International Journal of Innovative
Computing, Information and Control
Volume **19**, Number **6**, December **2023**

ICIC International ©2023 ISSN 1349-4198
pp. 1805–1825

AN ALTERNATIVE PARAMETER FREE CLUSTERING ALGORITHM USING DATA POINT POSITIONING ANALYSIS (DPPA) – COMPARISON WITH DBSCAN

S. M. F. D. Syed Mustapha

College of Technological Innovation
Zayed University
P.O.Box 19282, Dubai, United Arab Emirates
syed.duani@zu.ac.ae

Received March 2023; revised June 2023

Abstract. *DBSCAN is one of the most popular clustering algorithms that could handle clusters which have characteristics of arbitrary shape, multiple densities and noises. However, its accuracy depends on the right selection of the two parameters, MinPts and Eps. There have been numerous research works to overcome this issue by developing parameter free clustering algorithm. We propose a clustering algorithm which uses Data Point Positioning Analysis (DPPA) to analyze the relationship of each point to all points based on two nearest neighbor concepts, namely 1-NN and Max-NN. The algorithm is applied on 13 benchmark datasets that have been applied in many clustering algorithms with three-dimensional data and subsequently on higher dimensional data with sixteen attributes. The performance of the algorithm is visually compared with the three-dimensional graph plotting at various angles to determine the actual number of clusters. For the higher dimensional data, Silhouette coefficient is used to measure the performance. For both experimental results, the DPPA algorithm is compared against DBSCAN. The results show that the DPPA algorithm is comparable to the performance of DBSCAN algorithm such that it manages to detect arbitrary cluster shapes, identify the number of clusters and manage the data sets with noises.*

Keywords: Clustering algorithm, Unsupervised learning, Parameter free clustering algorithm, DBSCAN

1. **Introduction.** Clustering is an unsupervised machine learning technique that is extensively used for analyzing unlabeled data for different purposes such as image segmentation, text retrieval or business data analytics [1]. The traditional clustering techniques have been extensively applied to these problems, but they require some pre-conditions such as prior knowledge about the nature of data to initiate a good guess about the number of clusters such as in K-means algorithm [2], the number of neighbors such as KNN algorithm [3] and the feature selections to discriminate the clusters effectively when using algorithms such as hierarchical clustering or agglomerative clustering which require comparisons of data points based on their feature similarities or dissimilarities. Another challenge is that some of these traditional techniques do not exhibit optimal performance for clusters with irregular shapes and noises. Hence, more robust algorithms that do not depend on the prior knowledge of the number of clusters are developed such as DBSCAN [4], DDC [5], DSETS [6], Mean-shift algorithm [7,8] and Affinity Propagation [9]. The interests to reduce the dependability to initial parameters had been shown within the research on DBSCAN algorithm itself as well as other alternative algorithms which work towards parameter free algorithm. It is called as parameter free algorithm as the aim of

DOI: 10.24507/ijicic.19.06.1805

the algorithm is to bypass the manual guess in the initial parameter setting prior to its execution. Hence, the focus of this work is mainly on parameter free algorithm.

2. **Related Work.** The past researches in reducing the burden in setting the initial parameters for DBSCAN are classified into three types which are

- Automatic parameter determination algorithm – the standard *MinPts* and *Eps* are required but these parameters are determined automatically;
- Reduced DBSCAN parameter algorithm – either the *MinPts* or *Eps* parameters is avoided to reduce the dependencies to both parameters;
- Alternative parameter free determination algorithm – an alternative strategy to avoid using both *MinPts* and *Eps* or any manual parameter setting. Hence, some of these algorithms are alternative to DBSCAN and may be dependable to other parameters but free from the need of setting the parameters manually.
  - **Automatic parameter determination algorithm** – The work by Chen [10] seeks to improve the DBSCAN as it can handle multi-density data distribution. It scans each data to determine the M value which is the number of data points surrounding it using the *Eps* that is to automatically determine using the non-parametric kernel density estimation. The data object that has the largest M will be the core object for the cluster. The *MinPts* is determined based on the M value of the current core object and the maximum M value for the entire datasets. Each data point is analyzed in order to determine the M value and this process is repeated after the core object is known. The time complexity for this algorithm is $O(n^2)$. The strength of the algorithm is that it is adaptive to the variations of the density subject to the accuracy in determining the bandwidth h value of KDE. DSETS-DBSCAN is another type of algorithm that determines DBSCAN parameters automatically based on the concept of dominant set of data [11]. Dominant sets satisfy the constraints of high internal similarity and low external similarity. The process reveals partitions of data sets which are then fed to DBSCAN as cluster. Subsequently, DBSCAN procedure is applied using fixed $MinPts = 3$ and *Eps* that is derived from dominant sets. A similar attempt by Sharma and Sharma [12] proposes the use of KNN method (instead of dominant sets) as an alternative to find the kernel of the density. The kernel that has been determined is converted as cluster in which the standard DBSCAN procedure is applied. The *MinPts* is calculated based on the size of the clusters and *Eps* is calculated based on the maximum distance of 3rd nearest neighbor for all points. The values of *MinPts* and *Eps* can be automatically determined using an optimization method such as genetic algorithm. Azhir et al. [13] introduce NSGA-II (Non-dominated Sorting Genetic Algorithm) which starts with initial generation of population. DBSCAN method is applied to generating possible clusters. The objective functions are calculated using the fitness function and the standard genetic algorithm procedures (mutation, selection and crossover) are performed iteratively. The DBSCAN procedure is applied when data point is assigned incrementally to cluster. The process is repeated until the best value of *MinPts* and *Eps* are obtained. Another strategy in determining the two DBSCAN parameters automatically is to perform geometrical analysis on the data sets. Kinsuk et al. [14] apply Convex Hull and Voronoi to constructing the corresponding Empty Circle (EC) in which the radius (*Eps*) is determined and uses Silhouette Score to determine the *MinPts*. Another recent work is done by Mu et al. [15] where each data is profiled with KNN graph. The valley of the graph is used to calculate the *Eps* and *MinPts*. The fitness function is applied to determining

the best values of both parameters. Another interesting work in determining the two DBSCAN parameters is done by Hossain et al. [16] in which the mean value of the distance for a data point to all other points is calculated and this is repeated for all data points. The data point with the smallest minimum value is considered as having the highest density. Subsequently the *MinPts* is calculated by incrementally increasing the *Eps* value until the local minima is reached.

- **Reduced DBSCAN parameter algorithm** – The radius is the most challenging parameter to be determined as the distance is subject to the scale of x and y axes. Kinsuk et al. [14] apply computational geometry such as Voronoi and Convex Hull and empty circles to determining the *Eps* value while the *MinPts* is a fixed value. Another work that focuses on one of the DBSCAN parameters which is *Eps* itself is discussed by Lai et al. [17] using Multiverse Optimization (MVO) method. It is also worth to mention the work by Lv et al. [18] in automating the determination of nearest neighbor using locality sensitive hashing method to create an influence space. This approach reduces the computational time for searching the nearest neighbor as in the traditional DBSCAN.
- **Alternative parameter free determination algorithm** – Frey and Dueck introduce affinity propagation as the clustering technique which has been successfully implemented in various pattern recognition problems [19]. Each data point communicates with each other in exchanging messages (called responsibility and availability) and this is done iteratively to determine good exemplars. Some criteria preferences must be set to determine the affinity of the exemplars in the network. Even though there are no initial parameters that need to be pre-determined, the performance of this algorithm depends on the well-defined criteria preferences to achieve its optimal goal. Another weakness is that it is not capable of handling arbitrary shapes of clusters. Due to this reason, Chen et al. [20] combine DBSCAN and Affinity Propagation techniques called APSCAN to take the advantage of both capabilities. APSCAN uses AP technique to determine the exemplars of the datasets in which these exemplars are used to determine the densities. The density has two pieces of information which are the Radius and Number, whereby the Radius is the average distance of the exemplar to all points and the Number is the number of points that fall in between the Radius-neighborhood. The approach has used both parameters (Radius and Number) as an alternative to *Eps* and *MinPts*. Ding et al. [21] propose EPFC algorithm where the average value of nearest neighbor points is calculated based on all data sets. Each data point is examined based on its distance with the nearest neighbor, and two decisions will either be made to split the group or add the data points depending on the distance being calculated. Yaşar et al. [22] propose another parameter free algorithm that is independent of the two standard DBSCAN parameters called FN-DBSCAN-GM. Gaussian Means is used to determine the center of the data sets. K-means algorithm is deployed to assign each new data point. A new cluster is added if the new data point does not follow the Gaussian distribution. That means the core point is determined by the center of Gaussian and radius is calculated accordingly. The remaining unassigned data points are considered as noises. Another work that is worth to mention is using natural nearest neighbor as reported by Wu [23] and Bryant and Cios [24]. Each point searches for its nearest neighbor and vice versa, reverse nearest neighbor data point to that point and establish the relationship. This process is repeated for all data points and the point that has the most reverse

nearest neighbor is of high density. The eigenvalues determine whether the cluster has outliers or nicely dense cluster. The K-means algorithm is applied as the indications on the number of clusters are known. Mean-shift algorithm is another popular algorithm being widely used for image processing and clustering which depends on the estimation of kernel density and mapping the surrounding points to the central points with high density. It can handle arbitrary shape but perform poorly on ill-defined window size as the density is dependable to the window scaling [25]. Another work that proposed a parameter free algorithm is discussed in [26] where the radius is determined automatically based on the window size and the data distribution.

Based on the brief literature review, the two dominant research attempts are, firstly to find methods in automating the determination of two DBSCAN parameters and secondly to find alternative to the DBSCAN parameters. The example of the former is using non-parametric KDE to find *MinPts*, DSETS-DBSCAN to determine the core objects of the clusters, NSGA-II to generate possible clusters for DBSCAN and Convex Hull and Voronoi to determine the *Eps*. For these approaches, the burden of determining the two DBSCAN initial parameters has been reduced to either *Eps* or *MinPts* only. The examples of the latter are related to finding the alternative to *MinPts* and *Eps*. That means, the *MinPts* and *Eps* are avoided entirely, and the examples of such works are AP (Affinity Propagation), APSCAN, EPFC, FN-DBSCAN-GM, Mean-shift algorithm. Our interest in this research falls under this category which is to find an alternative to parameter free algorithm by analyzing the data points and the relationships of the datapoints.

3. Data Point Positioning Analysis.

Our proposed algorithm falls under the third type of parameter free algorithm work in which alternative parameters are used and they are determined automatically. In developing this algorithm, we set the following goals which are derived based on the strength and capabilities of the previous algorithm in this category:

Goal 1 (arbitrary shape): the ability to handle an arbitrary shape of clusters;

Goal 2 (independent parameters): the determination of the parameters should not be dependent to any determination of another parameters that are manually determined. For example, the determination of kernel density is dependent to the h value of KDE function which is set manually;

Goal 3 (handling noise): able to detect data points that do not exhibit visually close to clusters or potentially form new clusters;

Goal 4 (window size): the performance of the algorithm should be independent to the window size or scaling of the axis.

3.1. Modifying DBSCAN with Data Point Positioning Analysis (DPPA).

***Eps*-neighborhood in DPPA** – the *Eps* is the radius that is used to connect to the nearest data points. Instead of using *Eps*, a radius range is used in which any distance between two points that fall between this radius range is considered as nearest neighbor. The radius range is the minimum and maximum value of the potential distance between any two points in the entire data set. To automatically determine the *Eps* value, the data points distribution needs to be analyzed to have a good estimation of the *Eps* value.

Definition 3.1. *(Radius Range):* $\lambda(\min(\phi), \max(\phi))$ *is the radius range,* λ*, for the data sets* X*.*

Definition 3.2. $(\min(\phi))$*: Given* α *– a set of all minimum distance for each point in data set* X*:*

$$\alpha = \{\forall x_i \in X, \forall x_j \in X | [(\min_{x_1}(dist(x_1, x_j), \ldots, dist(x_1, x_n))), \ldots, (\min_{x_m}(dist(x_m, x_j), \ldots, dist(x_m, x_n)))]\}$$

If there are N number of data points, there are N number of $\min_{x_j}$ as each data point is measured against all points in the data set X and $\min_{x_j}$ is the smallest distance for data point x_j. Hence, $\min(\phi)$ is the smallest value for $\min[\min_{x_1}, \min_{x_2}, \ldots, \min_{x_N}]$ as shown in Figure 1.

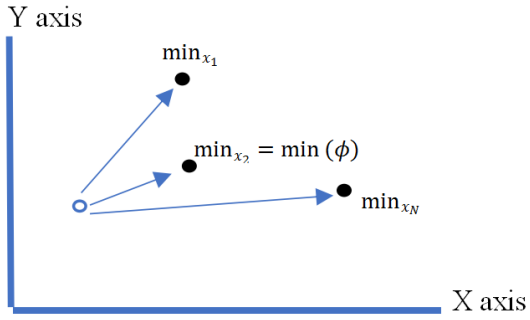


FIGURE 1. Determination of $\min(\phi)$ value

Definition 3.3. $(\max(\phi))$*:* $\max(\phi) = \max\{\min_{x_1}, \min_{x_2}, \ldots, \min_{x_N}\}$*. Among all the minimum value obtained from α (Definition 3.2), find the largest value and assigned to* $(\max(\phi))$*. Since the selection is made by selecting the highest value among the minimum distance, this ensures that all data points that are connected to other data points are only their nearest neighbor. Figure 2 shows several $\min_{x_j}$ for different data point and the $\max(\phi)$ will the highest value among the $\min_{x_j}$.*

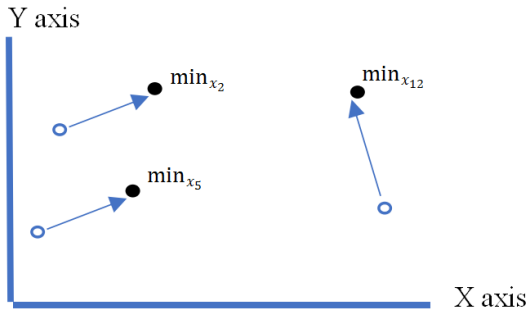


FIGURE 2. Determination of $\max(\phi)$ value

Border point in DPPA – the search for the neighboring data points is a one-way search. This is simply done by placing a mark to the data points that have been visited before. Hence, if a data point is at the border, it is most likely that it will not move forward as all the points that are pointing towards the border point have been marked as visited.

Definition 3.4. *Border point (x_i): if x_i is a border point, $\exists x_j$ where $j = \{1, \ldots, N\}$, $x_j \underset{dist \leq \lambda}{\longrightarrow} x_i$, mark x_j and x_i has no neighboring point to be visited.*

Figure 3 shows three data points pointing towards a data point (hollow blue color) which should be considered as border point. The data points x_i, x_j, and x_k register the border point as the neighboring point to be visited. These data points (x_i, x_j, and x_k) will be marked as visited and hence the border point will not have any neighboring data point to be visited and it will be considered as border point.

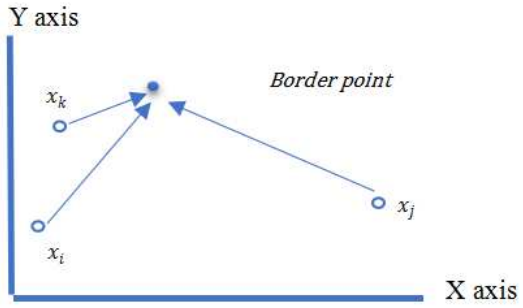


FIGURE 3. Determination of border point

Core points in DPPA – core points which are the data points at the high-density area are not used in DPPA. DPPA searches for the data point that has the longest link called Max-NN (Maximum Nearest Neighbor).

Definition 3.5. *(Max-NN): Given a data point* x_i, $\exists x_j, x_{i+\varepsilon} \in X$ *(datasets), where* $\max(\phi) \geq dist(x_i, x_{i+\varepsilon}) \geq \min(\phi)$. n_{x_i} *is the number of neighboring data point series presented as follows:* $n_{x_i} = \sum_{i=0}^{m} \begin{pmatrix} \min(\phi) \leq dist(x_i, x_{i+\varepsilon}) \leq \max(\phi) \rightarrow add\ 1 \\ add\ 0 \end{pmatrix}$. $x_{i+\varepsilon}$ *refers to any other data point where the distance between two data points* x_i *and* $x_{i+\varepsilon}$ *is within the largest distance and lowest distance that were calculated in Definition 3.2 and Definition 3.3.*

Figure 4 shows an example of Max-NN for a data point x_i. *There are several possible paths to establish data point series, and the path to the border point is called Max-NN. Every path from* x_i *is registered and the number of data points that are traversed will be counted, so Max-NN value is the total data points that it traverses. There are four solid blue dots which are the border points and the longest Max-NN for* x_i *is shown by the arrow.*

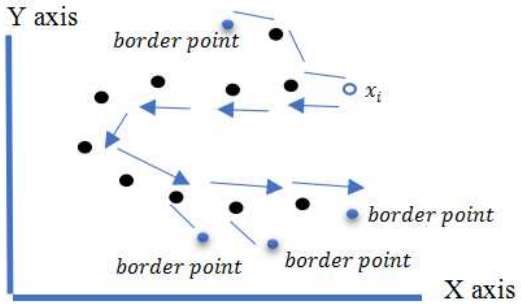


FIGURE 4. Determination of the Max-NN for data point x_i

Definition 3.6. *(1-NN): the first encountered nearest neighbor that is within the range of* λ. *A data point* x_i *could have one or more 1-NN.*

Merging clusters in DPPA – if x_i is a data point and x_j is 1-NN to x_i, then x_i and $x_j \in C$ cluster. Hence, $\forall x_k$ that are in Max-NN will be in C cluster.

Assume single cluster data points as shown in Figure 5, using the concepts discussed above, Table 1 is constructed to demonstrate the building of 1-NN and Max-NN.

The link denoted as $(\rightarrow)$, between two data points is only generated if both data points are within the radius range (λ). In the 1-NN, each data point will generate its nearest neighbor within the radius range as shown in Figure 5. For example, x_1 does not generate x_{14} in the 1-NN as its nearest neighbor as it does not consider as nearest neighbor. 1-NN

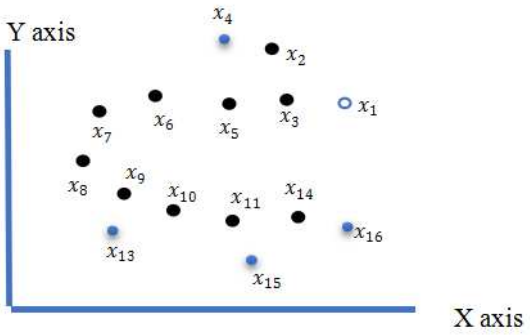


FIGURE 5. Single cluster data points

TABLE 1. The building of 1-NN and Max-NN

Data point	1-NN	Max-NN
x_1	x_2	$x_1 \to x_2 \to x_4$ (3 data points) $x_1 \to x_2 \to x_3 \to x_4$ (4 data points) $x_1 \to x_2 \to x_3 \to x_5 \to \cdots \to x_{13}$ (9 data points) $x_1 \to x_2 \to x_3 \to x_5 \to \cdots \to x_{15}$ (11 data points) $x_1 \to x_2 \to x_3 \to x_5 \to \cdots \to x_{16}$ (12 data points)
x_1	x_3	$x_1 \to x_3 \to x_5 \to x_6 \to \cdots \to x_{13}$ (8 data points) $x_1 \to x_3 \to x_5 \to x_6 \to \cdots \to x_{15}$ (10 data points) $x_1 \to x_3 \to x_5 \to x_6 \to \cdots \to x_{16}$ (11 data points)
x_2	x_1 – not generated as x_1 is already visited in 1-NN	Not generated as 1-NN is not available
x_2	x_4	Not generated as 1-NN is not available
x_2	x_3	$x_2 \to x_3 \to x_5 \to x_6 \to x_7 \to \cdots \to x_{13}$ (8 data points) $x_2 \to x_3 \to x_5 \to x_6 \to x_7 \to \cdots \to x_{15}$ (10 data points) $x_2 \to x_3 \to x_5 \to x_6 \to x_7 \to \cdots \to x_{16}$ (11 data points)
x_2	x_5	$x_2 \to x_5 \to x_6 \to x_7 \to \cdots \to x_{13}$ (7 data points) $x_2 \to x_5 \to x_6 \to x_7 \to \cdots \to x_{15}$ (9 data points) $x_2 \to x_5 \to x_6 \to x_7 \to \cdots \to x_{16}$ (10 data points)
x_3	x_5	$x_3 \to x_5 \to x_6 \to x_7 \to \cdots \to x_{13}$ (7 data points) $x_3 \to x_5 \to x_6 \to x_7 \to \cdots \to x_{15}$ (9 data points) $x_3 \to x_5 \to x_6 \to x_7 \to \cdots \to x_{16}$ (10 data points)
x_3	x_2 – not generated as x_2 is already visited in 1-NN	Not generated as 1-NN is not available
x_3	x_1 – not generated as x_1 is already visited in 1-NN	Not generated as 1-NN is not available
x_4	x_2 – not generated as x_2 is already visited in 1-NN	Not generated as 1-NN is not available
x_5	x_3 – not generated as x_3 is already visited in 1-NN	Not generated as 1-NN is not available
x_5	x_6	...
x_6	x_5 – not generated as x_5 is already visited in 1-NN	...
x_6	x_7	...
x_7	x_7 – not generated as x_7 is already visited in 1-NN	...
x_7	x_8	...
	...	...

will be constructed for all data points before Max-NN is generated. The generation of 1-NN will avoid any infinite loop as a data point will not be generated if it has been visited. That means, each data point will be visited once from the same path, for example, the path from $x_1 \to x_2$ will be considered the same as $x_2 \to x_1$; hence, it will not be generated as shown in the third row in Table 1. For Max-NN, the generation of data point series will depend on what has been declared in the 1-NN. For example, for x_1, there are two paths, $x_1 \to x_2$ and $x_1 \to x_3$, the next link will refer to x_2 and x_3 in the 1-NN column, respectively to check its next neighboring data point, and this step continues until it reaches the terminal data point or border point.

The merging of the data points into a common cluster will begin with the data point link that shows the longest path (for example, the longest link for x_1 is $x_1 \to x_2 \to x_3 \to x_5 \to \cdots \to x_{16}$ (12 data points)) and each data point will be stored in the first cluster C_i. This is followed by the data points in the subsequent link and the data points that are already in the cluster will be ignored. Based on the Max-NN in Table 1, all the data points are in the same cluster.

The following algorithm below describes the entire processes as discussed above.

1) Given a collection of data points $p_i \in \delta$, for each data point, calculate the distance $d(p_i, p_{i+1})$, for $\forall p_{i+1} \in \delta$ and $i = \{1, \ldots, m\}$ where m is the number of data points in δ. The distance is measured based on the coordinates x, y and z where
$$d(p_i, p_{i+1}) = \left(\sqrt[2]{\left(p_i^x - p_{i+1}^x\right)^2 + \left(p_i^y - p_{i+1}^y\right)^2 + \left(p_i^z - p_{i+1}^z\right)^2} \right).$$
2) For $\forall p_i$, determine $\min(d(p_i, p_{i+1}))$, $i = \{1, \ldots, m\}$
$$\xrightarrow{stores} \phi\left(\min(d(p_i)), \ldots, \min(d(p_m))\right).$$
3) Determine radius range, $\lambda(\min(\phi), \max(\phi))$ where ϕ is some scalar value.
4) For each point $p_i \in \delta$,
   a) find $n_{p_i} = \sum_{i=0}^{m} \begin{pmatrix} \min(\phi) \le d(p_i) \le \max(\phi) \to add\ 1 \\ add\ 0 \end{pmatrix}$ where n_{p_i} is the number of neighbors for data point p_i that has distance within radius range λ.
   b) build the neighbor-link table for 1-NN and Max-NN for each data point $d(p_i)$ as shown in Table 2.

Table 2. Neighbor-link for 1-NN and Max-NN

Data point	1-NN	Max-NN
p_i	p_a where a is some index	$p_i \to p_a \to p_d \to p_e$
p_i	p_b where b is some index	
p_i	p_c where c is some index	
p_a	p_d	
p_d	p_e	

Note: 1-NN: some data point p_i where $\min(\phi) \le d(p_i) \le \max(\phi)$ for each data point p_i where $i = \{1, \ldots, m\}$.
Max-NN: a linkage of data points begins with p_i and followed by every 1-NN of p_i until p_i has no 1-NN.

5) Sort the p_i in the neighbor-link table based on the n_{p_i} in an ascending order
   a) Place p_i in cluster C_i followed by all p_k that are in Max-NN.
   b) Place all p_j of 1-NN for p_i to C_i.
   c) Next p_{i+1}, if $p_{i+1} \in C_k$, then $C_i \leftarrow C_k$, $p_i \leftarrow p_{i+1}$ and go to a).
   d) Repeat until no more data point.

4. Dataset and Experimental Results.

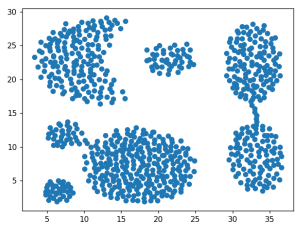
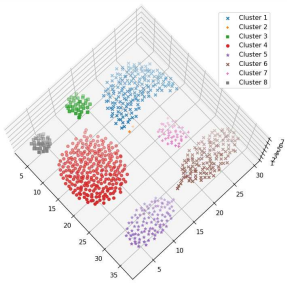
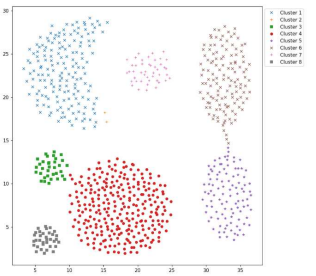
4.1. **DPPA's experiments on datasets.** The algorithm is tested on various datasets which are made available at (https://github.com/deric/clustering-benchmark/tree/master/src/main/resources/datasets/artificial). 13 datasets have been selected at random to test the performance of the algorithm. The objectives of the experiments are

a) to examine the ability of DPPA to find the clusters on three dimensions data sets (x, y, and z axes). The correct number of clusters can be determined by plotting into three-dimensional plot at various angles which will be shown for each result of the experiment;
b) to examine the ability of the DPPA to handle arbitrary shapes such as non-globular shape or contiguity-shape or elongated shape, ring-type shape;
c) to examine the ability of DPPA to handle various densities;
d) to examine the ability of DPPA to handle noises.

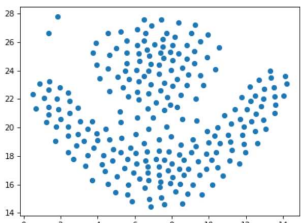
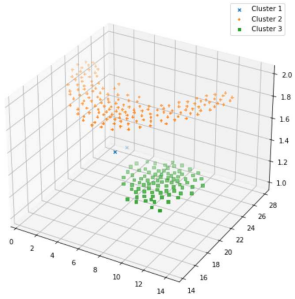
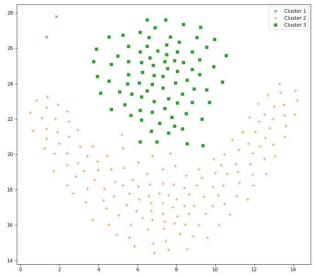
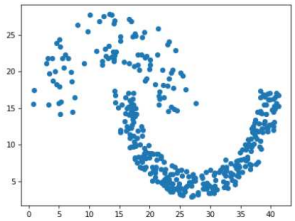
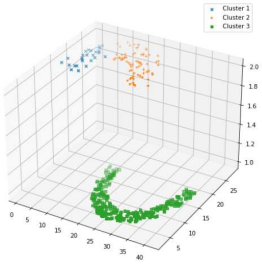
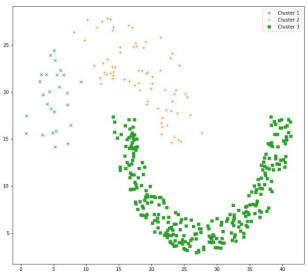
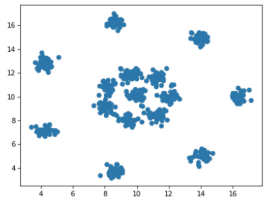
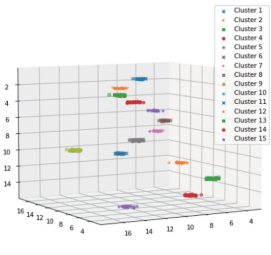
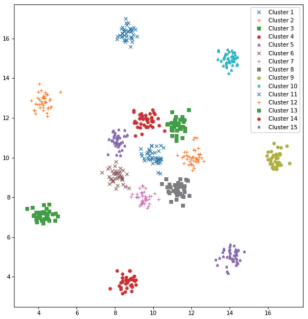
The evaluation of the DBSCAN clustering performance can be based on Silhouette Method or Visual Cluster Interpretation. Since the experimental data that are being used in three dimensional, the performance of DPPA will be based on Visual Cluster Interpretation. The experimental results are demonstrated by showing the original plotting of datasets in 2D graph (left column), 3D graph to show the number of clusters at different dimension (middle column) and the performance of DPPA algorithm (right column) as shown in Table 3.

In summary, the modified DPPA is able to detect the number of clusters as shown in the three-dimensional plots, to handle clusters with arbitrary shapes such as non-globular

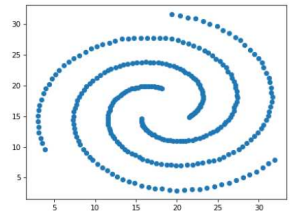
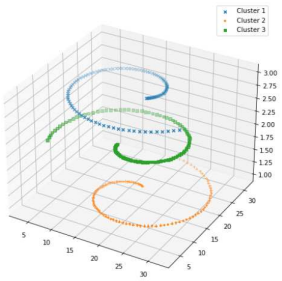
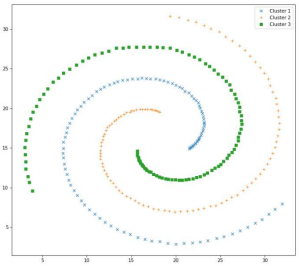
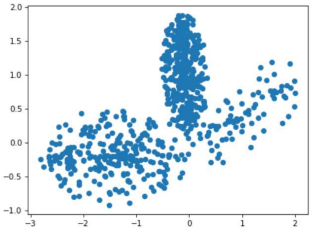
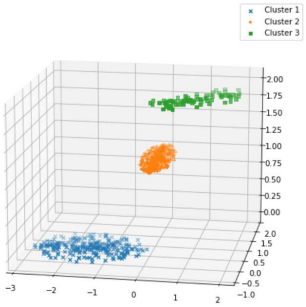
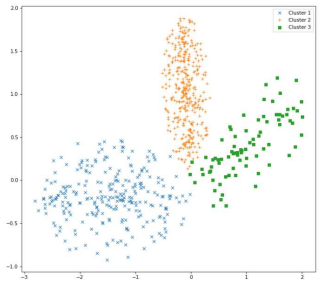
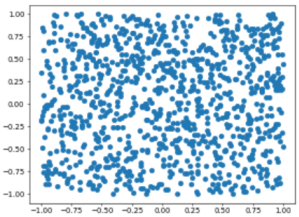
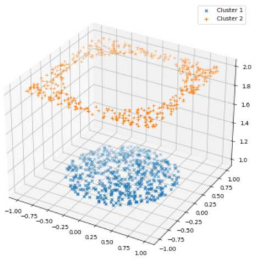
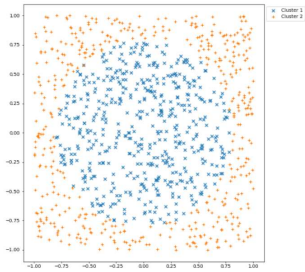
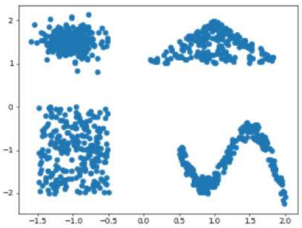
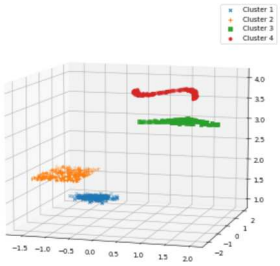
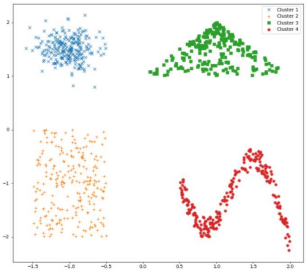
Table 3. Experimental results for DPPA

Data set name (Left Column)	Middle Column: 3D graph	Right Column: Result from DPPA algorithm
Data set name 1: Aggregation [27]	Analysis – Based on the three-dimensional plot, there are 7 main clusters with two data points (noises) that are treated as an additional cluster. These clusters are made of arbitrary shapes and various densities. The performance of DBSCAN has been studied and shown in [28] and [29].	
		
Data set name 2: Flame [30]	Analysis – Based on the three-dimensional plot, there are 2 main clusters with two data points (noises) that are treated as additional cluster. The cluster at the bottom is non-globular shape and these clusters have different densities.	

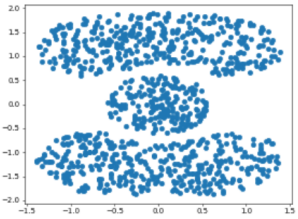
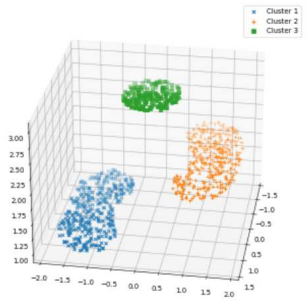
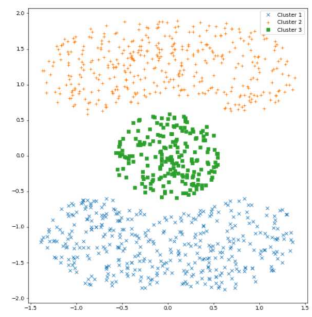
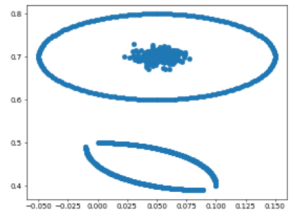
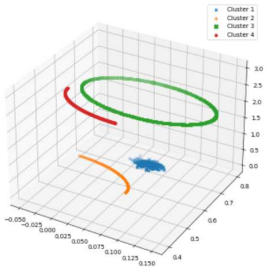
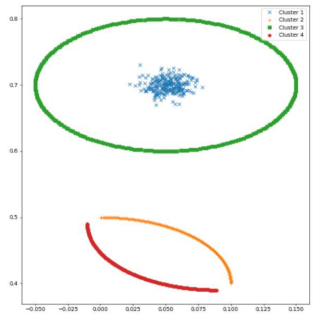
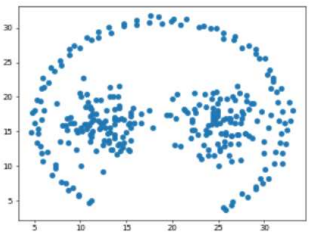
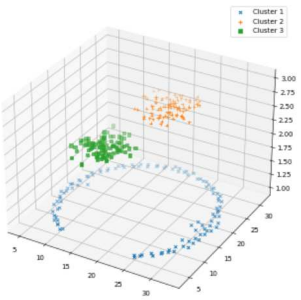
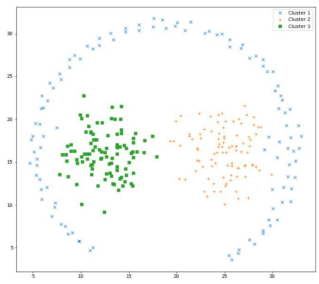
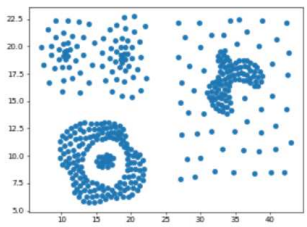
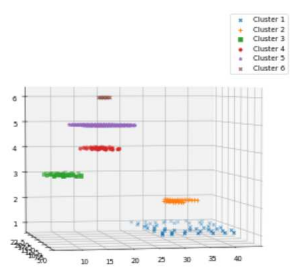
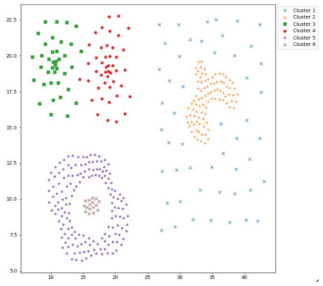
(Continued)

		
Data set name 3: Jain [31]	Analysis – Based on the three-dimensional plot, there are 3 main clusters. Cluster 1 and Cluster 2 are actually separated if it is viewed at different angles on three-dimensional plot. These clusters are made of arbitrary shapes (one of them is non-globular) and have various densities. The result shown by DBSCAN is shown in [29].	
		
Data set name 4: R15 [32]	Analysis – Based on the three-dimensional plot, there are 15 main clusters of similar shapes of similar densities. At two dimensional plots, eight inner clusters appear to be closed to each other, but they are separated by the third axis (Z axis) based on the three-dimensional view. The result shown by DBSCAN is shown in [29].	
		
Data set name 5: Spiral [33]	Analysis – The ability to determine clusters based on spiral shape has been the advantage to the DBSCAN compared to other types of clustering technique. The modified DPPA demonstrates similar capability. It can also handle contiguity-based cluster shape.	

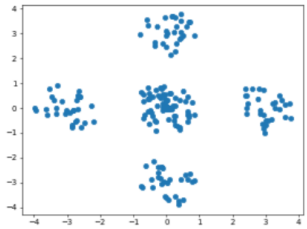
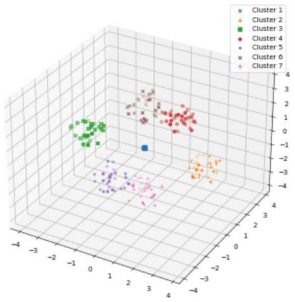
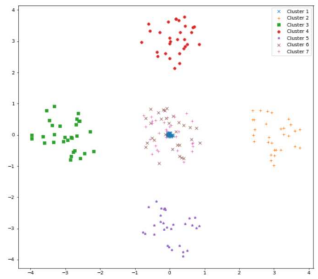
(Continued)

		
Data set name 6: Handl [34]	Analysis – Based on the three-dimensional plot, there are 3 main clusters, and they are mainly separated at Z axis. These clusters show different densities for each cluster.	
		
Data set name 7: Circle [35]		
		
Data set name 8: Shapes [36]	Analysis: There are 4 clusters of arbitrary shapes but spaced out by the Z axis as shown in the three-dimensional plot. These clusters also present different densities.	
		
Data set name 9: Cassini [37]	Analysis: There are 3 arbitrary shapes clusters and two of them are elongated.	

(Continued)

		
Data set name 10: Donutcurves [38]	Analysis – There are 4 clusters with different shapes and densities and they are identifiable as separate clusters. It can also handle ring-type cluster.	
		
Data set name 11: Pathbased [33]	Analysis – There are 3 main clusters with different shapes (contiguity-based) and with different densities.	
		
Data set name 12: Compound [39]	Analysis – There are 5 main clusters with an additional cluster that is actually noise (labeled as Cluster 1). The value of the Z dimension is significant to be able to separate the Cluster 2 and Cluster 1. The result also demonstrates the ability to handle noises, arbitrary shapes and various densities.	
		

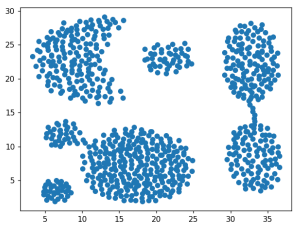
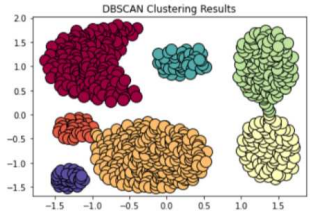
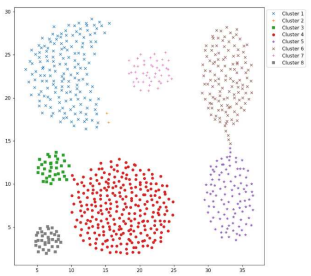
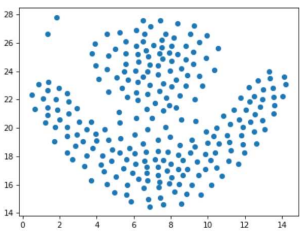
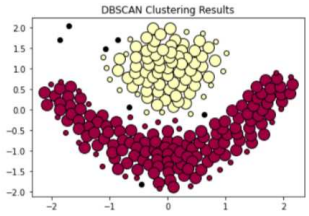
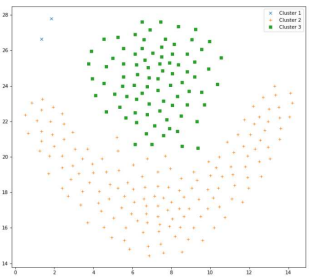
(Continued)

Data set name 13: Hepta [40]. This data set is strongly recommended to be tested for newly developed algorithm.	Analysis – DPPA is able to detect 7 clusters correctly and this is the minimum required capability of any newly developed algorithm to test using this data sets.	
		

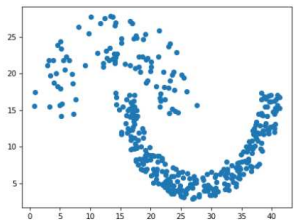
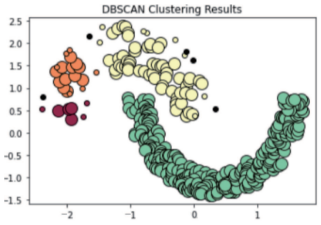
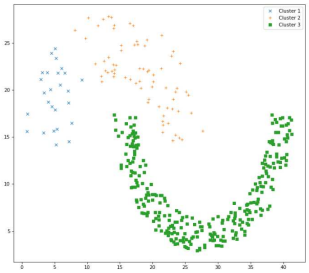
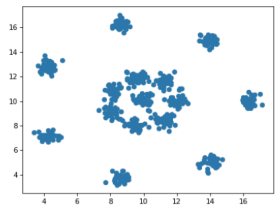
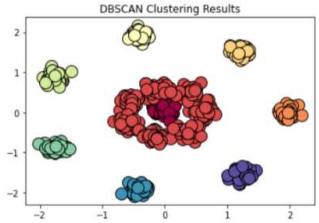
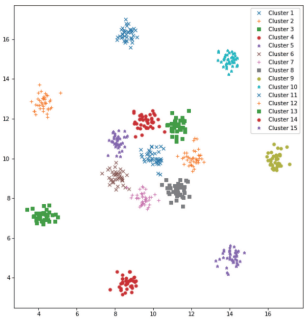
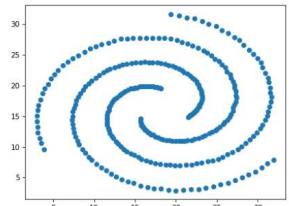
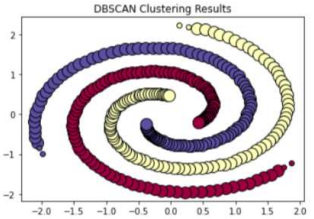
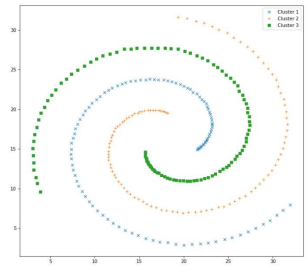
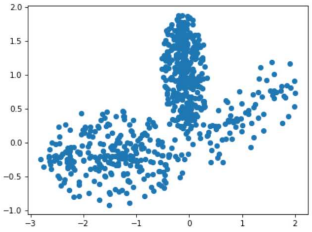
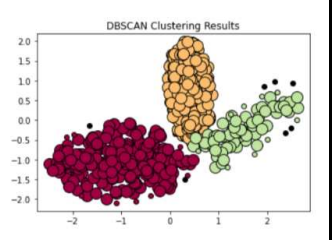
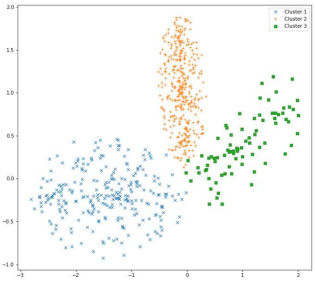
or congruity-based shapes and also to separate noises that surrounded the main cluster as shown in the data set Compound or Flame. The algorithm given is designed for three-dimensional data sets but extendable to higher dimensional data.

4.2. **Comparing DPPA with DBSCAN.** DBSCAN is a prominent clustering algorithm for clustering data with noises and arbitrary shapes. The proposed DPPA algorithm is compared to DBSCAN with $Eps = 0.3$ and $MinPts = 5$. The results and analyses are shown in Table 4.

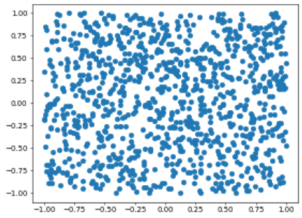
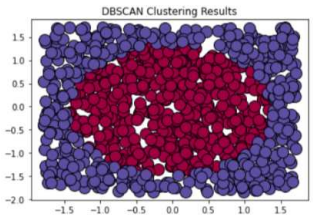
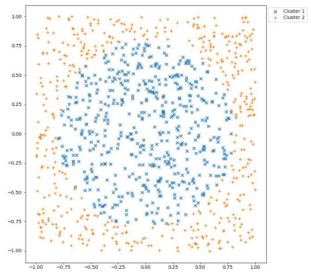
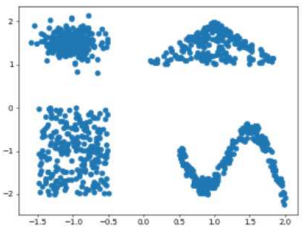
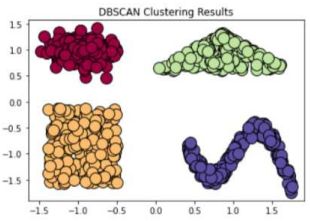
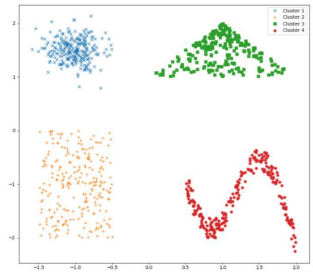
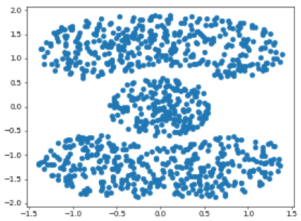
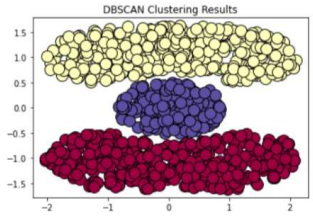
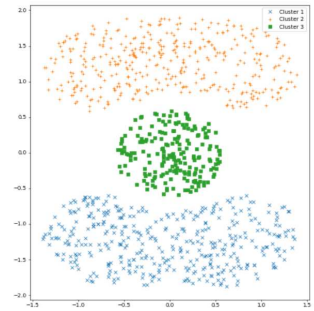
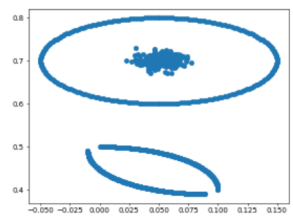
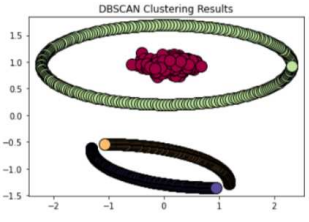
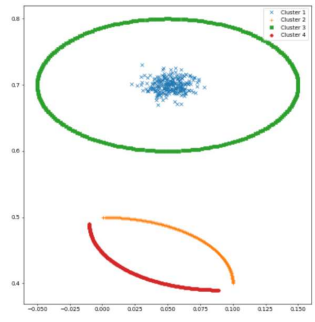
TABLE 4. Experimental results (Comparison between DBSCAN and DPPA)

Data set name (Left Column)	**Middle Column: DBSCAN**	**Right Column: DPPA**
Data set name 1: Aggregation [27]	Analysis – DBSCAN detects 7 clusters while DPPA detects 7 major clusters with two data points which are noises (labelled as cluster 8).	
		
Data set name 2: Flame [30]	Analysis – Both DBSCAN and DPPA detect 2 major clusters. DBSCAN identifies 7 data points as outliers while DPPA detects 2 data points as outliers.	
		

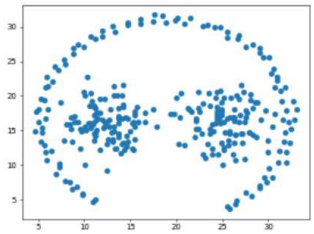
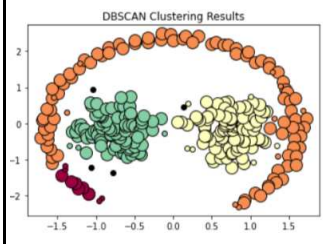
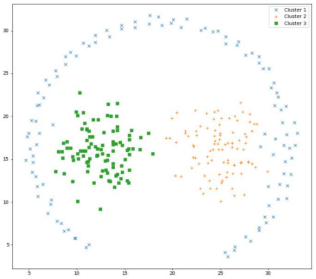
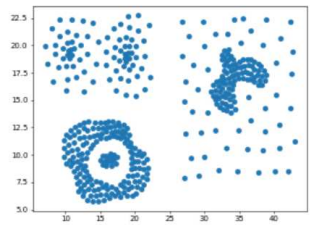
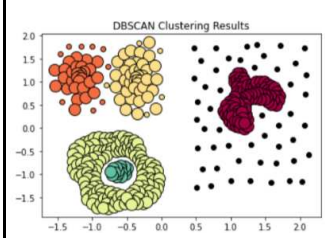
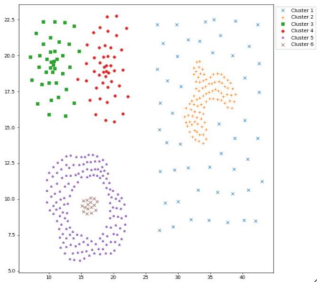
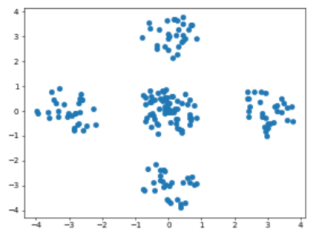
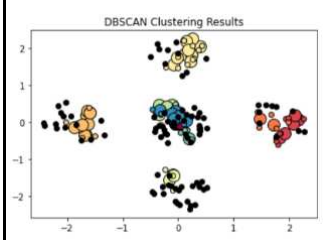
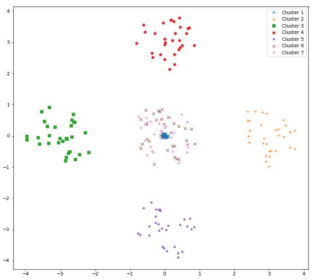
(Continued)

Data set name 3: Jain [31]	Analysis – DBSCAN detects 4 major clusters while DPPA detects 3 major clusters. In the previous Table 2, the 3-D shows there are 3 major clusters.
	 
Data set name 4: R15 [32]	Analysis – DBSCAN detects 9 major clusters while DPPA detects 15 major clusters. In the previous Table 2, the 3-D shows there are 15 major clusters.
	 
Data set name 5: Spiral [33]	Analysis – Both DBSCAN and DPPA detect 3 major clusters. Both algorithms are performing well on linked based clusters with clear separation between the clusters.
	 
Data set name 6: Handl [34]	Analysis – DBSCAN detects 3 major clusters with 6 data points labelled as outliers while DPPA detects 3 major clusters.
	 
Data set name 7: Circle [35]	Analysis – Both DBSCAN and DPPA detect 2 major clusters.

(Continued)

		
Data set name 8: Shapes [36]	Analysis – Both DBSCAN and DPPA detect 4 major clusters.	
		
Data set name 9: Cassini [37]	Analysis – Both DBSCAN and DPPA detect 3 major clusters.	
		
Data set name 10: Donutcurves [38]	Analysis – DBSCAN identifies 3 major clusters as the bottom circle is considered as connected while DPPA detects 4 major clusters. In the previous Table 2, the 3-D shows that the bottom circle is at different height and separated.	
		

(Continued)

Data set name 11: Pathbased [33]	Analysis – DBSCAN detects 4 major clusters with data points that are labelled as outliers while DPPA identifies 3 major clusters. The previous Table 2 shows that there are 3 major clusters.	
		
Data set name 12: Compound [39]	Analysis – Both DBSCAN and DPPA detect 6 major clusters.	
		
Data set name 13: Hepta [40]. This data set is strongly recommended to be tested for newly developed algorithm.	Analysis – DPPA detects 6 major clusters while DBSCAN shows overlappings of clusters of various densities.	
		

There are 7 datasets which show similar results while 5 datasets show discrepancies between DBSCAN and DPPA. 3-D diagram which is shown in previous Table 3 is referred, in order to visually inspect the position of the data points and the clusters.

4.3. **Comparing DBSCAN and DPPA at higher dimensions.** DPPA and DBSCAN are also tested on datasets that consist of 100K data with 8 attributes. The data is made available at (https://www.kaggle.com/datasets/iammustafatz/diabetes-prediction-dataset?resource=download).

Both algorithms are applied on the original data and the attributes are applied without any feature selection. Both algorithms show 8 clusters as shown in Figure 6.

The Silhouette coefficient for both algorithms (DBSCAN and DPPA) is 0.485, which is reasonable since the clusters shown in Figure 6 are close to each other and they are not

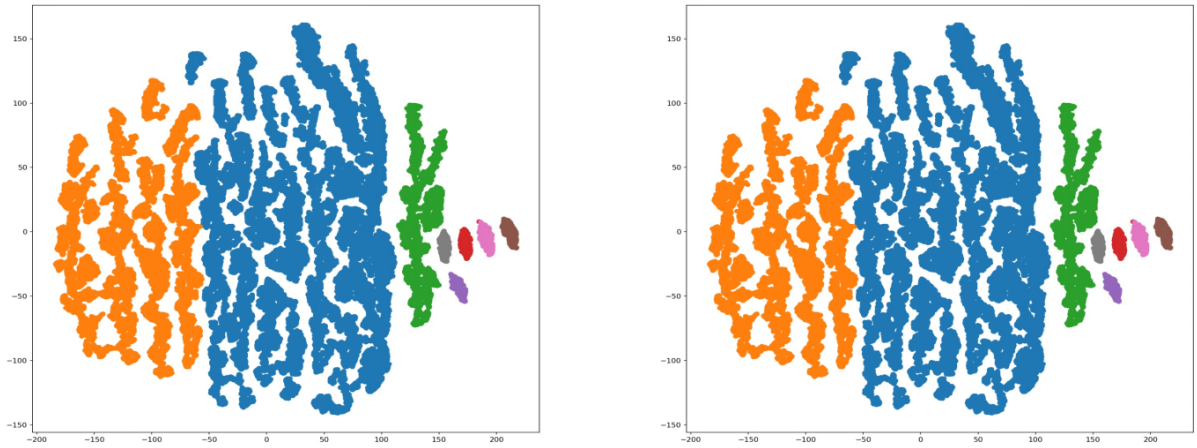


FIGURE 6. DPPA (left) and DBSCAN (right) on high dimension data. DBSCAN *Eps* – 15 and *MinPts* – 5

departed with huge distance based on the two-dimensional plot. It can also be concluded that the performance of DPPA is equally comparable with DBSCAN.

4.4. **Time complexity analysis T( ) for DPPA algorithm.** Time complexity is being used to analyze the performance of an algorithm [41]. Similar step is taken to evaluate the behaviour of the DPPA algorithm with respect to the time complexity. The following is the analysis and explanation on the complexity analysis of the algorithm as shown in Table 5.

Based on the calculation above, time complexity is taken by the largest value of the summation of all time complexity analysis which reveals $O\left(\frac{n^2}{2}\right)$. Hence, it has better time complexity than the worst case of traditional DBSCAN which is $O(n^2)$. Nevertheless, DBSCAN performs better in the best-case scenario where the time complexity is $O(n \log n)$.

5. **Conclusion.** This work falls under the third category of many research work in managing the issues of DBSCAN initial parameterization which are *MinPts* and *Eps*. Since DPPA does not depend on both parameters, the new algorithm in searching for the nearest neighbor has been changed which uses 1-NN and Max-NN concepts. With these concepts, to find the suitable radius *Eps* and the minimum number of points, *MinPts* is no longer needed as the radius range, λ, is automatically calculated by analyzing the positioning of the data points and the distancing between them. The calculation involves measuring the minimum distance of each data point to all data points and repeat for all available data points. The lowest and highest values of all minimum distance will be assigned as the radius range. Instead of searching the nearest neighbor in a circle manner based on the *Eps*, the new algorithm searches the links of data points based on its nearest neighbor that falls within the radius range. Hence, two clusters can be connected as long as there are few data points from both clusters that are near within the λ range. The proposed DPPA algorithm is applied to 13 benchmark data sets, and it is demonstrated that it manages to identify the right number of clusters, manage clusters with arbitrary shapes and handle the data sets with noises. The 13 data sets are used as benchmark for newly proposed clustering algorithm. For each data set, the DPPA algorithm automatically calculates the $(\max(\phi))$ and $(\min(\phi))$. The robustness is measured based on the ability to detect the number of clusters, the ability to cluster of arbitrary shape (like non-globular) which is shown at every data set in the experimental result. The experiment is extended to test a

TABLE 5. Analysis of DPPA algorithm

Algorithm	**Explanation**	**Time complexity analysis**
1. Given a collection of data points $p_i \in \delta$, for each data point, calculate the distance $d(p_i, p_{i+1})$, for $\forall p_{i+1} \in \delta$ and $i = \{1, \ldots, m\}$ where m is the number of data points in δ. The distance is measured based on the coordinates x, y and z where $d(p_i, p_{i+1}) = \left(\sqrt[2]{\left(p_i^x - p_{i+1}^x\right)^2 + \left(p_i^y - p_{i+1}^y\right)^2 + \left(p_i^z - p_{i+1}^z\right)^2}\right)$.	This is performed with nested loop (two-for loop). The outer loop and inner loop have the same loop-size which is n, number of data. The worst case and best case are the same as this step is independent on the position of the data.	$\frac{n(n-1)}{2} = \mathrm{T}\left(\frac{n^2}{2}\right)$
2. For $\forall p_i$, determine $\min(d(p_i,\ p_{i+1}))$, $i = \{1, \ldots, m\}$ $\overset{stores}{\longrightarrow} \phi(\min(d(p_i)), \ldots, \min(d(p_m)))$.	Since the distance is already calculated in Step 1, the searching will be Worst case: $\mathrm{T}(n-1)$ Best case: $\mathrm{T}(1)$	The constant will be ignored; hence we adopt $\mathrm{T}(n)$.
3. Determine radius range, $\lambda(\min(\phi), \max(\phi))$ where ϕ is some scalar value.	The searching for $\lambda(\min(\phi), \max(\phi))$ will be made in linear. Worst case: $\mathrm{T}(n-1)$ Best case: $\mathrm{T}(1)$	The constant will be ignored; hence we adopt $\mathrm{T}(n)$.
4. For each point $p_i \in \delta$, a) Find $n_{p_i} = \sum_{i=0}^{m} \begin{pmatrix} \min(\phi) \le d(p_i) \le \max(\phi) \rightarrow add\ 1 \\ add\ 0 \end{pmatrix}$ where n_{p_i} is the number of neighbors for data point p_i that has distance within radius range λ. b) Build the neighbor-link table for 1-NN and Max-NN for each data point $d(p_i)$ as shown below. Note: 1-NN: Some data point p_i where $\min(\phi) \le d(p_i) \le \max(\phi)$ for each data point p_i where $i = \{1, \ldots, m\}$. Max-NN: A linkage of data points begins with p_i and followed by every 1-NN of p_i until p_i has no 1-NN.	Worst case: A data point builds a link to all data points (note: this happens to a case where the point that links end to end between two extreme sides in the cluster and having the highest Max-NN such that Max-NN $< n$. Hence, the algorithm behavior will be $\mathrm{T}(n)$. Best case: The smallest link of Max-NN is to have only one neighbor; hence the behaviour is constant.	Worst case: $\mathrm{T}(n)$ Best case: $\mathrm{T}(1)$
5. Sort the p_i in the neighbor-link table based on the n_{p_i} in an ascending order a) Place p_i in cluster C_i followed by all p_k that are in Max-NN. b) Place all p_j of 1-NN for p_i to C_i. c) Next p_{i+1}, if $p_{i+1} \in C_k$, then $C_i \leftarrow C_k$, $p_i \leftarrow p_{i+1}$ and goto a). Repeat until no more data point.	Each link with the longest link will be placed in a cluster together with all the points connected to its chain. This will be repeated until no more data points are available. This step has the same performance for both best and worst cases.	$\mathrm{T}(n)$ as the performance is influenced by the number of data, n.

higher dimensional data set with 16 attributes and 100K data, the Silhouette coefficient for both DPPA and DBSCAN is the same.

DPPA relies on the distance calculated between the data points to determine its nearest neighborhood. There are two implications to this method which are i) the distance calculated by measurement may not correctly reflect the distance visually measured; hence what is considered as near distance in visual sense may not be the same as calculated and ii) the data points differences probably will be calculated more accurately if each point has feature values associated to it, such as financial or business information. It will be an interesting investigation which is left as the future work [41]. The weakness of the DPPA algorithm is that it has to test all points to ensure that it has measured the $(\max(\phi))$ and $(\min(\phi))$ accurately. The performance of the algorithm is still acceptable with 100K data; however, it may face challenges with trillions of data. Hence, the proposed future research work is to perform partitions on the datasets and merge the 1-NN and Max-NN.

REFERENCES

[1] W. Wei, X. Zhang and L. Yang, Full-cycle state evaluation of S700K switch machine based on residual network and fuzzy clustering, *International Journal of Innovative Computing, Information and Control*, vol.18, no.4, pp.1203-1216, 2022.

[2] A. A. Aldino, D. Darwis, A. T. Prastowo and C. Sujana, Implementation of K-means algorithm for clustering corn planting feasibility area in south lampung regency, *Journal of Physics: Conference Series*, 2021.

[3] L. Xiong and Y. Yao, Study on an adaptive thermal comfort model with K-nearest-neighbors (KNN) algorithm, *Building and Environment*, vol.202, no.1, DOI: 10.1016/j.buildenv.2021.108026, 2021.

[4] M. Ester, H. P. Kriegel, J. Sander and X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, *KDD*, vol.96, no.34, pp.226-231, 1996.

[5] A. Rodriguez and L. Alessandro, Clustering by fast search and find of density peaks, *Science*, vol.344, no.6191, pp.1492-1496, 2014.

[6] M. Pavan and M. Pelillo, Dominant sets and pairwise clustering, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.29, no.1, pp.167-172, 2006.

[7] K. G. Derpanis, Mean shift clustering, *Lecture Notes*, vol.32, pp.1-4, 2005.

[8] M. Zhao, A. Jha, Q. Liu, B. A. Millis, A. Mahadevan-Jansen, L. Lu, B. A. Landman, M. J. Tyska and Y. Huo, Faster Mean-shift: GPU-accelerated clustering for cosine embedding-based cell segmentation and tracking, *Medical Image Analysis*, vol.71, 102048, 2021.

[9] B. J. Frey and D. Dueck, Clustering by passing messages between data points, *Science*, vol.315, pp.972-976, 2007.

[10] F. Chen, An improved DBSCAN algorithm for adaptively determining parameters in multi-density environment, *2021 2nd International Conference on Artificial Intelligence and Information Systems (ICAIIS'21)*, Chongqing, 2021.

[11] J. Hou, H. Gao and X. Li, DSets-DBSCAN: A parameter-free clustering algorithm, *IEEE Transactions on Image Processing*, vol.25, no.7, pp.3182-3193, 2016.

[12] A. Sharma and A. Sharma, KNN-DBSCAN: Using k-nearest neighbor information for parameter-free density based clustering, *2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, 2017.

[13] E. Azhir, N. J. Navimipour, M. Hosseinzadeh, A. Sharifi and A. Darwesh, An efficient automated incremental density-based algorithm for clustering and classification, *Future Generation Computer Systems*, pp.665-678, 2021.

[14] G. Kinsuk, T. K. Biswas and P. Sarkar, ECR-DBSCAN: An improved DBSCAN based on computational geometry, *Machine Learning with Applications*, 2021.

[15] B. Mu, M. Dai and S. Yuan, DBSCAN-KNN-GA: A multi density-level parameter free clustering algorithm, *IOP Conf. Series: Materials Science and Engineering*, 2020.

[16] M. Z. Hossain, M. J. Islam, M. W. R. Miah, J. H. Rony and M. Begum, Develop a dynamic DBSCAN algorithm for solving initial parameter selection problem of the DBSCAN algorithm, *Indonesian Journal of Electrical Engineering and Computer Science*, vol.23, no.3, pp.1602-1610, 2021.

[17] W. Lai, M. Zhou, F. Hu, K. Bian and Q. Song, A new DBSCAN parameters determination method based on improved MVO, *IEEE Access*, pp.1-11, 2019.

[18] Y. Lv, T. Ma, M. Tang, J. Cao and Y. Tian, An efficient and scalable density-based clustering algorithm for datasets with complex structures, *Neurocomputing*, vol.171, pp.9-22, 2016.
[19] B. J. Frey and D. Dueck, Clustering by passing messages between data points, *Science*, vol.315, pp.972-976, 2007.
[20] X. Chen, W. Liu, H. Qiu and J. Lai, APSCAN: A parameter free algorithm for clustering, *Pattern Recognition Letters*, pp.973-986, 2011.
[21] Z. Ding, H. Xie and P. Li, Evolutionary parameter-free clustering algorithm, *2021 IEEE 2nd International Conference on Pattern Recognition and Machine Learning*, 2021.
[22] G. F. Yaşar, S. UTKU and G. Ulutagay, FN-DBSCAN-GM: A parameter free and robust version of DBSCAN algorithm, *International Conference on Research in Education and Science (ICRES)*, Kuşadası, Turkey, 2017.
[23] Y. Wu, Parameter free clustering algorithm based on density and natural nearest neighbor, *2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM)*, 2019.
[24] A. Bryant and K. Cios, RNN-DBSCAN: A density-based clustering algorithm using reverse nearest neighbor density estimates, *IEEE Transactions on Knowledge and Data Engineering*, vol.30, no.6, pp.1109-1121, 2018.
[25] D. Demirovic, An implementation of the mean shift algorithm, *Image Processing On Line*, vol.9, pp.251-268, 2019.
[26] S. M. F. D. S. Mustapha, B. Theruvil and M. Madanan, Visual comparison of clustering using link-based clustering method (LbCM) without predetermining initial centroid information, *ICIC Express Letters, Part B: Applications*, vol.12, no.4, pp.317-323, 2021.
[27] A. Gionis, H. Mannila and P. Tsaparas, Clustering aggregation, *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol.1, no.1, pp.1-30, 2007.
[28] S. Weng, J. Gou and Z. Fan, *h*-DBSCAN: A simple fast DBSCAN algorithm for big data, *Proc. of the 13th Asian Conference on Machine Learning (ACML 2021)*, Virtual Conference, 2021.
[29] L. Xu, J. Zhao, A. Shi and Z. Chen, Density peak clustering based on cumulative nearest neighbors degree and micro cluster merging, *Journal of Signal Processing Systems*, vol.91, no.10, pp.1219-1236, 2019.
[30] L. Fu and E. Medico, FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data, *BMC Bioinformatics*, vol.8, DOI: 10.1186/1471-2105-8-3, 2007.
[31] A. K. Jain and M. H. C. Law, Data clustering: A user's dilemma, in *Pattern Recognition and Machine Intelligence. PReMI 2005. Lecture Notes in Computer Science*, S. K. Pal, S. Bandyopadhyay and S. Biswas (eds.), Berlin, Heidelberg, Springer, 2005.
[32] C. J. Veenman, M. J. T. Reinders and E. Backer, A maximum variance cluster algorithm, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.24, no.9, pp.1273-1280, DOI: 10.1109/TPAMI.2002.1033218, 2002.
[33] H. Chang and D. Yeung, Robust path-based spectral clustering, *Pattern Recognition*, vol.41, no.1, pp.191-203, 2008.
[34] J. Handl and J. Knowles, *Multiobjective Clustering with Automatic Determination of the Number of Clusters*, Technical Report TR-COMPSYSBIO-2004-02, UMIST, Manchester, 2004.
[35] *Deric, Clustering-Benchmark, 11 November 2015*, https://github.com/deric/clustering-benchmark/blob/master/src/main/resources/datasets/artificial/circle.arff, Accessed on 14 July 2022.
[36] *Deric, Clustering-Benchmark, 11 November 2015*, https://github.com/deric/clustering-benchmark/blob/master/src/main/resources/datasets/artificial/shapes.arff, Accessed on 14 July 2022.
[37] *Deric, Clustering-Benchmark, 11 November 2015*, https://github.com/deric/clustering-benchmark/blob/master/src/main/resources/datasets/artificial/cassini.arff, Accessed on 2022 July 14.
[38] *Deric, Clustering-Benchmark, 10 November 2015*, https://github.com/deric/clustering-benchmark/blob/master/src/main/resources/datasets/artificial/donutcurves.arff, Accessed on 14 July 2022.
[39] C. Zahn, Graph-theoretical methods for detecting and describing gestalt clusters, *IEEE Transactions on Computers*, vol.100, no.1, pp.68-86, 1971.
[40] A. Ultsch, Clustering with SOM: U*C, *Proc. of Workshop on Self Organizing Feature Maps*, Paris, 2005.
[41] W. Xiang, Analysis of the time complexity of quick sort algorithm, *2011 International Conference on Information Management, Innovation Management and Industrial Engineering*, Shenzhen, China, 2011.

Author Biography

S. M. F. D. Syed Mustapha received the B.Sc. (Computer Science) from University of Texas at Permian Basin, USA in 1991; M.Phil. and Ph.D. from University of Wales, UK in 1995 and 1997, respectively in the area of ontology and reasoning.

He is a full professor at the College of Technological Innovation, Zayed University, UAE. His research interest covers knowledge modelling, information retrieval, clustering and knowledge sharing. He has published more than 100 papers in the reputable journal and conferences.