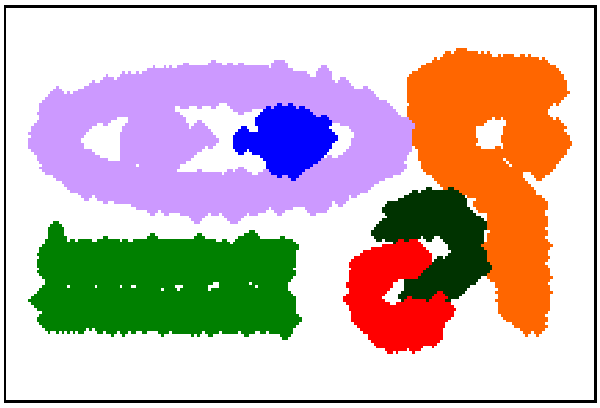
MOTIVATIONS AND OBSERVATIONS

In this research, two input parameters in DBSCAN are considered as shown in Figure 1: the optimal neighborhood radius, *r*, and the minimum number of neighbors, *k*. In fact, *k* is the size of the least group of cluster and is set to 4 in DBSCAN (Limwattanapibool & Arch-int, 2017). TURN* also sets it to a fixed value (Gui & Cheng, 2013). By experiments, we found the clustering results are similar

for a big range of k except when there are chaining noises between clusters. On the other hand, a small change of *r* can lead to a very different result. Figure 1 shows three clustering results with *r* = *7*, 8, and 9 respectively. When *r* = *7*, there are too many clusters. When r = 8, nine clusters are clearly separated. When *r* = *9*, three pairs of clusters are merged together.



(a) r = 7 **(b) r = 8**



(c) r = 9

Figure 1: Clustering results with different neighborhood *radii*

OBSERVATIONS ON DATA SET WITH THREE DENSITIES

Observation 1: We define *R* as the minimum neighborhood radius of *x* with *k* neighbors (*k* = *7*). The points are then arranged based on R in ascending order. The *R-x* shows the cleanness of the data which shows the distribution of the data set DS1 and its *R-x* graph. *DS1* is reproduced from a data set used by *CHAMELEON* (Wen et al., 2017). In order to analysis our proposed algorithm; we include additional facts in the three groups of clusters on the left top corner. The range of *DS1* is *16.5K*. As we can observe from Figure 2, if there is noise in the data set, there will be a turning point in the *R-x* grid where *R* begins to modify vividly (Pietrzykowski, 2017). The experimental results present mainly points on the right side of the turning point as noise. If the data set is uncontaminated, there will be no turning point in the graph.

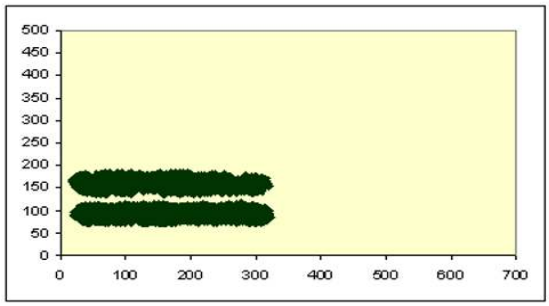
Observation 2: Set the neighborhood radius *r* as the radius at the turning point, *R*. We analyses the group of neighbors for every point within *R*, denoted as *K*, arrange the targeted points in reverse order, and acquire the arranged *K-x* graph.

Observation 3: We partition the sorted data set based on *K-x* graph above. We partition it into three subsets at two "knees". The two "knees" are at positions of X_{10052} and X_{16558}. Therefore, the three partitions are (a) Partition X_0 - X_{10052}, (b) Partition X_{10053} – X_{16558}, and (c) Partition X_{16559} – X_{17524}.

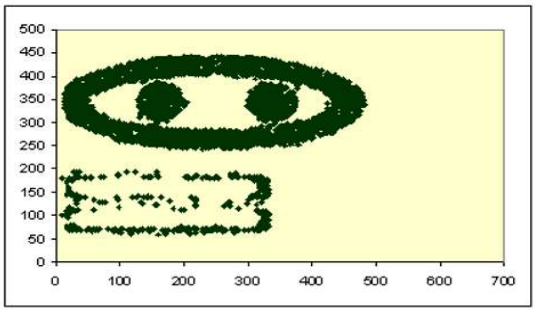
OBSERVATIONS ON DATA SET WITH FOUR DENSITIES

This section shows experiments on another data set, *DS2* with four different density groups. The sorted *K- x* graph. As we can see there are three "knees" in the graph to divide the data set into four different density groups.

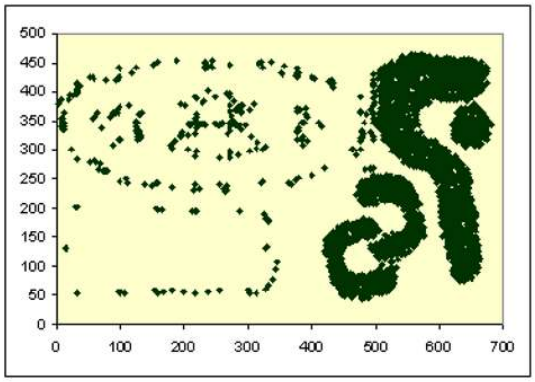
We then partition the data set into four subsets at three "knees" in Figure 2. The three "knees" are at positions of $X_{14800,}$ $X_{25000,}$ and X_{29800}. Therefore, the four partitions are (a) X_1 – X_{14800}, (b) X_{14801} – X_{25000}, (c) X_{25001} – X_{29800}, and (d) X_{29801} – X_{30500}, which are shown in Figure 2. Partition (a) consists of the densest clusters; partition (b) consists of the second densest clusters; partition (c) consists of the third densest clusters; and partition (d) is mainly noise. In summary, if a data set consists of η different density subsets, there will be η-*1* "knees" in the sorted *K-x* graph, which divide the data set in η density groups.



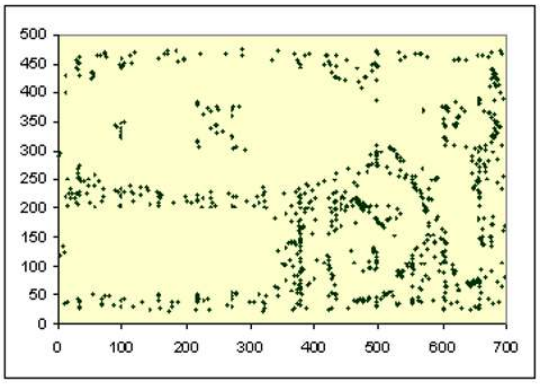
(a) The densest group: X_1 – X_{14800}



(b) The second densest: X_{14801} – X_{25000}



(c) The third densest: X_{25001} – X_{29800}



(d) Noise: X_{29801} – X_{30500}

Figure 2: Partitions of the sorted data set *DS2* by three "knees"

We then partition the data set into four subsets at three "knees" in Figure 2. The three "knees" are at positions of $X_{14800,}$ $X_{25000,}$ and X_{29800}. Therefore the four partitions are (a) X_1 - X_{14800}, (b) X_{14801} - X_{25000}, (c) X_{25001} - X_{29800}, and (d) X_{29801} - X_{30500}. Partition (a) consists of the densest clusters; partition (b) consists of the second densest clusters; partition (c) consists of the third densest clusters; and partition (d) is mainly noise. In summary, if a data set consists of η different density subsets, there will be η-*1* "knees" in the sorted *K-x* graph, which divide the data set in η density groups.

ITERATIVE DENSITY-BASED CLUSTERING METHOD

As mentioned in the previous section, IDCUP is hierarchical but not in the sense of the traditional hierarchical clustering. Rather we divide the data set into groups based on density and cluster the groups further using density-based clustering (Ding et al., 2015; Yen et al., 2017). In the next section, we developed MINR to divide the data set into density groups and calculate the optimal neighborhood radius for each group. In this section, we integrate MINR with e-DBSCAN into an iterative IDCUP. Given the optimal neighborhood radii, $r_1, r_2 \ldots r_m$ for m density groups, which are calculated in MINR, we start clustering using e-DBSCAN iteratively. First set $r = r_1$, the densest cluster(s) are formed (see Figure 3).

We then set $r = r_2$, and only process those uncluttered points. The second densest cluster(s) are obtained (see Figure 4). The process continues until $r = r_m$. The sparser cluster is formed at the later stage. The remaining unclustered points are noise.

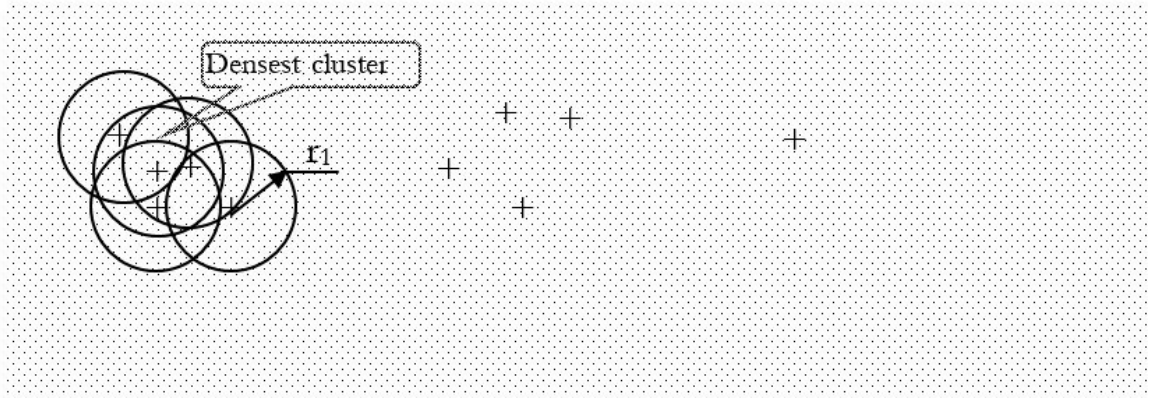


Figure 3: Clustering with r_1: the densest cluster is formed

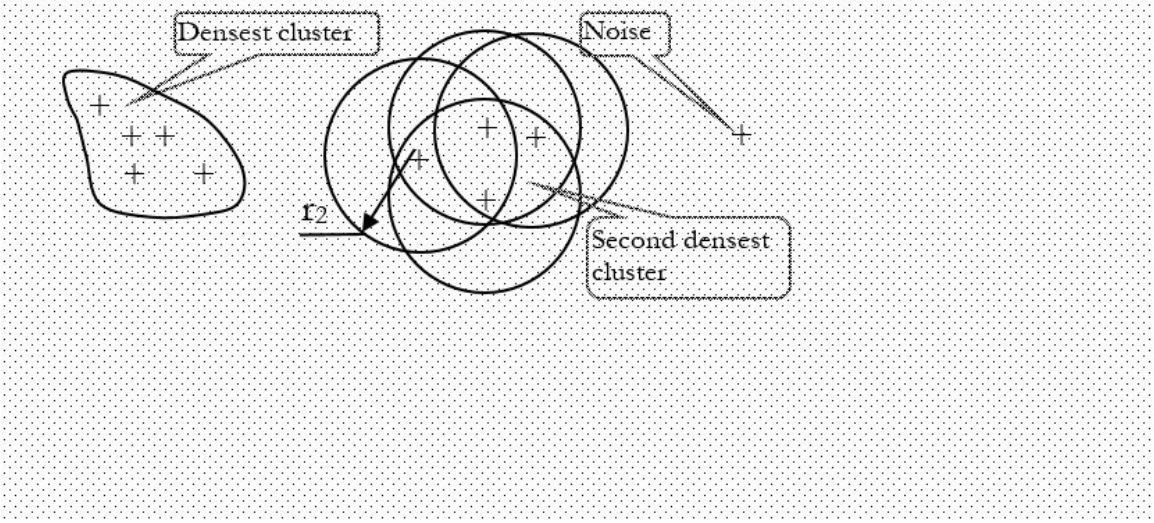


Figure 4: Clustering with r_2: the second densest cluster is formed

The whole process of IDCUP is summarized in Figure 5.

IDCUP Proposed algorithm
Input: *the data set X*
Output: *Labeled clusters and noise*
// C_i – Cluster set for radius r_i
// B_i – Boundary set for C_i
1. *Calculate a number of the neighborhood radii for different density clusters with MINR ( ): r_1, r_2 … r_m*
2. *Iterative Clustering with DBCUP*
   For i = 1 to m do

   C_i = e-DBSCAN (X, r_i, B_i);

   End for

3. *Check the boundaries of each pair of clusters. If the size of Bi ∩ Bj is close to that of either Bi or Bj, merge these two clusters.*

Figure 5: Proposed IDCUP algorithm

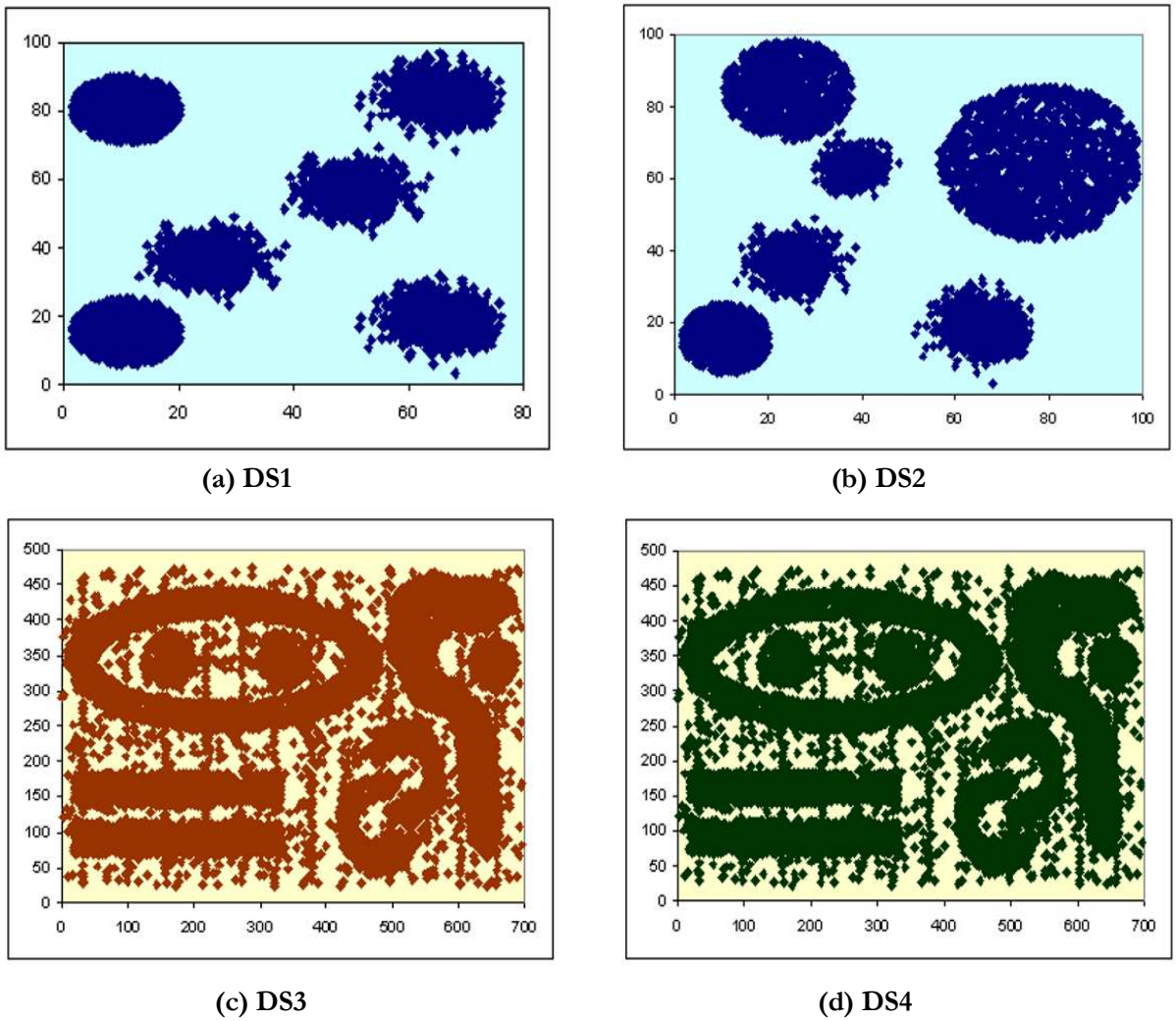
EXPERIMENTAL RESULTS

In this section, we show our experimental study with regards to clustering results and runtimes. We compare IDCUP with DBSCAN (Lammersen et al., 2014), TURN* (Guo & Zhang, 2014), and K-means (Fahy et al., 2019) using four different data sets. The data sets are arranged in different orders for different runs, in order to test the robustness of the clustering results. We found that among the four methods, only K-means method is sensitive to the order of inputs. K-means and DBSCAN both need input parameters. K-means needs the number of clusters, *k*, and DBSCAN asks for the neighborhood radius *r*. For *K-means*, we set *k* equal to the real number of clusters in the data set to get the best clustering result of the algorithm. For DBSCAN, we choose a reasonable neighborhood radius based on our knowledge of the data distribution.

DATA SETS

We use four typical data sets: *DS1, DS2, DS3*, and *DS4*. The distribution of the data sets is shown in Figure 6. Data set *DS1* and *DS2* are reproduced and enlarged based on the data set used by OPTICS (Fahy, Yang, & Gongora, 2019). Software tool PDflex is used to analyse the datasets. Data set *DS3* and *DS4* are reproduced and enlarged based on the data set used by CHAMELEON (Wen et al., 2017). The main features of each data set are:

(a) DS1 has the size of 6K. It contains 6 spherical clusters which are in similar densities and similar sizes. It doesn't have much noise.
(b) DS2 has the size of 25K. It contains 6 spherical clusters which are in similar densities but in different sizes.
(c) DS3 has the size of 17.5K. It contains 9 arbitrary-shape clusters which are in two different densities. The three clusters at the left top corner have higher density than the other clusters. It is noisy data.
(d) DS4 has the size of *31K*. It contains 9 arbitrary-shape clusters which are in three different densities. The two bar-shape clusters at the left bottom corner have the highest density. The three clusters at the left top corner have the second highest density. The other four clusters have the lowest density. It is noisy data.



(a) DS1 **(b) DS2**

(c) DS3 **(d) DS4**

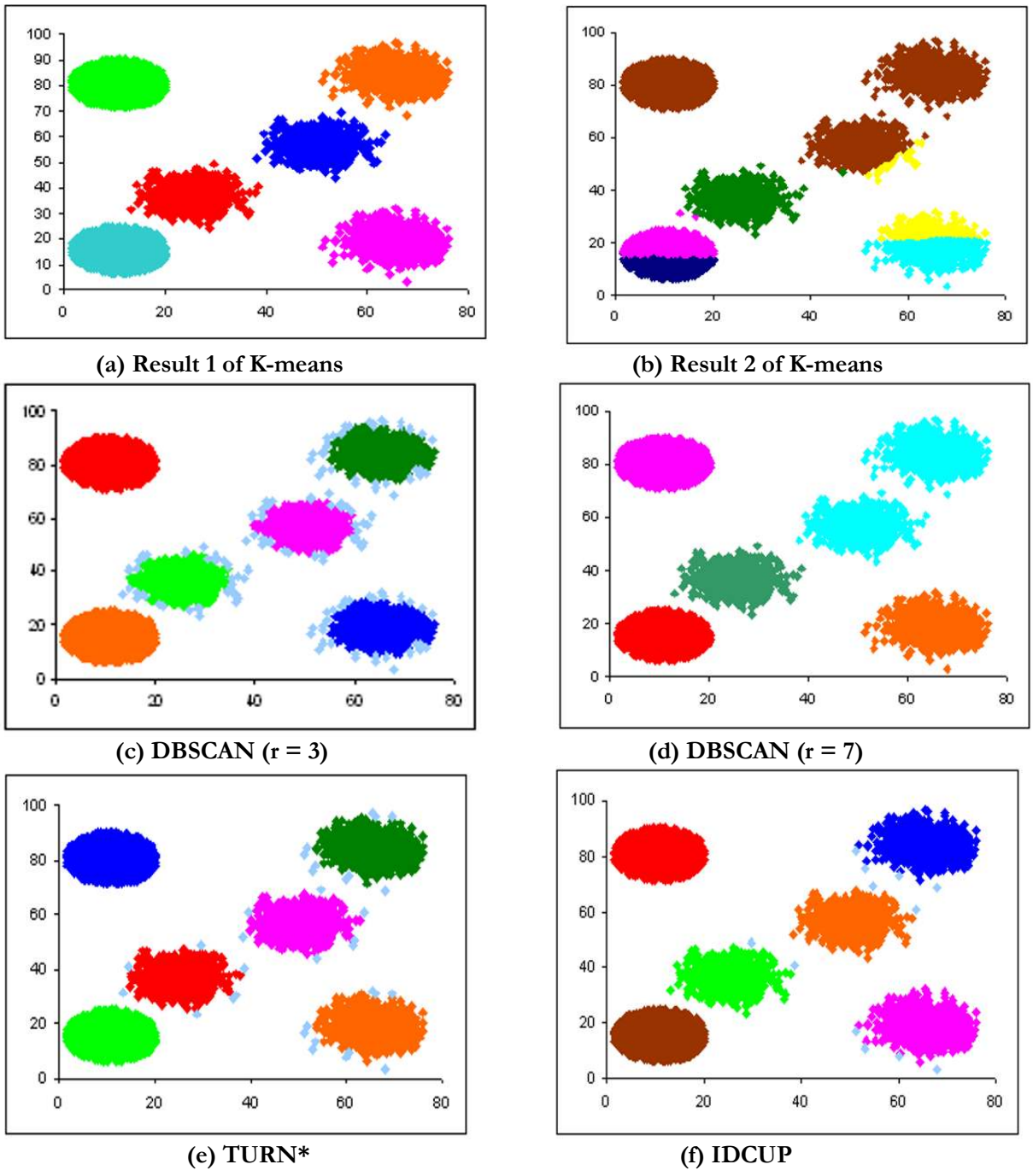
Figure 6: Visualization of four data sets

CLUSTERING RESULTS

In this section, we show the clustering results visually in Figures 7, 8, 9, and 10. Each figure is from the same data set. Points of the same color are in the same cluster. Noise is shown in grayish color in all figures. We show six clustering results for each data set, two from K-means algorithm, two from *DBSCAN*, one from *TURN**, and one from *IDCUP*.

Figure 7 shows the clustering results on DS1. DS1 is a data set with six clusters in similar densities. We can see from the figure that every method produces fairly good results except two cases: 1) when $r = 7$, DBSCAN merges two close clusters; and 2) Result 2 of 6-means when initial means are not selected properly.

Figure 8 presents the clustering results on DS2. DS2 is a data set that contains six clusters in different sizes. K-means merges two pair of clusters and break the largest cluster into four in result 1. In result 2, 6-means merges two clusters together and divide the largest cluster. When $r = 1.5$, DBSCAN breaks the largest cluster into many clusters, while when $r = 4$, DBSCAN merges two close clusters at the left top corner. TURN* and IDCUP both produce good results.



(a) Result 1 of K-means

(b) Result 2 of K-means

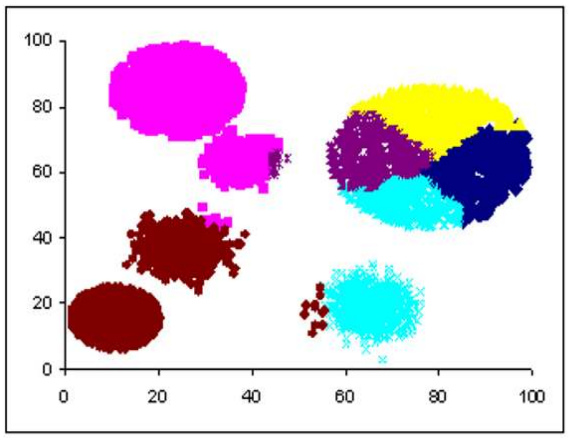
(c) DBSCAN (r = 3)

(d) DBSCAN (r = 7)

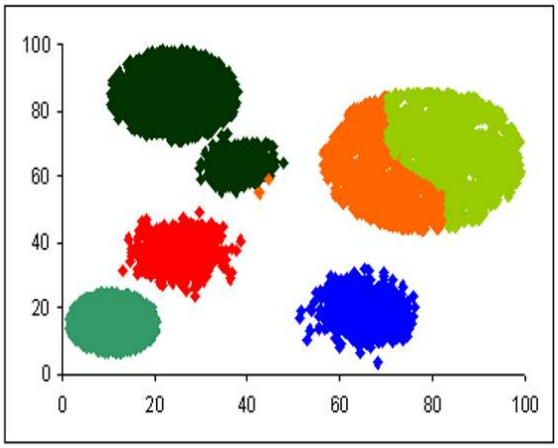
(e) TURN*

(f) IDCUP

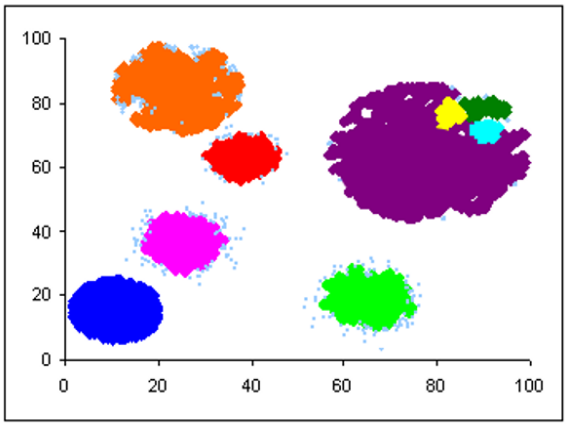
Figure 7: Clustering results on DS1



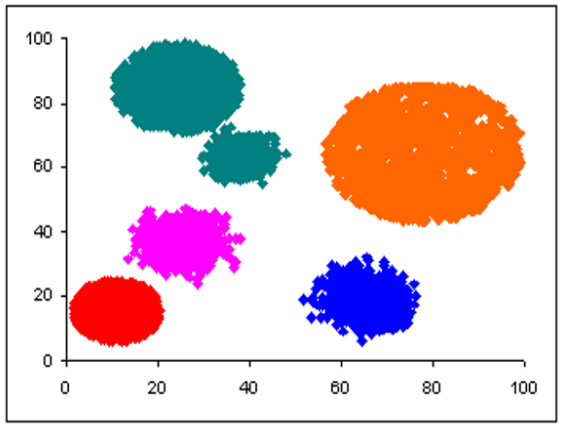
(a) Result 1 of K-means



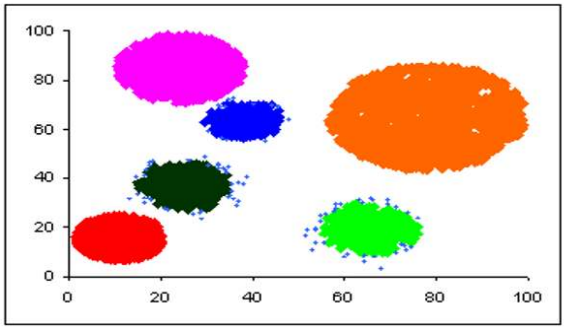
(b) Result 2 of K-means



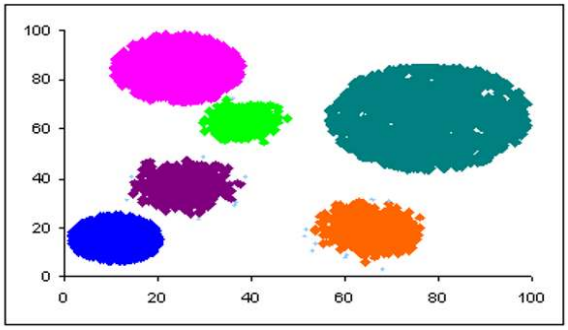
(c) DBSCAN (r = 1.5)



(d) DBSCAN (r = 4)



(e) TURN*



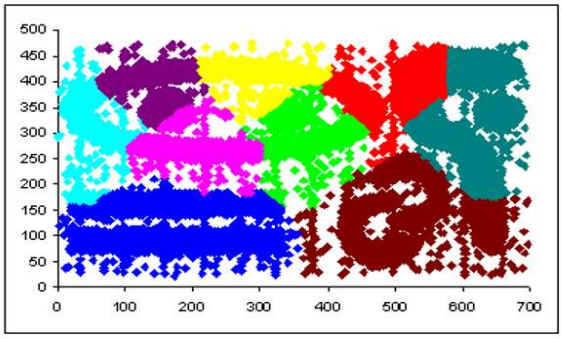
(f) IDCUP

Figure 8: Clustering results on DS2

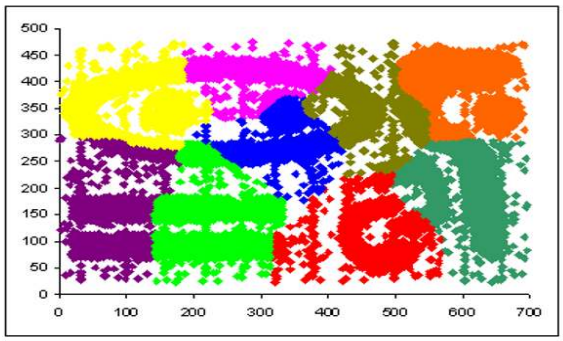
Figure 9 demonstrates the clustering results of *DS3*. *DS3* is a noisy data set which contains nine arbitrary-shape clusters and noise. The clusters are in two different densities. The big oval ring cluster and two round shape clusters inside it have higher density than the other clusters. 9-means shows another drawback in this case. Although it generates nine clusters according to the input parameter we set, the clustering results don't uncover any real cluster. When the input is in a different order, it generates different results. When $r = 7$, DBSCAN is able to generate the three dense clusters well but breaks the other clusters into pieces. When $r = 12$, DBSCAN merges five clusters including the three

dense clusters into one cluster and also merges the two bar-shape clusters into one. TURN* generates the three dense clusters well but breaks some other clusters and marks many boundary points as noise. IDCUP produces very good clusters except a few noise points are attached to clusters.

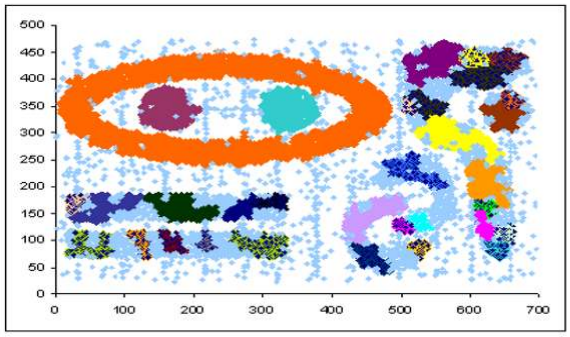
Figure 10 illustrates the clustering results of *DS4*. *DS4* is similar to *DS3*. Only the difference is the clusters of *DS4* are in three densities instead of two. The bar-shape clusters at the left bottom are the densest clusters. The big oval ring cluster and two round shape clusters inside it are the second densest clusters. The rest of clusters are the least dense ones. The cases of 9-means are similar to the cases above. For DBSCAN, when $r = 7$, it generates the five dense clusters well, but breaks the other clusters into pieces. When $r = 12$, DBSCAN merges five clusters including the three second dense clusters and also merges the two densest bar-shape clusters. TURN* generates the five dense clusters well, but breaks some other clusters and marks many boundary points as noise. IDCUP produces very good clusters except a few noise points are attached to clusters.



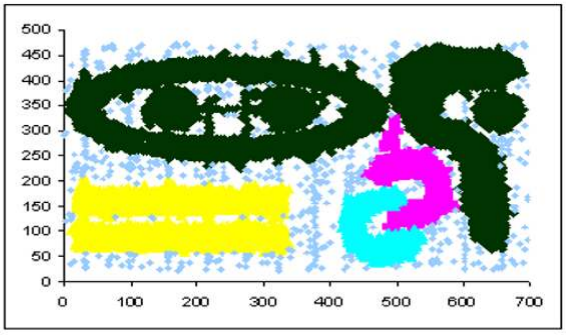
(a) Result 1 of K-means



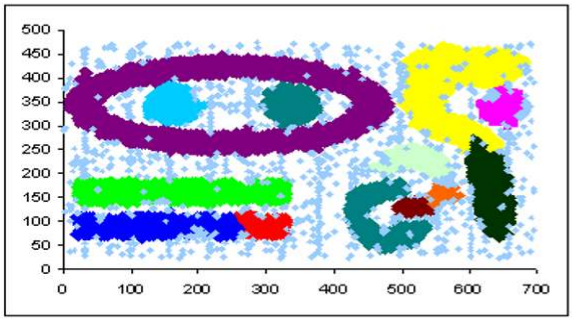
(b) Result 2 of K-means



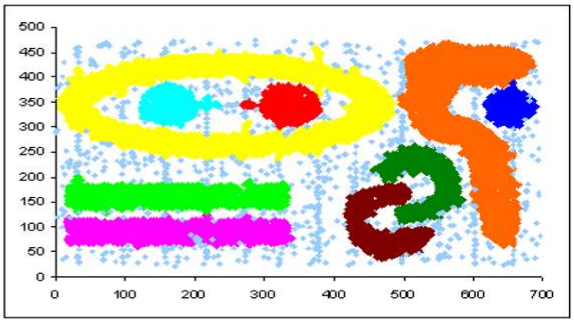
(c) DBCAN (r = 7)



(d) DBSCAN (r = 12)

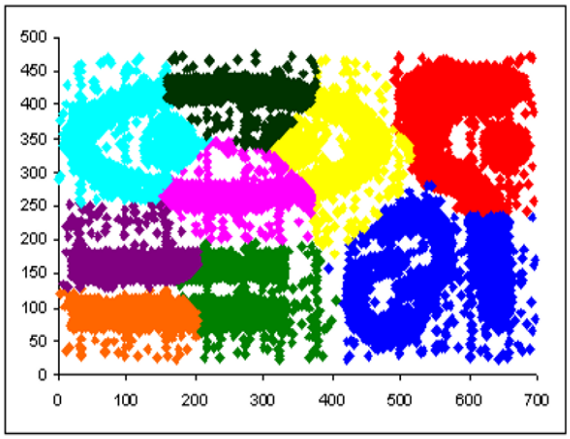


(e) TURN*

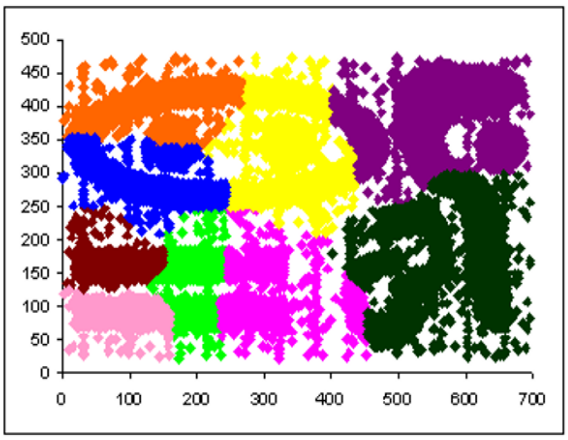


(f) IDCUP

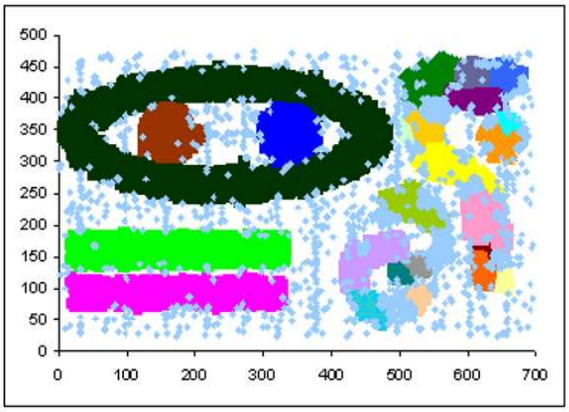
Figure 9: Clustering results on DS3



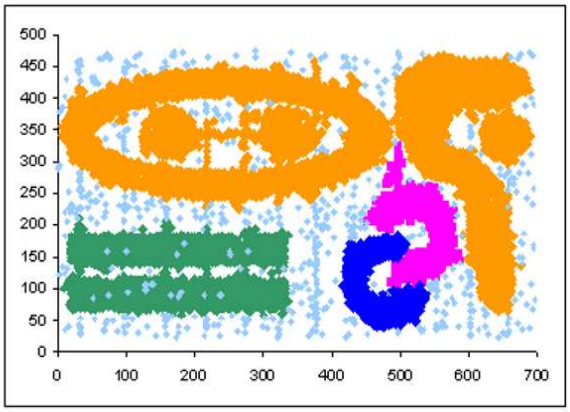
(a) Result 1 of K-means



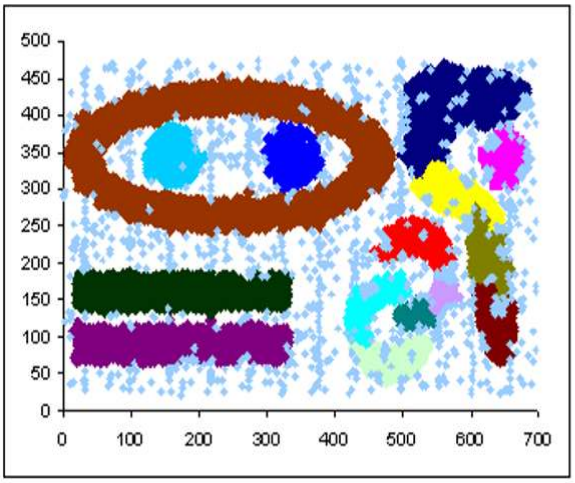
(b) Result 2 of K-means



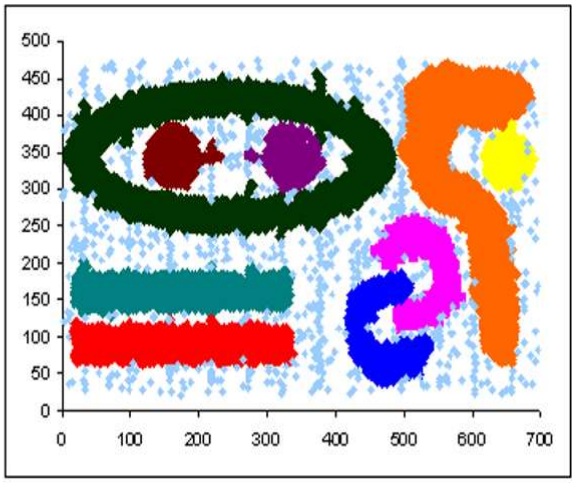
(c) DBSCAN (r = 7)



(d) DBSCAN (r = 12)



(e) TURN*



(f) IDCUP

Figure 10: Clustering results on DS4

In summary, K-means generates better results in DS1 and DS2 when clusters are spherical and in similar sizes than the other cases. But the results of K-means are not robust to the choices of initial partitions. DBSCAN is good for clusters in arbitrary shapes and in similar densities. But the results

are very sensitive to the input parameter. TURN* works well for arbitrary clusters without input parameter. However, it can only generate good clustering results when clusters are in similar densities. When clusters are in different densities, TURN* can generate the denser clusters well but tends to split the sparse clusters. IDCUP outperforms TURN* remarkably when the difference of cluster densities is immense.

RUNTIME COMPARISON

In this section, we compare K-means, DBSCAN, TURN*, and IDCUP. Table 1 shows run time on four data sets in milliseconds. The size of each data set is noted with the data set name.

Table 1: Run time on four data sets (*ms*)

Method	*DS1* (6K)	*DS2* (25K)	*DS3* (18K)	*DS4* (31K)
K-means	210	820	720	1290
DBSCAN	2060	61740	26680	101390
TURN*	13360	511750	196760	725480
IDCUP	4120	111132	48024	182502

K-means and DBSCAN both are parameter dependent. Their accuracy depends on correct choice of input parameter. They are not accurate in most cases. Therefore, it is not fair to compare the run time of parameter-dependent methods, K-means and DBSCAN, with the parameter-less methods, TURN* and IDCUP.

From Table 1, we can see that K-means and DBSCAN are both fast. TURN* is the most expensive method. IDCUP is more efficient than turn*. IDCUP is faster than TURN* because of the different mechanisms of these two approaches. The determination of the optimal radii in IDCUP takes place as a pre-process, while the determination of r in TURN* is carried out through the whole iterative clustering process.

DISCUSSION

This research focused on large data sets based on clustering has the following challenges: (1) clusters with arbitrary shapes; (2) less knowledge discovery process to decide the possible input features; (3) scalability for large data sizes. Density-based clustering has been known as a dominant method for determining the arbitrary-shape clusters. Briefly, the density-based clustering approach groups the data samples into a set of associated dense factors that are separated by low-density data sets. Density-based clustering applies a local cluster criterion and are very prevalent for mining clusters with arbitrary shape. However, the other two challenges remain in most existing clustering algorithms (Fahy et al., 2019). This paper is to explore and compare an Iterative Density-Based Clustering Method (IDCUP) to meet the current challenges. We meet the second challenge by reducing input parameters in density-based partitioning algorithm. We solve the third challenge by means of pruning and hashing techniques. Most clustering algorithms are designed for clusters with similar shapes, similar sizes, or similar densities. However, a data set may consist of clusters with different shapes, sizes, or densities. Furthermore, we can observe from results that K-means generates better results in *DS1* and *DS2* when clusters are spherical and in similar sizes than the other cases. But the results of K-means are not robust to the choices of initial partitions. DBSCAN is good for clusters in arbitrary shapes and in similar densities. But the results are very sensitive to the input parameter. TURN* works well for arbitrary clusters without input parameter. However, it can only generate good clustering results when clusters are in similar densities. When clusters are in different densities, TURN* can

generate the denser clusters well but tends to split the sparse clusters. IDCUP outperforms TURN* remarkably when the difference of cluster densities is immense.

From Table 1, we can observe that K-means is the fastest method. However, it doesn't generate good results in most cases. DBSCAN is the second fastest method. But the quality of its clustering results is highly dependent on the user input, which is usually a luxury in most applications. TURN* can generate good results when clusters are in similar densities, but it is very computationally expensive. IDCUP can generate good results in different cases.

IMPLICATIONS FOR PRACTICE

Experimental results in the previous section shows the discussed algorithms. The following recommendations are formulated the strengthen the concept:

- The proposed algorithm IDCUP can be effectively be used when the densities of clusters are enormous in large datasets.
- The proposed method has better accuracy and prediction than the DBSCAN with four times better execution speed in dense condition.
- IDCUP is more scalable than other density based methods as rather than the work on whole datasets, it utilized only partitions in clustering.
- The proposed quantity of partitions for the pre-processing time is suggested to be comparatively great.

This is supposed to the convention of the divide-and-conquer rule in available partitioning dataset which leads to less deviousness to cluster different objects.

LIMITATIONS

This paper suffers from some limitations. One of them is related to the sample to utilize datasets with only three densities. Secondly, un-clustered points are considered as only one type of noise to reshape the clusters. It only contains 6 spherical clusters which are in similar densities and similar sizes. It doesn't have much noise level. Finally, only four techniques are considered with compare IDCUP based on not too much large datasets.

CONCLUSION AND FUTURE WORK

In this paper, we have examined different density-based clustering methods for data sets with nested clusters of varying density. Two such data sets were used to evaluate some of the commonly cited algorithms found in the literature. Nested clusters were found to be challenging for the existing algorithms. In utmost cases, the targeted algorithms either did not detect the largest clusters or simply divided large clusters into non-overlapping regions. But, it may be possible to detect all clusters by doing multiple runs of the algorithm with different inputs and then combining the results. This work is considered three challenges of clustering methods. IDCUP is proposed and experimentally analyze with others the clustering algorithm to prove the concept and dealing with arbitrary shapes problems. It first automatically determines the optimal neighborhood radius for each density group based on data distribution. It then iteratively carries out density-based clustering using the neighborhood radii. Generally, IDCUP is more accurate than other methods, such as K-means, DBSCAN, and TURN*. When the density of clusters is similar, TURN* and IDCUP can generate good results. But when the clusters are in different densities, IDCUP is much better than all other methods. IDCUP is more efficient than TURN* in terms of computational time.

As future recommendations of this research, we are concerned with the exploration of further available challenges of the knowledge discovery process in clustering along with complex data sets with more time. A hybrid approach based on density-based and model-based clustering algorithms needs

to compare to achieve maximum performance accuracy and avoid the arbitrary shapes related problems including optimization. It is anticipated that the comparable kind of the future suggested process will attain improved performance with analogous precision in identification of clustering shapes.

ACKNOWLEDGMENTS

The authors would like to thanks to AUT University and Ministry of Science and Innovation New Zealand for funding the research.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interests regarding the publication of this paper.

REFERENCES

Agarwal, P., & Mehta, S. (2019). Subspace clustering of high dimensional data using differential evolution. In H. Banati, S. Mehta, & P. Kaur (Eds.), *Nature-inspired algorithms for big data frameworks* (pp. 47-74). IGI Global. https://doi.org/10.4018/978-1-5225-5852-1.ch003

Ding, K., Huo, C., Xu, Y., Zhong, Z., & Pan, C. (2015). Sparse hierarchical clustering for VHR image change detection. *IEEE Geoscience and Remote Sensing Letters*, *12*(3), 577-581. https://doi.org/10.1109/LGRS.2014.2351807

Fahy, C., Yang, S., & Gongora, M. (2019). Ant colony stream clustering: A fast density clustering algorithm for dynamic data streams. *IEEE Transactions on Cybernetics*, *49*(6), 2215-2228. https://doi.org/10.1109/TCYB.2018.2822552

Guo, K., & Zhang, Q. (2014). Fast clustering-based anonymization algorithm for data streams. *Journal of Software*, *24*(8), 1852-1867. https://doi.org/10.3724/SP.J.1001.2013.04330

Gui, Q., & Cheng, X. (2013). Clustering-based approach for multi-level anonymization. *Jisuanji Yingyong / Journal of Computer Applications*, *33*(2), 412-416. https://doi.org/10.3724/SP.J.1087.2013.00412

Hou, R., Zhu, B., Feng, M., Shi, X., & Lu, Y. (2013). Prediction model for lightning now casting based on DBSCAN. *Jisuanji Yingyong / Journal of Computer Applications*, *32*(3), 847-851. https://doi.org/10.3724/SP.J.1087.2012.00847

Heredia, L. C. C., & Mor, A. R. (2019). Density-based clustering methods for unsupervised separation of partial discharge sources. *International Journal of Electrical Power & Energy Systems*, *107*, 224–230. https://doi.org/10.1016/j.ijepes.2018.11.015

Lammersen, C., Schmidt, M., & Sohler, C. (2014). Probabilistic k-median clustering in data streams. *Theory of Computing Systems*, *56*(1), 251-290. https://doi.org/10.1007/s00224-014-9539-7

Limwattanapibool, O., & Arch-int, S. (2017). Determination of the appropriate parameters for K-means clustering using selection of region clusters based on density DBSCAN (SRCD-DBSCAN). *Expert Systems*, *34*(3), e12204. https://doi.org/10.1111/exsy.12204

Mohebi, A., Aghabozorgi, S., Wah, T. Y., Herawan, T., & Yahyapour, R. (2015). Iterative big data clustering algorithms: A review. *Journal of Software: Practice and Experience*, *46*(1), 107–129. https://doi.org/10.1002/spe.2341

Pietrzykowski, M. (2017). Local regression algorithms based on centroid clustering methods. *Procedia Computer Science*, *112*, 2363-2371. https://doi.org/10.1016/j.procs.2017.08.210

Pai, F., & Chang, H. (2013). The effects of knowledge sharing and absorption on organizational innovation performance – A dynamic capabilities perspective. *Interdisciplinary Journal of Information, Knowledge, and Management*, *8*, 83-97. https://doi.org/10.28945/1904

Ram, A., Jalal, S., Jalal, A., & Kumar, M. (2010). A density based algorithm for discovering density varied clusters in large spatial databases. *International Journal of Computer Applications*, *3*(6), 1-4. https://doi.org/10.5120/739-1038

Wei, Q., & Zhu, J. (2014). Kernel based CSP algorithm for classifying multi-class EEG data by combining k-means clustering and Nystrom approximation. *Journal of Information and Computational Science*, *11*(9), 2859-2871. https://doi.org/10.12733/jics20103700

Wen, F., Wang, X., & Zhang, G. (2017). Evolutionary-based automatic clustering method for optimizing multi-level network. *Cluster Computing*, *20*(4), 3161-3172. https://doi.org/10.1007/s10586-017-1030-1

Wang, J., Zhu, C., Zhou, Y., Zhu, X., Wang, Y., & Zhang, W. (2018). From partition-based clustering to density-based clustering: Fast find clusters with diverse shapes and densities in spatial databases. *IEEE Access*, *6*, 1718-1729. https://doi.org/10.1109/ACCESS.2017.2780109

Yang, X., Jin, L., Ye, W., Xiao, J., Zhang, D., & Xu, X. (2018). Laplacian centrality peaks clustering based on potential entropy. *IEEE Access*, *6*, 55462-55472. https://doi.org/10.1109/ACCESS.2018.2871500

Yen, T., Lim, T., Yoon, T., & Lai, S. (2017). Studying the varied shapes of gold clusters by an elegant optimization algorithm that hybridizes the density functional tight-binding theory and the density functional theory. *Computer Physics Communications*, *220*, 143-149. https://doi.org/10.1016/j.cpc.2017.07.002

Zhang, T., & Yuan, B. (2018). Density-based multiscale analysis for clustering in strong noise settings with varying densities. *IEEE Access*, *6*, 25861-25873. https://doi.org/10.1109/ACCESS.2018.2836389

Zhu, L., Lei, J., Bi, Z., & Yang, J. (2014). Soft subspace clustering algorithm for streaming data. *Journal of Software*, *24*(11), 2610-2627. https://doi.org/10.3724/SP.J.1001.2013.04469

BIOGRAPHIES

Saud Altaf received the PhD degree in computer science from Auckland University of Technology, New Zealand, in 2015. He has had more than fifteen years teaching experience in education sector of New Zealand and Pakistan. He is author of 12 Journal publications and 21 conference publication in the field of IT and computer engineering. He is member of different professional societies around world. His interests are in the area of wireless sensor networks, artificial intelligence and visible light communication for research, teaching and learning.

Muhammad Waseem Soomro is a Professional Engineer and member of Engineering New Zealand with a demonstrated history of working in the Manufacturing and Education industry. Skilled in Risk Management, Demand Side Management and Project Management, Health and Safety Management, Manufacturing processes and light materials, FEA Analysis, Risk Management, Demand Side Management and Project Management. Strong community, technical and social services professional with a Doctor of Philosophy (PhD) focused on Lightweight materials and smart manufacturing processes from AUT University, Auckland. Over the course of my career, I have seen full career progression while showcasing a broad and varied skillset that has enabled my success and the success of my teams and students. With a passion for cultivating and fostering lasting relationships based on a foundation of achieving excellence, growing and developing, I have established a demonstrable track record of success in identifying and pursuing new challenges and thriving in high pressure environments.

Laila Kazmi is a research student in the Shaheed Zulfikar Ali Bhutto Institute of Science and Technology, Islamabad Pakistan. She has BS in Computer Science, focusing in IoT. Her field of research addresses topics related to the Embedded system, data mining and wireless networks.