Hindawi
Complexity
Volume 2020, Article ID 1731075, 17 pages
https://doi.org/10.1155/2020/1731075

Research Article

Clustering by Detecting Density Peaks and Assigning Points by Similarity-First Search Based on Weighted K-Nearest Neighbors Graph

Qi Diao,[1] **Yaping Dai**,[1] **Qichao An**,[1] **Weixing Li**,[1] **Xiaoxue Feng**,[1] **and Feng Pan**[1,2]

[1]*Beijing Institute of Technology, School of Automation, Beijing 100081, China*
[2]*Kunming-BIT Industry Technology Research Institute INC, Kunming 650106, China*

Correspondence should be addressed to Feng Pan; panfeng@bit.edu.cn

Received 29 April 2020; Revised 14 June 2020; Accepted 18 June 2020; Published 12 August 2020

Guest Editor: Kailong Liu

This paper presents an improved clustering algorithm for categorizing data with arbitrary shapes. Most of the conventional clustering approaches work only with round-shaped clusters. This task can be accomplished by quickly searching and finding clustering methods for density peaks (DPC), but in some cases, it is limited by density peaks and allocation strategy. To overcome these limitations, two improvements are proposed in this paper. To describe the clustering center more comprehensively, the definitions of local density and relative distance are fused with multiple distances, including K-nearest neighbors (KNN) and shared-nearest neighbors (SNN). A similarity-first search algorithm is designed to search the most matching cluster centers for noncenter points in a weighted KNN graph. Extensive comparison with several existing DPC methods, e.g., traditional DPC algorithm, density-based spatial clustering of applications with noise (DBSCAN), affinity propagation (AP), FKNN-DPC, and K-means methods, has been carried out. Experiments based on synthetic data and real data show that the proposed clustering algorithm can outperform DPC, DBSCAN, AP, and K-means in terms of the clustering accuracy (ACC), the adjusted mutual information (AMI), and the adjusted Rand index (ARI).

1. Introduction

The natural ecosystem has the characteristics of diversity, complexity, and intelligence, which provide infinite space for data-driven technology. As a new research focus, the data-driven prediction method has been widely used in energy, transportation, finance, and automobiles [1–7]. Clustering algorithm is an important branch of data-driven technology, which provides important information for further data analysis through mining the internal association of data [8, 9].

Due to the different definitions of clustering, different clustering strategies have been reported. Among them, the K-means algorithm is a simple and effective clustering algorithm. It preselects K initial clustering centers and then iteratively assigns each data point to the nearest clustering center [10]. Since the initial clustering center has certain impacts on the clustering results of K-means, the works [11, 12] provided several methods for selecting the initial clustering center and improving the accuracy of clustering. Since the K-means and its variants are based on the idea that data points are assigned to the nearest clustering center, these methods cannot facilitate the nonspherical clustering task well. Unlike the K-means algorithm, affinity propagation (AP) [8] has been developed based on the similarity between data points, and it can complete clustering by exchanging information between them. Hence, the AP algorithm does not need to determine the number of clusters in advance, and it has the time advantage in completing the clustering task of large-scale datasets [13]. However, for complex datasets, the AP method may also lead to performance degradation as the K-means method [14].

To address the aforementioned problems, density-based clustering methods have been proposed, which can find clusters of various shapes and sizes in noisy data, where the

high-density regions are considered as the clusters and separated by low-density regions [15–19]. In this line, density-based spatial clustering of applications with noise (DBSCAN) [15, 16] was proposed as an effective density-based clustering method. It needs to determine two parameters about the density of points (ε and MinPts) to achieve clustering of arbitrary shapes, where ε is the neighborhood radius and MinPts is the number of points contained in the neighborhood radius ε [15]. However, choosing a suitable threshold is a challenging task for these methods [15, 17]. Subsequently, Rodriguez and Laio [20] proposed a novel density-based clustering algorithm through fast search and density peaking (named as DPC). The DPC algorithm uses the local density and the relative distance of each point to establish a decision graph, finds the cluster centers according to the decision graph, and then assigns the noncenter point to the cluster of the nearest higher density neighbor. Although the DPC algorithm is simple and effective for detecting arbitrary shape clustering, several issues are limiting its practical application. Firstly, DPC is sensitive to the cutoff distance d_c, implying that the parameter d_c is set suitably to retain satisfactory performance, which is not a trivial task. Secondly, the clustering centers should be manually selected, which may not be feasible and convenient for some datasets. Moreover, the allocation error of high-density points will directly affect the allocation of low-density points around it, which may also contribute to propagating in the subsequent allocation process continuously.

To overcome these issues, the main advanced DPC algorithm has recently been studied. To avoid the influence of the cutoff distance d_c, the concept of K-nearest neighbors (KNN) has been introduced into the DPC algorithm, which proposed two different density measures, e.g., DPC-KNN [19] and FKNN-DPC [9]. Although both algorithms are based on the K-nearest neighbor information, they have been developed separately. Moreover, to solve the problem of manual selection of clustering centers, Li et al. [21] proposed a density peak clustering method to automatically determine the clustering centers. In this algorithm, the potential clustering centers are determined by the γ ranking graph, and then the true clustering centers are filtered out using the cutoff distance d_c. To remedy the allocation error transmission, FKNN-DPC [9] and SNN [22] both adopted a two-step allocation strategy to allocate noncentral points. In the first step, they use the breadth-first search to assign nonoutlier points. In the second step, FKNN-DPC uses the fuzzy weighted K-nearest neighbor technology to allocate the remaining points, and the SNN is based on whether the number of shared neighbors reaches the threshold to determine the cluster of the remaining points.

This paper proposed an improved clustering algorithm based on the density peaks (named as DPC-SFSKNN). It has the following new features: (1) the local density and the relative distance are redefined, and the distance attributes of the two neighbor relationships (KNN and SNN) are fused. This method can detect the low-density clustering center. (2) A new allocation strategy is proposed. A similarity-first search algorithm based on weighted KNN graphs is designed to allocate noncenter points. It has to be ensured that the allocation strategy is fault tolerant.

In general, this paper is organized as follows: Section 2 briefly mainly introduces the DPC algorithm and its development and analyzes the DPC algorithm in detail. Section 3 introduces the DPC-SFSKNN algorithm in detail and gives a detailed analysis. Section 4 tests the proposed algorithm on several synthetic and real-world datasets and compares its performance with DPC, DBSCAN, AP, FKNN-DPC, and K-means in terms of several very popular criteria for testing a clustering algorithm, namely, clustering accuracy (ACC), adjusted mutual information (AMI), and adjusted Rand index (ARI). Section 5 draws some conclusions.

2. Related Work

The density peak clustering algorithm (DPC) was proposed by Alex and Alessandro in 2014. The core idea of the DPC algorithm lies in the characterization of the cluster center, which has the following two characteristics: the cluster center point has a higher local density, which is surrounded by neighbor points with lower local density; the cluster center point is relatively far from other denser data points. These characteristics of the cluster center are related to two quantities: the local density ρ_i of each point i and its relative distance δ_i, which represents the closest distance from the point to larger density points.

2.1. DPC Algorithm and Improvements. Suppose X is a dataset for clustering and d_{ij} represents the Euclidean distance between data points i and j. The calculation of local density and relative distance depends on the distance d_{ij}. The DPC algorithm introduces two methods for calculating local density: the "cutoff" kernel method and the Gaussian kernel method. For a data point i, its local density ρ_i is defined in (1) with the "cutoff" kernel method and in (2) with the Gaussian kernel method:

$$\rho_i = \sum_j \chi\left(d_{ij} - d_c\right), \qquad \chi(\varsigma) = \begin{cases} 1, & \varsigma < 0, \\ 0, & \varsigma \geq 0, \end{cases} \tag{1}$$

$$\rho_i = \sum_j \exp\left(-\frac{d_{ij}^2}{d_c^2}\right), \tag{2}$$

where d_c is defined as a cutoff distance, which represents the neighborhood radius of the data point. The most significant difference between the two methods is that ρ_i calculated by the "cutoff" kernel is a discrete value, while ρ_i calculated by the Gaussian kernel is a continuous value. Therefore, the probability of conflict (different data points correspond to the same local density) in the latter is relatively smaller.

Moreover, d_c is an adjustable parameter in (1) and (2), which is defined as

$$d_c = d_{N*2\%}, \tag{3}$$

where d_c represents the average number of neighbors for each point, which is between 1 and 2 of all points [20]; N is the serial number of the last data point after the ascending order of all the distances d_{ij}, and it is also the total number of points; 2 in

formula (3) is the empirical parameter provided in reference [20], which can be adjusted according to different datasets.

The relative distance δ_i represents the minimum distance between the point i and any other higher density points and is mathematically expressed as

$$\delta_i = \begin{cases} \min\limits_{j:\ \rho_j > \rho_i} \left(d_{ij}\right), & \rho_i < \max\limits_k (\rho_k), \\ \max\limits_j \left(d_{ij}\right), & \rho_i = \max\limits_k (\rho_k), \end{cases} \tag{4}$$

where d_{ij} is the distance between points i and j. When the local density ρ_i is not the maximum density, the relative distance δ_i is defined as the minimum distance between the point i and any other higher density points; when ρ_i is the maximum density, δ_i takes the maximum distance to all other points.

After calculating the local density and relative distance of all data points, the DPC algorithm establishes a decision graph through the set of points ρ_i and δ_i. The point with high values of ρ_i and δ_i is called a peak, and the center of the cluster is selected from the peaks. Then, the DPC algorithm directly assigns the remaining points to the same cluster as the nearest neighbor peak.

For the DPC algorithm, the selection of d_c has a great influence on the correctness of the clustering results. Both DPC-KNN and FKNN-DPC schemes introduce the concept of K-nearest neighbors to eliminate the influence of d_c. Hence, two different local density calculations are provided.

The local density proposed by DPC-KNN [19] and FKNN-DPC [9] is given in (5) and (6), respectively:

$$\rho_i = \exp\left(-\frac{1}{K} \sum_{j \in \mathrm{knn}(i)} d_{ij}^2\right), \tag{5}$$

$$\rho_i = \sum_{j \in \mathrm{knn}(i)} \exp\left(-d_{ij}\right), \tag{6}$$

where K is the total number of nearest neighbors and $\mathrm{KNN}(i)$ represents the set of K-nearest neighbors of point i. These two methods provide a unified density metric for datasets of any size through the idea of K-nearest neighbors and solve the problem of nonuniformity of DPC's density metric for different datasets.

Based on K-nearest neighbors, SNN-DPC proposes the concept of shared-nearest neighbors (SNN) [22], which is used to represent the local density ρ_i and the relative distance δ_i. The idea of SNN is that if there are more same neighbors in the K-nearest neighbors of two points, the similarity of two points is higher, and the expression is given by

$$\mathrm{SNN}(i, j) = \mathrm{KNN}(i) \cap \mathrm{KNN}(j). \tag{7}$$

Based on the SNN concept, the expression of SNN similarity is as follows:

$$\mathrm{Sim}_{i,j} = \begin{cases} \dfrac{|\mathrm{SNN}(i, j)|^2}{\sum_{p \in \mathrm{SNN}(i,j)} \left(d_{ip} + d_{jp}\right)}, & \text{if } i, j \in \mathrm{SNN}(i, j), \\ 0, & \text{otherwise}, \end{cases} \tag{8}$$

where d_{ip} is the distance between points i and p and d_{jp} is the distance between points j and p. The condition for calculating SNN similarity is that points i and j appear in each other's K-nearest neighbor set. Otherwise, the SNN similarity between the two points is 0.

Next, the local density ρ_i of point i is expressed by SNN similarity. Suppose point i is any point in the dataset X, then $S(i) = x_1, x_2, \ldots, x_k$ represents the set of k points with the highest similarity with point i. The expression of local density is

$$\rho_i = \sum_{j \in S(i)} \mathrm{Sim}(i, j). \tag{9}$$

At the same time, the equation for the relative distance δ_i of the point i is as follows:

$$\delta_i = \begin{cases} \min\limits_{j:\ \rho_j > \rho_i} \left[d_{ij} \left(\sum\limits_{p \in \mathrm{knn}(i)} d_{ip} + \sum\limits_{q \in \mathrm{knn}(j)} d_{jp} \right) \right], & \rho_i < \max\limits_k (\rho_k), \\ \max\limits_{j \in (X - i)} (\delta_i), & \rho_i = \max\limits_k (\rho_k). \end{cases} \tag{10}$$

The SNN-DPC algorithm not only redefines the local density and relative distance, but also changes the data point allocation strategy. The allocation strategy divides the data points into two categories: "unavoidable subordinate points" and "probable subordinate points." The two types of data points have their allocation algorithms. Compared with the DPC algorithm, this allocation strategy method is better for the clustering of clusters with different shapes.

2.2. DPC Algorithm Analysis. The DPC algorithm proposes a very simple and elegant clustering algorithm. However, due to its simplicity, DPC has the following two potential problems to be further addressed in practice.

2.2.1. DPC Ignores Low-Density Points. When the density difference between clusters is large, the performance of the DPC algorithm can be significantly degraded. To show this issue, we take the dataset Jain [23] as an example, and then the clustering results calculated using the truncated kernel distance of the DPC are shown in Figure 1. It can be seen that the cluster distribution in the upper left is relatively sparse, and the cluster distribution in the lower right is relatively close. The red star in the figure represents the cluster centers in the upper left corner. Under the disparity in density, the clustering centers selected by the DPC are all on the tightly distributed cluster below. Due to the incorrect selection of the clustering centers, the subsequent allocations are also incorrect.

Analyzing the local density and the relative distance separately, from Figures 2(a) and 2(b), it can be seen that the ρ value and the δ value of point A of the false cluster center are much higher than that of the true cluster center C. The results of Gaussian kernel distance calculation are the same, and the correct clustering center cannot be selected on the dataset Jain. Therefore, how to increase the ρ value and the δ

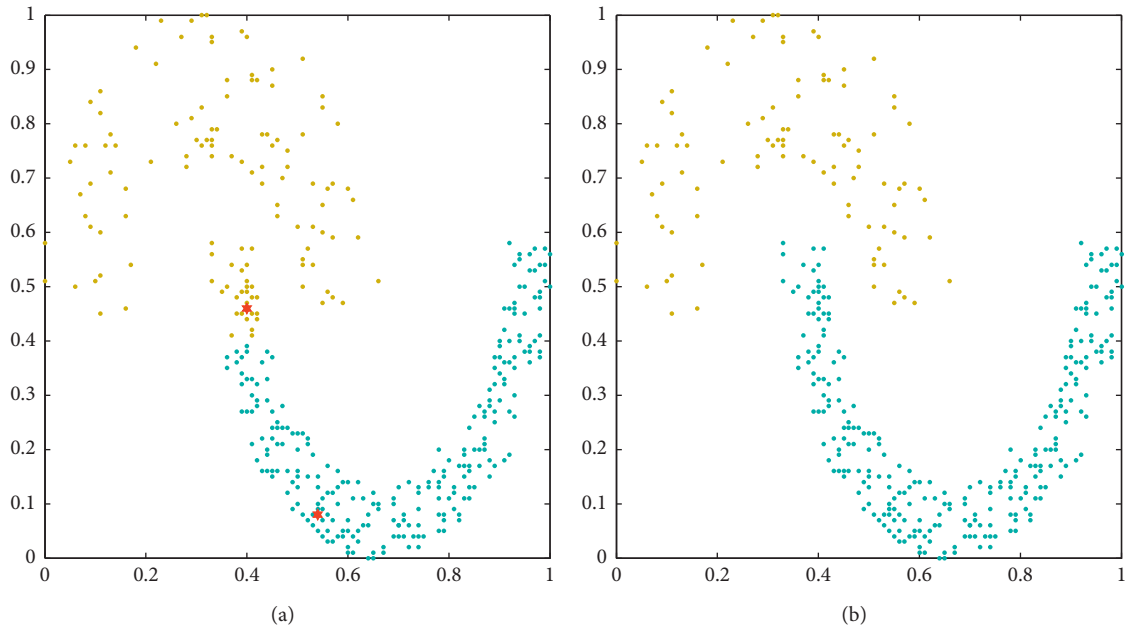


Figure 1: Results of the traditional DPC algorithm on the Jain dataset. (a) Clustering of Jain by DPC. (b) Ground truth.

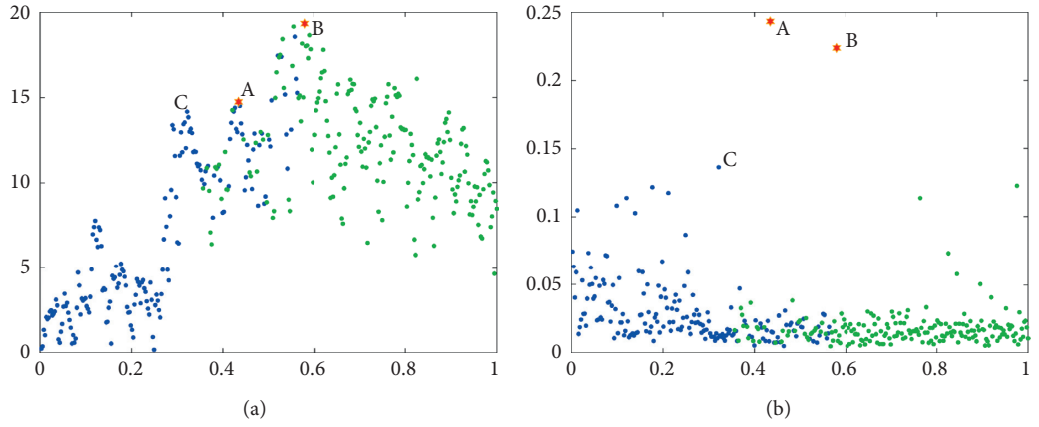


Figure 2: ρ and δ values of the result of the traditional DPC algorithm on the Jain dataset. (a) Clustering of Jain by DPC. (b) Ground truth.

value of the low-density center and make it stand out in the decision graph is a problem that needs to be considered.

2.2.2. DPC Ignores Low-Density Point Allocation Strategy with Low Fault Tolerance. The fault tolerance of the allocation strategy of the DPC algorithm is not satisfactory, mainly because the allocation of points receives a higher impact than the density of points. Hence, if a high-density point allocation error occurs, it will directly affect the subsequent allocation of points with a lower density. Taking the Pathbased dataset [24] as an example, Figure 3 shows the clustering result calculated by the DPC algorithm by using the "cutoff" kernel distance. It can be seen from the figure that the DPC algorithm can find a suitable clustering center, but the allocation results of most points are incorrect. The same is true of the results using the Gaussian kernel distance calculation. The results of point assignment on the Pathbased dataset are similar to those of "cutoff" kernel clustering. Therefore, the fault tolerance of the point allocation strategy should be further improved. Moreover, the points are greatly affected by other points during the allocation, which is also an issue to be further addressed.

3. Proposed Method

In this section, the DPC-SFSKNN algorithm is introduced in detail. The DPC-SFSKNN algorithm is proposed, where the five main definitions of the algorithm are introduced, and the entire algorithm process is introduced. Moreover, the complexity of the DPC-SFSKNN algorithm is analyzed.

3.1. The Main Idea of DPC-SFSKNN. The DPC algorithm relies on the distance between points to calculate the local density and the relative distance and is also very sensitive to

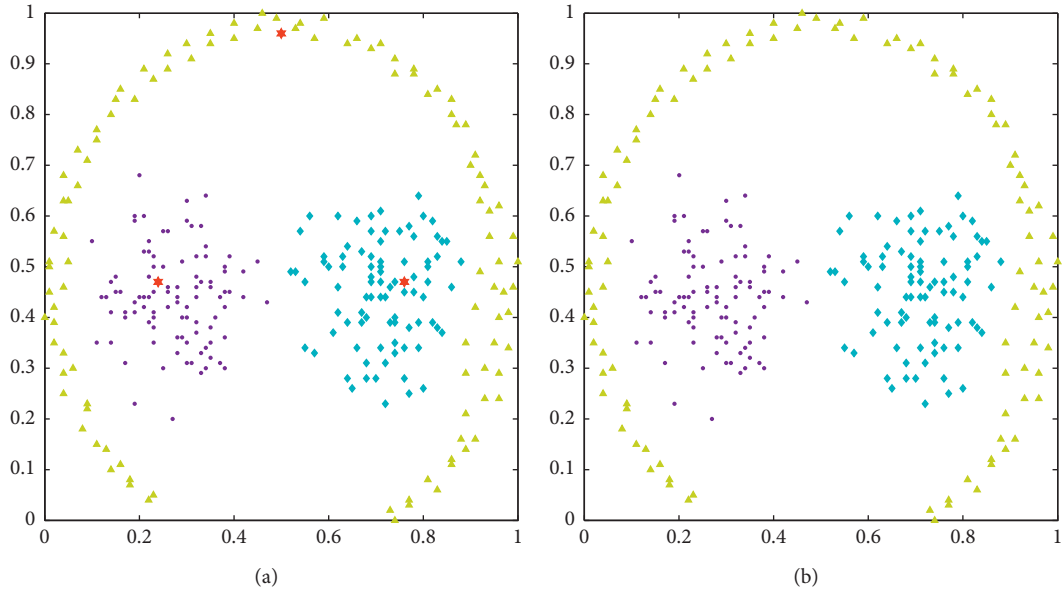


Figure 3: Results of the traditional DPC algorithm on the Pathbased dataset.

the choice of the cutoff distance d_c. Hence, the DPC algorithm may not be able to correctly process for some complex datasets. The probability that a point and its neighbors belong to the same cluster is high. Adding attributes related to neighbors in the clustering process can help to make a correct judgment. Therefore, we introduce the concept of shared-nearest neighbor (SNN) proposed in [22], when defining the local density and the relative distance. Its basic idea is that if they have more common neighbors, the two points are considered to be more similar as said above (see equation (7)).

Based on the above ideas, we define the average distance $\overline{d}_{\text{snn}}(i, j)$ of the shared-nearest neighbor between point i and point j and the similarity between the two points.

Definition 1 (average distance of SNN). For any points i and j in the dataset X, the shared-nearest neighbor set of two points is $\text{SNN}(i, j)$, and the average distance of SNN $\overline{d}_{\text{snn}}(i, j)$ is expressed as

$$\overline{d}_{\text{snn}}(i, j) = \frac{\sum_{p \in \text{SNN}(i,j)} \left(d_{ip} + d_{jp}\right)}{2S}, \tag{11}$$

where point p is any point of $\text{SNN}(i, j)$ and S is the number of members in the set $\text{SNN}(i, j)$. $\overline{d}_{\text{snn}}(i, j)$ shows the spatial relationship between point i and point j more comprehensively by calculating the distances between two points and shared-nearest neighbor points.

Definition 2 (similarity). For any points i and j in the dataset X, the similarity $\text{Sim}(i, j)$ between point i and j can be expressed as

$$\text{Sim}(i, j) = \frac{S}{K} * 100\%, \tag{12}$$

where K is the number of nearest neighbors. K is selected from 4 to 40 until the optimal parameter appears. The lower bound is 4 because a smaller K may cause the algorithm to become endless. For the upper bound, it is found by experiments that a large K will not significantly affect the results of the algorithm. The similarity is defined according to the aforementioned basic idea "if they have more common neighbors, the two points are considered to be more similar," and the similarity is described using the ratio of the number of shared-nearest neighbors to the number of nearest neighbors.

Definition 3 (K-nearest neighbor average distance). For any point i in the dataset X, its K-nearest neighbor set is $\text{KNN}(i)$, and then the expression of K-nearest neighbor average distance $\overline{d}_{\text{knn}}(i)$ is as follows:

$$\overline{d}_{\text{knn}}(i) = \frac{\sum_{p \in \text{knn}(i)} d_{ip}}{K}, \tag{13}$$

where point p is any point in $\text{KNN}(i)$ and the number of nearest neighbors of any point is K. K-nearest neighbor average distance can describe the surrounding environment of a point to some extent. Next, we use it to describe local density.

Definition 4 (local density). For any point i in the dataset X, the local density expression is

$$\rho_i = \sum_{j \in \text{knn}(i)} \frac{S}{\overline{d}_{\text{knn}}(i) + \overline{d}_{\text{knn}}(j)}, \tag{14}$$

where point j is a point in the set $\text{KNN}(i)$ and $\overline{d}_{\text{knn}}(i)$ and $\overline{d}_{\text{knn}}(j)$ are the K-nearest neighbor average distances of point i and point j, respectively. In formula (14), the numerator (the number of shared-nearest neighbor S) represents the similarity between the two points, and the denominator (the sum of the average distances) describes the environment around them. When S is a constant and if

the value of the sum of the average distances ($\overline{d}_{\text{knn}}(i) + \overline{d}_{\text{knn}}(j)$) is small, the local density ρ_i of point i is large. Point j is one of the K-nearest neighbors of point i. When the values of $\overline{d}_{\text{knn}}(i)$ and $\overline{d}_{\text{knn}}(j)$ are small, it means i and j are closely surrounded by their neighbors. If $\overline{d}_{\text{knn}}(i)$ has a larger value (point j is far away from point i) or $\overline{d}_{\text{knn}}(j)$ has a larger value (when the neighboring points of the distance are far away from the point j), the local density of the point i becomes smaller. Therefore, only the average distances of the two points are small, and it can be expressed that the local density of point i is large. Moreover, when the sum of the average distances of the two points is constant and if the number of shared-nearest neighbors of the two points is large, the local density is large. A large number of shared neighbors indicate that the two points have a high similarity and a high probability of belonging to the same cluster. The higher the similarity points around a point, the greater its local density and the greater the probability of becoming a cluster center. This is beneficial to those low-density clustering centers. A large number of shared neighbors can compensate for the loss caused by their large distance from other points so that their local density is not only affected by distance. Next, we define the relative distance of the points.

Definition 5 (relative distance). For any point i in the dataset X, the relative distance can be expressed as

$$\delta_i = \begin{cases} \min\limits_{j:\ \rho_j > \rho_i} \left[d_{ij} + \overline{d}_{\text{knn}}(i) + \overline{d}_{\text{knn}}(j) \right], & \rho_i < \max\limits_k (\rho_k), \\ \max\limits_{j \in (X-i)} (\delta_i), & \rho_i = \max\limits_k (\rho_k), \end{cases} \tag{15}$$

where point j is one of the K-nearest neighbors of point i, d_{ij} is the distance between points i and j, and $\overline{d}_{\text{knn}}(i)$ and $\overline{d}_{\text{knn}}(j)$ are the average distance from the nearest neighbor of points i and j. We can use the sum of the three distances to represent the relative distance. Compared to the DPC algorithm which only uses d_{ij} to represent the relative distance, we define the concept of relative distance and K-nearest neighbor average distances of two points. The new definition can not only express the relative distance, but also be more friendly to low-density cluster centers. Under the condition of constant d_{ij}, the average distance of the nearest neighbors of the low-density points is relatively large, and its relative distance will also increase, which can increase the probability of low-density points being selected.

The DPC-SFSKNN clustering center is selected in the same way as the traditional DPC algorithm. The local density ρ and relative distance δ are used to form a decision graph. The n points with the largest local density and relative distance are selected as the clustering centers.

For DPC-SFSKNN, the sum of the distances from points of a low-density cluster to their K-neighbors may be large; thus, they receive a greater compensation for their δ value. Figures 4(a) and 4(b) show the results of DPC-SFSKNN on the Jain dataset [23]. Compared to Figure 2(b), the δ values of points in the upper branch are generally larger than those of the lower branch. This is because the density of the upper branch is significantly smaller and the distances from the points to their respective K-nearest neighbors are larger; thus, they receive a greater compensation. Even if the density is at a disadvantage, the higher δ value still makes the center of the upper branch distinguished in the decision graph. This shows that the DPC-SFSKNN algorithm can correctly select low-density clustering centers.

3.2. Processes. The entire process of the algorithm is divided into two parts: the selection of clustering centers and the allocation of noncenter points. The main step of our DPC-SFSKNN and a detailed introduction of the proposed allocation strategy are given in Algorithm 1.

Line 9 of the DPC-SFSKNN algorithm establishes a weighted K-nearest neighbor graph, and Line 11 is a K-nearest neighbor similarity search allocation strategy. To assign noncenter points in the dataset, we designed a similarity-first search algorithm based on the weighted K-nearest neighbor graph. The algorithm uses the breadth-first search idea to find the cluster center with the highest similarity for the noncenter point. The similarity of noncenter points and their K-nearest neighbors is sorted in an ascending order, the neighbor point with the highest similarity is selected as the next visited node, and it is pushed into the path queue. If the highest similarity point is not unique, the point with the smallest SNN average distance is selected as the next visited node. The visiting node also needs to sort the similarity of its K-nearest neighbors and select the next visiting node. The search stops until the visited node is the cluster center point. Algorithm 2 describes the entire search process. Finally, each data point except the cluster centers is traversed to complete the assignment.

Similarity-first search algorithm is an optimization algorithm based on breadth-first search according to the allocation requirements of noncenter points. Similarity is an important concept for clustering algorithms. Points in the same cluster are similar to each other. Two points with a higher similarity have more of the same neighbors. Based on the above ideas, the definition of similarity is proposed in (12). In the process of searching, if only similarity is used as the search criteria, it is easy to appear that the highest similarity point is not unique. Therefore, the algorithm chooses the average distance of the SNN as the second criterion, and a smaller $\overline{d}_{\text{snn}}$ point means that the two points are closer in space.

The clustering results of the DPC-SFSKNN algorithm based on the Pathbased dataset are shown in Figure 5. Figure 3 clearly shows that although the traditional DPC algorithm can find cluster centers on each of the three clusters, there is a serious bias in the allocation of noncenter points. From Figure 5, we can see the effectiveness of the noncentral point allocation algorithm of the DPC-SFSKNN algorithm. The allocation strategy uses similarity-first search to ensure that the similarity from the search path is the highest, and a gradual search to the cluster center to avoid the points with low similarity is used as a reference. Besides, the similarity-first search allocation strategy based on the weighted K-nearest neighbor graph considers neighbor

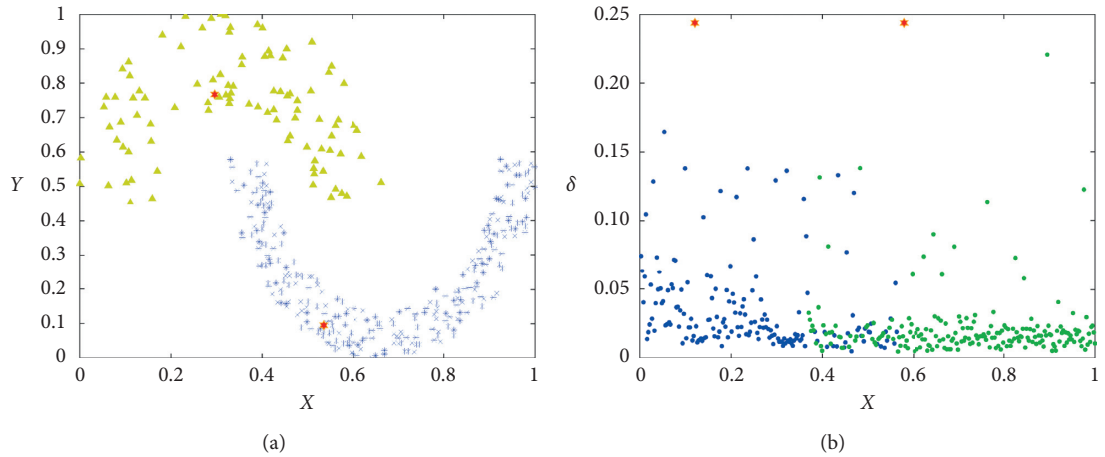


Figure 4: Result and ρ value of the DPC-SFSKNN algorithm on the Jain dataset.

Require: dataset X, parameter K
Ensure: clustering result C
(1) Data preprocessing: normalize the data
(2) Calculate the Euclidean distance between the points
(3) Calculate the K-nearest neighbors of each point $i \in X$
(4) Calculate the average distance of K-nearest neighbors of each point $\overline{d}_{\text{knn}}(i)$ according to (13)
(5) Calculate the local density ρ_i of each point $i \in X$ according to (14)
(6) Calculate the relative distance δ_i of each point $i \in X$ according to (15)
(7) Find the cluster center by analyzing the decision graph composed of ρ and δ and use the cluster center as the set CC
(8) Calculate the similarity between point i and its K-nearest neighbors according to (12)
(9) Connect each point in the dataset X with its K-nearest neighbors and use the similarity as the connection weight to construct a weighted K-nearest neighbor graph
(10) Calculate the average distance of SNN $\overline{d}_{\text{snn}}(i, j)$ between point i and its shared-nearest neighbors according to (11)
(11) Apply Algorithm 2 to allocate the remaining points

Algorithm 1: DPC-SFSKNN.

Require: $w \in X$, set of cluster centers CC, number of neighbors K, similarity matrix $S^{n*n} = \text{sim}(i, j)^{n*n}$, and SNN average distance matrix $\text{DSNN}^{n*n} = \overline{d}_{\text{snn}}(i, j)^{n*n}$
Ensure: point $w \in$ CC
(1) Initialize the descending queue Q and the path queue P. The K-nearest neighbors of point w are sorted in the ascending order of similarity and pushed into Q. Push M into P.
(2) **while** tail point of P $P \in$ CC **do**
(3) **if** the highest similarity point is unique. **then**
(4) Pop a point *this* at Q's tail
(5) **else**
(6) Select a point *this* with the smallest DSNN
(7) **end if**
(8) Empty descending queue Q
(9) The K-nearest neighbors of *this* are sorted in the ascending order of similarity and pushed into Q.
(10) Push *this* into P
(11) **end while**

Algorithm 2: Similarity-first search allocation strategy.

information. When the point of the highest similarity is not unique, the point with the shortest average distance of the shared neighbors is selected as the next visited point.

3.3. Complexity Analysis. In this section, the complexities of the DPC-SFSKNN algorithm are analyzed, including time complexity and space complexity. Suppose the size of the

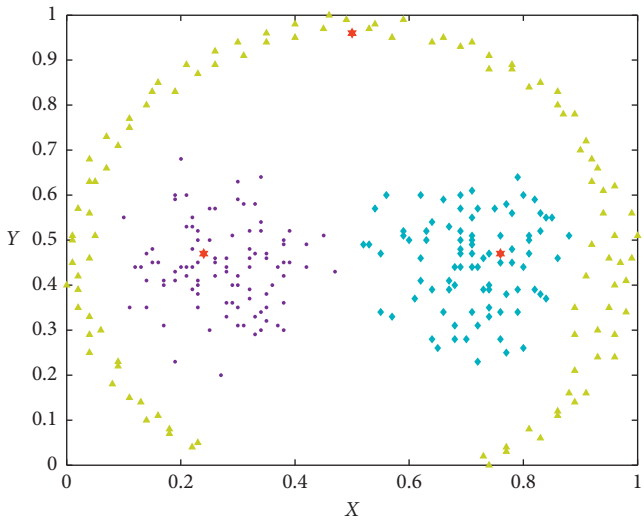


Figure 5: Results of the traditional DPC-SFSKNN algorithm on the Pathbased dataset.

dataset is n, the number of cluster centers is m and the number of neighbors is k.

3.3.1. Time Complexity. The time complexity analysis of DPC-SFSKNN is as follows.

Normalization requires a processing complexity of approximately $O(n)$; the complexities of calculating the Euclidean distance and similarity between points are $O(n^2)$; the complexity of computing the K-nearest neighbor average distance $\overline{d}_{\text{knn}}$ is $O(n^2)$; similarly, the complexity of the average distance $\overline{d}_{\text{snn}}$ between the calculation point and its shared-nearest neighbors does not exceed $O(n^2)$ at most; the calculation process of calculating the local density ρ_i and distance δ_i of each point needs to acquire the KNN information complexity of each point as $O(kn)$, so the complexities of local density ρ and distance δ are $O(kn^2)$; the point allocation part is the search time of one point, and in the worst case, searching all points requires $O(n)$. There are n points in the dataset, and the total time does not exceed $O(n^2)$. In summary, the total approximate time complexity of DPC-SFSKNN is $O(kn^2)$.

The time complexity of the DPC algorithm depends on the following three aspects: (a) the time to calculate the distance between points; (b) the time to calculate the local density ρ_i for point i, and (c) the time to calculate the distance δ_i for each point i. The time complexity of each part is $O(n^2)$, so the total approximate time complexity of DPC is $O(n^2)$.

The time complexity of the DPC-SFSKNN algorithm is k times higher than that of the traditional DPC algorithm. However, k is relatively small compared to n. Therefore, they do not significantly affect the run time. In Section 4, it is demonstrated that the actual running time of DPC-SFSKNN does not exceed k times of the running time of the traditional DPC algorithm.

3.3.2. Space Complexity. DPC-SFSKNN needs to calculate the distance and similarity between points, and its complexity is $O(n^2)$. Other data structures (such as ρ and δ arrays and various average distance arrays) are $O(n)$. For the allocation strategy, in the worst case, its complexity is $O(n^2)$. The space complexity of DPC is $O(n^2)$, which is mainly due to the distance matrix stored.

The space complexity of our DPC-SFSKNN is the same as that of traditional DPC, which is $O(n^2)$.

4. Experiments and Results

In this section, experiments are performed based on several public datasets commonly used to test the performance of clustering algorithms, including synthetic datasets [23–27] and real datasets [28–34]. In order to visually observe the clustering ability of DPC-SFSKNN, the DPC [20], DBSCAN [15], AP [8], FKNN-DPC [9], and K-means [10] methods are all tested for comparison. Three popular benchmarks are used to evaluate the performance of the above clustering algorithms, including the clustering accuracy (ACC), adjusted mutual information (AMI), and adjusted Rand index (ARI) [35]. The upper bounds of the three benchmarks were all 1. The larger the benchmark value, the better the clustering effect. The codes for DPC, DBSCAN, and AP were provided based on the corresponding references.

Table 1 lists the synthetic datasets used in the experiments. These datasets were published in [23–27]. Table 2 lists the real datasets used in the experiments. These datasets include the real-world dataset from [29–34] and the Olivetti face dataset in [28].

To eliminate the influence of missing values and differences in different dimension ranges, the datasets need to be preprocessed before proceeding to the experiments. We replace the missing values by the mean of all valid values of the same dimension and normalize the data using the min-max normalization method shown in the following equation:

$$\overline{x}_{ij} = \frac{x_{ij} - \min\left(x_j\right)}{\max\left(x_j\right) - \min\left(x_j\right)}, \tag{16}$$

where x_{ij} represents the original data located in the ith row and jth column, $\overline{x}_{ij}$ represents the rescaled data of x_{ij}, and x_j represents the original data located in the jth column.

Min-max normalization method processes each dimension of the data and preserves the relationships between the original data values [36], therefore decreasing the influence of the difference in dimensions and increasing the efficiency of the calculation.

To fairly reflect the clustering results of the five algorithms, the parameters in the algorithms are adjusted to ensure that their satisfactory clustering performance can be retained. For the DPC-SFSKNN algorithm, the parameter K needs to be specified in advance, and an initial clustering center is manually selected based on a decision graph composed of the local density ρ and the relative distance δ. It can be seen from the experimental results in Tables 3 and 4 that the value of parameter K is around 6, and the value of parameter K for the dataset with dense sample distribution is more than 6. In addition to manually select the initial clustering center, the traditional DPC algorithm also needs

Table 1: Synthetic datasets.

Dataset	Records	Attributes	Clusters	Source
Pathbased	300	2	3	[24]
Jain	373	2	2	[23]
Flame	240	2	2	[25]
Aggregation	788	2	7	[26]
DIM512	1024	512	16	[27]
DIM1024	1024	1024	16	[27]

Table 2: Real-world datasets.

Dataset	Records	Attributes	Clusters	Source
Iris	150	4	3	[29]
Libras movement	360	90	15	[31]
Wine	178	13	3	[29]
Parkinsons	197	23	2	[29]
WDBC	569	30	2	[34]
Pima-Indians-diabetes	768	8	2	[29]
Segmentation	2310	19	7	[29]
Dermatology	366	33	6	[29]
Seeds	210	7	3	[30]
Ionosphere	351	34	2	[33]
Waveform	5000	21	3	[32]
Waveform (noise)	5000	40	3	[32]
Olivetti face	400	92∗112	40	[28]

Table 3: The comparison of ACC, AMI, and ARI benchmarks for 6 clustering algorithms on synthetic datasets.

Algorithm	AMI	ARI	ACC	EC/AC	Par	AMI	ARI	ACC	EC/AC	Par
		Pathbased						Jain		
DPC-SFSKNN	0.926	0.910	0.925	3/3	6	**1.000**	**1.000**	**1.000**	2/2	7
DPC	0.521	0.463	0.742	3/3	2	0.609	0.713	0.853	2/2	3
DBSCAN	0.781	0.522	0.667	—	0.05/6	0.883	0.985	0.918	—	0.08/10
AP	0.679	0.475	0.783	3/3	10	0.681	0.812	0.882	2/2	40
FKNN-DPC	**0.941**	**0.960**	**0.987**	3/3	5	0.056	0.132	0.793	—	10
K-means	0.568	0.461	0.772	—	3	0.492	0.577	0.712	—	2
	Aggregation					Flame				
DPC-SFSKNN	0.942	0.951	0.963	7/7	6	0.873	0.934	0.956	2/2	6
DPC	**1.000**	**1.000**	**1.000**	7/7	4	**1.000**	**1.000**	**1.000**	2/2	5
DBSCAN	0.969	0.982	0.988	—	0.05/8	0.867	0.936	0.981	-	0.09/8
AP	0.795	0.753	0.841	7/7	7.7	0.452	0.534	0.876	3/3	35
FKNN-DPC	0.995	0.997	0.999	3/3	8	**1.000**	**1.000**	**1.000**	2/2	5
K-means	0.784	0.717	0.786	—	7	0.418	0.465	0.828	—	2
	DIM512					DIM1024				
DPC-SFSKNN	**1.000**	**1.000**	**1.000**	16/16	8	**1.000**	**1.000**	**1.000**	16/16	9
DPC	**1.000**	**1.000**	**1.000**	16/16	2	**1.000**	**1.000**	**1.000**	16/16	0.01
DBSCAN	**1.000**	**1.000**	**1.000**	—	0.3/7	**1.000**	**1.000**	**1.000**	—	10/8
AP	**1.000**	**1.000**	**1.000**	16/16	20	**1.000**	**1.000**	**1.000**	16/16	30
FKNN-DPC	**1.000**	**1.000**	**1.000**	16/16	8	**1.000**	**1.000**	**1.000**	16/16	10
K-means	0.895	0.811	0.850	—	1	0.868	0.752	0.796	—	16

to determine d_c. Based on the provided selection range, d_c is selected so that the number of neighbors is between 1 and 2% of the total number of data points [20]. The two parameters that DBSCAN needs to determine are ε and minpts as in [15]. The optimal parameters are determined using a circular search method. The AP algorithm only needs to determine a preference, and the larger the preference, the more the center points are allowed to be selected [8]. The general method for selecting parameters is not effective, and only multiple experiments can be performed to select the optimal parameters. The only parameter of K-means is the number of clusters. The true number of clusters in the dataset is used in this case. Similarly, FKNN-DPC needs to determine the nearest neighbors K.

Table 4: Comparison of ACC, AMI, and ARI benchmarks for 6 clustering algorithms on real-world datasets.

Algorithm	AMI	ARI	ACC	EC/AC	Par	AMI	ARI	ACC	EC/AC	Par
	Iris					Libras movement				
DPC-SFSKNN	0.896	0.901	0.962	3/3	6	**0.547**	**0.368**	**0.510**	10/15	8
DPC	0.812	0.827	0.926	3/3	2	0.535	0.304	0.438	9/15	0.5
DBSCAN	0.792	0.754	0.893	—	0.14/9	0.412	0.183	0.385	—	0.96/5
AP	0.764	0.775	0.911	3/3	6	0.364	0.267	0.453	10/15	2.5
FKNN-DPC	0.**912**	**0.922**	**0.973**	3/3	7	0.508	0.308	0.436	10/15	9
K-means	0.683	0.662	0.823	—	3	0.522	0.306	0.449	—	15
	Wine					Parkinsons				
DPC-SFSKNN	**0.843**	**0.851**	**0.951**	3/3	6	0.193	0.380	0.827	2/2	6
DPC	0.706	0.672	0.882	3/3	2	0.210	0.114	0.612	2/2	5
DBSCAN	0.612	0.643	0.856	—	0.42/10	0.205	0.213	0.674	—	0.4/6
AP	0.592	0.544	0.781	3/3	6	0.142	0.127	0.669	2/2	15
FKNN-DPC	0.831	0.852	0.949	3/3	7	**0.273**	**0.391**	**0.851**	2/2	5
K-means	0.817	0.838	0.936	—	3	0.201	0.049	0.625	—	2
	WDBC					Ionosphere				
DPC-SFSKNN	0.432	0.516	0.857	2/2	6	0.361	0.428	0.786	3/2	7
DPC	0.002	−0.004	0.602	2/2	9	0.238	0.276	0.681	3/2	0.65
DBSCAN	0.397	0.538	0.862	—	0.27/7	**0.544**	**0.683**	**0.853**	—	0.2/7
AP	0.598	0.461	0.854	2/2	40	0.132	0.168	0.706	2/2	15
FKNN-DPC	**0.679**	**0.786**	**0.944**	2/2	7	0.284	0.355	0.752	2/2	8
K-means	0.611	0.730	0.928	—	2	0.129	0.178	0.712	—	2
	Segmentation					Pima-Indians-diabetes				
DPC-SFSKNN	**0.665**	**0.562**	**0.746**	6/7	6	0.037	0.083	0.652	2/2	6
DPC	0.650	0.550	0.684	6/7	3	0.033	0.075	0.647	2/2	4
DBSCAN	0.446	0.451	0.550	—	0.25/10	0.028	0.041	0.577	—	0.15/6
AP	0.405	0.436	0.554	7/7	25	0.045	0.089	0.629	3/2	35
FKNN-DPC	0.655	0.555	0.716	7/7	7	0.001	0.011	0.612	2/2	6
K-means	0.583	0.495	0.612	—	6	**0.050**	**0.102**	**0.668**	—	2
	Seeds					Dermatology				
DPC-SFSKNN	0.753	0.786	0.919	3/3	7	**0.862**	**0.753**	0.808	7/6	6
DPC	0.727	0.760	0.918	3/3	2	0.611	0.514	0.703	4/6	2
DBSCAN	0.640	0.713	0.874	—	0.17/8	0.689	0.690	**0.815**	—	0.7/3
AP	0.598	0.682	0.896	3/3	10	0.766	0.701	0.762	7/6	5
FKNN-DPC	**0.759**	**0.790**	**0.924**	3/3	8	0.847	0.718	0.768	7/6	7
K-means	0.671	0.705	0.890	—	3	0.796	0.680	0.702	—	6
	Waveform					Waveform (noise)				
DPC-SFSKNN	0.355	**0.382**	**0.725**	3/3	5	0.267	**0.288**	**0.651**	3/3	6
DPC	0.320	0.269	0.586	3/3	0.5	0.104	0.095	0.502	3/3	0.3
DBSCAN	—	—	—	—	—	—	—	—	—	—
AP	—	—	—	—	—	—	—	—	—	—
FKNN-DPC	0.324	0.350	0.703	3/3	5	0.247	0.253	0.648	3/3	5
K-means	**0.363**	0.254	0.501	-	3	**0.364**	0.252	0.512	—	3

4.1. Analysis of the Experimental Results on Synthetic Datasets. In this section, the performance of DPC-SFSKNN, DPC [20], DBSCAN [15], AP [8], FKNN-DPC [9], and K-means [10] is tested with six synthetic datasets given in Table 1. These synthetic datasets are different in distribution and quantity. Different data situations can be simulated to compare the performance of six algorithms in different situations. Table 3 shows AMI, ARI, ACC, and EC/AC of the five clustering algorithms on the six comprehensive datasets, where the best results are shown in bold and "—" means no value. Figures 6–9 show the clustering results of DPC-SFSKNN, DPC, DBSCAN, AP, FKNN-DPC, and K-means based on the Pathbased, Flame, Aggregation, and Jain datasets, respectively. The five algorithms achieve the optimal clustering on DIM512 and DIM1024 datasets, so that the clustering of the two datasets is not shown. Since the cluster centers of DBSCAN are relatively random, only the positions of clustering centers of the other three algorithms are marked.

Figure 6 shows the results of the Pathbased dataset. DPC-SFSKNN and FKNN-DPC can complete the clustering of the Pathbased dataset correctly. From Figures 6(b), 6(d), and 6(f), it can be seen that the clustering results of DPC, AP, and K-means are similar. The clustering centers selected by DPC, AP, DPC-SFSKNN, and FKNN-DPC are highly similar, but the clustering results of DPC and AP are not satisfactory. For the DPC algorithm, the low fault tolerance rate of its allocation strategy is the cause of this result. A

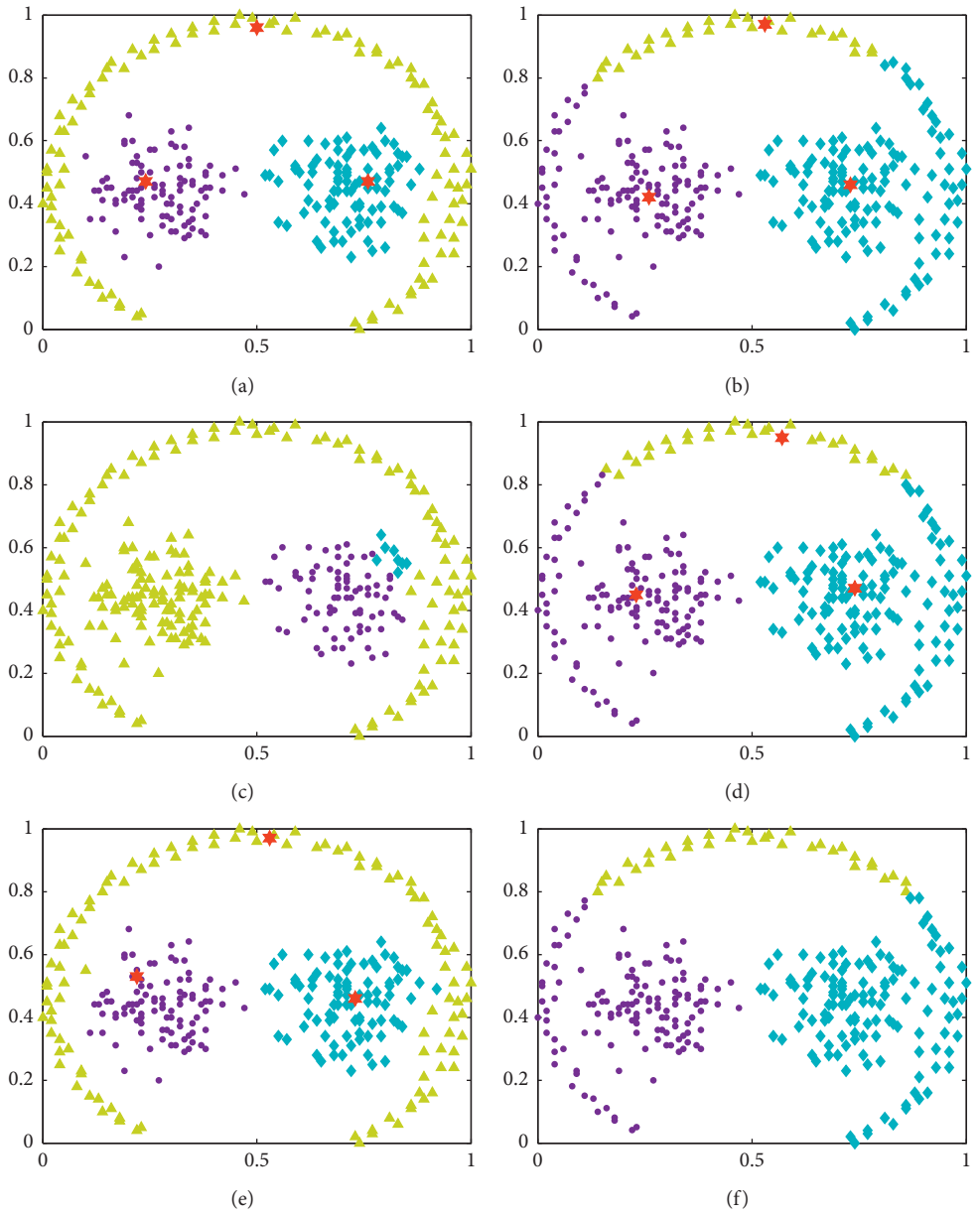


Figure 6: The clustering of Pathbased by 6 clustering algorithms. (a) DPC-SFSKNN, (b) DPC, (c) DBSCAN, (d) AP, (e) FKNN-DPC, and (f) K-means.

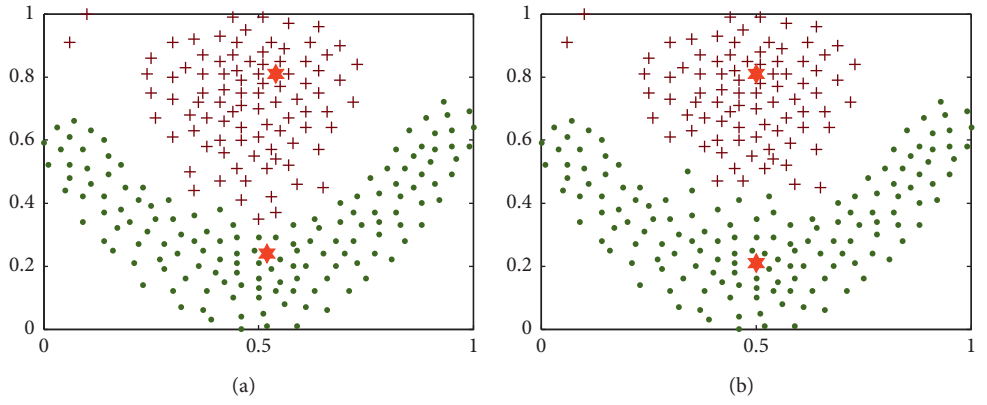


Figure 7: Continued.

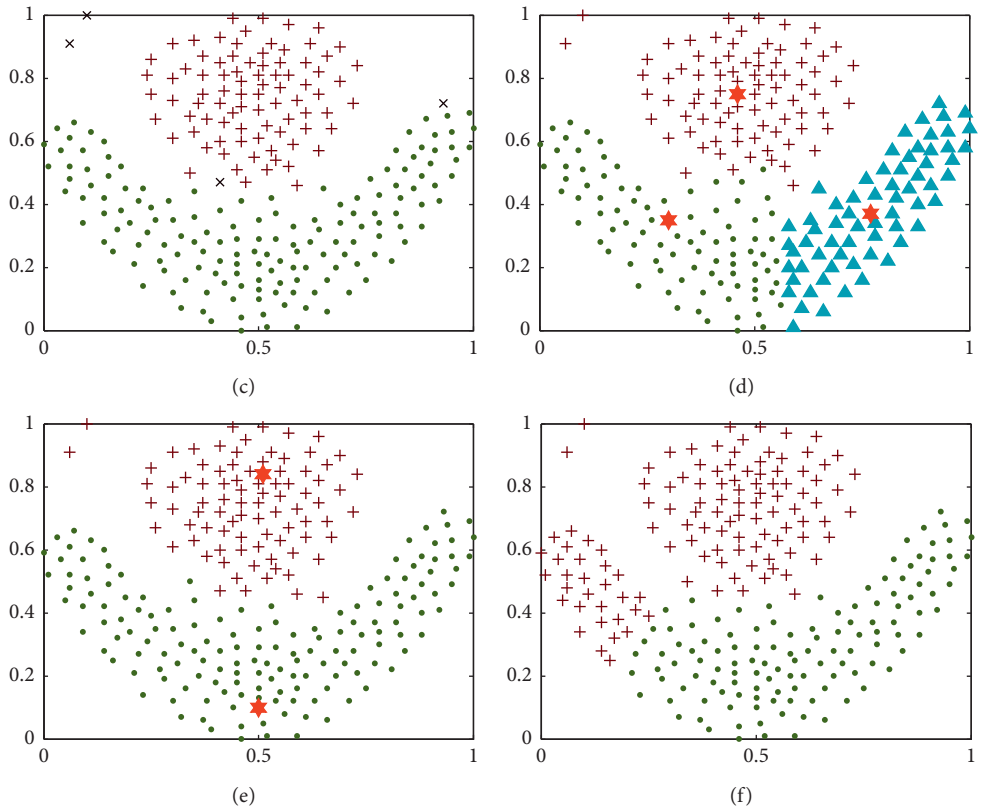


Figure 7: The clustering of Flame by 6 clustering algorithms. (a) DPC-SFSKNN, (b) DPC, (c) DBSCAN, (d) AP, (e) FKNN-DPC, and (f) K-means.

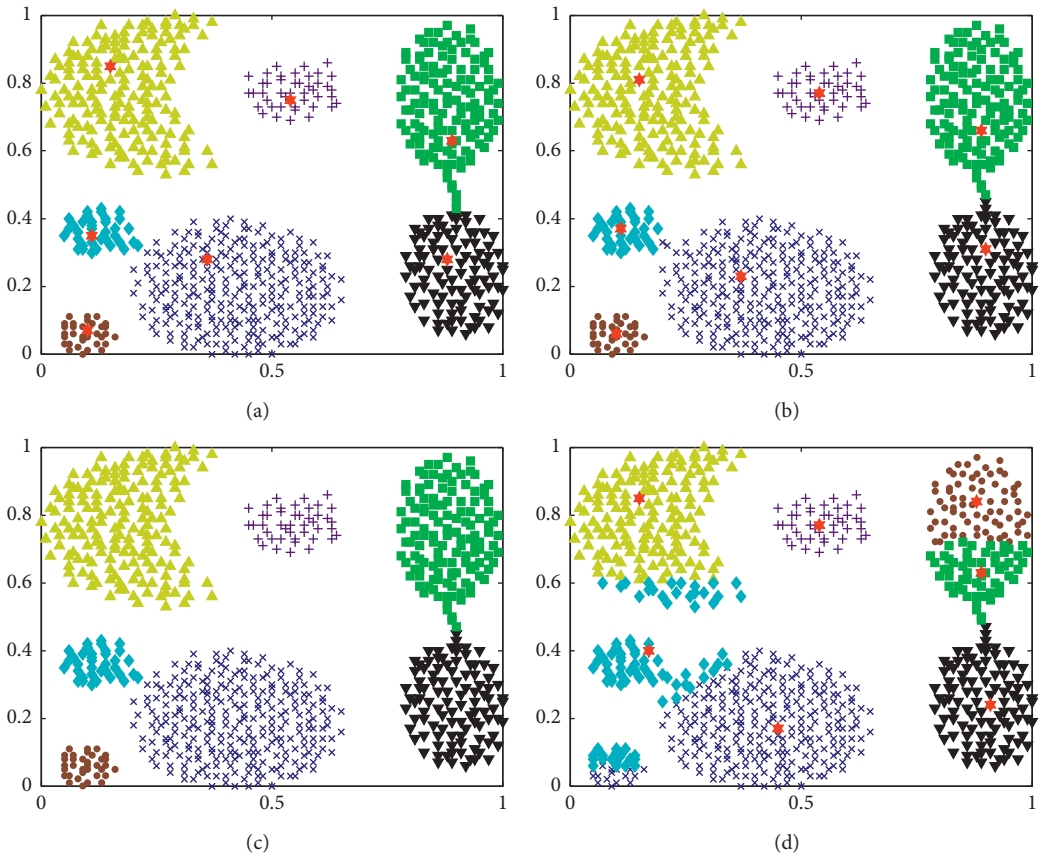


Figure 8: Continued.

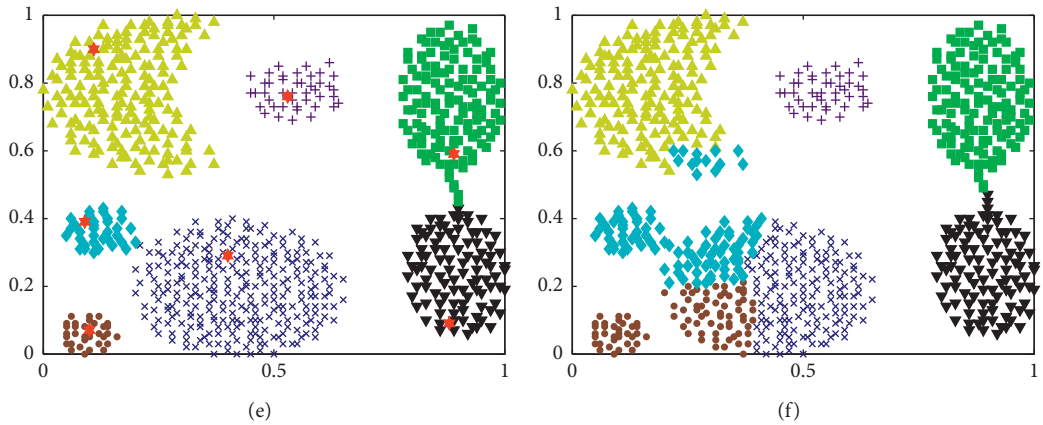


Figure 8: The clustering of Aggregation by 6 clustering algorithms. (a) DPC-SFSKNN, (b) DPC, (c) DBSCAN, (d) AP, (e) FKNN-DPC, and (f) K-means.

high-density point allocation error will be transferred to low-density points, and the error propagation will seriously affect the clustering results. AP and K-means algorithms are not good at dealing with irregular clusters. The two clusters in the middle are too attractive to the points on both sides of the semicircular cluster, which leads to clustering errors. DBSCAN can completely detect the semicircular cluster, but the semicircular cluster and the cluster on the left of the middle are incorrectly classified into one category, and the cluster on the right of the middle is divided into two clusters. The similarities between points and manually prespecified parameters may severely affect the clustering. DPC-SFSKNN and FKNN-DPC algorithms perform well on the Pathbased dataset. These improved algorithms that consider neighbor relationships have a great advantage in handling such complex distributed datasets.

Figure 7 shows the results of four algorithms on the Flame dataset. As shown in the figure, DPC-SFSKNN, DPC, FKNN-DPC, and DBSCAN can correctly detect two clusters, while AP and K-means cannot completely correct clustering. Although AP can correctly identify higher clusters and select the appropriate cluster center, the lower cluster is divided into two clusters. Both clusters of K-means are wrong. The clustering results in Figure 8 show that the DPC-SFSKNN, DPC, FKNN-DPC, and DBSCAN algorithms can detect 7 clusters in the Aggregation dataset, but AP and K-means still cannot cluster correctly. DPC-SFSKNN, DPC, and FKNN-DPC can identify clusters and centers. Although the cluster centers are not marked for DBSCAN, the number of clusters and the overall shape of each cluster are correct. The AP algorithm successfully found the correct number of clusters, but it chose two centers for one cluster, which divided the cluster into two clusters. The clustering result of K-means is similar to that of AP.

The Jain dataset shown in Figure 9 is a dataset consisting of two semicircular clusters of different densities. As shown in the figure, the DPC-SFSKNN algorithm can completely cluster two clusters with different densities. However, DPC, AP, FKNN-DPC, and K-means incorrectly assign the left end of the lower cluster to the higher cluster, and the cluster centers of the DPC are all on the lower cluster. Compared with that, the distribution of the cluster centers of the AP is more reasonable. For the DBSCAN algorithm, it can accurately identify lower clusters, but the left end of the higher cluster is incorrectly divided into a new cluster so that the higher cluster is divided into two clusters.

According to the benchmark data shown in Table 3, it is clear that the performance of DPC-SFSKNN is very effective among the six clustering algorithms, especially in the Jain dataset. Although DPC and FKNN-DPC perform better than DPC-SFSKNN on Aggregation and Flame datasets, DPC-SFSKNN can find the correct clustering center of the aggregation and can complete the clustering task correctly.

4.2. Analysis of Experimental Results on Real-World Datasets. In this section, the performance of the five algorithms is still benchmarked according to AMI, ARI, ACC, and EC/AC, and the clustering results are summarized in Table 4. 12 real-world datasets are selected to test DPC-SFSKNN's ability to identify clusters on different datasets. DBSCAN and AP algorithm cannot get effective clustering results on waveform and waveform (noise). The symbol "—" represents no result.

As shown in Table 4, in terms of benchmarks AMI, ARI, and ACC, DPC-SFSKNN outperforms all five other algorithms on the Wine, Segmentation, and Libras movement datasets. At the same time, FKNN-DPC performs better than the other five algorithms on the Iris, Seeds, Parkinsons, and WDBC datasets. It can be seen that the overall performance of DPC-SFSKNN is slightly better than DPC on 11 datasets except for Parkinsons. On the Parkinsons, DPC-SFSKNN is slightly worse than DPC in AMI but better than DPC in ARI and ACC. Similarly, DPC-SFSKNN had a slightly better performance in addition to Iris, Parkinsons, WDBC, and Seeds of eight sets of data in FKNN-DPC, and DPC-SFSKNN is slightly worse than DPC in AMI, ARI, and ACC. DBSCAN gets the best results on the Ionosphere. K-means is the best on Pima-Indians-diabetes, and K-means is the best in AMI on waveform and waveform (noise) datasets. In general, the clustering results of DPC-SFSKNN in real-world datasets are satisfactory.

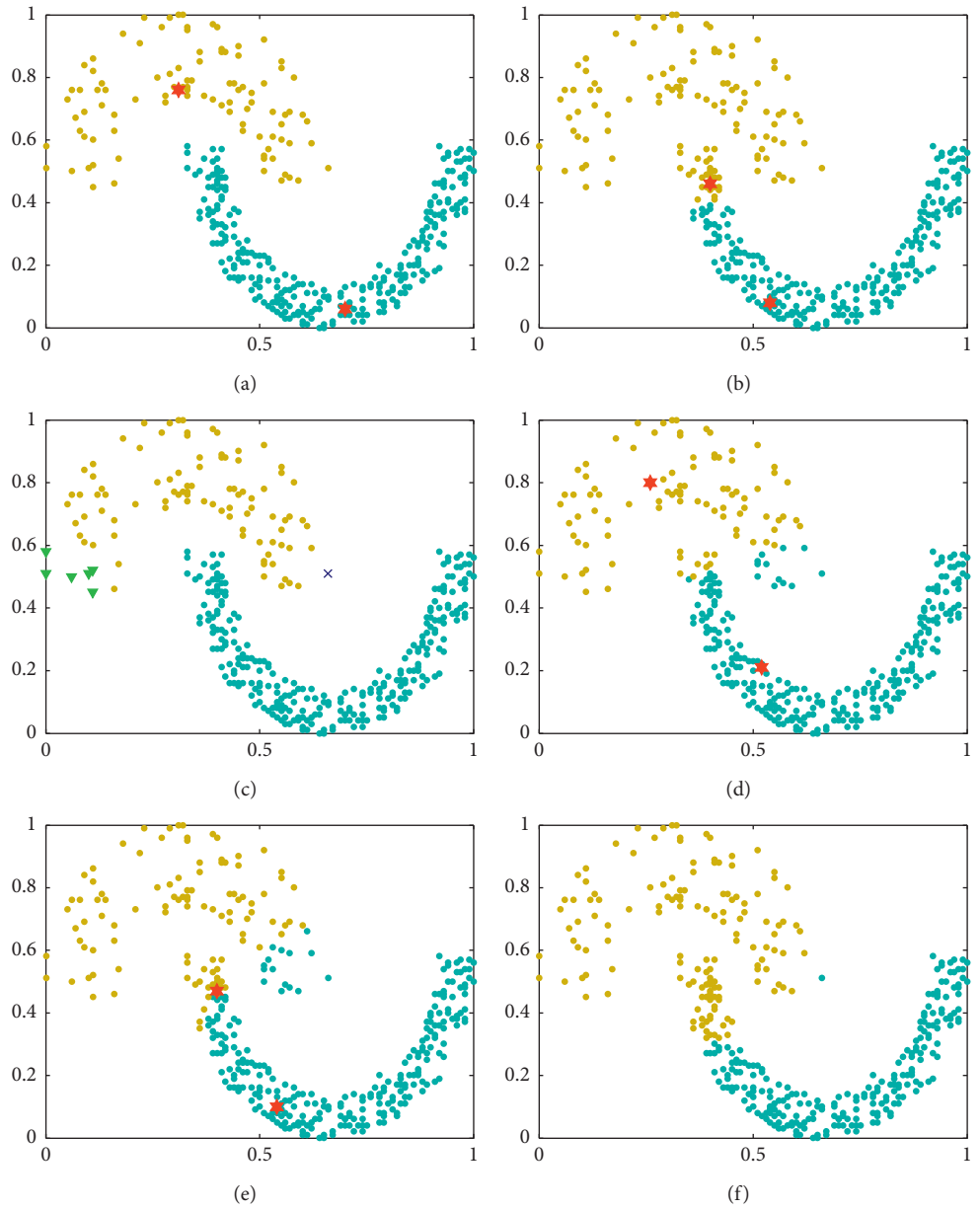
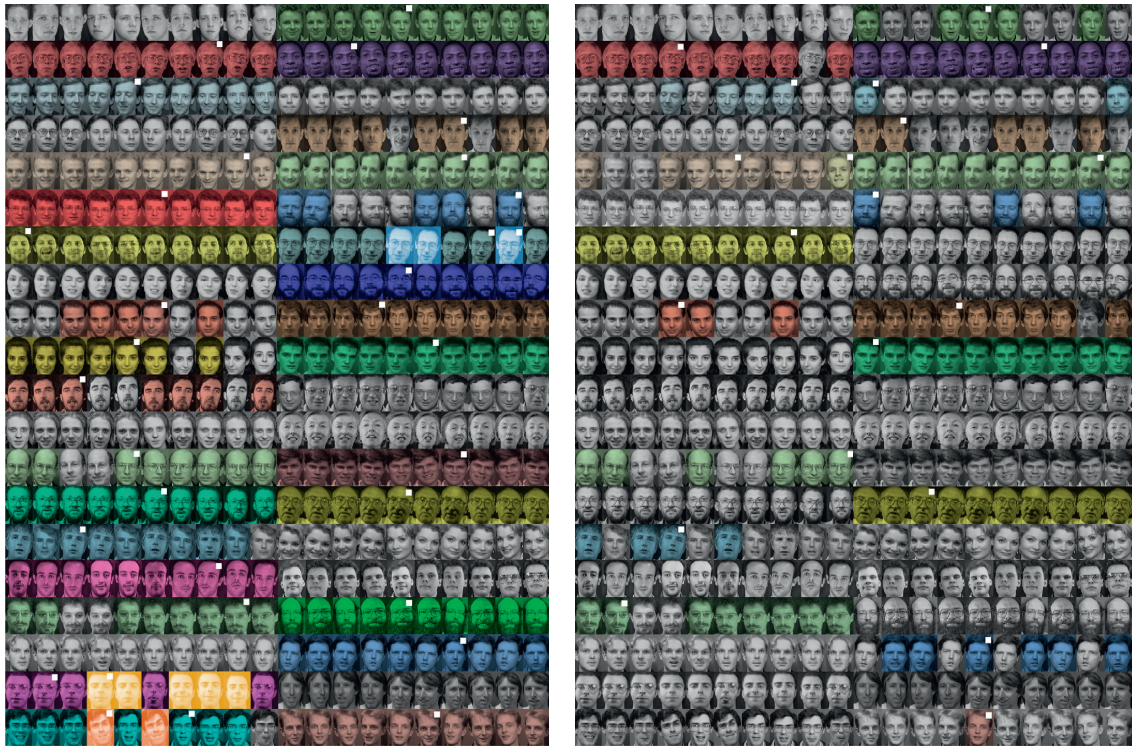


Figure 9: The clustering of Jain by 6 clustering algorithms. (a) DPC-SFSKNN, (b) DPC, (c) DBSCAN, (d) AP, (e) FKNN-DPC, and (f) K-means.

4.3. Experimental Analysis of Olivetti Face Dataset. Olivetti face dataset [28] is an image dataset widely used by machine learning algorithms. Its purpose is to test the clustering situation of the algorithm without supervision, including determining the number of clusters in the database and the members of each cluster. The dataset contains 40 clusters, each of which has 10 different images. Because the actual number of clusters (40 different clusters) is equal to the number of elements in the dataset (10 different images, each cluster), the reliability of local density becomes smaller, which is a great challenge for density-based clustering algorithms. To further test the clustering performance of DPC-SFSKNN, DPC-SFSKNN performed experiments on the Olivetti face database and compared it with DPC, AP, DBSCAN, FKNN-DPC, and K-means.

The clustering results achieved by DPC-SFSKNN and DPC for the Olivetti face database are shown in Figure 10, and white squares represent the cluster centers. The 32 clusters corresponding to DPC-SFSKNN found in Figure 10(a) and the 20 clusters found by DPC in Figure 10(b) are displayed in different colors. Gray images indicate that the image is not assigned to any cluster. It can be seen from Figure 10(a) that DPC-SFSKNN found that the 32 cluster centers were covered 29 clusters, and as shown in Figure 10(b), the 20 cluster centers found by DPC were scattered in 19 clusters. Similar to DPC-SFSKNN, DPC may divide one cluster into two clusters. Because DPC-SFSKNN can find much more density peaks than DPC, it is more likely to identify a cluster as two different clusters. The same situation occurs with the FKNN-DPC algorithm. However,



(a) (b)

Figure 10: The clustering of Olivetti by two clustering algorithms. (a) DPC-SFSKNN and (b) DPC.

Table 5: Performance comparison of algorithms by clustering criteria for the Olivetti face database.

Metric	DPC-SFSKNN	DPC	DBSCAN	AP	FKNN-DPC	K-means
ACC	0.786	0.665	0.648	0.763	0.818	0.681
AMI	0.792	0.728	0.691	0.737	0.832	0.742
ARI	0.669	0.560	0.526	0.619	0.714	0.585
EC/AC	32/40	20/40	—	28/40	36/40	—
Par	6	0.5	0.6/4	2.1	4	40

the performance of FKNN-DPC is better than that of DPC-SFSKNN in AMI, ARI, ACC, and EC/AC. From the data in Table 5, based on AMI, ARI, ACC, and EC/AC, the clustering results of these algorithms are compared. The performance of DPC-SFSKNNC is slightly superior to the performance of the other four algorithms except FKNN-DPC.

4.4. Running Time. This section shows the comparison of the time performance of DPC-SFSKNN with DPC, DBSCAN, AP, FKNN-DPC, and K-means on real-world datasets. The time complexity of DPC-SFSKNN and DPC has been analyzed in Section 3.3.1, and the time complexity of DPC is $O(n^2)$ and the time complexity of DPC-SFSKNN is $O(kn^2)$, where n is the size of the dataset. However, the time consumed by DPC mainly comes from calculating the local density and the relative distance of each point, while the time consumed by DPC-SFSKNN comes mainly from the calculation of K-nearest neighbors and the division strategy of noncenter points. Table 6 lists the running time (in seconds) of the six algorithms on the real datasets. It can be seen that although the time complexity of DPC-SFSKNN is approximately k times that of DPC, their execution time on actual datasets is not k times.

In Table 6, it can be found that on a relatively small dataset, the running time of DPC-SFSKNN is about twice or more times that of DPC, and the difference mainly comes from DPC-SFSKNN's allocation strategy. Although the computational load of the local densities for points grows very quickly with the size of a dataset, the time consumed by the allocation strategy in DPC-SFSKNN increases randomly with the distribution of a dataset. This leads to an irregular gap between the running time of DPC and DPC-SFSKNN.

FKNN-DPC has the same time and space complexity as DPC, but the running time is almost the same as DPC-SFSKNN. It takes a lot of running time to calculate the relationship between K-nearest neighbors. The time complexity of DBSCAN and AP is approximate to $O(n^2)$, and the parameter selection of both cannot be determined by simple methods. When the dataset is relatively large, it is difficult to find their optimal parameters, which may be the reason that

Table 6: Running time of 6 clustering algorithms in seconds on UCI datasets.

Dataset	DPC-SFSKNN	DPC	DBSCAN	AP	FKNN-DPC	K-means
Iris	0.241	0.049	0.059	0.565	0.148	0.014
Wine	0.238	0.048	0.098	0.832	0.168	0.013
WDBC	0.484	0.092	0.884	6.115	0.464	0.018
Seeds	0.244	0.050	0.122	0.973	0.164	0.014
Libras movement	0.602	0.068	0.309	3.016	2.602	0.075
Ionosphere	0.325	0.064	0.349	2.018	0.309	0.021
Segmentation	1.569	0.806	8.727	66.79	0.313	0.062
Dermatology	0.309	0.063	0.513	2.185	0.409	0.007
Pima-Indians-diabetes	0.792	0.126	2.018	9.709	0.892	0.009
Parkinsons	0.255	0.048	0.114	0.866	0.263	0.003
Waveform	16.071	3.511	—	—	7.775	0.067
Waveform (noise)	17.571	3.784	—	—	7.525	0.109

the two algorithms have no running results on the waveform dataset. The approximate time complexity of K-means is $O(n)$, and Table 6 proves its efficiency. K-means has almost no loss of accuracy under the premise of fast speed, which makes it a very popular clustering algorithm, but K-means is not sensitive to irregularly shaped data.

5. Conclusions and Future Work

A new clustering algorithm is proposed based on the traditional DPC algorithm in this paper. This algorithm proposes a density peak search algorithm that takes into account the surrounding neighbor information and develops a new allocation strategy to detect the true distribution of the dataset. The proposed clustering algorithm performs fast search, finds density peaks, say cluster centers of a dataset of any size, and recognizes clusters with any arbitrary shape or dimensionality. The algorithm is called DPC-SFSKNN, which means that it calculates the local density and the relative distance by using some distance information between points and neighbors to find the cluster center, and then the remaining points are assigned using similarity-first. The search algorithm is based on the weighted KNN graph to find the owner (clustering center) of the point. The DPC-SFSKNN successfully addressed several issues arising from the clustering algorithm of Alex Rodriguez and Alessandro Laio [20] including its density metric and the potential issue hidden in its assignment strategy. The performance of DPC-SFSKNN was tested on several synthetic datasets and the real-word datasets from the UCI machine learning repository and the well-known Olivetti face database. The experimental results on these datasets demonstrate that our DPC-SFSKNN is powerful in finding cluster centers and in recognizing clusters regardless of their shape and of the dimensionality of the space in which they are embedded and of the size of the datasets and is robust to outliers. It performs much better than the original algorithm DPC. However, the proposed algorithm has some limitations: the parameter K needs to be manually adjusted according to different datasets; the clustering centers still need to be manually selected by analyzing the decision graph (like the DPC algorithm); the allocation strategy improves the clustering accuracy but takes time and cost. How to improve the degree of automation and allocation efficiency of the algorithm needs further research.

Data Availability

The synthetic datasets are cited at relevant places within the text as references [23–27]. The real-world datasets are cited at relevant places within the text as references [29–34]. The Olivetti face dataset is cited at relevant places within the text as references [28].

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (6160303040 and 61433003), in part by the Yunnan Applied Basic Research Project of China (201701CF00037), and in part by the Yunnan Provincial Science and Technology Department Key Research Program (Engineering) (2018BA070).

Supplementary Materials

It includes the datasets used in the experiments in this paper. (*Supplementary Materials*)

References

[1] K. L. Liu, Y. L. Shang, Q. Ouyang, and W. D. Widanage, "A data-driven approach with uncertainty quantification for predicting future capacities and remaining useful life of lithium-ion battery," *IEEE Transactions on Industrial Electronics*, p. 1, 2020.

[2] X. P. Tang, K. L. Liu, X. Wang et al., "Model migration neural network for predicting battery aging trajectories," *IEEE Transactions on Transportation Electrification*, vol. 6, no. 2, pp. 363–374, 2020.

[3] X. Tang, K. Liu, X. Wang, B. Liu, F. Gao, and W. D. Widanage, "Real-time aging trajectory prediction using a base model-oriented gradient-correction particle filter for Lithium-ion

batteries," *Journal of Power Sources*, vol. 440, Article ID 227118, 2019.

[4] K. Liu, Y. Li, X. Hu, M. Lucu, and W. D. Widanage, "Gaussian process regression with automatic relevance determination kernel for calendar aging prediction of lithium-ion batteries," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 3767–3777, 2020.

[5] K. Liu, X. Hu, Z. Wei, Y. Li, and Y. Jiang, "Modified Gaussian process regression models for cyclic capacity prediction of lithium-ion batteries," *IEEE Transactions on Transportation Electrification*, vol. 5, no. 4, pp. 1225–1236, 2019.

[6] L. Cai, J. Meng, D.-I. Stroe, G. Luo, and R. Teodorescu, "An evolutionary framework for lithium-ion battery state of health estimation," *Journal of Power Sources*, vol. 412, pp. 615–622, 2019.

[7] L. Cai, J. H. Meng, D. I. Stroe et al., "Multi-objective optimization of data-driven model for lithium-ion battery SOH estimation with short-term feature," *IEEE Transactions on Power Electronics*, p. 1, 2020.

[8] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[9] J. Xie, H. Gao, W. Xie, X. Liu, and P. W. Grant, "Robust clustering by detecting density peaks and assigning points based on fuzzy weighted K-nearest neighbors," *Information Sciences*, vol. 354, pp. 19–40, 2016.

[10] F. S. Samaria and A. C. Harter, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Berkeley Symposium On Mathematical Statistics and Probability*, pp. 281–297, Berkeley, CA, USA, 1967.

[11] S. Kant, T. L. Rao, and P. N. Sundaram, "An automatic and stable clustering algorithm," *Pattern Recognition Letters*, vol. 15, no. 6, pp. 543–549, 1994.

[12] D. Arthur and S. Vassilvitskii, "K-Means++: the advantages of careful seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 7–9, New Orleans, LA, USA, 2007.

[13] Y. Zhao, W. Halang, and X. Wang, "Rough ontology mapping in E-business integration," *E-Service Intelligence, BMC Bioinf*, vol. 8, pp. 75–93, 2007.

[14] Y. Xiao and J. Yu, "Semi-supervised clustering based on affinity propagaiton algorithm," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, 2007.

[15] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference On Knowledge Discovery and Data Mining*, pp. 226–231, Portland, OR, USA, 1996.

[16] R. J. G. B. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," *Advances in Knowledge Discovery and Data Mining*, vol. 7819, pp. 160–172, 2013.

[17] Z. Liang and P. Chen, "Delta-density based clustering with a divide-and-conquer strategy: 3DC clustering," *Pattern Recognition Letters*, vol. 73, pp. 52–59, 2016.

[18] M. Ankerst, M. M. Breuning, H. P. Kriegel, and J. Sander, "OPTICS: ordering points to identify the clustering structure," in *Proceedings of the 1999 ACM SIGMOD-International Conference on Management of Data*, pp. 49–60, Philadelphia, PA, USA, 1999.

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

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

[21] T. Li, H. W. Ge, and S. Z. Su, "Density peaks clustering by automatic determination of cluster centers," *Journal of Computer Science and Technology*, vol. 10, no. 11, pp. 1614–1622, 2016.

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

[23] R. A. Jarvis and E. A. Patrick, "Clustering using a similarity measure based on shared near neighbors," *IEEE Transactions on Computers*, vol. C-22, no. 11, pp. 1025–1034, 1973.

[24] H. Chang and D.-Y. Yeung, "Robust path-based spectral clustering," *Pattern Recognition*, vol. 41, no. 1, pp. 191–203, 2008.

[25] L. Fu and E. Medico, "Flame, a novel fuzzy clustering method for the analysis of DNA microarray data," *BMC Bioinformatics*, vol. 8, no. 1, 2007.

[26] A. Gionis, H. Mannila, and P. Tsaparas, "Clustering aggregation," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, p. 4, 2007.

[27] P. Franti, O. Virmajoki, and V. Hautamaki, "Fast agglomerative clustering using a k-nearest neighbor graph," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 11, pp. 1875–1881, 2006.

[28] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in *Proceedings of the 1994 IEEE Workshop On Applications Of Computer Vision*, pp. 138–142, Sarasota, FL, USA, 1994.

[29] K. Bache and M. Lichman, *UCI Machine Learning Repository*, http://archive.ics.uci.edu/ml, 2013.

[30] M. Charytanowicz, J. Niewczas, P. Kulczycki, P. A. Kowalski, S. Lukasik, and S. Zak, "Complete gradient clustering algorithm for features analysis of X-ray images," *Information Technologies in biomedicine. Advances in Intelligent and Soft Computing*, vol. 69, Berlin, Germany, Springer.

[31] D. B. Dias, R. C. B. Madeo, T. Rocha, H. H. Biscaro, and S. M. Peres, "Hand movement recognition for brazilian sign language: a study using distance-based neural networks," in *Proceedings of the 2009 International Joint Conference on Neural Networks*, pp. 697–704, Atlanta, GA, USA, 2009.

[32] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*, Routledge, New York, NY, USA, 1st edition, 1984.

[33] V. G. Sigillito, S. P. Wing, L. V. Hutton, and K. B. Baker, "Classification of radar returns from the ionosphere using neural networks," *Johns Hopkins APL*, vol. 10, no. 3, pp. 262–266, 1989.

[34] W. N. Street, W. H. Wolberg, and O. L. Mangasarian, "Nuclear feature extraction for breast tumor diagnosis," in *Proceedings of the SPIE 1905, Biomedical Image Processing and Biomedical Visualization*, San Jose, CA, USA, 1993.

[35] X. V. Nguyen, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: is a correction for chance necessary?" in *Proceedings of the ICML 2009, the 26th Annual International Conference On Machine Learning*, San Montreal, Canada, 2009.

[36] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques, the Morgan Kaufmann Series in Data Management Systems*, Morgan Kaufmann, Burlington, MA, USA, 3rd edition, 2011.