Combination of Density-Based Spatial Clustering of Applications with Noise Method with Grid Search to Improve Complexity Using Nash Equilibrium

Uranus Kazemi
Arak University
Seyfollah Soleimani
s-soleimani@araku.ac.ir

Arak University

Research Article

Keywords: Density-Based Spatial Clustering of Applications with Noise (DBSCAN)- Clustering- Grid Search- Density- Nash Equilibrium

Posted Date: March 15th, 2024

DOI: https://doi.org/10.21203/rs.3.rs-4087100/v1

Additional Declarations: No competing interests reported.

Combination of Density-Based Spatial Clustering of Applications with Noise Method with Grid Search to Improve Complexity Using Nash Equilibrium

Uranus Kazemi[1], Seyfollah Soleimani[1,*]
[1]Department of Computer Engineering, Faculty of Engineering, Arak University, Arak 38156-8-8349, Iran

Abstract - One of the important issues in data processing is clustering, the purpose of which is to find similar patterns in the data. Many clustering methods differ in their approaches and similarities. The density-based spatial clustering of applications with noise (*DBSCAN*) clustering method is one of the most practical density-based clustering methods that can identify training samples with different shapes, and for this reason, it has many applications in different fields. Although this method has its advantages, it has some weaknesses, such as the lack of proper performance in big data, the difficulty of determining Epsilons (*Eps*) and the Minimum number of points (*Minpts*) parameters for optimal clusters, etc. To solve these problems, in this paper, a dynamic method is used to solve the problem of identifying clusters with different densities, and another method is used to increase the speed of the algorithm and reduce the computational complexity. Testing the new method on several sets of data shows that the proposed method has a high efficiency in clustering and outperforms the density-based spatial clustering of applications with noise (*DBSCAN*) method in terms of complexity and efficiency.

Keywords: Density-Based Spatial Clustering of Applications with Noise (*DBSCAN*)- Clustering- Grid Search- Density- Nash Equilibrium.

1. Introduction

In the last decade, data production has experienced significant growth, and this data is being produced from various sources, including mobile phones, wireless sensor networks, etc. Managing this large amount of data has become a big challenge in today's era. Data clustering is proposed as a solution that groups data based on their similarity. In the following, we will discuss different aspects in three sections:

1.1. Background and Motivation

By entering the age of information and communication and starting to use data and information as the main assets in the scientific, economic, social, and cultural movements of societies, organizations, and various companies, the development of people's participation in the world of the internet and network communications in the world became a concern. This concern was related to the type of data that was produced every day and at a terrible speed in the world and in various fields where information technology is the same as the previous time, and how to handle this volume and variety of data and information with Paying attention to the structures that exist in the information technology space can be managed, controlled, and processed, and it can be used to improve the structures and increase profitability.

So far, many methods have been proposed for data processing, from greedy algorithms to heuristic algorithms, data clustering, and using search to find different parameters for data clustering. In

* s-soleimani@araku.ac.ir

other words, one of the most important methods for extracting data from the data is clustering. Clustering divides a set of data into different groups so that the samples that are in the same group are very similar to each other, but the similarity of other samples is different. The similarity or dissimilarity of the samples is measured by criteria based on distance, etc. [1]. It has been suggested that the DBSCAN clustering method is a density-based clustering method that determines the structure of the clusters according to the data distribution. It automatically determines the number of final clusters according to the nature of the data, is sensitive to noise, and is compatible with any cluster shape. What is ignored in many of these solutions is the fact that, due to the challenges in the data, such as processing, management, security, etc., it is not possible to find a suitable search for the parameters in the DBSCAN clustering and the cost of the search. And the complexity of the calculations increases; also, this clustering method is not performed with high accuracy due to the dependency on the primary parameters.

Therefore, according to the application areas, this clustering method can be applied to many clustering problems because it has good performance in different fields such as analysis, urban planning, system development, etc. Our focus is to be able to process data correctly, then provide a solution in this process, to search for DBSCAN clustering parameters so that we can process data with appropriate speed and accuracy, and then reduce the search cost and the complexity of calculations. In other words, in this article, a combination of parameter-based search methods in game theory and density-based methods is used to reduce the complexity of calculations and increase accuracy and speed, and DBSCAN clustering is used as part of the main algorithm of this article.

1.2. Literature Review

As mentioned, the purpose of clustering is to quickly and reliably access related information and identify the logical relationship between them. Each cluster has a representative that represents the cluster and often represents the center of the cluster. The degree of similarity of the data to the center of the cluster is generally determined by a parameter called the similarity criterion. In other words, clustering as a tool in data mining has many uses, including biological data division, big data, data reduction, noise filtering, outlier data discovery, etc. [2]. Due to their great use, there have been extensive studies in this field, which have led to the discovery of various methods. In general, these methods can be divided into four categories:

- Partition Clustering: This clustering method classifies information into several groups based on the characteristics and similarity of the data. Data analysts determine the number of clusters that should be generated for clustering methods [3]. In the partition clustering method, when the database *(D)* contains several objects (*N*), the partition method creates the user-specified partition (*K*) from the data, in which each partition represents a cluster and a specific region, and each object will belong to only one cluster. These methods work based on the distance between objects; that is, they receive n objects and divide them into k groups. One of the famous algorithms for these methods is mentioned as *K-means* [4].
- Hierarchical methods: In these methods, a set of data is divided into different levels. These methods are divided into two categories based on how they work: bottom-up and top-down. The important features of these methods are that they used to find clusters with a spherical shape. In general, it can be said that hierarchical clustering is a clustering method whose purpose is to build a hierarchy of clusters. In the hierarchical clustering method, each level of the hierarchy displays a category of data that can be viewed in the form of a tree, where

the leaves of the tree represent an initial observation and the root of the tree is the collection of all observations [5].

- Density-based methods: Density-based clustering refers to unsupervised learning methods that identify distinct clusters in the data. This type of clustering is inherently defined for continuous space, unlike other clustering methods that rely on distance criteria to cluster objects. These methods use the density of objects in a small area for clustering [6]. Based on the idea that a cluster in a data space is a continuous region with high point density, other clusters are separated by contiguous regions of low point density. The density of data points is usually considered noise in the separating areas with low point density [7]. In this method, points located in a certain range (a certain neighborhood radius) are placed in a cluster. A minimum density is usually considered in the methods, and clustering is done in the areas where this minimum is met, and it is considered the best way to find clusters with an arbitrary and varied shape. One of the most famous of these methods is density-based clustering (*DBSCAN*) [8].
- Network-based methods (Grid): In these methods, the data is divided into a limited number of cells; in other words, they are divided into cells, and a grid structure is created that can perform clustering operations for each of the cells. [9]. These methods can be combined with other algorithms, such as density-based methods where each point has a high density and hierarchical methods [10].

Clustering algorithms discover natural structures in data sets. In recent years, several algorithms have been proposed for data clustering [11–13]. Many researchers develop different clustering algorithms or modify existing approaches. In most of these algorithms, the number of clusters must be determined in advance, and the algorithm itself cannot obtain the optimal number of clusters. This is because the number of clusters is not known for many real data sets, and even an approximation of their number cannot be determined [14]. As mentioned, data clustering is used in various fields, but the key issue is the correct selection of input parameters because the same algorithm can provide different results depending on the applied parameters [15]. These problems can be solved by using different indicators for clustering. Due to the existence of different clustering methods and their advantages and disadvantages, changes in these methods make the created clusters not suitable for data density, but in the *DBSCAN* clustering method, clusters of different shapes and sizes are discovered and need to be determined by *Eps* and *Minpts* parameters; the determination of these parameters is very important for the correct operation of this method [16]. To reduce the complexity of *DBSCAN*, many methods have been provided, and in this section, we briefly review these methods:

In network-based methods, it is tried to reduce the complexity of *DBSCAN* by dividing the feature space of the network to reduce the search time by only considering the adjacent network. In the *GridDBSCAN* method, the feature space is divided into grids with equal sizes, and the points in each grid and the existing points around the grid are considered a group. For each point in a particular grid, its neighbors are placed in a group, which reduces the search time. This method requires a parameter given by the user [10].

Gunawn presented a network-based method called *G13* [17]. In the worst possible case, it has a runtime complexity of *O(nlogn)* for two-dimensional data. A network must have at least *Minpts* main points, and using the network structure, only points in neighboring networks are considered when finding. Considering that *G13* was only suitable for two-dimensional data, Gan and *Tayo* [18] extended this method to more than two-dimensional data so that the *DBSCAN* method could

be executed in sub-quadratic time and presented an approximate algorithm for *DBSCAN*. to be executed in *O(n)* time.
Other methods, such as *TI-DBSCAN* [19] and *TI-DBSCAN-REF* [20], have similar clustering to the *DBSCAN* method. Unlike the *DBSCAN* method, these methods use spatial indexes and triangular inequality to reduce the search space. In the [21] method, *ST-DBSCAN* adds three solutions to *DBSCAN*, including the identification of the core data and noise data of neighboring clusters, and improves *DBSCAN* in two ways. (1) clusters spatial and temporal data based on non-spatial spatial and temporal features; (2) This method assigns a density factor to each cluster to make noisy data in clusters that have different densities, unlike *DBSCAN*, detectable. The *P-DBSCAN* method [22] uses a series of labels to analyze the location of data. The main purpose of this method is to find locations with the help of a large number of photos and labels. Li Ma presented the *MRG-DBSCAN* method [23], in which the implementation of the *DBSCAN* method and the generation of central points are done using Map-Reduce.
Nash equilibrium has wide applications, including in the social sciences [24], engineering problems [25], intelligent networks [26], big data [27], etc. As an example, it can look for non-cooperative game problems that include equality and inequality constraints. In addition, it is used to solve network problems on large scales with big data [28], where the data, the objective function, and the feasible set of each player are maintained by each representative of the players. In many cases, Nash equilibrium seeks to solve problems in the real world [29]. On the other hand, existing functions or constraints may be indistinguishable. On the other hand, non-smooth methods may perform better than convergence features [30].

1.3. Contribution

What we present in this article is a clustering method based on grid search with the density-based method. The proposed method improves the *DBSCAN* clustering method in several ways. In other words, it uses a dynamic method to solve the problem of identifying clusters with different densities and another method to increase the speed of the algorithm and reduce the computational complexity. Therefore, the innovations in this article are as follows:

- Using a dynamic method to solve the problem of identifying clusters with different densities.
- Increasing the speed of *DBSCAN* clustering by determining its appropriate parameters in different data.
- Using the Nash equilibrium to find the *DBSCAN* parameters.
- Reducing computational complexity in clustering different data.

2. Definition of the problem

In this section, we define the problem:
Definition 1: Suppose there is a set of data that we want to cluster, clustering is the division of the data set $A= \{a_1, a_2, \dots, a_n\}$ into $K=\{c_1, c_2, \dots, c_K\}$ clusters that meet the following conditions:

- Each data point must be assigned to a cluster.
- Each cluster must be assigned at least one data point.
- Each data point must be assigned to only one cluster.
- Data that are similar to each other—in other words, their distance from each other is small—should be placed in one cluster as much as possible, and data that are far apart should be placed in different clusters.

The DBSCAN clustering method, which works based on density, depends on two parameters: the neighborhood radius and the minimum number of points [31–32]. The general idea of this algorithm is that it starts from a random point, and if that point has the minimum density (in other words, there are at least a certain number of points in its neighborhood radius), it assigns itself to a new cluster and then checks all the points that are in its neighborhood. Each of the points with a high density is recursively examined, and this continues until all points are assigned to clusters. In the following, some important concepts in this method are formally defined:

Definition 2: Neighborhood radius is one of the two parameters of the *DBSCAN* method, which is denoted by *Eps*. This parameter specifies at what radius from a point there should be a sufficient number of points.

Definition 3: The minimum number of points that must exist in the neighborhood of a point is indicated by the *Minpts* parameter.

Definition 4: A point is called a core point if there are at least *Minpts* points within the radius *Eps* from it.

Definition 5: Directly accessible points: A point a_i is directly accessible from a point like a_j if:

- a_j is in the neighboring distance *Eps* from a_i.
- a_i is a central point.

Definition 6: Reachable Points: A point a_j is reachable from a point a_i if there is a set of central points $a_i, \dots, a_j$ from a_i to a_j.

Definition 7: Connected points: two points a_i and a_j are connected if there is a central point a_0 such that both points a_i and a_j are reachable from a_0.

In the *DBSCAN* method, both points are connected in one cluster according to definition 7. This method traverses the data once. In the simple version of this method, to find points close to a point, the distance of each point must be measured, so the complexity of this method will be of the order of $O(n^2)$ [33]. Using the same index and optimal data structure. The complexity of this method is reduced to $O(n \log n)$.

One of the problems of the mentioned algorithm is that it depends on the neighborhood radius parameters and the minimum number of points, and these parameters depend on the neighborhood radius and the minimum number of points, and these parameters are considered the same for all points. Therefore, if the nature of the initial data set should consist of several clusters with different densities, in other words, it includes very dense clusters and clusters with less density, the algorithm will not be able to correctly identify the clusters. On the other hand, the time complexity of the algorithm is relatively high, and if the number of points is too high, its execution time will be high. In the proposed method, these two defects have been solved.

Definition 8: The data can be divided into grids, that is, the data space is first divided into grids to create a spatial index. Based on grids, considering only the data in adjacent grids of a given data accelerates the search objective. According to definition 1, if $A = \{a_1, a_2, \dots, a_n\}$ is a set of data, this data is normalized in d-dimensional space and from zero to one. The length of the network L is a maximum value between $\rho = 0^{\infty} \left\{\frac{1}{2^{\rho}}\right\}$ less than $\frac{\varepsilon}{\sqrt{d}}$ d. Each network has a unique key and a combination of network commands for each dimension [34].

Definition 9: Nash equilibrium which is known as Nash solution [35]. In game theory [36] it is an outcome in a non-cooperative game for two or more players that cannot be improved by changing the strategy of any player. Nash equilibrium is a key concept in game theory, where it defines the solution of non-cooperative games with n players. If each game consists of n players and each player i has a strategy set S_i and each player has a cost functionπ_i: $S \to R$, then a strategy$t_i \to S_i$

is the best answer. If the other strategy does not produce a cost function in S_i, then $\{s_1, s_2, \dots, s_n\}$ is a complete strategy for each player. If a strategy is the best response among other strategies of the characteristic, then this characteristic is a Nash equilibrium [37].

3. Proposed method

Due to the problems in the *DBSCAN* clustering method, such as the lack of a specific order for selecting points and the presence of different data densities, this method cannot provide proper clustering [38-39]. The proposed method is a combination of network-based and density-based methods with the idea of Nash equilibrium, which is used for proper clustering of data with different shapes, increasing clustering speed, and reducing computational complexity. In this approach, to solve cluster identification with different densities, Nash equilibrium along with dynamic radius and selecting cells from the network according to Nash equilibrium are used to increase the speed of the algorithm. In the following, the steps of the proposed method are described, and then its formal definition is discussed:

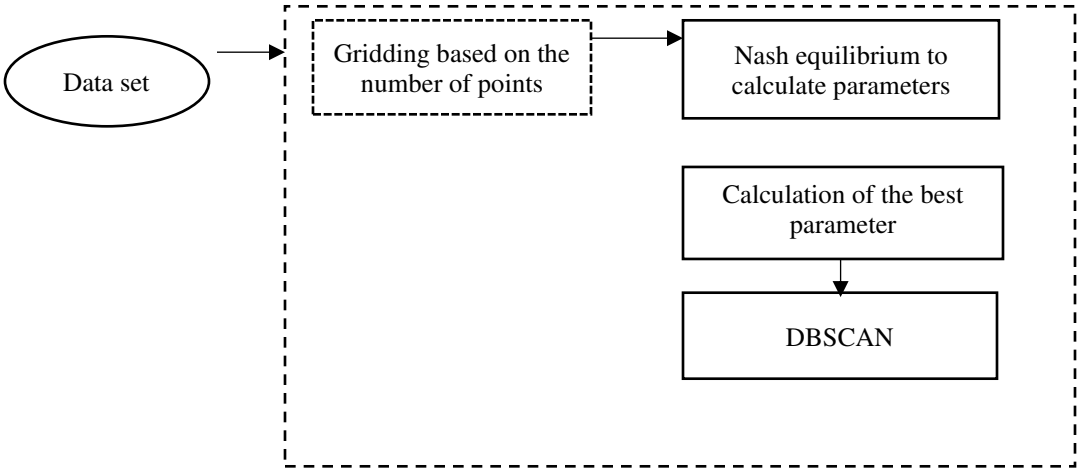

Figure 1- An overview of the proposed method

First step: the number of grid cells is determined based on the number of points. Then the clustering starts from the cells that have the highest number of points. After finding the clusters that have a very high density, the radius value for clustering is determined based on the grid search mentioned in the first algorithm. In this step, we assume that the number of points in the data set is *n*. We consider the dimensions of the grid structure as $\sqrt{n} \times \sqrt{n}$, then we place each point in its corresponding cell in the grid structure. which can be seen in Figure 2.

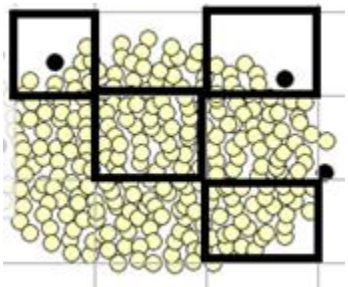

Figure 2- An example of data where each point is assigned to the corresponding cell.

Second step: based on the grid search that is done to determine the value of the radius, it first trains each data set with the provided values and provides the best values for the meta-parameters. Considering that the entire data as input has been received and we want to find the best meta-parameter among them, in this case, the value of the training and test data is determined by validating, and then the highest accuracy is considered for the calculation of the meta-parameter, which is used to calculate the highest accuracy after training from Through the Euclidean distance [40] and the silhouette criterion [41], the values are evaluated, and in each network that we have specified at the beginning, we have the values of the Euclidean distance and the silhouette criterion. Then, according to the definition of Nash equilibrium, we first determine the type of game according to If the number of players is more than one and each player can perform several possible actions, the game is static with complete information, and then the game is written according to the strategic form:

- Set of players: samples (number of players: 2).
- Strategy: parameters based on Euclidean distance and scores based on the silhouette criterion.
- The outcome of the players (utility): the best parameter and the best score.
- Increasing utility: the number of states that are checked in a two-dimensional array.

Nash equilibrium is used to increase speed and accuracy and prevent local and global optimal. In this case, in the interval where the value of points and the value of parameters exist, it can predict which value should be chosen so that we can select the best one with appropriate accuracy. score and parameter so that the method can find clusters with a smaller number of points (less density).

Third step: Now we find the cells that have the maximum number of points. Then we run the *DBSCAN* algorithm based on the parameters of the grid search on those cells. In the tests performed on this solution, five steps are considered. As mentioned, the algorithm uses the cells that have the maximum number of points (the highest density) to select the appropriate interval in each step. Then the number of points is divided by the number of steps to estimate the interval for each step. For example, suppose the maximum number of points in the grid structure is *100*. The amount of quota reduction is obtained by dividing 100 by 5 (20). Therefore, in the first stage, cells with a minimum of 80 and a maximum of 100 points are selected. That is, the range [80, 100] in the next stage will be the optimal range in the form of [60, 80], and it will continue in the same way.

Fourth step: As mentioned, clustering is done using the *DBSCAN* algorithm according to the grid search in each of the grids. In this step, the coordinates of the grids are moved. This is done by moving each cell up and to the right by half the length of that cell; in other words, the cell is moved diagonally by half the length of a cell. Then we run the algorithm again on each cell, or *DBSCAN* cell. The shape of the clusters can be seen in Figure 2.

Fifth step*:* In this step, the results of the previous two steps are merged. Since the previous step is performed at the level of a single cell in the network, it detects small clusters. At this step, larger clusters are formed by merging small clusters. The merging method is based on sharing points between two clusters. In other words, both clusters that have more than a certain number of common points are merged and form a larger cluster. Suppose we have two clusters of c_1 and c_2, where c_1 is the result of clustering in the first stage and c_2 is the result of clustering in the second step. If the number of points that have the same label in c_2 but have different labels in $c_1$are more than a certain limit (such as *Minpts*), they can be merged. Figure 3 shows the steps applied to the proposed method:

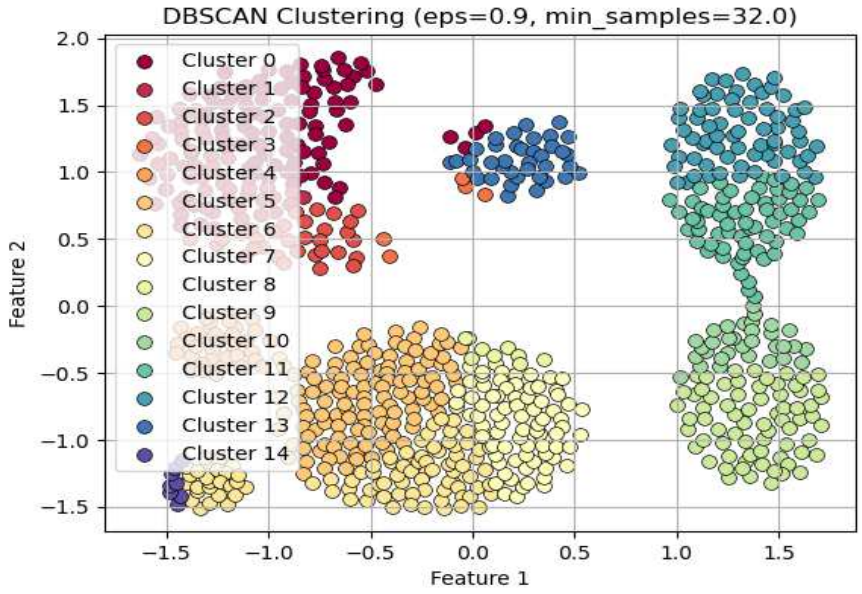

Figure 3- Proposed method on a sample of data

3.1. Formal definition of the proposed method

We divide the proposed method into three parts. The main code grid of the proposed method can be seen in Figures 4 to 6:

Algorithm 1: Improved Grid Search

1: Input: Data set
2: Output: best parameters key: value
3: import Data set
4: For all Datasets **do**
Set the dimensions of the grid structured n*n
Define *k* fold cross validation
Split Data set into Train and Test
D1←Train Data
D2←Test Data
5: End For
6: For j ∈ values of each value Є any grid **do**
7: For k fold cross validation **do**
Get max accuracy value with:
D11← $\sqrt{\sum_{j=1}^{n}(P_j - q_j)^2}$
D2← $\frac{b(j)-a(j)}{\max\{a(j),b(j)\}}$
D1←model.fit(D1)
y_pried ←model.predict(D2)
n_clusters ← len(y_pried)
8: If n_clusters > 1
score←silhouette_score(D2, y_pried)
else
score ← -1
9: End If
10: End For
11: End For
12: //Step to not search all points

13: Definition Scores [ ]←score, step←0 , i←0 , best_prams←None, best_score←float('-info')
14: While True:
i←i+step
15: **If** ((i)>len (Data in the grid)) **then** break
scored←Data in grid[i]
16: **End If**
17: **If** scored > best_score then best_score ← scored
best_prams ← Data in grid[i]
18: **End If**
19: **If**(step>0) then step←step-1 , i←i+1
else:
i← i+ 1, step ←step+1
20: **End If**
21: return best_prams, best_score
22: End While
23: hyper parameters Dictionary← (best_prams, best_score)
24: For all hyper parameters, the Dictionary calls Nash Equilibrium (NE)
25: **For** all best parameters key
26: **while True:** // for predicting the best parameters and More speed to find the best score and the best parameter
Definition Nash [][]
Insert hyper parameters Dictionary Nash
If $u_i(x_i^*,\ x_{-i}^*) \geq u_i(x_i^*,\ x_{-i}^*)$ **then**$x^* \epsilon\ X$ **is** call Nash Equilibrium
Calculate $u_i(x_i^*,\ x_{-i}^*) = max_{x_i \epsilon X_i}^{u_i(x_i^*,\ x_{-i}^*)}$ for all i
27: **End If**
28: **End While**
29: **End For**
30: End For

Figure 4- Proposed pseudo-code - First part

Algorithm 2: IGS-DBSCAN Algorithm

1: Input: Dateset D, Minpts, Eps
2: Output: a Set of Clusters
3: set the dimensions of the grid-structure $\sqrt{n} \times \sqrt{n}$
4: For each unvisited point p in the data set D **do**
Put point p in the corresponding cell in the grid
5: End For
6: c_1 ← GRID CLUSTERING (grid)
7: shift the cells of the grid up and right by shape_size(cell) / 2
8: c_2← GRID CLUSTERING (grid)
9: create a matrix M with size n ×n and fill it with 0
10: For each point p in D do
11: **If** P label in c_i != Plabel in c_2 then
increment the corresponding element in M by 1
12: End if
13: End for
14: For each element in M **do**
15: If element ≥ Minpts then
for the corresponding row and column Merge Clusters in c_i
16: End if
17: End for
18: return c_1

Figure 5- Proposed pseudo-code - Second part

Algorithm 3: GRID CLUSTERING Function

1: Function GRID CLUSTERING (grid)
2: result = []// no label for each poin**t**
3: For each cell in the grid with a maximum point **do**
Call Algorithm 1
update result with DBSCAN (cell,Eps, Minpts)
4: End For
5: max_point = size of the cell that has the maximum number of points
threshold=$\frac{\text{mar-point}}{5}$
6: i=1
7: while i ≤ 5 **do**
Eps= Eps + 0.1×Eps
For each unvisited cell with number of points in range (max_point - i× threshold, max_point + (i -1) × threshold) **do**
update result with DBSCAN (cell, Eps, Minpts)
8: End For
9: i+=1
10: End while
11: return result
12: End Function

Figure 6- Proposed pseudo-code - Third part

4- Evaluation

This section contains a variety of studies on various data sets, using the DBSCAN algorithm as the basis for all clustering methods. As had been covered in the preceding sections, the *Eps* and *Minpts* parameters are important and have a big impact on how well this kind of clustering is done. As such, a comparison between the techniques and the new approach that was introduced in the previous section has been conducted. In addition, studies have been done on how these methods' accuracy is assessed. It should be mentioned that there are many different sizes and kinds of data. However, a fresh method of clustering data with different dimensions is investigated.

4.1. Data set

A few datasets—some labeled and others unlabeled—have been utilized to assess the suggested strategy. Data sets are described in terms of the number of samples and size. Most datasets are unlabeled since labeling can be an intensive and costly process; nevertheless, medium-sized datasets are measured using datasets with labels. Table 1 shows the characteristics of the data set [41] in the proposed method.

Table 1- Characteristics of the data set

Datasets name		Number of data	Number of Dimensions
Unlabeled datasets	T4.8k	8000	2
	Brich2	11000	2
	Flame	240	2
Labeled datasets	R15	600	3
	D31	3100	2
	Unbalance	6500	2
	JSI	600	3
	Yeast	1484	8
	Breast	699	9
	Iris	150	4
	Wine	178	13
	Glass	240	9
	Diabetes	768	8
	Bupa	345	5

4.2. Datasets evaluation

The proposed method is evaluated using labeled datasets in regard to *purity*, which counts the number of instances in a cluster with the same label. The *Fisher*, *Davis-Bouldin*, and *Silhouette* criteria are used to evaluate the clustering performance for unlabeled datasets [42-44]. An explanation of these requirements is offered here.

A percentage of every sample that is correctly recognized within a given range is known as the *purity* criteria [0, 1]. A value that is close to a single implies a greater level of data clustering accuracy and may be computed as follows:

$$\text{Purity} = \sum_{r=1}^{k} \frac{n_r}{n} P(S_r) \qquad (1)$$

where $P(S_r)$ checks the accuracy of cluster *r* and the number of samples in cluster *r* and n shows the total number of samples. In this sense, we consider the maximum distribution of samples for a cluster.

Fisher's criterion for clustering methods shows the number of groups that are both distant from one another and centered around their mean. To calculate the Fisher value, divide the trace of the intra-class dispersion matrix trace (S_b) by the trace of the between-class dispersion matrix trace (S_W). In this way, every cluster is regarded as a class. Below are the definitions of between-class and within-class scatter matrices:

$$S_b = \frac{1}{N} \sum_{i=1}^{c} L_i S_{bi} \qquad (2)$$

$$S_{bi} = (x - \mu_i)(x - \mu_i)^T \qquad (3)$$

where L_i and μ_i are the number of samples and the mean of the i^{th} cluster, respectively. In addition, S_w is the summation of all within-class scatter matrices and S_{wi} is the within-class scatter matrix of the i^{th} cluster.

$$S_w = \frac{1}{N} \sum_{i=1}^{c} S_{wi} \qquad (4)$$

$$S_{wi} = \sum_{j \in l_i} (x_j - \mu_i)(x_j - \mu_i)^T \qquad (5)$$

$$F\_value = \frac{trace\ (S_B)}{trace(S_W)} \qquad (6)$$

Davies-Bouldering (DB) index measures the status of two-by-two clusters and for each cluster, the worst value is selected. The final value of this index is an average over the worst values of all clusters. The similarity measure R_{ij} for the i^{th}, j^{th} clusters is determined as follows:

$$R_{ij} = \frac{s_i + s_j}{D_{ij}} \quad (7)$$

where S_i and S_j are the variances of the i^{th}, and j^{th} clusters, respectively and D_{ij} is the distance between their means. The worst case for the i^{th} cluster maximizes the R_{ij} over the clusters. The *DB* index is determined as follows:

$$DB_i = \overset{Max}{j} R_{ij} \quad , \quad DB = \frac{1}{C}\sum_{i=1}^{C} DB_i \quad (8)$$

where C is the number of clusters. *Silhouette* value is determined for each sample and measures its belonging to its cluster compared to other clusters. This index is defined below:

$$S(i) = \frac{b(i) - a(i)}{\max [a(i), b(i)]} \quad (9)$$

where $a(i)$ is the average distance of simple i with the other samples in the same cluster and $b(i)$ is the minimum distance of sample i with all samples in other clusters. *S(i)* can be in the interval of [-1, 1], where a negative value implies that this sample does not belong to its cluster and vice versa. A positive value in the summation of silhouette values of samples within each cluster shows its validity while if this summation is negative, it shows that this cluster should be removed and its samples should be assigned to its neighbor clusters.

4.3. Evaluation results

The results of applying the proposed method and the compared methods to the considered data sets of the purity opinion (for the labeled datasets) and the other three criteria (for the unlabeled datasets) are shown in Figures 7 and 10. In this section, we present the findings of our method in comparison with other methods, including *DBSCAN*, *STDBSCAN*, and *PDBSCAN*, on the data sets described according to the mentioned criteria. Keep in mind that cross-validation is used to determine the optimal parameter for each set of data.

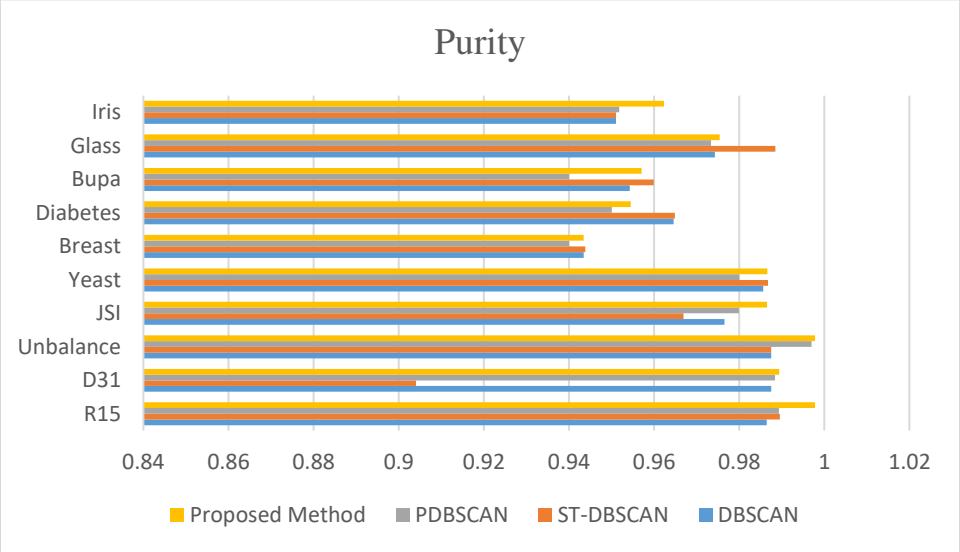

Figure 7- Purity criteria of the proposed method compared to other methods

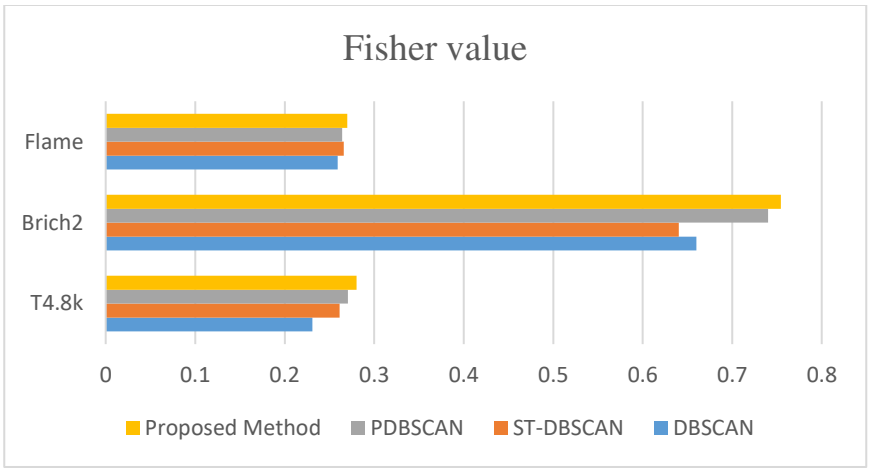

Figure 8- Clustering results of the proposed method and other methods with Fisher's criterion

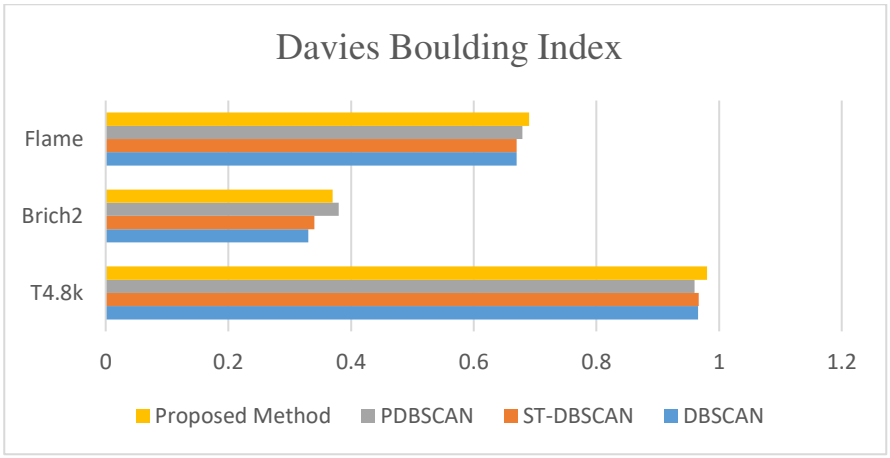

Figure 9- Clustering results of the proposed method and other methods with the David Boldin criterion

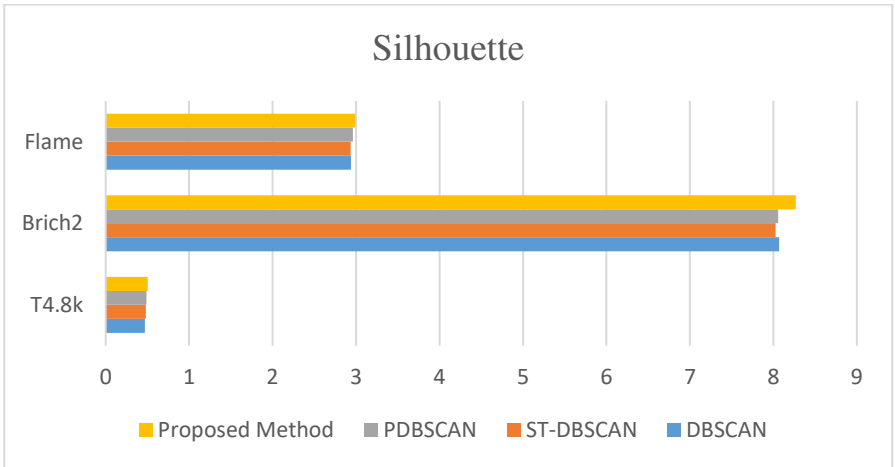

Figure 10- Clustering results of the proposed method and other methods with the Silhouette criterion

Table 2 shows the silhouette criterion in this section, together with the results of the grid search and the two most important DBSCAN method parameters using the proposed method:

Table 2- Parameters of the proposed method on different data

Datasets name	Best parameter		best score	Silhouette
	EPS	Minpts		
T4.8k	0.9	34	0.001	0.34
Brich2	0.5	34	0.020	0.235
Flame	0.8	31	0.324	0.054
R15	0.9	15	0.291	0.421
D31	0.9	33	0.211	0.541
Unbalance	0.9	15	0.001	0.654
JSI	0.7	34	0.241	0.032
Yeast	0.6	32	0.310	0.041
Breast	0.9	34	0.117	0.011
Iris	0.2	15	0.102	0.081
wine	0.1	15	0.050	0.091
Glass	0.2	15	0.002	0.032
Diabetes	0.3	27	0.001	0.732
Bupa	0.1	36	0.010	0.401

4.4. Time complexity analysis

One of the disadvantages of the *DBSCAN* clustering method is that it is of the order of time O(n^2). In the proposed method, the proposed method uses a network structure whose dimensions are $\sqrt{n} \times \sqrt{n}$. We assume that the data is uniformly distributed in the cells. In this case, the number of points in each cell will be almost equal to $\sqrt{n}$. If we use the *DBSCAN* algorithm for each cell, the time complexity of each cell will be *O(n)*. Therefore, the time complexity of the proposed method will be approximately $O(n \times n^2)$. If the improved *DBSCAN* methods are also used, the time complexity of the proposed method will be reduced by the same proportion.

5. Conclusion

The *DBSCAN* algorithm is considered one of the best density-based clustering algorithms that can detect clusters with irregular shapes. This algorithm is especially useful for spatial and temporal data clustering. Although the time complexity of this algorithm is of the second order, its execution time in big data is high. On the other hand, it is not very suitable for identifying clusters that have different densities and is not able to identify them correctly. In the method presented in this article, the time complexity is reduced by classifying the data, dividing them in a grid, implementing the clustering algorithm in each of the grid cells according to Nash equilibrium, and then integrating the results. The main idea is that points can be located in a cluster that is also in the neighborhood. Therefore, it is not necessary to compare all the points, and it is enough to compare the points that are placed in each cell. The output of the proposed algorithm shows that it can perform better than the basic algorithm. On the other hand, its time complexity is less than the square of the number of points. It is the basic algorithm.

References

[1] Li, J., Ma, R., Deng, M., Cao, X., Wang, X., & Wang, X. (2024). A comparative study of clustering algorithms for intermittent heating demand considering time series. Applied Energy, 353, 122046.

[2] Baradaran, A. A., & Rabieefar, F. (2023). NEECH: New Energy-Efficient Algorithm Based on the Best Cluster Head in Wireless Sensor Networks. Iranian Journal of Science and Technology, Transactions of Electrical Engineering, 1-16.
[3] Duan, J., Yang, X., Gao, S., & Yu, H. (2024). A partition-based problem transformation algorithm for classifying imbalanced multi-label data. *Engineering Applications of Artificial Intelligence*, *128*, 107506.
[4] Jia, Y., Lu, K., Li, X., & Hao, C. (2022). SRG: a clustering algorithm based on scale division and region growing. Cluster Computing, 1-21.
[5] Oyewole, G. J., & Thopil, G. A. (2023). Data clustering: Application and trends. *Artificial Intelligence Review*, *56*(7), 6439-6475.
[6] Fahim, A. (2023). A varied density-based clustering algorithm. *Journal of Computational Science*, *66*, 101925.
[7] Tian, Q., Cheng, Y., He, S., & Sun, J. (2024). Unsupervised multi-source domain adaptation for person re-identification via feature fusion and pseudo-label refinement. *Computers and Electrical Engineering*, *113*, 109029.
[8] Kazemi, U., & Boostani, R. (2021). FEM-DBSCAN: AN efficient density-based clustering approach. Iranian Journal of Science and Technology, Transactions of Electrical Engineering, 1-14.
[9] Fu, N., Ni, W., Hu, H., & Zhang, S. (2023). Multidimensional grid-based clustering with local differential privacy. *Information Sciences*, *623*, 402-420.
[10] Huang, X., Ma, T., Liu, C., & Liu, S. (2023). GriT-DBSCAN: A spatial clustering algorithm for very large databases. *Pattern Recognition*, *142*, 109658.
[11] Sadigov, R., Yıldırım, E., Kocaçınar, B., Patlar Akbulut, F., & Catal, C. (2023). Deep learning-based user experience evaluation in distance learning. Cluster Computing, 1-13.
[12] Ahmad, S., Mehfuz, S., Urooj, S., & Alsubaie, N. (2024). Machine learning-based intelligent security framework for secure cloud key management. Cluster Computing, 1-27.
[13] Huang, A. C., Meng, S. H., & Huang, T. J. (2023). A survey on machine and deep learning in semiconductor industry: methods, opportunities, and challenges. Cluster Computing, 26(6), 3437-3472.
[14] Manchanda, A. (2024). Computational Intelligence for Big Data Analysis. In *Computational Science and Its Applications* (pp. 199-230). Apple Academic Press.
[15] Gao, X. (2024). A clustering (DBSCAN+ GMM) investigation of the young open cluster NGC 6649. *Monthly Notices of the Royal Astronomical Society*, *527*(2), 1784-1793.
[16] Cheng, D., Xu, R., Zhang, B., & Jin, R. (2023). Fast density estimation for density-based clustering methods. *Neurocomputing*, *532*, 170-182.
[17] Gunawan, A., & de Berg, M. (2013). A faster algorithm for DBSCAN. *Master's thesis*.
[18] Gan, J., & Tao, Y. (2015, May). DBSCAN revisited: Mis-claim, un-fixability, and approximation. In *Proceedings of the 2015 ACM SIGMOD international conference on management of data* (pp. 519-530).
[19] Kryszkiewicz, M., & Lasek, P. (2010, June). TI-DBSCAN: Clustering with DBSCAN by Means of the Triangle Inequality. In *International Conference on Rough Sets and Current Trends in Computing* (pp. 60-69). Berlin, Heidelberg: Springer Berlin Heidelberg.
[20] Ohadi, N., Kamandi, A., Shabankhah, M., Fatemi, S. M., Hosseini, S. M., & Mahmoudi, A. (2020, April). Sw-dbscan: A grid-based dbscan algorithm for large datasets. In *2020 6th International Conference on Web Research (ICWR)* (pp. 139-145). IEEE.

[21] Birant D, Kut A. ST-DBSCAN: An algorithm for clustering spatial–temporal data. Data & knowledge engineering. 2007 Jan 1;60(1):208-21.
[22] Kisilevich, S., Mansmann, F., & Keim, D. (2010, June). P-DBSCAN: A density based clustering algorithm for exploration and analysis of attractive areas using collections of geo-tagged photos. In *Proceedings of the 1st international conference and exhibition on computing for geospatial research & application* (pp. 1-4).
[23] Ma, L., Gu, L., Li, B., Qiao, S., & Wang, J. (2015). Mrg-dbscan: An improved dbscan clustering method based on map reduce and grid. *International Journal of Database Theory and Application*, *8*(2), 119-128.
[24]eiss, U., & Agassi, J. (2023). *Games to Play and Games Not to Play: Strategic Decisions via Extensions of Game Theory* (Vol. 469). Springer Nature.
[25] Ye, M., Han, Q. L., Ding, L., & Xu, S. (2023). Distributed Nash equilibrium seeking in games with partial decision information: a survey. *Proceedings of the IEEE*, *111*(2), 140-157.
[26] Hanafi, N., & Saadatfar, H. (2022). A fast DBSCAN algorithm for big data based on efficient density calculation. *Expert Systems with Applications*, *203*, 117501.
[27] Zhang, Y., Qu, Y., Gao, L., Luan, T. H., Jolfaei, A., & Zheng, J. X. (2023). Privacy-preserving data analytics for smart decision-making energy systems in sustainable smart community. *Sustainable Energy Technologies and Assessments*, *57*, 103144.
[28] Wu, X., Wu, T., Khan, M., Ni, Q., & Dou, W. (2017). Game theory based correlated privacy preserving analysis in big data. *IEEE Transactions on Big Data*, *7*(4), 643-656.
[29] Daskalakis, C., Fabrikant, A., & Papadimitriou, C. H. (2006). The game world is flat: The complexity of Nash equilibria in succinct games. In *Automata, Languages and Programming: 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part I 33* (pp. 513-524). Springer Berlin Heidelberg.
[30] Chen, S., Liu, G., Zhou, Z., Zhang, K., & Wang, J. (2023). Robust multi-agent reinforcement learning method based on adversarial domain randomization for real-world dual-uav cooperation. *IEEE Transactions on Intelligent Vehicles*.
[31] Sadhukhan, P., Halder, L., & Palit, S. (2024). Approximate DBSCAN on obfuscated data. *Journal of Information Security and Applications*, *80*, 103664.
[32] Schubert, E., Sander, J., Ester, M., Kriegel, H. P., & Xu, X. (2017). DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. *ACM Transactions on Database Systems (TODS)*, *42*(3), 1-21.
[33] Scitovski, R., & Sabo, K. (2020). DBSCAN-like clustering method for various data densities. *Pattern Analysis and Applications*, *23*(2), 541-554.
[34] Schikuta, E. (1996, August). Grid-clustering: An efficient hierarchical clustering method for very large data sets. In *Proceedings of 13th international conference on pattern recognition* (Vol. 2, pp. 101-105). IEEE.
[35] Holt, C. A., & Roth, A. E. (2004). The Nash equilibrium: A perspective. *Proceedings of the National Academy of Sciences*, *101*(12), 3999-4002..
[36] Traulsen, A., & Glynatsi, N. E. (2023). The future of theoretical evolutionary game theory. *Philosophical Transactions of the Royal Society B*, *378*(1876), 20210508.
[37] Ye, M., Han, Q. L., Ding, L., & Xu, S. (2023). Distributed Nash equilibrium seeking in games with partial decision information: a survey. *Proceedings of the IEEE*, *111*(2), 140-157.
[38] Ienco, D., & Bordogna, G. (2018). Fuzzy extensions of the DBScan clustering algorithm. *Soft Computing*, *22*(5), 1719-1730.

[39] Pedroche, D. S., Herrero, J. G., & López, J. M. M. (2024). Context learning from a ship trajectory cluster for anomaly detection. *Neurocomputing*, *563*, 126920.
[40] Crook, O. M., Cucuringu, M., Hurst, T., Schönlieb, C. B., Thorpe, M., & Zygalakis, K. C. (2024). A linear transportation lp distance for pattern recognition. *Pattern Recognition*, *147*, 110080.
[41] http://cs.uef.fi/sipu/datasets/
[42]Yang, J., Yang, J. Y., & Zhang, D. (2002). What's wrong with Fisher criterion?. *Pattern recognition*, *35*(11), 2665-2668.
[43] Ganj, A., Ebadpour, M., Darvish, M., & Bahador, H. (2023). LR-Net: A Block-based Convolutional Neural Network for Low-Resolution Image Classification. Iranian Journal of Science and Technology, Transactions of Electrical Engineering, 47(4), 1561-1568.
[43] Goudarzi, S., Jafari, M. J., & Afsar, A. (2017). A hybrid model for portfolio optimization based on stock clustering and different investment strategies. *International Journal of Economics and Financial Issues*, *7*(3), 602-608.
[44] Campello, R. J., & Hruschka, E. R. (2006). A fuzzy extension of the silhouette width criterion for cluster analysis. *Fuzzy Sets and Systems*, *157*(21), 2858-2875.