An Adaptive Clustering Algorithm Based on Local-Density Peaks for Imbalanced Data Without Parameters

Wuning Tong, Yuping Wang, *Senior Member, IEEE*, and Delong Liu

Abstract—Imbalanced data clustering is a challenging problem in machine learning. The main difficulty is caused by the imbalance in both cluster size and data density distribution. To address this problem, we propose a novel clustering algorithm called LDPI based on local-density peaks in this study. First, an initial sub-cluster construction scheme is designed based on a 3-dimensional (3-D) decision graph that can easily detect the initial sub-cluster centers and identify the noise points. Second, a sub-cluster updating strategy is designed, which can automatically identify the false sub-cluster centers and update the initial sub-clusters. Third, a sub-cluster merging scheme is designed, which merges the updated initial sub-clusters into final clusters. Consequently, the proposed algorithm has three advantages: 1) It does not require any input parameters; 2) It can automatically determine the cluster centers and number of clusters; 3) It is suitable for imbalanced datasets and datasets with arbitrary shapes and distributions. The effectiveness of LDPI is demonstrated experimentally and the superiority of LDPI is identified by comparison with 5 state-of-the-art algorithms.

Index Terms—Data clustering, density peaks, imbalanced data, multiple centers

1 Introduction

Clustering is a process of dividing a set of unlabeled data/objects into multiple clusters of similar data/objects. It assigns data/objects with similar attributes to the same cluster. It is routinely used in machine learning applications, e.g., disease classification [1], [2], online community discovery [3], financial fraud detection [4], image segmentation[5], [6], etc. Many classic clustering methods have been proposed in the past (e.g., in [7], [8], [9], [10], [11]). Most of them were suitable for balanced datasets, i.e., datasets with balanced cluster sizes and uniform density distributions. However, many real-world datasets are imbalanced (i.e., they have imbalanced cluster sizes or/and non-uniform density distributions), for example, datasets of people with/without a certain disease[12], datasets of financial fraud transactions and legal transactions [13], [14], etc. Effectively handling the imbalanced data clustering problems remains a prominent issue in machine learning.

• *Wuning Tong is with the School of Computer Science and Technology, Xidian University, Xian, Shaanxi 710071, China, and also with the Department of Science and Technology, Shaanxi University of Chinese Medicine, Xianyang, Shaanxi 712046, China. E-mail: tongwuning@sntcm.edu.cn.*
• *Yuping Wang and Delong Liu are with the School of Computer Science and Technology, Xidian University, Xi'an, Shaanxi 710071, China. E-mail: ywang@xidian.edu.cn, dlliu_1@stu.xidian.edu.cn.*

Manuscript received 10 Feb. 2021; revised 6 Dec. 2021; accepted 23 Dec. 2021. Date of publication 28 Dec. 2021; date of current version 7 Mar. 2023.
This work was supported in part by the National Natural Science Foundation of China under Grant 61872281, and in part by the National Key Research and Development Program of China under Grant 2017YFC1703506.
(Corresponding author: Yuping Wang.)
Recommended for acceptance by V. Moreira.
Digital Object Identifier no. 10.1109/TKDE.2021.3138962

Current researches involving imbalanced datasets have been focused mostly on classification and not on clustering. For example, [15], [16] dealt with the imbalanced classification problem through under-sampling the major class, while [17], [18] tackled the imbalanced classification problem via over-sampling the minor class. Research on the clustering of imbalanced datasets has become prominent in recent years.

For datasets with non-uniform density distributions, Rodriguez *et al.* [19] proposed a density peaks clustering (DPC) algorithm. When the distance threshold d_c is assigned appropriately, the algorithm will work well on convex datasets with slightly non-uniform density distributions. However, d_c is determined based on experience, which makes it difficult to determine an appropriate value for d_c. To determine d_c more accurately, Mehmood *et al.* proposed a clustering algorithm (called CFSFDP-HD)[20], which uses the optimal bandwidth of the kernel density in the heat diffusion equation as the distance threshold d_c. Wang *et al.* [21] proposed a DPC-based algorithm that uses the optimal value of the potential entropy of the dataset as the distance threshold d_c. These algorithms avoid assigning d_c manually, but they do not work well on non-convex datasets with non-uniform density distributions. Duan *et al.* [22] proposed a density-based clustering algorithm with different radius thresholds for different datasets. This algorithm is effective when the density distribution between clusters is non-uniform. However, it fails when the density within a cluster varies drastically. Liu *et al.* proposed a shared-nearest-neighbor-based clustering algorithm [23] that constructs clusters by reasonably allocating the neighbors to the cluster centers. It is effective on clusters with non-uniform density distributions; however, it requires the cluster centers to be manually assigned. Some researchers (e.g.[24], [25]) proposed clustering algorithms that automatically assign cluster centers. Bie *et al.* [25] proposed a clustering method named Fuzzy-CFSFDP that does not assign cluster centers manually. The algorithm assigns the points whose upward distances (the minimum distance between a point and other denser points) are twice the standard deviation of all points' upward distances and whose densities are greater than the average density of all points as local cluster centers. Then, it assigns the remaining points to their nearest local cluster centers and local clusters are formed. After that, it merges certain local clusters to obtain the final clusters. However, this algorithm is not applicable to datasets with drastically imbalanced density distributions because it omits sparse clusters. Bryant *et al.* [26] proposed another density-based clustering algorithm that uses the reverse nearest neighbors to estimate the density of points. It needs only one parameter K and can handle issues caused by drastically varying cluster densities. However, it is not effective on datasets with imbalanced cluster sizes.

Regarding imbalanced cluster sizes, Wen *et al.* [27] proposed an algorithm called one-step spectral rotation clustering for imbalanced high-dimensional data. This algorithm clusters points through recursively learning the category information of the known points. Consequently, the algorithm requires a certain amount of learning data as prior knowledge. Liang *et al.* [28] proposed a multi-center clustering algorithm (called MC) based on the K-means algorithm for imbalanced datasets. The algorithm first divides objects into several small sub-groups that are similar in size to the minor clusters to reduce the "uniform effect". (The uniform effect means that the points of major clusters are mis-clustered into minor clusters, and the resulting clusters have relatively uniform sizes.) Then, it merges some nearby sub-groups into a cluster. The algorithm demonstrates good performance when processing imbalanced datasets, provided that the number of sub-groups to be merged is provided in advance. Inappropriately setting the number of sub-groups would produce incorrect clustering results. Wang *et al.* [29] proposed an imbalanced data clustering algorithm based on the multiple center points found in a dataset. The algorithm determines whether to merge sub-clusters into a cluster by measuring the degree of overlapping between each pair of sub-clusters. This algorithm also requires the number of sub-clusters to be assigned in advance. Cheung *et al.* [30] proposed a clustering method (called RPCCL) that does not need the number of clusters specified in advance; it can obtain cluster centers automatically. Moreover, Lu *et al.* [31] developed a clustering method (called SMCL) based on the adaptive multi-prototype using K-means for imbalanced data clustering. This method, an improvement of the RPCCL method, can automatically select the seed points of an imbalanced dataset. Each seed point represents one sub-cluster, and the sub-clusters are merged by a specific scheme to obtain the final clusters. The pitfall of this method is that it cannot effectively distinguish small clusters from noise points.

To overcome the deficiencies of the aforementioned algorithms when working with imbalanced datasets, namely, (a) manually setting parameter values, (b) manually selecting cluster centers, (c) being ineffective for datasets with non-uniform density distributions in some clusters, (d) ineffectively distinguishing small clusters from noise points, this study proposes an adaptive clustering algorithm based on local-density peaks (briefly denoted as LDPI).

First, we design an initial sub-cluster construction scheme which selects the noise points and initial sub-cluster centers based on the density, upward distance, and number of reverse nearest neighbors. For each sub-cluster center, we construct an initial sub-cluster by assigning it and its nearest remaining points to this initial sub-cluster. Then, we design a sub-cluster updating strategy that can automatically distinguish the false sub-cluster centers from the initial sub-cluster centers to produce updated sub-clusters. Finally, we design a sub-cluster merging scheme that can choose proper updated sub-clusters and merge them. If there exists any noise point, it is assigned into the nearest cluster.

The contributions of this study are summarized as follows: 1) A novel parameter-free clustering algorithm LDPI is proposed for imbalanced data clustering. 2) A method for adaptively and automatically determining the distance threshold d_c is proposed, which is crucial for density computation. 3) An improved density calculation method is proposed, which can better deal with imbalanced data. 4) New methods for determining noise and boundary points are proposed. 5) A new 3-dimensional decision graph is proposed. It can identify the cluster centers and noise points correctly in imbalanced datasets, while the existing 2-dimensional decision graph cannot.

The remainder of this article is organized as follows. Section 2 presents an overview of related work. The proposed algorithm LDPI is described in detail in Section 3. Section 4 discusses the experimental results, and conclusions are drawn in Section 5.

2 Overview of Related Work

This section briefly introduces the density peaks clustering algorithm and related concepts of the k-nearest neighbor algorithm.

2.1 The Density Peaks Clustering Algorithm

The density peaks clustering (DPC) algorithm [19] is a simple and efficient clustering algorithm. Its main idea is choosing the density peak points in the dataset as cluster centers. A density peak point is the point satisfying the following conditions: (1) The density of this point is greater than that of its neighbor points; (2) The upward distance of this point (the minimum distance between it and other denser points) is relatively large. A point with lower density and larger upward distance is classified as a noise point.

The main process of the algorithm is stated as follows: Given a dataset $s = \{x_i\}_{i=1}^n$ with n points, where each point $x_i = [x_{i1}, x_{i2}, \cdots, x_{im}]$ has m features, let d_{ij} denote the euclidean distance between points x_i and x_j. The density of point x_i is defined as

$$\rho_i = \sum_{j \neq i} X(d_{ij} - d_c), \tag{1}$$

where $X(x) = 1$ if $x < 0$, otherwise $X(x) = 0$. d_c is the distance threshold assigned to ensure that the average number of neighbors is approximately 2% the number of the total data points.

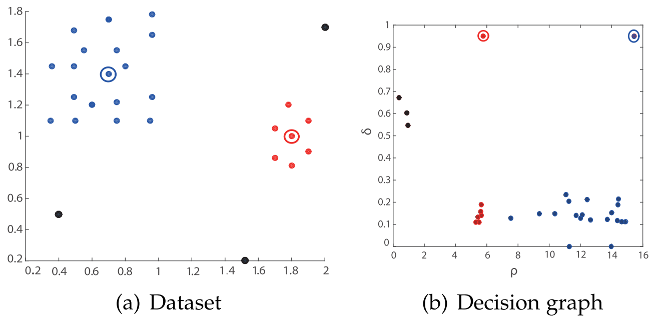


Fig. 1. Dataset and its decision graph.

The upward distance of point x_i is defined as

$$\delta_i = \begin{cases} \max_j\{d_{ij}\}, & if\ \rho_i > \rho_j\ for\ \forall j \\ \min_j\{d_{ij}|\rho_j > \rho_i\}, & otherwise. \end{cases} \quad (2)$$

Fig. 1 illustrates the basic process of the DPC algorithm, where Fig. 1a represents the dataset, and Fig. 1b displays a 2-dimensional decision graph (refer to [19] for details), with the density ρ and upward distance δ of each data as the two coordinates of a point. Therefore, each data corresponds to one point in the decision graph. The points with a relatively high density and big upward distance (indicated with blue and red circles, respectively) in Fig. 1b are assigned as cluster centers (indicated with blue and red circles, respectively) in Fig. 1a. The points with big upward distances and low densities (represented in black) in Fig. 1b correspond to the noise points (represented in black) in Fig. 1a. Finally, the remaining data in Fig. 1a is classified into the corresponding clusters according to the nearest neighbor principle.

The DPC algorithm performs well for datasets with balanced cluster sizes. However, for datasets with imbalanced cluster sizes, its performance is not satisfactory.

Fig. 2 demonstrates the clustering process of DPC on a simple 2-dimensional imbalanced dataset containing three clusters indicated by blue, green, and red points in Fig. 2a. The DPC algorithm [19] cannot obtain the correct clusters. Note that DPC selects the points with a higher density ρ and a larger upward distance δ as cluster centers based on the decision graph presented in Fig. 2b. However, it is difficult to know how large a density and upward distance should be selected, which depends on the experience of users. Thus, different people may select different cluster centers and get different clustering results.

If two points (inside the red rectangular area) are selected as cluster centers, as depicted in Fig. 2c, the dataset will be divided into two clusters, as depicted in Fig. 2d, in which the smallest cluster is mistakenly merged into the largest one. This means that DPC overlooks the minor clusters if the number of selected centers is smaller than the corrected value. When three points (inside the green rectangular area) are selected as cluster centers, three clusters are obtained, as depicted in Fig. 2e, where the largest cluster on the left is incorrectly divided into two clusters, and the smallest cluster is merged into a larger cluster. In other words, DPC cannot identify the smallest cluster. When four points (inside the blue rectangular area) are selected as cluster centers, DPC correctly identifies the smallest cluster but divides the largest cluster into two clusters, as shown in Fig. 2f.

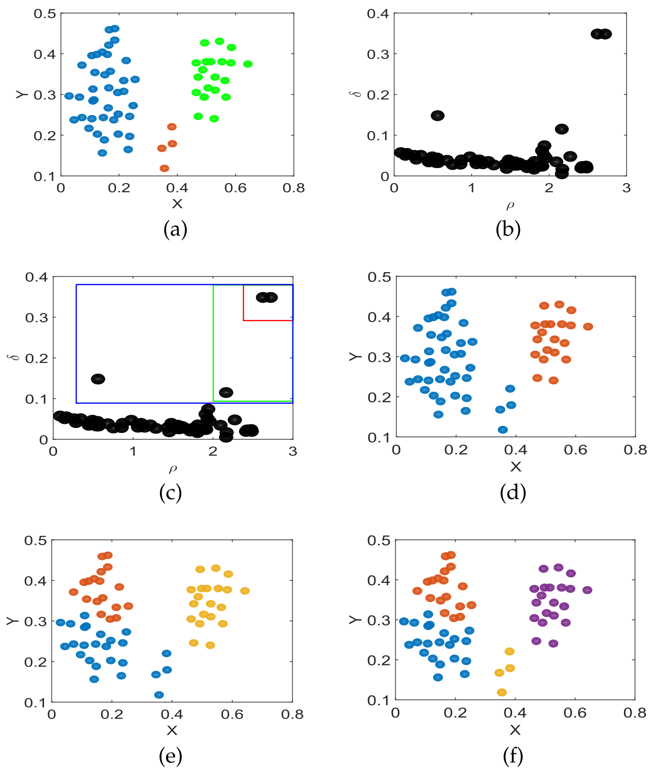


Fig. 2. The clustering process using the DPC algorithm. (a) Dataset. (b) Decision graph. (c) Center selection. (d) Clustering results with two centers. (e) Clustering results with three centers. (f) Clustering results with four centers.

Fig. 3a illustrates the clustering process of DPC on a dataset with three clusters whose density distributions are imbalanced. The density of the cluster constructed with blue points is lower than that of the other two clusters. The density of the cluster center at point 3 is 0 according to Eq. (1). (d_c is assigned to ensure that the average number of the neighbors is approximately 2% the number of the total data points, as suggested by authors of [19].) As a result, the center point 3 is misjudged as a noise point and DPC returns an incorrect result.

2.2 Related Concepts of the K-Nearest Neighbor Algorithm

The k-nearest neighbor (KNN) algorithm is widely used in classification [32], [33] and clustering [34], [35] tasks. However, it is ineffective when applied to imbalanced datasets. To overcome this pitfall, some researchers used the reverse nearest neighbors in clustering with good clustering results

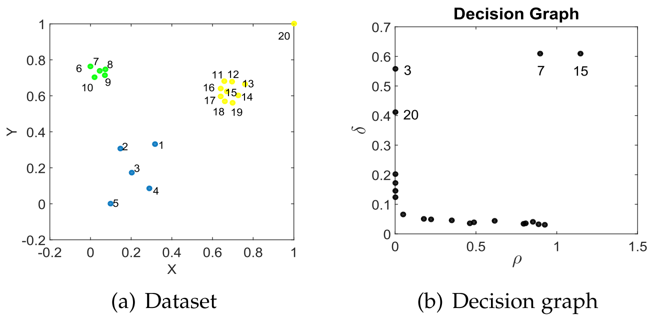


Fig. 3. Noise point identification using the DPC algorithm.

(e.g., [26], [36]). We adopt the concepts of the k nearest neighbors and the reverse nearest neighbors in this study, which can be defined as follows: Given a dataset $X = \{x_i\}_{i=1}^{n}$, d_{ij} denotes the distance between two points x_i and x_j.

Definition 1. *[26] The k nearest neighborhood of a point x_i is defined by a sub-set $N_k(x_i)$ of X satisfying the following conditions:*

1) $N_k(x_i) \subseteq X/\{x_i\}$.
2) $|N_k(x_i)| = k$.
3) $\forall x_j \in N_k(x_i), x_l \in X/\{N_k(x_i) \cup x_i\} : d_{ij} \leq d_{il}$.

Definition 2. *[26], [36] The reverse nearest neighborhood of point x_i is defined by a sub-set $RN_k(x_i)$ of X satisfying the following conditions:*

1) $RN_k(x_i) \subseteq X/\{x_i\}$.
2) *if* $x_j \in N_k(x_i)$ *then* $x_i \in RN_k(x_j)$.

3 Proposed Method

The proposed algorithm LDPI comprises three stages: 1) An initial sub-cluster generation scheme is designed, which can automatically identify the noise points and initial sub-cluster centers. The sub-clusters are generated by classifying the remaining points into the sub-cluster centers according to the nearest neighbor principle. 2) A sub-cluster updating scheme is designed, which can identify the false sub-cluster centers among the initial sub-cluster centers. Subsequently, the scheme removes them to obtain the updated sub-cluster centers and corresponding updated sub-clusters. 3) A sub-cluster merging strategy is applied to merge the updated sub-clusters and generate the final clusters.

Definition 3. *The imbalanced data clustering refers to the clustering of a dataset with imbalanced cluster sizes or/and non-uniform density distributions (imbalanced density distributions).*

3.1 Determining the Proper Distance Threshold d_c and Density

As discussed before, manually assigning d_c in DPC usually results in incorrect clusters, especially with imbalanced datasets. Thus, the method of determining the proper distance threshold d_c is crucial to clustering. In this subsection, we propose a novel method that can determine the proper distance threshold d_c.

For a given dataset $S = \{x_i\}_{i=1}^{n}$ with n points, let D_i be the distance between point i and its nearest neighbor. Then the maximal value of all distances between every point and its nearest neighbor, calculated by Eq. (3)

$$d_c = \max_{i=1}^{n}\{D_i\}, \tag{3}$$

can be seen as the radius of the smallest sparse cluster, which can be used as the distance threshold d_c.

Another important issue is to define the proper density of points. DPC [19] uses Eq. (1) to calculate the density of points, but its accuracy is not satisfactory. Multiple points may have the same density, which hinders distinguishing the density of these points. To tackle this issue, the algorithm (called DPADN) in[37] uses Eq. (4) to calculate the local densities of points.

$$\rho_i = \sum_{j \in S, j \neq i} e^{-(\frac{d_{ij}}{d_c})^2} \tag{4}$$

DPADN achieves good accuracy. However, it requires heavy computational effort because it involves all the available points in the calculation. Moreover, the major clusters have a great impact on the minor clusters. To overcome the drawbacks, it is reasonable to determine the local density of each point by its neighboring points instead of all the available points. From this observation, we design a novel formula that calculates the local density of each point as follows

$$\rho_i = \sum_{j \in S/\{i\}, d_{ij} \leq d_c} e^{-(\frac{d_{ij}}{d_c})^2}. \tag{5}$$

Note that we only use the points in the neighborhood with a radius d_c to compute the local density. Both the computational effort and the influence of major clusters on the density of minor clusters are reduced greatly; thus, the local density computed with this formula is more reasonable.

3.2 Determining the Noise Points by a Novel Decision Graph

Noise points have a strong negative impact on the clustering results. Therefore, they need to be identified. In DPC, points with lower densities and larger upward distances are considered as noise points, but with the imbalanced datasets, some small cluster centers will be mistakenly assigned as noise points despite the fact that they have lower densities and larger upward distances. For example, Fig. 3a shows an imbalanced dataset with three clusters, and Fig. 3b depicts the decision graph of this dataset according to DPC. In Fig. 3b point 3 is classified as a noise point. However, notice from Fig. 3a that it is a cluster center. To properly identify the noise points in an imbalanced dataset, we design a new decision graph (a 3-dimensional decision graph) to identify the noise points/cluster centers in this subsection, that is, we use the number of reverse nearest neighbors of each point as an extra identifier (except for density and upward distance) to assess whether a point is a cluster center or noise point. First, we use the k-d tree to calculate the k nearest neighbors of each point. Then, we calculate the reverse nearest neighbors of each point according to Definition 2 and count the number of reverse nearest neighbors, where k can be automatically determined using the method described in [38]. Take the dataset described in Fig. 3 as an example. The value $k = 3$ is automatically calculated based on the dataset structure (refer to [38] for details). The k nearest neighbors of each point in the dataset are reported in Table 1, and the number and members of the reverse nearest neighbors of each point are reported in Table 2.

Observe from Table 2 that the points with numerous reverse nearest neighbors are mostly center points, while the points with few reverse nearest neighbors are mostly boundary or noise points. For the center points of sparsely distributed clusters, even if there is a larger upward distance and lower density, the number of reverse nearest neighbors is still relatively large. Thus, if we consider the number of reverse nearest neighbors as an extra identifier

TABLE 1
The K Nearest Neighbors of the Dataset

Points	k nearest neighbors			Points	k nearest neighbors		
	1	2	3		1	2	3
1	2	3	4	11	12	16	15
2	3	1	4	12	11	15	13
3	4	2	1	13	12	14	15
4	3	5	1	14	19	15	13
5	3	4	2	15	16	17	18
6	7	10	8	16	15	11	17
7	8	9	10	17	18	15	16
8	7	9	10	18	17	19	15
9	8	7	10	19	18	14	17
10	7	9	6	20	13	12	11

other than the density and upward distance in the original 2-dimensional decision graph, we can design a 3-dimensional decision graph to effectively deal with imbalanced datasets and identify noise points. Based on this observation, we can update the definition of noise points as follows:

Given a dataset $X = \{x_i\}_{i=1}^{n}$, ρ is the density of the point defined in Eq. (5), δ is the upward distance of the point defined in Eq. (2), and RNN is the number of the reverse nearest neighbors of the point.

Definition 4. *The point x_i is a noise point if x_i satisfies the following conditions:*

1) $\delta_i > \mu(\delta) + \sigma(\delta)$.
2) $\rho_i < \mu(\rho) - \sigma(\rho)$.
3) $RNN_i < \mu(RNN) - \sigma(RNN)$.

where $\mu(A)$ and $\sigma(A)$ are the mean and standard deviation of A of all points, respectively, and A is the upward distances, the densities, and the numbers of reverse nearest neighbors, respectively.

TABLE 2
The Reverse Nearest Neighbors of the Dataset

Points	RNN	Reverse nearest neighbors						
		1	2	3	4	5	6	7
1	3	2	3	4	-	-	-	-
2	3	1	3	5	-	-	-	-
3	4	2	4	5	1	-	-	-
4	4	3	5	1	2	-	-	-
5	1	4	-	-	-	-	-	-
6	1	10	-	-	-	-	-	-
7	4	6	8	10	9	-	-	-
8	3	7	9	6	-	-	-	-
9	3	7	8	10	-	-	-	-
10	4	6	7	8	9	-	-	-
11	3	12	16	20	-	-	-	-
12	3	11	13	20	-	-	-	-
13	3	20	12	14	-	-	-	-
14	2	13	19	-	-	-	-	-
15	7	16	12	14	17	11	13	18
16	3	15	11	17	-	-	-	-
17	4	18	15	16	19	-	-	-
18	3	17	19	15	-	-	-	-
19	2	14	18	-	-	-	-	-
20	0	-	-	-	-	-	-	-

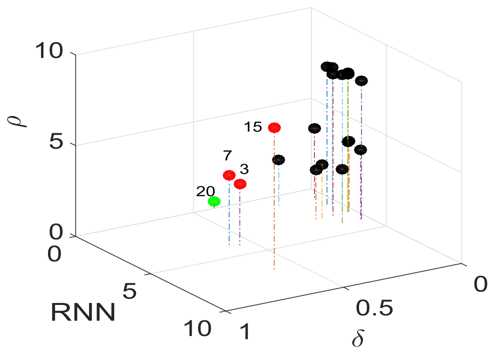


Fig. 4. Determination of the noise points, where the green point indicates the noise point.

Based on these, we design a 3-dimensional decision graph (the conventional decision graph in DPC is 2-dimensional), as shown in Fig. 4. This new decision graph ameliorates the identification process of noise points and cluster centers. A point with a small ρ_i and RNN_i as well as a large δ_i is a noise point according to Definition 4, while a point with a large δ and RNN or a large δ and ρ is very likely to be a cluster center. Obviously, the green point 20 depicted in the figure is a noise point and the red points 3, 7, and 15 depicted in the figure are cluster centers. If one uses the conventional DPC decision graph (2-dimensional), it is impossible to identify the correct cluster centers and noise points.

3.3 Initial Sub-Cluster Generation Scheme

To avoid assigning the number of clusters manually in advance and enable the algorithm to make automatic clustering on imbalanced datasets, we first set a relatively large number of sub-cluster centers and then construct the initial sub-clusters. The detailed procedure is as follows:

After excluding noise points, the remaining points whose upward distances are greater than $\mu(\delta) + \sigma(\delta)$ are chosen as initial sub-cluster centers. It is a slack rule to ensure that the distances between different centers are sufficiently large and centers of minor clusters with lower densities can be selected. Subsequently, each initial sub-cluster center i corresponds to an initial sub-cluster C_i. All remaining points are sorted in a descending order of their densities. Suppose that the sequence is $y_1, y_2, \cdots, y_r$. For the first remaining point y_1, the nearest center point i with a density higher than that of y_1 is identified. Then, y_1 is assigned to the initial sub-cluster C_i. For the second remaining point y_2, the nearest point x with a density higher than that of y_2 is identified. Then, y_2 is assigned to the initial sub-cluster to which x belongs. The third remaining point y_3 can be assigned to an initial sub-cluster using the same method for y_2. In this way, all remaining points can be assigned to their initial sub-clusters.

The pseudo code of the initial sub-cluster construction algorithm is reported in Algorithm 1. In lines 1-2, the algorithm calculates the density and upward distance values. In lines 4-10, the noise points are automatically obtained, and in lines 11-19 the initial sub-cluster centers are obtained. Between lines 20-26 the initial sub-clusters are produced by assigning the remaining points to the corresponding initial sub-clusters.

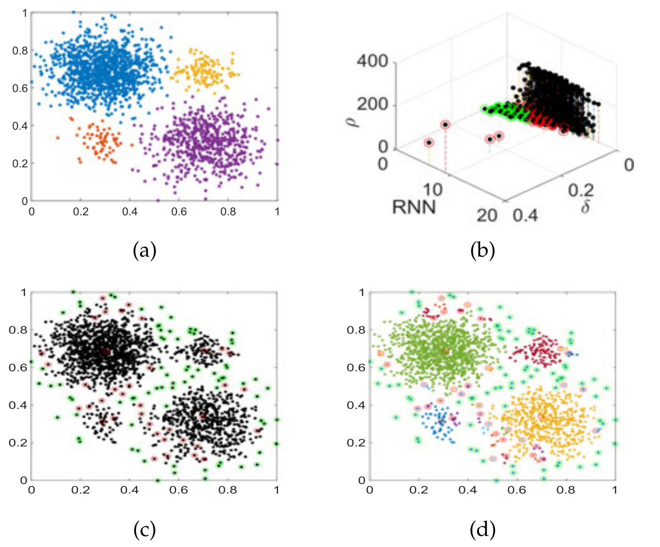


Fig. 5. An example of generating initial sub-cluster centers and the corresponding sub-clusters. (a) Dataset. (b) 3-D decision graph. (c) Initial sub-cluster centers. (d) Initial sub-clusters.

Algorithm 1. Initial Sub-Cluster Construction Algorithm

```
Input: Dataset $S = \{x_1, x_2, \cdots, x_n\}$.
Output: Points label $C$, the number of initial sub-clusters $ICC$.
 1: Calculate densities $\{\rho_i\}_{i=1}^{n}$ by Eq. (5);
 2: Use $\{\rho_i\}_{i=1}^{n}$ to calculate upward distances $\{\delta_i\}_{i=1}^{n}$ by Eq. (2);
 3: $ICC = 0, C = -1, j = 0$;
 4: determine the noise points;
 5: for $i = 1$ to $n$ do
 6:     if $\delta(i) > \mu(\delta) + \sigma(\delta)$ and $RNN(i) < \mu(RNN) - \sigma(RNN)$
        and $\rho(i) < \mu(\rho) - \sigma(\rho)$ then
 7:         $C_i = 0$;
 8:     end if
 9:     $i = i + 1$;
10: end for
11: determine the initial sub-cluster centers;
12: for $i = 1$ to $n$ do
13:     if $\delta(i) > \mu(\delta) + \sigma(\delta)$ and $C_i \neq 0$ then
14:         $j = j + 1$;
15:         $ICC = ICC + 1$;
16:         $C_i = j$;
17:     end if
18:     $i = i + 1$;
19: end for
20: Assign the remaining points to the initial sub-clusters;
21: for $i = 1$ to $n$ do
22:     if $C_i == -1$ then
23:         Assign the label of the nearest higher density points to
            $C_i$;
24:     end if
25:     $i = i + 1$;
26: end for
27: return initial sub-clusters C;
```

Fig. 5 demonstrates an example of determining the initial sub-cluster centers and corresponding sub-clusters, where Fig. 5a shows the Gaussian dataset, Fig. 5b illustrates that the 38 points in red circles are sub-cluster centers and the 58 points in green circles are noise points in the decision graph, Fig. 5c depicts that the 38 points in red circles are sub-cluster centers and the 58 points in green circles are noise points in the dataset, and Fig. 5d demonstrates the 38 initial sub-clusters with their center points in red circles and noise points in green circles.

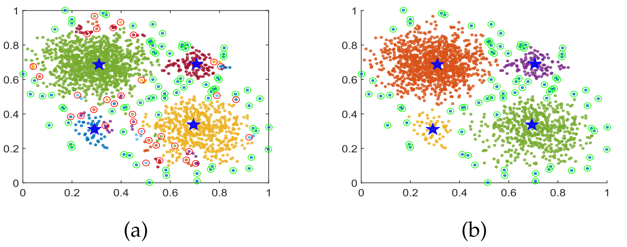


Fig. 6. The initial sub-cluster updating process for the Gaussian dataset. (a) Noise points, false sub-cluster centers, and sub-cluster centers. (b) Noise points and sub-clusters.

Algorithm 2. Sub-Cluster Updating Algorithm

```
Input: Dataset $S$, Points label C, $ICC$.
Output: Points label $C$, the number of sub-clusters $ICC$.
 1: Delete the false sub-cluster centers from the initial sub-clus-
    ter center set;
 2: for $i = 1$ to $ICC$ do
 3:     if $N_i^C < 0.5 * N_i^{d_c}$ then
 4:         $C_i = -1$;
 5:         $ICC = ICC - 1$;
 6:     end if
 7:     $i = i + 1$;
 8: end for
 9: Assign the points in the false sub-clusters to the sub-
    clusters;
10: for $i = 1$ to $n$ do
11:     if $C_i == -1$ then
12:         Assign the label of the nearest higher density points to
            $C_i$;
13:     end if
14:     $i = i + 1$;
15: end for
16: return final sub-clusters C;
```

As explained above, this scheme is a loose procedure and likely to assign some false sub-cluster centers. To solve this problem, a sub-cluster updating scheme is proposed in the following subsection.

3.4 Sub-Cluster Updating Scheme

First, we identify the false sub-clusters as follows: Suppose that the *i-th* initial sub-cluster C_i contains N_i^C points, and the neighborhood of the *i-th* sub-cluster center with radius d_c contains $N_i^{d_c}$ points. If C_i contains less than half of $N_i^{d_c}$ points, that is, $N_i^C < 0.5 * N_i^{d_c}$, then more than half of the points in this neighborhood are attracted by other sub-clusters. This means that such a sub-cluster is not a true cluster and should be removed. Then, the sub-clusters are updated by assigning the points in the false sub-clusters to their nearest neighboring sub-clusters.

Fig. 6 gives an example of the sub-cluster updating process. Fig. 6a shows 4 updated sub-cluster centers (blue stars), 58 noise points (green circles), and 34 false sub-cluster centers (red circles). Fig. 6b shows 4 updated sub-clusters and 58 noise points (green circles). After the sub-cluster

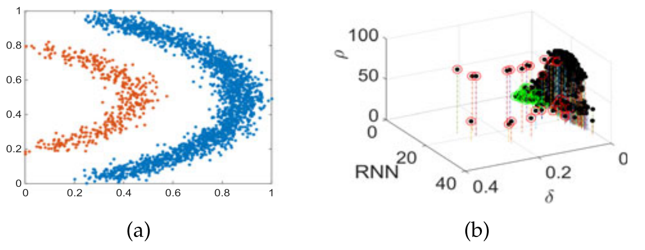
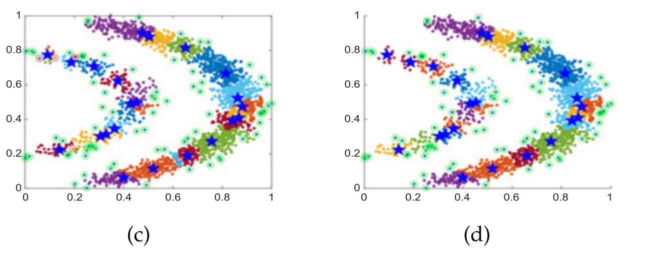



Fig. 7. The initial sub-cluster updating process for the Lithuanian dataset. (a) Dataset. (b) 3-D decision graph. (c) Noise points, false sub-cluster centers, and sub-cluster centers. (d) Noise points and sub-clusters.

updating process, the resulting number of sub-clusters is usually the number of true clusters.

Algorithm 3. Sub-Cluster Merging Algorithm

```
Input: Dataset $S = \{x_1, x_2, \cdots, x_n\}$, $ICC$.
Output: Final clusters $C$.
 1: Get the inner points set $In(S)$;
 2: for $m = 1$ to $ICC - 1$ do
 3:     for $n = m + 1$ to $ICC$ do
 4:         if $\min_{x_i \in C_m, x_j \in C_n}(d_{ij}) < r$ then
 5:             if $x_i \in In(C_m)$ and $x_j \in In(C_n)$ then
 6:                 Record $C_m$ and $C_n$;
 7:             else if $\rho_i + \rho_j > \rho_{C_{mn}}$ then
 8:                 Record $C_m$ and $C_n$;
 9:             end if
10:         end if
11:         $n = n + 1$;
12:     end for
13:     $m = m + 1$;
14: end for
15: Merge the corresponding clusters into one;
16: Assign the noise points to the clusters;
```

The pseudo code of the sub-cluster updating algorithm is reported in Algorithm 2. The false sub-cluster centers are deleted in lines 1-8. In lines 9-15 the points in the false sub-clusters are assigned to the remaining sub-clusters.

However, when a cluster contains multiple density peak points after the sub-cluster updating process, the resulting number of sub-clusters may not be the number of true clusters. We use Lithuanian to demonstrate this case in Fig. 7, where 54 noise points and 28 initial sub-cluster centers are depicted as green and red circles, respectively, in the decision graph shown in Fig. 7b; while only 22 true sub-cluster centers indicated as blue stars with 6 false sub-cluster centers depicted as red circles are shown in Fig. 7c. (54 noise points are depicted as green circles as well.) After the updating process, the 6 false sub-clusters are removed, and the 22 sub-cluster centers indicated as blue stars are left in Fig. 7d. However, there are only 2 true clusters. To obtain the true clusters, we propose a sub-cluster merging strategy in the following subsection.

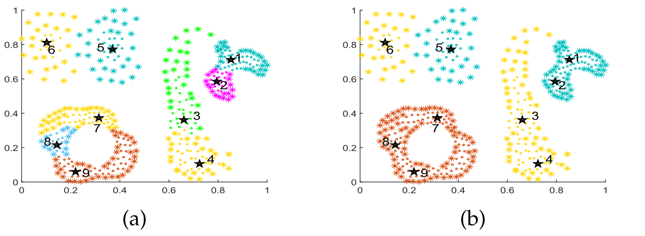


Fig. 8. Sub-cluster merging procedure, boundary points shown by $*$.

3.5 Sub-Cluster Merging Strategy

First, the boundary points of each sub-cluster are identified. The boundary points are defined as the points whose densities are lower than the average density of the given sub-cluster, that is, for $\forall x_i \in C_j$, if $\rho_i < average(\rho_{C_j})$, then x_i is a boundary point of the sub-cluster C_j. As shown in Fig. 8a, points marked with $*$ are the boundary points of each sub-cluster.

To merge the updated sub-clusters, let S denote the dataset, $No(S)$ denote the set of noise points in S, and $In(X)$ denote a sub-set of X excluding all boundary points of X. The merging radius r is defined by Eq. (6) as follows

$$r = \begin{cases} d_c, & if\ No(S) = \emptyset \\ \max_{x_i \in S/No(S)}\{D_i\}, & otherwise. \end{cases} \tag{6}$$

where D_i has been known in computing d_c in Eq. (3). Therefore, it is unnecessary to re-compute it. For sub-clusters C_m and C_n to be merged, denote $d(C_m, C_n) = \min\{d_{ij} | x_i \in C_m, x_j \in C_n\}$, and two cases need to be considered:

1) If $d(C_m, C_n) > r$, then C_m and C_n are far away and not merged.
2) Otherwise,

i) if $d(C_m, C_n) = \min\{d_{ij} | x_i \in In(C_m), x_j \in In(C_n)\}$, then C_n and C_m are very close and can be merged directly. As shown in Fig. 8a, sub-clusters 1 and 2 satisfy the above condition and are merged. Similarly, sub-clusters 3 and 4 also satisfy the above condition and are merged.
ii) Otherwise,

A) denote $(x_t, x_s) = arg\min_{x_i \in C_m, x_j \in C_n}(d_{ij})$. If the sum of the densities of x_t and x_s is larger than the average density of their sub-cluster centers, that is, if $\rho_t + \rho_s > \rho_{C_{mn}}$, then C_m and C_n are merged, where ρ_t is the density of x_t and $\rho_{C_{mn}}$ is the average density of the centers of C_m and C_n. As shown in Fig. 8a, sub-clusters 7 and 8 meet the condition for merging and are merged.
B) Otherwise, the two sub-clusters are not merged. For example, sub-clusters 5 and 6 in Fig. 8a do not meet any merging conditions and are not merged.

This process repeats until no pair of sub-clusters meets the merging conditions. After the merging process, if there remains no noise points, the clustering is complete. Otherwise, each noise point is assigned to its nearest cluster.

TABLE 3
Information Regarding the Twelve Datasets

Datasets	Points	Features	Clusters	Cluster sizes
Lithuanian	2400	2	2	2000,400
Banana	2400	2	2	2000,400
Spiral	1000	2	2	500,500
Flame	240	2	2	153,87
Gaussian	2000	2	4	1212,606,121,61
Ids2	3200	2	5	2000,400,400,200,200
Breast cancer	683	9	2	444,239
Ecoli	327	5	5	143,77,52,35,20
Vote	232	13	2	124,108
Thyroid	215	5	3	150,35,30
Vehicle	846	18	4	218,217,212,199
Robotnavigation	5456	24	4	2205,2097,826,328

Fig. 8b demonstrates the final five clusters. A brief description of the merging algorithm is shown in Algorithm 3.

3.6 The Framework and Time Complexity of the Proposed Algorithm

The framework of the proposed algorithm is described in Algorithm 4. To analyze the time complexity of the proposed algorithm, suppose that the dataset is $S = \{x_1, x_2, \cdots, x_n\}$. The time complexity of calculating all distances between all points is $O(n^2)$. When computing the k nearest neighbors, the complexity is $O(k * n * logn)$ with $k \ll n$; when computing the reverse nearest neighbors, the complexity is $O(n)$. Thus, the total complexity for assessing the noise points is $O(k * n * logn)$. The complexity of d_c calculation is $O(n)$. The complexity of density calculation is $O(mn)$, where m is the number of points in the neighborhood, usually $m \ll n$. When calculating the upward distance, the complexity is $O(n)$. When selecting the initial cluster center, the complexity is $O(n)$. When assessing the false sub-cluster centers, the complexity is $O(c)$, where c is the number of initial clusters, usually $c \ll n$. When assessing the boundary points, the complexity is $O(\bar{c}n)$; when merging the initial clusters, the complexity is $O(\bar{c}^2)$, where $\bar{c}$ is the number of updated initial clusters and $\bar{c} \leq c$. Therefore, the overall time complexity of the algorithm is $O(n^2)$.

Algorithm 4. The Framework of the Proposed Algorithm LDPI

```
1: Determine the noise points, generate the initial sub-cluster
   centers and corresponding initial sub-cluster using Algorithm 1.
2: Update the initial sub-cluster centers and corresponding
   sub-clusters using Algorithm 2.
3: Merge the sub-clusters to get the final clusters using Algorithm 3.
```

4 Experiments

4.1 Test Datasets and the Benchmark Algorithms

To evaluate the performance of the proposed LDPI algorithm, 12 test datasets (including six synthetic datasets and six real datasets from [31], [39], [40], [41], [42]) listed in Table 3 were used in the experiments. These datasets include both the non-convex datasets depicted in Figs. 9, 10, 11, and 12 and the convex datasets depicted in Figs. 13 and 14. The datasets depicted in Figs. 9, 10, 13, and 14 are imbalanced. The density distributions of the datasets demonstrated in Figs. 9 and 10 are non-uniform. These datasets have different characteristics of features, clusters, density distributions and degrees of cluster imbalance. Thus, they can represent a wide spectrum of real-life scenarios and be used to critically test the capabilities of the clustering algorithms. Detailed information on these datasets is listed in Table 3. The real datasets were processed using principal component analysis, and the features of all datasets were normalized to [0, 1].

To test the performance of the proposed LDPI algorithm, five commonly used state-of-the-art clustering algorithms, DPC[19], DPADN[37], Fuzzy-CFSFDP[25], MC[28] and SMCL[31], were used as benchmarks, where DPC is a classic density peaks algorithm and Fuzzy-CFSFDP and DPADN are two improved versions of the DPC algorithm. These algorithms can automatically assign the clustering centers.

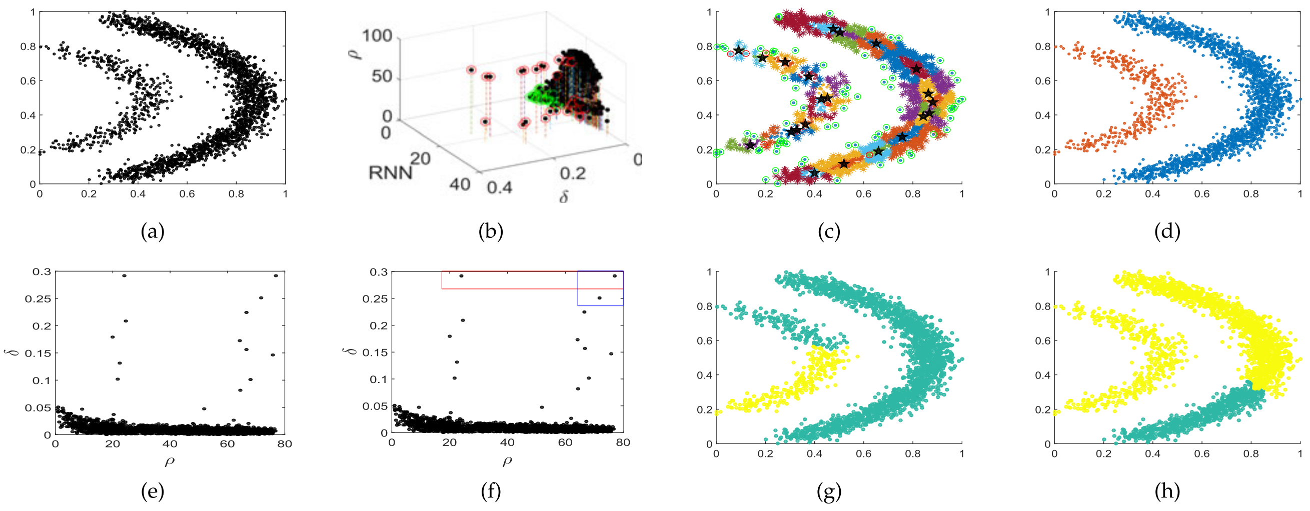


Fig. 9. Dataset Lithuanian. (a) Dataset. (b) 3-D decision graph. (c) Initial cluster centers, boundary points, and noise points. (d) Clustering result. (e) 2-D decision graph. (f) Center selection. (g) Clustering result with 2 centers in the red rectangle. (h) Clustering result with 2 centers in the blue rectangle.

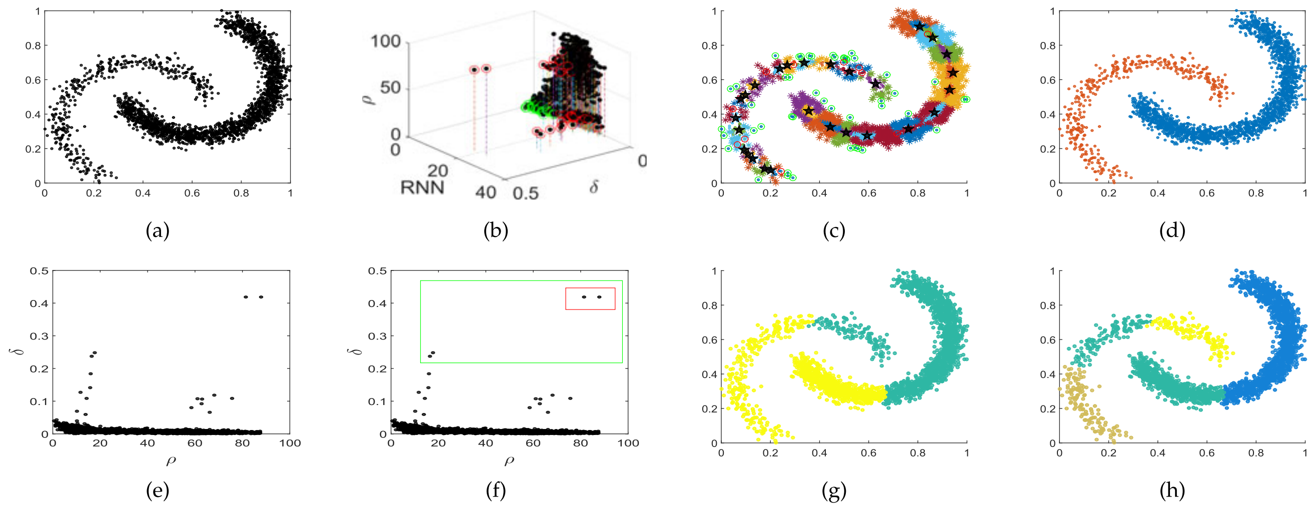


Fig. 10. Dataset Banana. (a) Dataset. (b) 3-D Decision graph. (c) Initial cluster centers, boundary points, and noise points. (d) Clustering result. (e) 2-D decision graph. (f) Center selection. (g) Clustering result with 2 centers. (h) Clustering result with 4 centers.

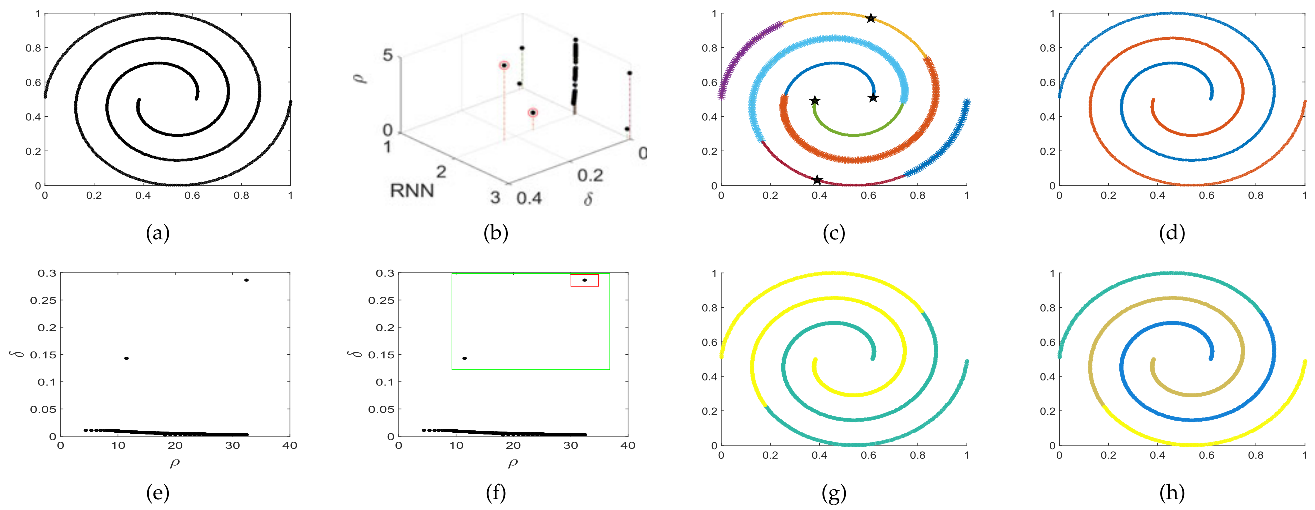


Fig. 11. Dataset Spiral. (a) Dataset. (b) 3-D decision graph. (c) Initial cluster centers, boundary points, and noise points. (d) Clustering result. (e) 2-D decision graph. (f) Center selection. (g) Clustering result with 2 centers. (h) Clustering result with 4 centers.

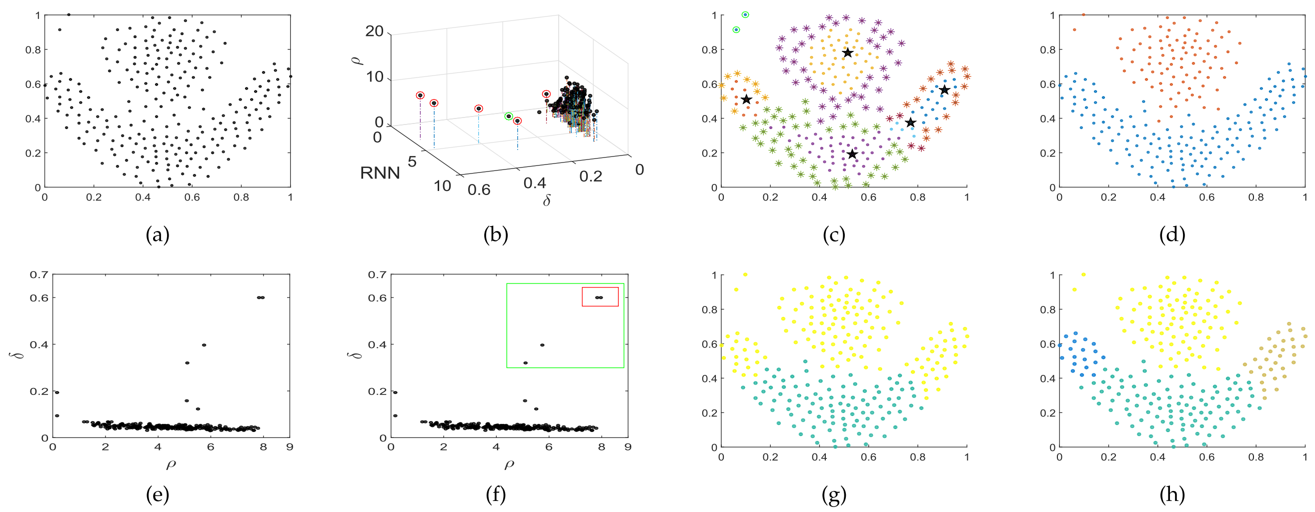


Fig. 12. Dataset Flame. (a) Dataset. (b) 3-D decision graph. (c) Initial cluster centers, boundary points, and noise points. (d) Clustering result. (e) 2-D decision graph. (f) Center selection. (g) Clustering result with 2 centers. (h) Clustering result with 4 centers.

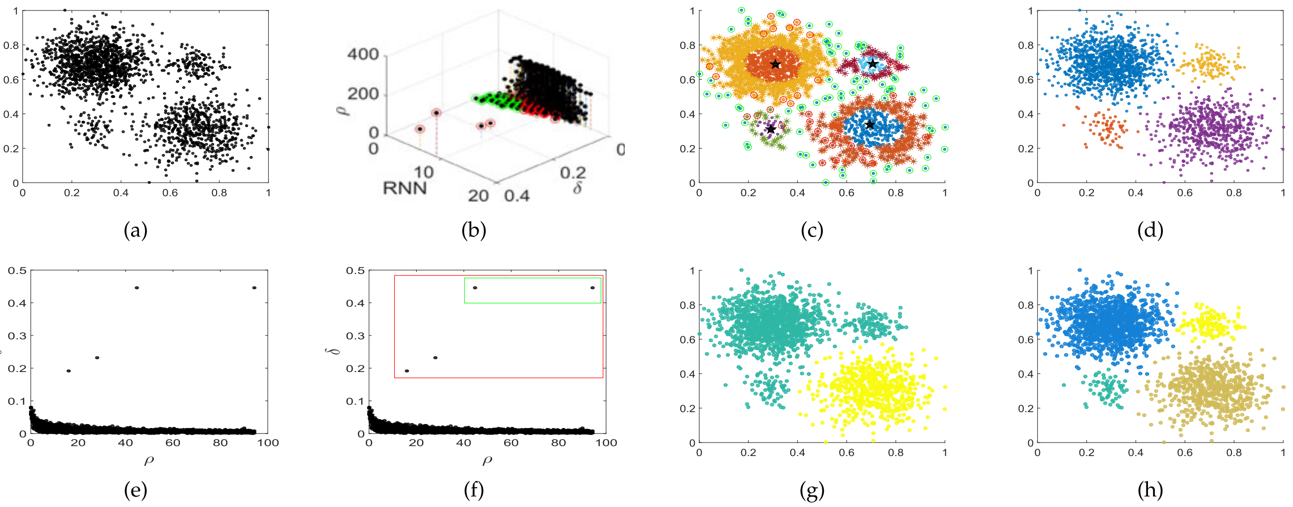


Fig. 13. Dataset Gaussian. (a) Dataset. (b) 3-D decision graph. (c) Initial cluster centers, boundary points, and noise points. (d) Clustering result. (e) 2-D decision graph. (f) Center selection. (g) Clustering result with 2 centers. (h) Clustering result with 4 centers.

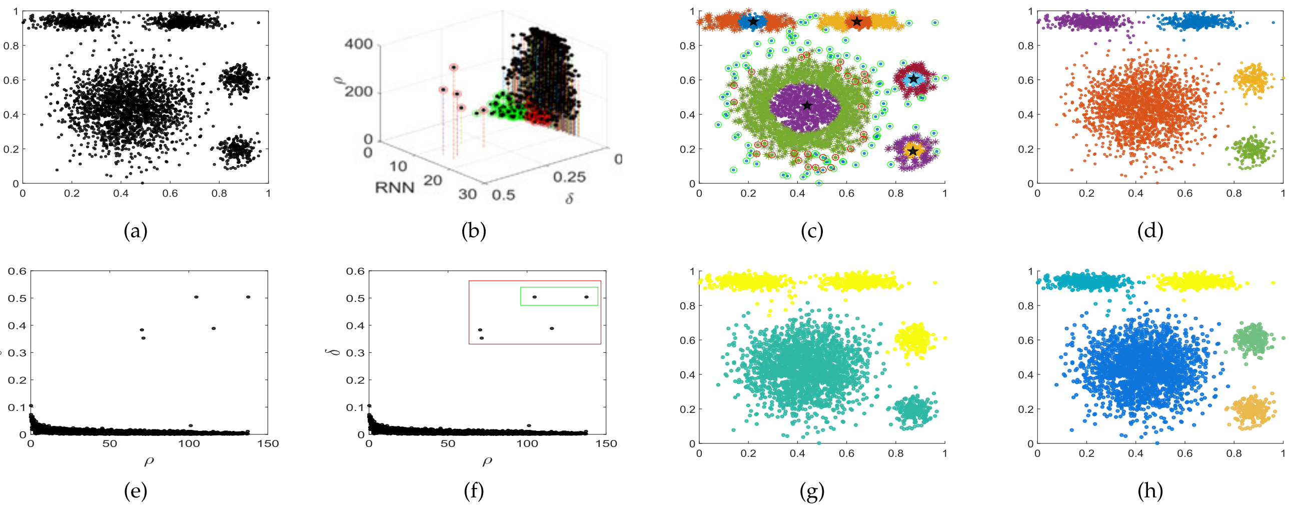


Fig. 14. Dataset lds2. (a) Dataset. (b) 3-D decision graph. (c) Initial cluster centers, boundary points, and noise points. (d) Clustering result. (e) 2-D decision graph. (f) Center selection. (g) Clustering result with 2 centers. (h) Clustering result with 5 centers.

MC and SMCL are clustering algorithms for imbalanced datasets. Parameters of these five benchmark algorithms were set to the suggested values mentioned in researchers' respective original researches, while the proposed algorithm did not need any parameter setting. The experiments were conducted on a PC with an Intel i7 2.9 GHz CPU, 8 GB RAM, and Windows 10 operating system.

4.2 Performance Comparison

Five widely used performance metrics, accuracy (ACC), recall, normalized mutual information (NMI), the number of clusters obtained (Clusters) and time cost, were used to evaluate the performance of all the algorithms: DPC, DPADN, Fuzzy-CFSFDP, MC, SMLC, and LDPI.

4.2.1 Visual Demonstration of the Clustering Process of LDPI on Different Datasets

To intuitively demonstrate the performance of the proposed algorithm LDPI, the clustering processes of LDPI and DPC on synthetic datasets are demonstrated in Figs. 9, 10, 11, 12, 13, and 14. Sub-figures (a)-(d) in Figs. 9, 10, 11, 12, 13, and 14 illustrate the clustering process using LDPI, while sub-figures (e)-(h) depict the clustering process using DPC.

Figs. 9a, 9b, 9c, and 9d demonstrate the clustering process on Lithuanian using the LDPI algorithm. The ratio of the number of points in the smallest cluster to the number of points in the largest cluster is 1:5. Not only the number of points but also the density of point distributions in the two clusters is different. LDPI first automatically selected 54 points as noise points (indicated with green circles) and 28 points as initial sub-cluster centers (indicated with red circles) from the decision graph in Fig. 9b. Then, 6 false sub-cluster centers were removed from the initial sub-cluster centers; the remaining 22 true sub-cluster centers were depicted as black five-stars in Fig. 9c. Finally, the algorithm identified the boundary points marked by asterisks $*$ in Fig. 9c and merged the 22 sub-clusters according to the sub-cluster merging algorithm. The correct clustering result is presented in Fig. 9d. Figs. 9e, 9f, 9g, and 9h show the clustering process on Lithuanian using the DPC algorithm. Fig. 9e depicts a 2-dimensional decision graph. This dataset has

TABLE 4
Information Regarding the Clustering Process of the Synthetic Datasets

Datasets	Num-Noi[1]	Num-Ini[2]	Num-Fal[3]	Num-Sub[4]	Num-cl[5]	Ratio[6]	Convex	Den- dis[7]
Lithuanian	54	28	6	22	2	1:5	Non-convex	non-uniform
Banana	33	38	11	27	2	1:5	Non-convex	non-uniform
Spiral	0	4	0	4	2	1:1	Non-convex	partially uniform
Flame	2	5	0	5	2	1:1.76	Non-convex	partially uniform
Gaussian	58	38	34	4	4	1:19.88	convex	partially uniform
Ids2	71	33	28	5	5	1:10	convex	partially uniform

[1]*Num-Noi is the number of noise.*
[2]*Num-Ini is the number of initial sub-clusters.*
[3]*Num-Fal is the number of false sub-cluster centers.*
[4]*Num-Sub is the number of sub-clusters.*
[5]*Num-cl is the number of clusters.*
[6]*Ratio is the ratio of the number of samples in the smallest cluster to the number of samples in the largest one.*
[7]*Den-dis is the density distributions of the dataset.*

two clusters. Suppose that we know a priori that there are two clusters. (In fact, the number of clusters is unknown in advance.) According to the DPC algorithm, the two points with larger density and upward distance values were selected as cluster centers. If one selects two points shown in the red rectangular area depicted in Fig. 9f as cluster centers, the clustering result is shown in Fig. 9g. In Fig. 9f, if one selects two points shown in the blue rectangular area as cluster centers, the corresponding clustering result is shown in Fig. 9h. Notice that none of these clustering results is correct.

The same procedure was applied to the other datasets, whose results are presented in Figs. 10, 11, 12, 13, and 14 in a similar manner to that described in Fig. 9. Detailed information regarding the number of noise points, number of false sub-cluster centers, density distributions, and ratio of the number of samples in the smallest cluster to the number of samples in the largest cluster for each dataset is listed in Table 4. The DPC algorithm will face two cases when selecting the number of clusters, i.e., it selects the correct number of clusters (the points inside the red rectangular area) or the incorrect number of clusters (the points inside the green rectangular area), as shown in Figs. 10, 11, 12, 13, and 14 (f). Note that two points inside the red rectangular area coincide and become one point, and, likewise, four points inside the green rectangular area become two points in Fig. 1 1f. Special attention should be paid to Figs. 10, 11, and 12 (f)-(h), from which we can see that all clustering results obtained by DPC are incorrect no matter whether the number of clusters is correct. In Figs. 13 and 14 (f)-(h), if the number of clusters is selected correctly, the clustering results obtained by DPC are correct; otherwise, they are incorrect.

Observe from the results depicted in Figs. 9, 10, 11, 12, 13, and 14 and Table 4 that LDPI produces correct clustering results for all the test datasets. It has excellent clustering capability for datasets with different cluster densities and different levels of data imbalance. For the DPC algorithm, when the number of clusters is selected incorrectly, the clustering results are wrong. Even when the number of clusters is selected correctly, there are still four datasets with wrong clustering results.

4.2.2 Comparison to Five Benchmark Algorithms

The comparison results on the four metrics, accuracy (ACC), recall, normalized mutual information (NMI), and number of clusters obtained (Clusters), are reported in Table 5, where the best and second-best values are indicated with bold face and underline, respectively. For non-convex datasets (Lithuanian, Banana, Spiral, and Flame), LDPI achieved the best results on the first three datasets among all the benchmark algorithms and the second-best result on Flame. In fact, LDPI obtained the correct clusters precisely on the first three datasets. For Flame, although LDPI performed marginally worse compared to DPADN and Fuzzy-CFSFDP with respect to ACC, recall, and NMI, it achieved very good values on ACC, recall, and NMI (0.9917, 0.9935, and 0.9359, respectively). The MC algorithm achieved less satisfactory clustering results on the first three datasets and a reasonably good result on Flame. Flame has a less non-convex characteristic while the first three datasets have a stronger non-convex characteristic. This indicates that MC performs unsatisfactorily on strong non-convex datasets. This is because MC is based on the K-means algorithm and inherits the ineffective characteristics of the K-means algorithm on non-convex datasets. SMCL performed very well on Lithuanian, relatively well on Banana and Flame, and poorly on Spiral. SMCL is based on the K-means algorithm as well but it adopts the multiple center technique, which results in better performance to some extent on non-convex datasets. DPADN achieved good clustering results on Spiral and Flame, but a poor result on Lithuanian. DPC failed to process Lithuanian, Banana, Spiral and Flame. Fuzzy-CFSFDP performed well on Spiral and Flame, but poorly on Lithuanian and Banana. DPC, DPADN, and Fuzzy-CFSFDP perform poorly because they are greatly affected by d_c. In addition, the center point selection condition for Fuzzy-CFSFDP tends to ignore the minor clusters of some imbalanced datasets.

Gaussian and Ids2 are convex and imbalanced datasets. LDPI performed the best on these two datasets. For Gaussian, DPC achieved the best result on ACC and the second-best results on recall and NMI. LDPI achieved the second-best result on ACC and the best results on recall and NMI. For Ids2, LDPI achieved the best results on ACC, recall, and NMI. DPC achieved the second-best results on ACC and NMI. LDPI, SMCL, and MC correctly assigned the number of clusters on both datasets, while Fuzzy-CFSFDP and DPADN incorrectly assigned them on the two datasets. Generally, LDPI has good performance on convex datasets

TABLE 5
Results of Clustering

Datasets	Measures	DPC	DPADN	Fuzzy-CFSFDP	MC	SMCL	LDPI
Lithuanian	Accuracy	0.5967	0.6946	0.5946	0.6602	**1.0000**	**1.0000**
	Recall	0.5165	0.4168	0.5168	0.6782	**1.0000**	**1.0000**
	NMI	0.0001	0.0644	0.0001	0.2065	**1.0000**	**1.0000**
	Clusters	-	**2.0000**	**2.0000**	**2.0000**	**2.0000**	**2.0000**
Banana	Accuracy	0.7092	0.9923	0.6923	0.7515	0.9996	**1.0000**
	Recall	0.8225	0.9949	0.8049	0.6802	0.9998	**1.0000**
	NMI	0.2740	0.9878	0.2678	0.7152	0.9928	**1.0000**
	Clusters	-	**2.0000**	**2.0000**	**2.0000**	**2.0000**	**2.0000**
spiral	Accuracy	0.7360	**1.0000**	**1.0000**	0.5996	0.5320	**1.0000**
	Recall	0.7363	**1.0000**	**1.0000**	0.6213	0.7559	**1.0000**
	NMI	0.1673	**1.0000**	**1.0000**	0.0912	0.0404	**1.0000**
	Clusters	-	**2.0000**	**2.0000**	**2.0000**	**2.0000**	**2.0000**
Flame	Accuracy	0.7878	**1.0000**	**1.0000**	0.9892	0.9875	0.9917
	Recall	0.8333	**1.0000**	**1.0000**	0.9892	0.9853	0.9935
	NMI	0.4132	**1.0000**	**1.0000**	0.9001	0.8994	0.9359
	Clusters	-	**2.0000**	**2.0000**	**2.0000**	**2.0000**	**2.0000**
Gaussian	Accuracy	**0.9910**	0.6070	0.6060	0.9690	0.9810	0.9895
	Recall	0.9897	0.5607	0.7644	0.9695	0.9070	**0.9926**
	NMI	0.9337	0.2075	0.2539	0.9023	0.9101	**0.9371**
	Clusters	-	3.0000	2.0000	**4.0000**	**4.0000**	**4.0000**
Ids2	Accuracy	0.9953	0.6250	0.8703	0.9898	0.9931	**0.9959**
	Recall	0.9881	0.2000	0.8934	0.9898	0.9835	**0.9983**
	NMI	0.9752	0.0011	0.7985	0.9512	0.9652	**0.9778**
	Clusters	-	2.0000	4.0000	**5.0000**	**5.0000**	**5.0000**
Breast	Accuracy	0.6810	0.6515	0.3499	0.8892	**0.9678**	0.9561
	Recall	0.6922	0.5021	0.5000	0.8012	**0.9638**	0.9459
	NMI	0.1823	0.0092	0.0001	**0.8127**	0.7830	0.7283
	Clusters	-	3.0000	1.0000	3.0000	**2.0000**	**2.0000**
Ecoli	Accuracy	0.6433	0.4373	0.4373	0.6503	0.6843	**0.6911**
	Recall	0.5093	0.2012	0.2000	0.5501	0.6524	**0.6849**
	NMI	0.3740	0.0156	0.0001	0.5023	0.5132	**0.5684**
	Clusters	-	2.0000	1.0000	2.0000	2.0000	**3.0000**
Vote	Accuracy	0.8126	0.5302	0.8060	0.7885	0.8664	**0.8966**
	Recall	0.8628	0.4960	0.8048	0.6203	0.8771	**0.9008**
	NMI	0.5301	0.0075	**0.5589**	0.4167	0.4812	0.5514
	Clusters	-	**2.0000**	3.0000	**2.0000**	**2.0000**	**2.0000**
Thyroid	Accuracy	0.6077	0.6977	0.6977	0.6516	0.6799	**0.7164**
	Recall	0.3333	0.3333	0.3333	0.3218	0.3676	**0.3778**
	NMI	0.0001	0.0425	0.0001	0.2045	0.2154	**0.2799**
	Clusters	-	**2.0000**	1.0000	**2.0000**	**2.0000**	**2.0000**
Vehicle	Accuracy	0.2400	0.2612	0.3413	0.3077	0.3463	**0.3676**
	Recall	0.2544	0.2548	0.3019	0.3582	0.3216	**0.3790**
	NMI	0.1198	0.0187	0.1045	0.1217	0.1179	**0.1466**
	Clusters	-	**3.0000**	2.0000	2.0000	2.0000	**3.0000**
Robotnavigation	Accuracy	0.3085	0.3091	0.3064	0.3134	0.1514	**0.4545**
	Recall	0.2634	0.2020	0.2760	0.2819	0.1514	**0.4876**
	NMI	0.0058	0.0001	0.0046	0.0073	0.0001	**0.1243**
	Clusters	-	2.0000	2.0000	2.0000	1.0000	**3.0000**

The best and second-best results are indicated with bold face and underline, respectively.

and datasets with imbalanced cluster sizes. DPC, MC, and SMCL have relatively good performance on these datasets. DPADN and Fuzzy-CFSFDP have poor performance on datasets with large imbalance ratios because they can hardly identify the centers of minor clusters.

For the remaining six real datasets, each has a different ratio of the number of samples in the smallest cluster to the number of samples in the largest one. These datasets are totally analyzed according to 24 performance indicator values (each with 4 indicator values). LDPI achieved 20 best performance indicator values (briefly best values) and 3 second-best performance indicator values (briefly second-best values). SMCL achieved 5 best values and 11 second-best values, and thus performed the second best overall. DPC, DPADN, Fuzzy-CFSFDP, and MC performed worse on these 6 datasets. They got 0, 3, 1, and 3 best values, respectively, and 2, 4, 3, and 6 second-best values, respectively. Concretely, for Breast, SMCL achieved the best values on

TABLE 6
Run-Time (in Second) of LDPI and Benchmark Algorithms

Datasets	DPC	DPADN	Fuzzy-CFSFDP	MC	SMCL	LDPI
Lithuanian	9.4865	5.8309	4.9349	155.6602	154.5697	14.7325
Banana	7.4576	22.5070	8.8576	118.6502	124.6125	13.3937
Spiral	6.2334	0.4267	0.4506	452.7820	356.8269	1.0085
Flame	6.1923	0.4894	0.3931	9.7632	10.8509	0.4702
Gaussian	7.5425	17.8587	5.1544	196.2310	119.1607	10.0920
Ids2	8.8381	34.2195	6.1643	1203.1968	525.6128	21.2560
Breast	6.7499	2.5501	3.3519	105.2309	374.9465	7.8430
Ecoli	7.3265	2.7109	2.1342	22.6891	15.4769	1.0555
Vote	5.9244	0.9360	1.1972	187.4625	104.6480	0.7414
Thyroid	8.4722	0.2371	3.1765	6.8710	5.2933	0.6628
Vehicle	9.8756	3.9987	4.6634	480.6896	504.7623	3.6378
Robotnavigation	17.3470	97.0040	22.1637	15672.4642	8364.5231	41.0657

ACC and recall, and the second-best value on NMI. MC achieved the best value on NMI. LDPI achieved the second-best values on ACC and recall, which is a very competitive result. For Ecoli, LDPI identified 3 final clusters, Fuzzy-CFSFDP identified 1 final cluster, and the other 4 algorithms identified 2 final clusters. (The correct result is 5 clusters.) Thus, LDPI performed the best on this dataset. For Vote, LDPI achieved the best values on ACC and recall, and the second-best value on NMI. For Thyroid, LDPI achieved the best values on ACC, recall, and NMI. DPADN and Fuzzy-CFSFDP achieved the second-best value on ACC, and SMCL achieved the second-best values on recall and NMI. For Vehicle, LDPI achieved the best values on ACC, Recall, and NMI. In terms of number of clusters, LDPI and DPADN found the closest number to the true value. For Robotnavigation, LDPI achieved the best values on ACC, Recall, and NMI. DPC achieved the second-best value on NMI, and MC achieved the second-best values on ACC and recall. For the correct number of clusters obtained, LDPI got the closest number to the true value.

In summary, the proposed LDPI performs the best on a variety of datasets with different levels of density distributions and imbalance ratios and performs much better than the 5 benchmark state-of-the-art algorithms on ACC, recall, NMI, and Clusters.

Time cost comparison. Observe from Table 6 that LDPI requires much less time than the imbalance clustering algorithms MC and SMCL on all datasets. For example, on the synthetic dataset Spiral, the time cost of MC and SMCL is approximately 450 and 350 times that of LDPI, respectively. On the real dataset Robotnavigation, the time cost of MC and SMCL is approximately 380 and 200 times that of LDPI, respectively. This is because many iterations are needed by MC and SMCL in many situations. To compare LDPI to the three DPC type algorithms, DPC, DPADN, and Fuzzy-CFSFDP, regarding time cost, observe from Table 6 that Fuzzy-CFSFDP used less time than LDPI did on 8 datasets and more time than LDPI did on 4 datasets. DPADN used less time than LDPI did on 4 datasets and more time than LDPI did on 8 datasets. DPC used less time than LDPI did on 5 datasets and more time than LDPI did on 7 datasets. Overall, regarding time cost, LDPI performs marginally worse than Fuzzy-CFSFDP, and better than the other compared algorithms. However, as discussed before, LDPI achieves much better results than the benchmark algorithms in terms of the other 4 performance indicators. In general, LDPI performs the best, especially with imbalanced datasets.

5 Conclusion

In this study, we proposed a novel clustering algorithm called LDPI, which aims to effectively deal with datasets that are imbalanced in terms of both cluster size and data density distribution. In LDPI, first, new methods to compute parameter d_c and local density are designed. Then, a novel 3-dimensional decision graph is designed. The existing 2-dimensional decision graph cannot identify the cluster centers and noise points correctly for imbalanced datasets, while the new 3-dimensional decision graph can solve this problem. Furthermore, certain identification conditions for the noise points and boundary points are proposed. Based on these, an initial sub-cluster construction scheme is designed first, which can automatically determine the noise points and initial sub-cluster centers. Second, a sub-cluster updating strategy is proposed, which can identify and remove the false sub-cluster centers and update the initial sub-clusters. Third, a sub-cluster merging scheme is designed to automatically merge the updated sub-clusters to form the final clusters. The proposed algorithm has three advantages over the already existing algorithms: 1) It can automatically identify the noise points and determine the cluster centers; 2) It does not need any input parameters; 3) It can effectively process imbalanced datasets and datasets with arbitrary shapes and distributions.

References

[1] J. L. Bruse *et al.*, "Detecting clinically meaningful shape clusters in medical image data: Metrics analysis for hierarchical clustering applied to healthy and pathological aortic arches," *IEEE Trans. Biomed. Eng.*, vol. 64, no. 10, pp. 2373–2383, Oct. 2017.

[2] S. Zhou and K. Wang, "Localization site prediction for membrane proteins by integrating rule and SVM classification," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 12, pp. 1694–1705, Dec. 2005.

[3] R. Kuo, L. Ho, and C. M. Hu, "Integration of self-organizing feature map and K-means algorithm for market segmentation," *Comput. Operations Res.*, vol. 29, no. 11, pp. 1475–1493, 2002.

[4] Q. Lu and C. Ju, "Research on credit card fraud detection model based on class weighted support vector machine," *J. Convergence Inf. Technol.*, vol. 6, no. 1, pp. 62–68, 2011.

[5] L. T. Law and Y. M. Cheung, "Color image segmentation using rival penalized controlled competitive learning," in *Proc. Int. Joint Conf. Neural Netw.*, 2003, pp. 108–112.

[6] X. Liu, Y. M. Cheung, M. Li, and H. L. Liu, "A lip contour extraction method using localized active contour model with automatic parameter selection," in *Proc. 20th Int. Conf. Pattern Recognit.*, 2010, pp. 4332–4335.

[7] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proc. 15th Berkeley Symp. Math. Statist. Probability*, 1967, pp. 281–297.

[8] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch: An efficient data clustering method for very large databases," *ACM Sigmod Rec.*, vol. 25, no. 2, pp. 103–114, 1996.

[9] M. Ester *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discov. Data Mining*, 1996, pp. 226–231.

[10] W. Wang, J. Yang, and R. Muntz, "Sting: A statistical information grid approach to spatial data mining," in *Proc. 23rd Int. Conf. Very Large Data Bases*, 1997, pp. 186–195.

[11] D. H. Fisher, "Knowledge acquisition via incremental conceptual clustering," *Mach. Learn.*, vol. 2, no. 2, pp. 139–172, 1987.

[12] F. Thabtah, "An accessible and efficient autism screening method for behavioural data and predictive analyses," *Health Inform. J.*, vol. 25, no. 4, pp. 1739–1755, 2019.

[13] H. Chen, W. Chung, J. J. Xu, G. Wang, Y. Qin, and M. Chau, "Crime data mining: A general framework and some examples," *Computer*, vol. 37, no. 4, pp. 50–56, 2004.

[14] W. Wei, J. Li, L. Cao, Y. Ou, and J. Chen, "Effective detection of sophisticated online banking fraud on extremely imbalanced data," *World Wide Web*, vol. 16, no. 4, pp. 449–475, 2013.

[15] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *J. Big Data*, vol. 6, no. 1, pp. 1–54, 2019.

[16] S. Kotsiantis *et al.*, "Handling imbalanced datasets: A review," *GESTS Int. Trans. Comput. Sci. Eng.*, vol. 30, no. 1, pp. 25–36, 2006.

[17] S. Barua, M. M. Islam, X. Yao, and K. Murase, "MWMOTE–majority weighted minority oversampling technique for imbalanced data set learning," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 2, pp. 405–425, Feb. 2014.

[18] A. Estabrooks, T. Jo, and N. Japkowicz, "A multiple resampling method for learning from imbalanced data sets," *Comput. Intell.*, vol. 20, no. 1, pp. 18–36, 2004.

[19] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.

[20] R. Mehmood, G. Zhang, R. Bie, H. Dawood, and H. Ahmad, "Clustering by fast search and find of density peaks via heat diffusion," *Neurocomputing*, vol. 208, pp. 210–217, 2016.

[21] S. Wang, D. Wang, C. Li, Y. Li, and G. Ding, "Clustering by fast search and find of density peaks with data field," *Chin. J. Electron.*, vol. 25, no. 3, pp. 397–402, 2016.

[22] L. Duan, L. Xu, F. Guo, J. Lee, and B. Yan, "A local-density based spatial clustering algorithm with noise," *Inf. Syst.*, vol. 32, no. 7, pp. 978–986, 2007.

[23] R. Liu, H. Wang, and X. Yu, "Shared-nearest-neighbor-based clustering by fast search and find of density peaks," *Inf. Sci.*, vol. 450, pp. 200–226, 2018.

[24] R. Mehmood, H. Dawood, R. Bie, and H. Ahmad, "Fuzzy clustering by fast search and find of density peaks," in *Proc. Int. Conf. Identification Inf. Knowl. Internet Things*, 2015, pp. 258–261.

[25] R. Bie, R. Mehmood, S. Ruan, Y. Sun, and H. Dawood, "Adaptive fuzzy clustering by fast search and find of density peaks," *Pers. Ubiquitous Comput.*, vol. 20, no. 5, pp. 785–793, 2016.

[26] A. Bryant and K. Cios, "RNN-DBSCAN: A density-based clustering algorithm using reverse nearest neighbor density estimates," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 6, pp. 1109–1121, Jun. 2018.

[27] G. Wen, X. Li, Y. Zhu, L. Chen, Q. Luo, and M. Tan, "One-step spectral rotation clustering for imbalanced high-dimensional data," *Inf. Process. Manage.*, vol. 58, no. 1, 2021, Art. no. 102388.

[28] J. Liang, L. Bai, C. Dang, and F. Cao, "The -means-type algorithms versus imbalanced data distributions," *IEEE Trans. Fuzzy Syst.*, vol. 20, no. 4, pp. 728–745, Aug. 2012.

[29] Y. Wang and L. Chen, "Multi-exemplar based clustering for imbalanced data," in *Proc. 13th Int. Conf. Control Automat. Robot. Vis.*, 2014, pp. 1068–1073.

[30] Y. M. Cheung, "On rival penalization controlled competitive learning for clustering with automatic cluster number selection," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 11, pp. 1583–1588, Nov. 2005.

[31] Y. Lu, Y. M. Cheung, and Y. Y. Tang, "Self-adaptive multiprototype-based competitive learning approach: A k-means-type algorithm for imbalanced data clustering," *IEEE Trans. Cybern.*, vol. 51, no. 3, pp. 1598–1612, Mar. 2021.

[32] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967.

[33] T. Basu and C. A. Murthy, "Towards enriching the quality of k-nearest neighbor rule for document classification," *Int. J. Mach. Learn. Cybern.*, vol. 5, no. 6, pp. 897–905, 2014.

[34] M. Du, S. Ding, and H. Jia, "Study on density peaks clustering based on k-nearest neighbors and principal component analysis," *Knowl.-Based Syst.*, vol. 99, pp. 135–145, 2016.

[35] Y. Chen *et al.*, "Fast density peak clustering for large scale data based on knn," *Knowl.-Based Syst.*, vol. 187, 2020, Art. no. 104824.

[36] S. Vadapalli, S. R. Valluri, and K. Karlapalem, "A simple yet effective data clustering algorithm," in *Proc. 6th Int. Conf. Data Mining*, 2006, pp. 1108–1112.

[37] W. Tong, S. Liu, and X.-Z. Gao, "A density-peak-based clustering algorithm of automatically determining the number of clusters," *Neurocomputing*, vol. 458, pp. 655–666, 2021.

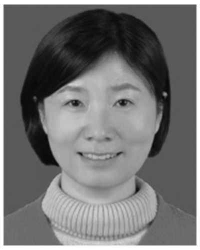
[38] Q. Zhu, J. Feng, and J. Huang, "Natural neighbor: A self-adaptive neighborhood method without parameter K," *Pattern Recognit. Lett.*, vol. 80, pp. 30–36, 2016.

[39] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[40] R. Duin, "PR-Tools4.1, a matlab toolbox for pattern recognition," 2017. [Online]. Available: http://prtools.org

[41] P. Franti and S. Sieranoja, "K-means properties on six clustering benchmark datasets," 2018. [Online]. Available: http://cs.uef.fi/sipu/datasets/

[42] A. L. Freire, G. A. Barreto, M. Veloso, and A. T. Varela, "Short-term memory mechanisms in neural network learning of robot navigation tasks: A case study," in *Proc. 6th Latin Amer. Robot. Symp.*, 2009, pp. 1–6.



Wuning Tong received the BE and ME degrees from the Shaanxi University of Science and Technology, Xi'an, China, in 2006 and 2009, respectively. She is currently working toward the PhD degree with the School of Computer Science and Technology of Xidian University, Xi'an, China and an associate professor with the School of Science with the Shaanxi University of Chinese Medicine. Her research interests include machine learning, optimization methods, and evolutionary computation.

Yuping Wang (Senior Member, IEEE) received the PhD degree from the Department of Mathematics, Xi'an Jiaotong University, Xi'an, China, in 1993. He has been a full professor since 1997 with the Department of Applied Mathematics and School of Computer Science and Technology, Xidian University, China. His research interests include optimization modeling for problems in computer science, evolutionary computation, and optimization algorithms. He has published more than 200 papers.

Delong Liu received the BE degree from the Wuhan University of Science and Technology, Wuhan, China, in 2019. He is currently working toward the postgraduate degree in computer science and technology at Xi'an Xidian University, Xi'an, China. His research interests include machine learning, optimization methods, and evolutionary computing.