IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 21, 2020, accepted June 10, 2020, date of publication June 17, 2020, date of current version June 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3003057

Adaptive Density Peaks Clustering Based on K-Nearest Neighbor and Gini Coefficient

DONG JIANG, WENKE ZANG, RUI SUN, ZEHUA WANG, AND XIYU LIU, (Member, IEEE)
School of Business, Shandong Normal University, Jinan 250014, China

Corresponding author: Wenke Zang (wink@sdnu.edu.cn)

This work was supported in part by the National Science Foundation of China under Grant 61876101.

ABSTRACT Density Peaks Clustering (DPC) is a density-based clustering algorithm that has the advantage of not requiring clustering parameters and detecting non-spherical clusters. The density peaks algorithm obtains the actual cluster center by inputting the cutoff distance and manually selecting the cluster center. Thus, the clustering center point is not selected on the basis of considering the whole data set. This paper proposes a method called G-KNN-DPC to calculate the cutoff distance based on the Gini coefficient and K-nearest neighbor. G-KNN-DPC first finds the optimal cutoff distance with Gini coefficient, and then the center point with the K-nearest neighbor. The automatic clustering center method can not only avoid the error that a cluster detects two center points but also effectively solve the traditional DPC algorithm defect that cannot handle complex data sets. Compared with DPC, Fuzzy C-Means, K-means, KDPC and DBSCAN, the proposed algorithm creates better clusters on different data sets.

INDEX TERMS Density peaks clustering, adaptive algorithm, K-nearest neighbor, Gini coefficient.

I. INTRODUCTION

Big data has been rapidly and widely used in the fields of physics, biological engineering, life medicine etc [1]. Exploring massive amounts of information and obtaining useful information about future actions is one of the most important applications of big data. Due to a large amount of diverse data, clustering algorithms for data processing and generalization research are imminent. Clustering is based on the similarity between data objects divide the sample of the data set into such clusters reasonable, clustering result objects within the same cluster have high similarity, but the similarity is low between clusters. The object of clustering is one of the most important objects for understanding the world. People who use clustering can discover knowledge from data and reveal hidden patterns and rules. Therefore, it is widely used in the fields of scientific data analysis and systems engineering. There is an ever-increasing interest in clustering algorithm that can automatically understand, process, and summarize the data.

The clustering algorithm includes partitioned clustering method, hierarchical clustering method, density-based clustering method, grid-based clustering method, and integrated clustering algorithm [2]–[4]. K-means is the most widely used partitioned clustering algorithm [4]. However, the clustering results of the K-means algorithm is heavily dependent on the center of the initial cluster, it is difficult to find non-convex clusters, which are sensitive to noise points and outliers, and the number of clusters needs to be set in advance. The fuzzy C-means clustering algorithm (FCM) has a large amount of calculation, because it has to calculate its distance to all known samples for each sample to be classified, in order to obtain its K-nearest neighbors.

Alex Rodriguez published a novel density peaks clustering algorithm (DPC) in the journal Science in 2014 [5]. Different from other clustering algorithms, this algorithm uses local density and distance to characterize the spatial distribution of each data point, and obtains the clustering center with a decision graph. Since the DPC algorithm has the advantages of a novel idea, easy implementation, and high clustering efficiency, it has been widely recognized in various fields.

Hou analyzed the influence of the kernel density estimation method in DPC algorithm on the clustering results, and used the K-nearest neighbor idea to redefine the local density, and designed a new cluster center selection by using the distance normalization principle [6]. Xie uses the local standard deviation of each data point to define its local density, which improves the accuracy of the algorithm to select the cluster center point from the sparse data [7]. The 3DC algorithm uses the recursive method to find the optimal number of clusters in

The associate editor coordinating the review of this manuscript and approving it for publication was Hong-Mei Zhang.

the original data [8]. Rashid Mehmood proposed the Fuzzy-CFSFDP algorithm by introducing fuzzy rules [9]. Wang proposed a method for obtaining the cutoff threshold d_c from the original data set by using the potential entropy of the data segment [10]. Mehmood proposed a CFSFDP HD algorithm based on the thermal diffusion theory [11]. Gao designed a method for calculating the parameter d_c using the standard deviation of each attribute of the original data [12]. Xu proposed a manifold density peaks clustering algorithm, which uses geodesic distance to calculate the manifold distance between each data point and input cluster by pre-input. The numbers allow the algorithm to automatically complete the clustering, and also introduces an equidistant mapping to map high-dimensional data sets to lower dimensions for dimensionality reduction [13].

K-nearest neighbor (KNN) is a classification algorithm, which is simple and efficient. It can not only deal with text and stream data classification problems [14], but also shows very well in clustering and strong skill, so this method is constantly introduced into the DPC algorithm. Liu proposed an adaptive density peaks clustering algorithm, which calculates the parameter by introducing KNN [15]. Du proposed DPC-KNN. The algorithm uses the KNN idea to estimate the density of each point and uses principal component analysis to reduce the dimensionality of the data, improving the processing ability of high-dimensional data and obtaining a good clustering effect [16]. Huang proposed QCC, which determines a cluster center with K nearest neighborhood or reverse K nearest neighborhood and defines a novel concept of similarity between clusters to solve the complex-manifold problem [17]. However, since the clustering process of G-DPC-KNN algorithm is the same as DPC, the defect of the DPC algorithm still exists in this algorithm. Although the quality of the classification is improved, the model is more complicated.

Based on many improvements for the density peaks clustering algorithm [18]–[26] and outliers detection strategy [27], [28], we propose a method to calculate the cut-off distance based on the Gini coefficient and find center points by KNN. The rest of this paper is organized as follows: In Sec.2, we describe the principle of the DPC and its analysis. In Sec.3, we make a detailed description of a new algorithm. In Sec.4, we present experimental results in synthetic data sets and UCI data sets compared with several well-known algorithms including DPC, KDPC, FCM, and K-means. And we test the adaptive of parameter K. Finally, we derive the conclusions given in the last section along with the expected future works.

II. RELATED WORK

The density clustering algorithm is easy to identify clusters of arbitrary shapes. The basic idea of this kind of algorithm is to treat data in high-density areas as the same cluster. Typical algorithms include Density-Based Spatial Clustering of Applications with Noise(DBSCAN) [29] and Ordering points to identify the clustering structure (OPTICS) [30]. Thereinto, there is no need to know in advance the number of cluster classes for the DBSCAN algorithm proposed by MMihael that overcomes the sensitivity of inputting parameters.

The basic idea of density peaks cluster is that cluster centers are surrounded by low-density points, and their distance from any point with a higher density is larger [5]. The DPC algorithm effectively finds the cluster center and assigns the remaining points to the appropriate clusters. Because the algorithm is simple in designed process and excellent in clustering performance, it has been applied in many domains.

The core idea of the clustering algorithm is to characterize the clustering center. Rodriguez and Laio believed that the clustering centers have the following two characteristics:

(1) They have a high density, that is, it was surrounded by neighbor points with a lower density;

(2) They are relatively far away from data points with higher density than them.

There are two main quantities to be calculated for density peaks clustering: local density ρ_i and the distance ξ_i between i and high-density points. The definitions of local density and relative distance are as follows: [5], local density ρ can be calculated in two ways:

Cut-off kernel:

Local density of sample point i:

$$\rho_i = \sum_j \chi\left(d_{ij} - d_c\right) \tag{1}$$

Here

$$\chi\left(x\right) = \begin{cases} 1 & \text{if x} < 0 \\ 0 & \text{if other} \end{cases}$$

Gaussian kernel:

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

The relative distance from the sample point i:

$$\xi_i = \begin{cases} \min\limits_{j \in I_S^i}\left(d_{ij}\right) & I_S^i \neq \phi \\ \max\limits_{j \in I_S^i}\left(d_{ij}\right) & I_S^i = \phi \end{cases} \tag{3}$$

where $I_S^i = \{k \in I : \rho_k > \rho_i\}$ when $\rho_i = \max\limits_{j \in I_S^i}\left(d_{ij}\right)$, $I_S^i = \phi$.

As for how DPC search and find the cluster center point. The main idea is to plot the decision graph and choose points which have higher ξ_i and ρ_i. The premise is to collect ξ_i and ρ_i for every point. The clustering centers can then be detected by analyzing the decision graph. The points of high ξ_i and relatively high ρ_i can be chosen as cluster centers. After finding the cluster centers, assign each remaining point to the same cluster as its nearest higher density neighbor. DPC performs efficiently because the allocation strategy is performed in a single step.

DPC defines the boundary area of each cluster. This is the set of points assigned to the cluster. The distance between these points and the points belonging to other clusters is d_c. Then, find the highest density point in its boundary area for each cluster and express its density by ρ_i. Cluster points with a density higher than ρ_i are considered as the core of the cluster and other points are cluster halo, which is understood as noise.

From the results of testing DPC on several different data sets, it is clear that DPC can perform well in many instances, though the following drawbacks are apparent: [31]–[40]

i) After the distance matrix is constructed, two important variables ξ_i and ρ_i need to be solved, and the solution of these two parameters is closely related to the only parameter d_c of this algorithm. The selection of the distance threshold is directly related to the quality of the clustering effect, especially on some small and medium-sized data sets, the density peaks clustering is sensitive to d_c anomaly.

ii) In the same cluster, two or more points with large density and distance often appear. When manually selecting the cluster center point in the decision graph, it is impossible to determine whether the two points are in the same cluster. The center point will likely split a cluster into two. And the algorithm allows the user to select the cluster center by providing a decision map which is subjectivity.

iii) After finding the clustering center, assigning the remaining points to the class of the nearest neighbor with higher density may bring error propagation, that is, once the points are misallocated, it may cause several points that are lower in density than the point to be misclassified, resulting in poor clustering effects.

Many scholars have optimized and improved the DPC algorithm from different angles. To remedy some limitations in DPC, we propose an algorithm named G-KNN-DPC. G-KNN-DPC first finds the optimal cutoff distance with the Gini coefficient and finds the center points with the K-nearest neighbors. Gini coefficient is calculated based on the potential energy of every data point in the data field, then get the threshold value d_c. The algorithm proposed was tested on several synthetic and real-world data sets which are often used to test the performance of a clustering algorithm. The results demonstrate that G-KNN-DPC can consider the true distribution of a data set and has better performance.

III. PROPOSED ALGORITHM

There are still some defects in DPC, including some of its improved algorithms. To solve these problems, we will improve DPC in three aspects by calculating cutoff distance with Gini coefficient, choosing a new way to select initial cluster centers with the K-nearest neighbors, and aggregating clusters if they meet a criterion. In this section, we will give the details of the algorithm we proposed and analyze its complexity theoretically.

A. CALCULATE THE CUTOFF DISTANCE BASED ON GINI COEFFICIENT

In the DPC algorithm, the selection of cutoff distance d_c is the key to compute density. It is found that cutoff distance d_c has an important influence on the result of clustering. If d_c is too large, the density value of each data point will be approximately equal, resulting in that the partition of all data points will be into the same class cluster. If d_c is too small, each class cluster will contain few sample points, and one cluster will likely be divided into several parts. Moreover, one threshold can't fit all data sets because of the different density of data points in different data sets. In order to cover the shortage of the DPC algorithm, it is necessary to find an index to describe the overall situation of the different data sets. So, in this paper, the Gini coefficient in which all the data points are considered is proposed to calculate the optimal cutoff distance.

Objects distributed in certain areas are often interrelated and interact with each other. There is radiation restriction between different objects. At the beginning of the 19th century, the physicist Ferrari proposed the term 'field' to describe the interaction between objects. In the data field, each object is affected by all other objects, and each is affected by other objects. The data field can be regarded as a space full of data energy, and the data emits energy to other data in the field through its own data field, thereby generating effects.

A potential field is usually described by a potential function. Potential functions vary with location and can be superimposed. The law of spatial distribution is usually depicted by equipotential lines. In the spatial domain, each data object contributes to the potential energy of any point, and the size of the contribution is inversely proportional to the square of the distance between them. The potential energy is stronger in data-intensive places. Otherwise, it will be relatively weak.

In the data set, the points with higher potential energy are in the dense region, while those with lower potential energy are in the sparse region. Given that, the potential energy in the data domain is similar to the local density of the points in the data set. Therefore, the potential energy of each point is used as an indicator of the overall distribution of the data set to estimate the potential energy of the whole data set.

For data set $\{x_1, x_2, \cdots, x_n\}$, the formula for the potential energy at each point is

$$\delta_i = \sum_{\mathrm{j}=1}^{n} \left(e^{-\left(\frac{\|x_{\mathrm{j}} - x_i\|}{\sigma}\right)^2} \right) \tag{4}$$

σ is the impact factor that controls the interaction between objects. The value of σ will directly affect the density degree. The smaller the value σ is, the more compact the network will be. On the contrary, if the parameter is large, the cluster structure will be relatively sparse. This characteristic is the same as that of d_c. If the points in the data set are dense, we will also choose a smaller d_c according to the distance between objects. If the points in the data set are sparse,

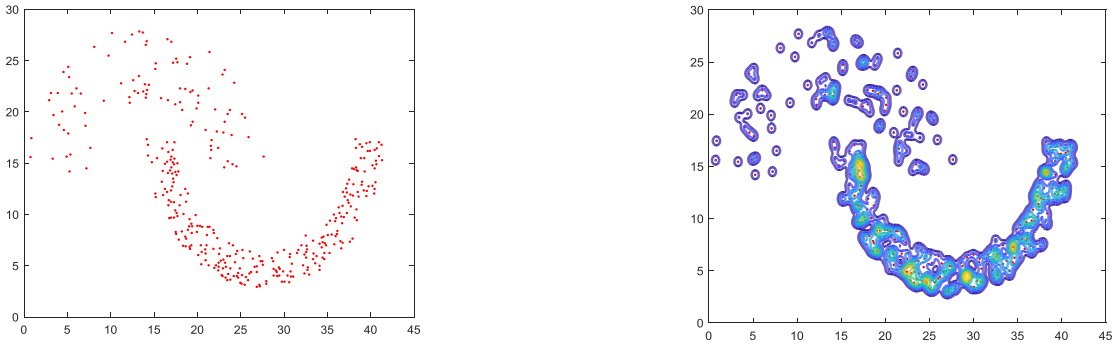

FIGURE 1. The original graph of Jain data set and the potential energy of Jain data set.

the threshold we choose will be larger. It is found that the method of calculating the potential energy of the point is similar to that of calculating the local density in the DPC algorithm when comparing Eq. (4) and Eq. (2).

In the data domain, if the data is evenly distributed, the uncertainty of the data distribution will be greater and the impurity of the data will be greater as well. If the data is not evenly distributed, the uncertainty of the data distribution will be small and as well as the impurity of the data. The impurity of the data can be described by the Gini coefficient. The calculation formula of the Gini coefficient in the data domain is as follows:

$$G = 1 - \sum_{i=1}^{n} \left(\frac{\delta_i}{Z} \right)^2 \tag{5}$$

$$Z = \sum_{i=1}^{n} \delta_i \tag{6}$$

Z is the total potential energy of all data points in the data domain, δ_i is the potential energy of each point. Combined with Eq. (4) and Eq. (5), when the influence factor σ increases gradually from 0, it changes with the change of Gini index G. When the Gini index value G takes the minimum value, it reflects that the data has the smallest impurity, the least uncertainty, the uneven potential energy distribution of the data, the more differences of the data potential energy, and the easier clustering, which is the ideal result. Combining the similarity of d_c and σ with the characteristics of Gini coefficient, the value of the optimal impact factor is obtained when the Gini index value is the smallest. The optimal value σ is used as the value of truncation distance d_c, so as to achieve the effect of adaptive truncation distance, $d_c = \sigma$.

As it is shown in Figure 1, the moon-shaped cluster below is denser than the one above, so the closed loops in yellow and green which represent higher potential appear more frequently than above one. Take Jain data set as an example, σ gain optional value which is 9. So, we take $d_c = \sigma = 9$. As we can see from Figure 2, it shows the relationship curve of Gini index G and impact factor σ in Jain dataset. With σ going up to infinity, G is going down first and σ reaches the

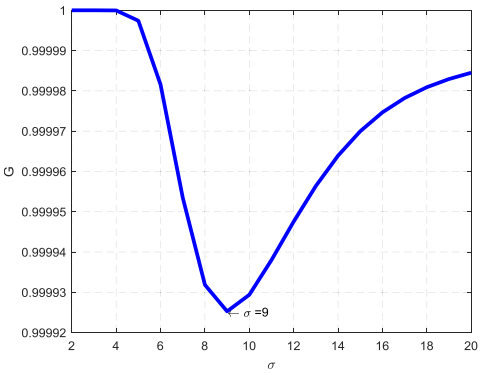

FIGURE 2. Change of influence factor σ of data set Jain.

optimal value 9, then G is going up. Experiments on Jain data set are described in detail in the experiments section.

B. FIND THE CENTER POINTS BY USING K-NEAREST NEIGHBOR

There are two key points to find center points by using the DPC algorithm. They are density and distance. In this new algorithm, the formula of density for each point is the same as DPC shown in Eq.(1). We compute distance based on K-nearest neighbors. K represents the number of the nearest neighbors. The set of K-nearest neighbors of a certain point is defined by Eq. (7)

$$NN_i = \left\{ j | \min_k \left(d_{ij} \right), i \neq j \right\} \tag{7}$$

The set of point i and its K-nearest neighbors is defined by Eq. (8)

$$O_i = \{NN_i, i\} \tag{8}$$

The set of the points which density is higher than i is defined by Eq. (9)

$$HP_i = \left\{ j | \rho_j > \rho_i, j \neq i \right\} \tag{9}$$

Some of the points have overlapping parts by comparing its NN_i and HP_i. This leads to inaccurate distance calculations. So we delete the overlapping parts in HP_i.

The distance of each point is defined by Eq. (10)

$$\xi_i = \min\{distance(HP_i, O_i)\}\, HP_i \cap O_i = \varnothing \tag{10}$$

The distance of G-DPC-KNN find distance between K-nearest neighbors of a certain point and those points whose density is denser than it.

The local density of point is given by Eq. (1). Then calculate $\rho_i \times \xi_i$ for each point, and sort them in descending order. Select n points in the upper order as the center points.

C. ASSIGN REMAIN POINTS TO DIFFERENT CLUSTER

Cluster centers are selected by order $\rho_i \times \xi_i$, and the remaining points are assigned to each cluster. The remaining point i is absorbed in the set which points with higher density and smaller distance from i belongs to. So, we assign points with higher density at the beginning. To prevent the interference of outliers, we checked each point before assignment. According to Eq. (11), if $N_i > 1$ for a certain point, it does not be assigned. The allocation process continues to iterate.

$$N_i = Count(\{j|\, d_{ij} < d_c\}) \tag{11}$$

j represents other points in the data set.

The steps of the algorithm are as follow:

Algorithm 1 Adaptive Density Peaks Clustering Based on K-Nearest Neighbor and Gini Coefficient

Input: Initial points, K, n

Output: The label vector of cluster index

Step 1: Calculate d_c

1. Calculate d_{ij} according to Initial points;
2. Determine d_c according to Eq. (4).

Step 2: Adaptive find cluster center

1. Calculate ρ_i based on Eq.(1);
2. Calculate ξ_i based on Eq. (10);
3. Compute $\rho_i \times \xi_i$, then sort them in descending order;
4. Find cluster center.

Step 3: Assign remain points to different cluster

1. Point i is absorbed to clusters and assigned;
2. Iterate until all points are assigned.

D. COMPLEXITY ANALYSIS

Supposed that the data set has n points and let CN denote the number of clusters. The space complexity of DPC is $O\left(n^2\right)$, where n is the size of the data set, which is mainly due to storing the distance matrix. In the process of G-KNN-DPC, there are three objects need storing spaces: First, the algorithm needs space to store the distance from each point to its K-nearest neighbors, it is K*N entries. Second, each point has two attributes as ρ and ξ, which needs 2N spaces. The space complexity of G-KNN-DPC does not increase more than $O\left(n^2\right)$ because the extra space required by our algorithm does not exceed $O\left(Cn^2\right)$. So, the space complexity of our algorithm is the same as DPC in [24].

The time complexity of DPC depends on the following three aspects: (a) the time for computing the distance between points; (b) the time to calculate the local density ρ for point i, and (c) the time used to compute the distance ξ for each point i. The time complexity of each is $O\left(n^2\right)$, so the total approximate time complexity of DPC is $O\left(n^2\right)$.

The time complexity of G-KNN-DPC depends on the following four parts: (a) compute the distance between points $O\left(n^2\right)$; (b) evaluate the local density for each point. To compute this we need to search the K-nearest neighbors of point i, whose time complexity is $O(n)$, so the time complexity for searching K-nearest neighbors for n points is $O\left(n^2\right)$. As a result the time complexity to calculate the local density for each point is of order $O\left(n^2\right)$; (c)evaluate the distance ξ of point i, which has time complexity $O\left(n^2\right)$; (d) assign points to their most appropriate clusters with time complexity also $O\left(n^2\right)$.

Therefore, the overall time complexity of G-KNN-DPC is $O\left(n^2\right)$ which is the same as DPC.

IV. EXPERIMENTS AND ANALYSIS

In this section, experimental results will show the superior performance of G-KNN-DPC. To achieve this, we use two kinds of data sets (1) six synthetic data sets [41]–[43] and (2) seven real-world data sets: the banknotes data set, the bcw data set, the seeds data set. the vertebral data set, the iris data set, the wholesale data set, and the wifilocalization data set. And all of the real-world data sets are obtained from the UCI repository. These data sets are chosen to test the ability of our algorithm in identifying clusters having arbitrary shapes without being affected by noise, size, or dimensions of the data sets. The numbers of features (dimensions), data points (instances), and clusters vary in each of the data sets. The details of the synthetic and real-world data sets are listed in Tables 1.

We compare the G-KNN-DPC with improved DPC algorithm, DPC algorithm based on kernel (KDPC) and the classical clustering algorithms, such as, K-means [20], Fuzzy C-means (FCM), Density-Based Spatial Clustering of Applications with Noise (DBSCAN). The code for FCM, K-means and DBSCAN are provided by their authors. The code of improved DPC is optimized based on the original code provided by Rodriguez and Laio. The code of DPC is optimized by using the kernel(KDPC) based on the original code [44]. Three popular criteria clustering accuracy (ACC), normalized mutual, information (NMI), rand index (RI) are used to evaluate the performances of the above clustering algorithms. Each benchmark value ranged from 0 to 1.0, and the larger it is, the better is the clustering. We only keep the valid data after the three decimal places, except the NMI parameters of the Wifilocalization data set, because it could not reflect the difference of data. We conduct experiments

TABLE 1. The details for thirteen data sets.

Data sets	Instances	Dimensions	Clusters
Aggregation	788	2	7
Spiral	312	2	3
D15	600	2	15
Jain	373	2	2
Four-lines	512	2	4
A3	266	2	3
Banknotes	1372	4	2
Bcw	699	9	2
Seeds	210	7	3
Vertebral	310	7	2
Iris	150	4	3
Wholesale	440	6	3
Wifilocalization	2000	7	4

on a desktop computer with a Core i7 DMI2-Intel 3.6 GHz processor and 16 GB RAM running MATLAB 2017A.

A. DECISION GRAPHS COMPARATIVE ANALYSIS

To evaluate the performance of the G-KNN-DPC method, we used the Spiral dataset as an example. In the Spiral dataset, some clusters are the composition of different densities. **a** is pictured according to Eq. (1) and Eq. (10). All DPC algorithm tends to find the largest dense point in each density and use it as the clustering center of the decision graph, as shown in Figure 4. To select the exact number of cluster centers from decision graph, humans should have the domain knowledge of a certain dataset. G-KNN-DPC will automatically select the clustering center point, before which some interference needs to be removed. As shown in Figure 4, we see that there is little interference point when selecting the center point in **a**, while there are many interference points in **b.** In particular, it eliminates the interference of some points like the center points.

B. EVALUATION ON METRICS

In this paper, three clustering metrics are used to measure the performance of our algorithm: ACC [45], and NMI [46]. RI [47].

The ACC is calculated as:

$$ACC = \frac{\sum_{k=1}^{n} \delta(l_k, map(g_k))}{n} \tag{12}$$

where n represents the number of data samples, g_k and l_k represent the cluster label obtained by our algorithm and the true classification label of the data sample $\mathbf{x}_k$, respectively. $map(\bullet)$ is a mapping function that maps cluster labels obtained by our algorithm to the true classification labels. When $l_k = map(g_k)$, the function value of $\delta(l_k = map(g_k))$ is 1, otherwise it is 0.

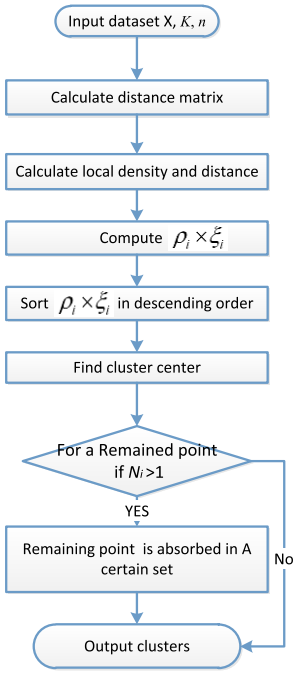

FIGURE 3. Flowchart for G-KNN-DPC.

The *NMI* is defined by

$$NMI = \frac{\sum_{i=1}^{c}\sum_{j=1}^{c} n_{ij} \log\left(\frac{n \cdot n_{ij}}{n_i \cdot n'_j}\right)}{\sqrt{\left(\sum_{i=1}^{c} n_i \log\left(\frac{n_i}{n}\right)\right)\left(\sum_{j=1}^{c} n'_j \log\left(\frac{n'_j}{n}\right)\right)}} \tag{13}$$

where c represents the number of clusters, n represents the number of samples in the data set, n_{ij} represents the number of data samples belonging to the *i-th* cluster in the label set obtained by the algorithm and belonging to the *j-th* cluster in the true label set, simultaneously. n_i is the number of data

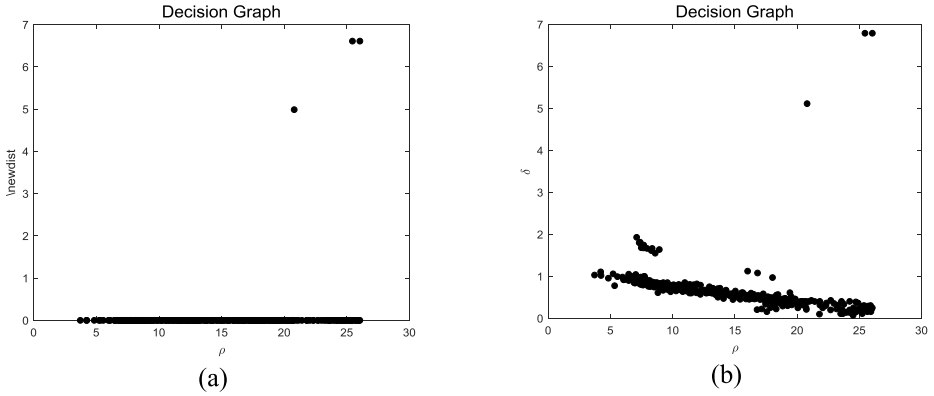

FIGURE 4. Decision graph created by G-KNN-DPC and DPC on Jain dataset. a Decision graph created by G-KNN-DPC. b Decision graph created by DPC.

samples belonging to the *i-th* cluster center obtained by the algorithm, and n_j' is the number of data samples belonging to the *j-th* cluster in the real case. *NMI* effectively measures the statistical information between the clustering result distribution obtained by the algorithm and the actual classification label distribution. The range of *NMI* values is [0, 1]. Generally speaking, the larger the *NMI* value, the better the clustering result.

Supposed g represents the actual category information, l represents the clustering result. *RI* is the rand index which is defined as:

$$RI = \frac{a+b}{C_2^n} \tag{14}$$

where a represents the number of pairs of points that are in the same cluster in g and l, and b represents the number of pairs of points in different clusters in g and l, n represents the number of samples in the data set, C_2^n indicates the number of pairs that can be composed in the data set. The value range of *RI* is [0,1], the larger the value, the more consistent the clustering result with the real situation.

C. ADAPTABILITY OF G-KNN-DPC

In the Eq. (7), the parameter K represents the nearest neighbors of the object. The choice of K will affect the distance calculated and thus the choice of the center point. In order to make the experiment more credible, we will select four different K to verify the adaptability of the algorithm in this part.

As shown in Figure 5, (a) and (b) are Four-lines graphs. The abscissa is different k, and the ordinate is the clustering accuracy (ACC) obtained from the results of the algorithm working on different data sets. (a) is the clustering results of different K on the artificial data set, and (b) is the clustering results of different K on the UCI data set. We found that G-KNN-DPC performed well on the synthetic data set. Except for the Spiral data set, the results of other data sets are stable, and all of them are the best results or close to the best results. Spiral has 3 clusters which embedded each other. KNN looks for its neighborhood in terms of the circle, so when K is bigger (such as $K = 17$), points with higher density belonging to other data sets are also included in the K-nearest neighbors. And in the allocation process, it is allocated to the class with the highest density points in the K-nearest neighbors. As a result, the clustering results obtained are not accurate. G-KNN-DPC working on UCI data sets is also more superior than other data set. As shown in (b), when G-KNN-DPC works on vertebral, seeds and bcw data set, the performance is stable. When $K = 7$ or $K = 10$, the performance of this algorithm on seven UCI data sets is very stable, and it is more superior when compared with other algorithms. The results show that G-KNN-DPC has strong robustness and superiority when K set in an appropriate range.

D. EXPERIMENTS ON SYNTHETIC DATA SETS AND RESULTS ANALYSIS

In this subsection, we show the performance of G-KNN-DPC, improved DPC, FCM, K-means, KDPC and DBSCAN on six synthetic data sets listed in Tables 1. The results of each algorithm on six synthetic data sets are displayed embedded in two-dimensional space as different marked and colored shapes, just as Figure 6-11 shown. The performance of each algorithm is benchmarked in terms of ARI, NMI, RI shown in Table 2, Table 3, and Table 4. And the best results are shown in boldface.

The Four-lines data set has 4 clusters of the shape of a line, they are all separated. Figure 6 shows the algorithm proposed, improved DPC and DBSCAN can both find cluster centers and the correct clusters. In Table 2, Table 3, and Table 4, the benchmarks data of the algorithm proposed are all almost 1.00 exactly and their values of G-KNN-DPC are also showed as 1.00 for data rounding. As shown in Figure 6, FCM, K-means and KDPC could not find the right clusters even some of them are departed.

Jain has 2 clusters with 373 points. The clusters distribute randomly on 2-dimensional space and some have mild overlapping. The algorithm proposed achieves

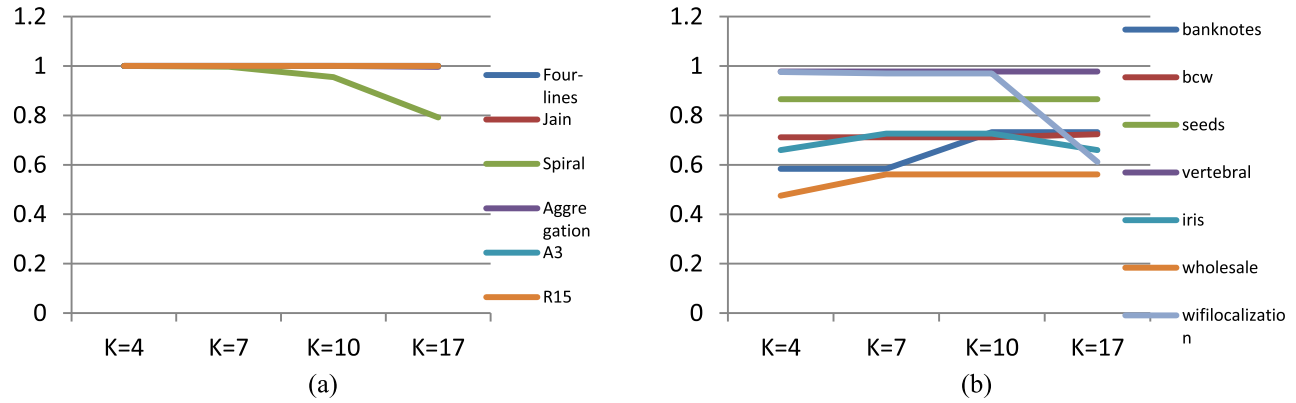

FIGURE 5. The accuracy of the dataset under different k.

TABLE 2. ACC for four algorithms on thirteen data sets.

Data sets	Proposed algorithm	DPC	FCM	K-means	KDPC	DBSCAN
Four-lines	**1.000**	**1.000**	**1.000**	0.670	0.699	**1.000**
parameter	k=7; n=4	dc=0.033	n=4	n=4	n=4	k=6; Eps=0.033
Jain	**1.000**	**1.000**	0.751	0.753	**1.000**	**1.000**
parameter	k=7; n=2	dc=13.2	n=2	n=2	n=2	k=10; Eps=0.08
Spiral	0.994	**1.000**	0.340	0.346	**1.000**	**1.000**
parameter	k=7; n=3	dc=13.6	n=3	n=3	n=3	k=4; Eps=2
Aggregation	**1.000**	**1.000**	0.632	0.736	0.741	0.957
parameter	k=7; n=7	dc=3.11	n=7	n=7	n=7	k=6; Eps=3
A3	**1.000**	**1.000**	**1.000**	0.629	0.711	0.670
parameter	k=7; n=3	dc=0.026	n=3	n=3	n=3	k=6; Eps=0.026
R15	**1.000**	1.000	0.948	0.737	0.997	**1.000**
parameter	k=7; n=15	dc=0.58	n=15	n=15	n=15	k=6; Eps=0.58
banknotes	**0.732**	0.674	0.677	0.671	0.626	0.555
parameter	k=10; n=2	dc=2.559	n=2	n=2	n=2	k=20; Eps=2.559
bcw	**0.711**	0.424	0.609	0.612	0.414	0.491
parameter	k=4; n=2	dc=1.414	n=2	n=2	n=2	k=20; Eps=1.414
seeds	**0.866**	0.227	0.539	0.600	0.746	0.314
parameter	k=17; n=3	dc=0.896	n=3	n=3	n=3	k=20; Eps=0.896;
vertebral	**0.978**	0.761	0.962	0.956	0.682	0.471
parameter	k=10; n=2	dc=16.511	n=2	n=2	n=2	k=20; Eps=16.511
Iris	**0.727**	0.640	0.527	0.547	0.567	0.333
parameter	k=10; n=3	dc=0.2	n=3	n=3	n=3	k=20; Eps=2.1;
Wholesale	**0.561**	0.514	0.545	0.532	0.382	0.532
parameter	k=7; n=3	dc=4356.541	n=3	n=3	n=3	k=20; Eps=4356.541
Wifilocalization	**0.970**	0.761	0.962	0.957	0.747	0.250
parameter	k=10; n=4	dc= 9.274	n=4	n=4	n=4	k=20; Eps=9.274

an optimal result on this data set. The optimal cut-off distance obtained from the adjustment gives a good result to Jain. The improved DPC, KDPC and DBSCAN work well while FCM and K-means have average performance.

Spiral has 3 clusters which embrace each other. From the results shown by Figure 8, we can see the algorithm proposed based on density get correct results while K-means and FCM are powerless. In (b) of Figure 8, we found that two blue points are misclassification. This is a phenomenon well worth discussing. As it shown in (a) of Figure 5, G-KNN-DPC get perfect result when K = 4. the special assignment strategy help explain this phenomenon. It may assign data points to clusters erroneously once a data point with a higher density is assigned to an incorrect cluster. But improved DPC, KDPC and DBSCAN overcome this defect. Aggregation has 788 points partitioned into 7 clusters. The results show G-KNN-DPC can find all cluster centers correctly and assign

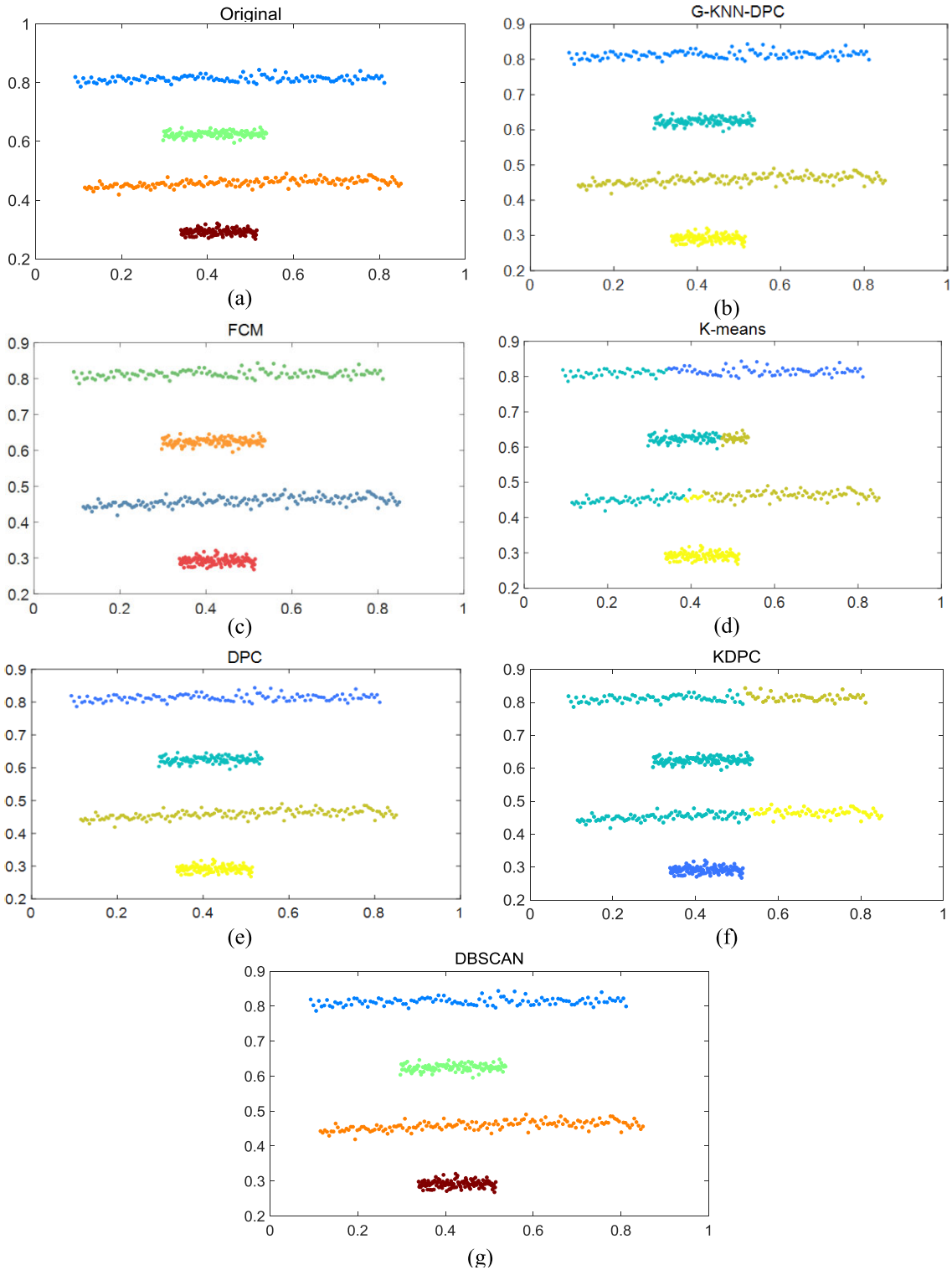

FIGURE 6. Clustering results of Four-lines data set on different algorithms.

almost all points to their corresponding clusters on this data set. Improved DPC can get similar results to our algorithm. It is worth to point out that the initial cluster centers are selected manually from the decision graph when processing the improved DPC algorithm. So sometimes the number of clusters will not be able to find out correctly. K-means and FCM just find two almost correct clusters exactly. KDPC find four correct clusters, which is better than K-means and FCM, but it divided one true cluster into two, and two clusters are merged into one cluster as shown in Figure 9(f). Figure 9(g)

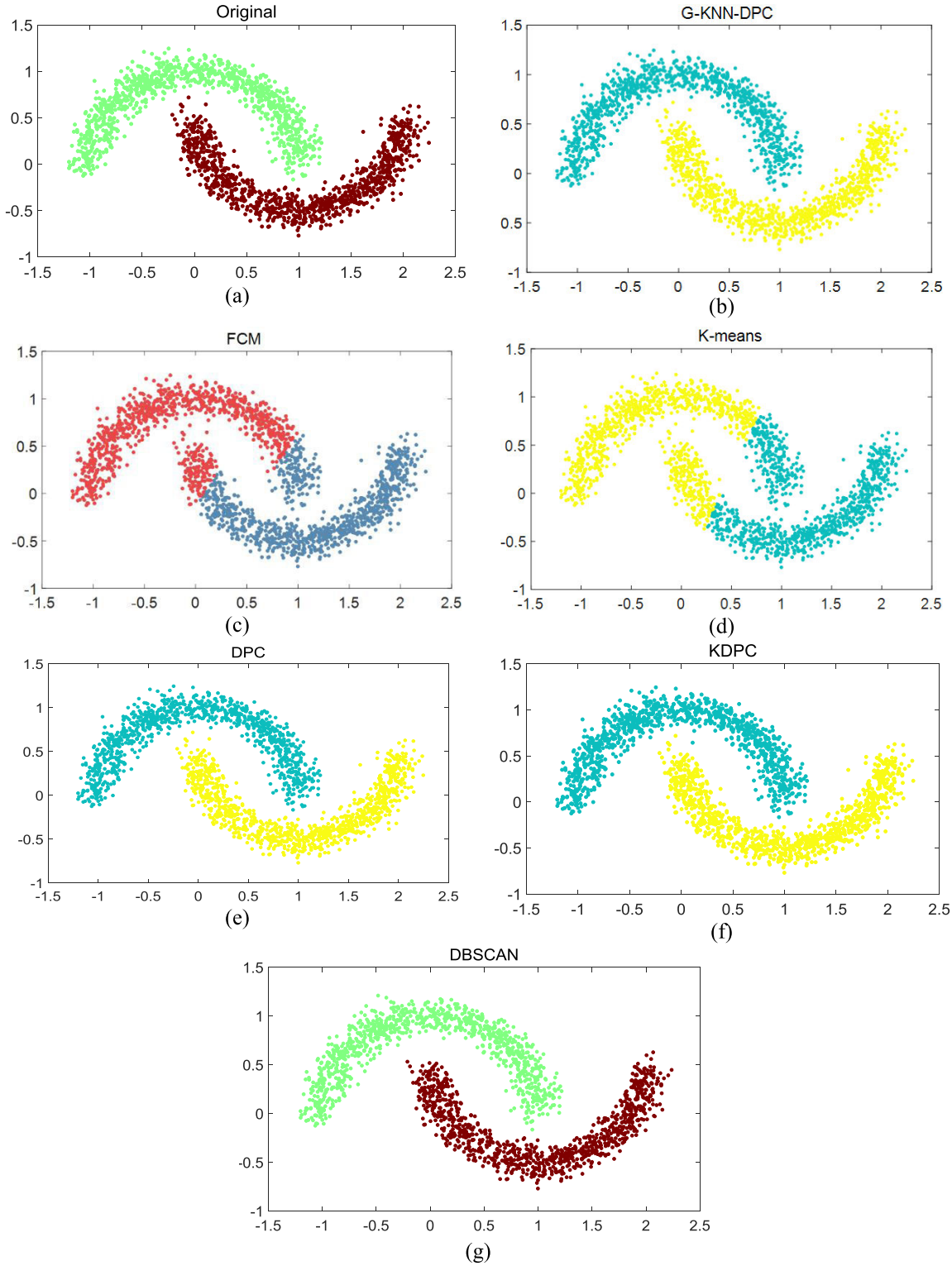

FIGURE 7. Clustering results of Jain data set on different algorithms.

shows that DBSCAN finds six clusters, which is too close to get perfect results.

A3 has three clusters with two kinds of different shapes. As Table 2 shows, G-KNN-DPC and improved DPC get the precise clustering results on this data set. As Figure 10 shows, K-means, FCM, KDPC and DBSCAN cannot find clusters correctly, one of the class is evenly divided into three categories.

The R15 data set has 15 clusters containing 600 points. It distributed in a 2-dimensional space and is overlapping

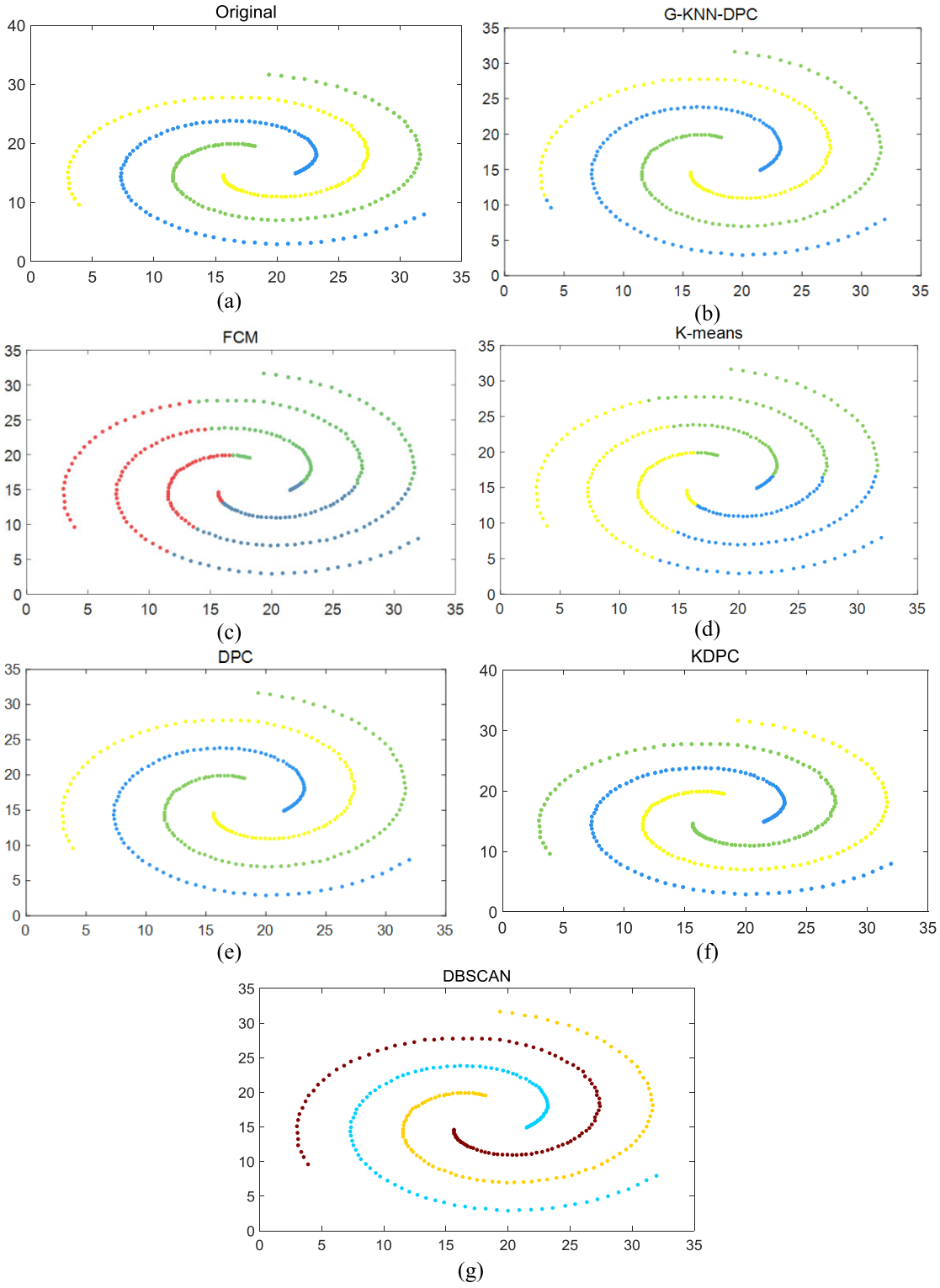

FIGURE 8. Clustering results of Spiral data set on different algorithms.

slightly. One of the clusters is surrounded by seven other clusters closely. The experimental results of the four algorithms are shown in Table 2, Table 3, and Table 4. The clustering results are displayed in Figure 11. G-KNN-DPC, improved DPC, FCM KDPC, and DBSCAN can both find the correct cluster centers. But K-means did not do well when assigning all data points to their corresponding clusters. And K-means did not find the right cluster centers.

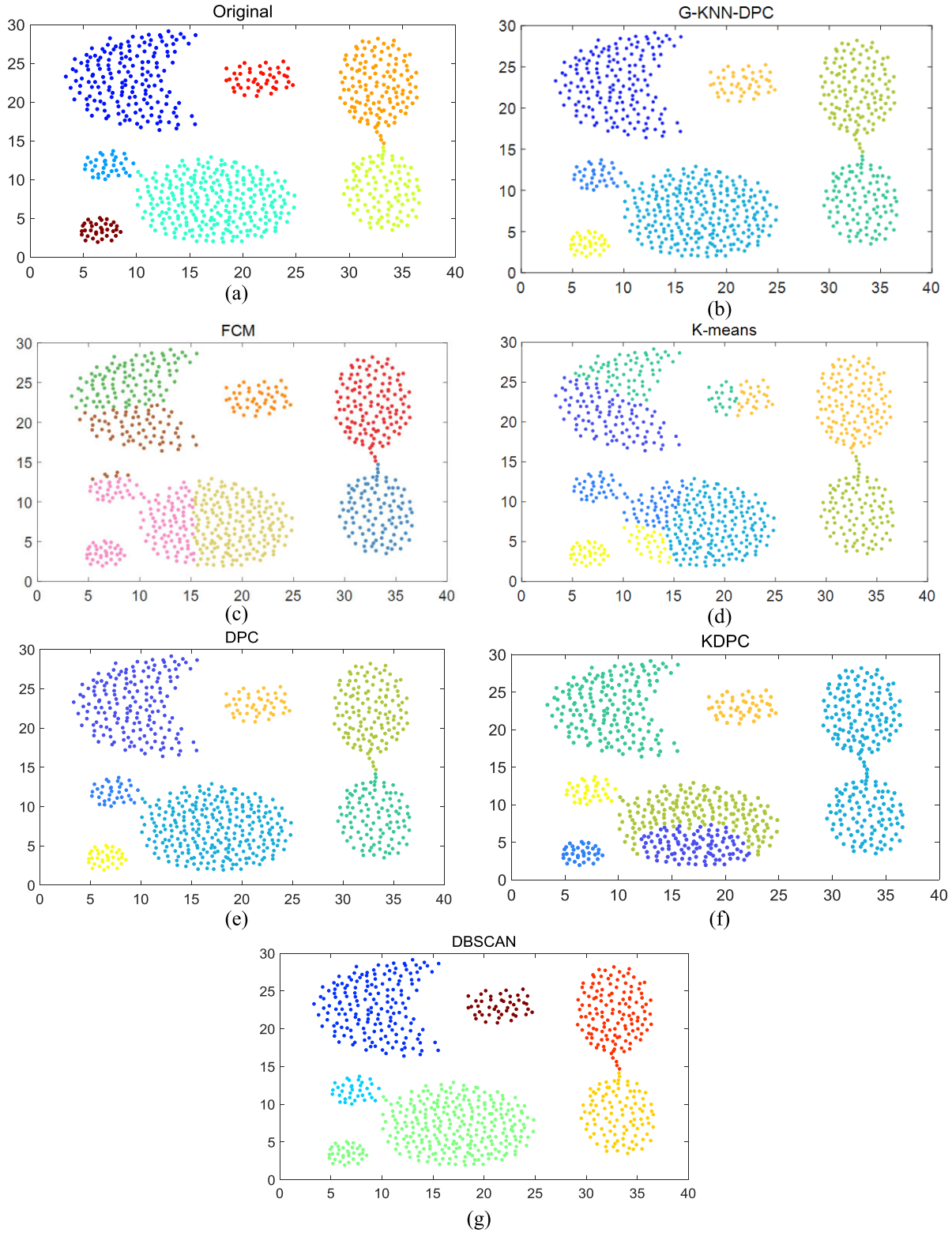

FIGURE 9. Clustering results of Aggregation data set on different algorithms.

E. EXPERIMENTS ON UCI DATA SETS AND RESULTS ANALYSIS

Seven UCI data sets were chosen to test the ability of G-KNN-DPC to recognize the clusters on varied data sets, which are commonly used in clustering or classification and all listed in Table 1. These seven data sets come from the UCI machine learning repository. In this subsection, the performance of each algorithm is benchmarked in terms of ARI,

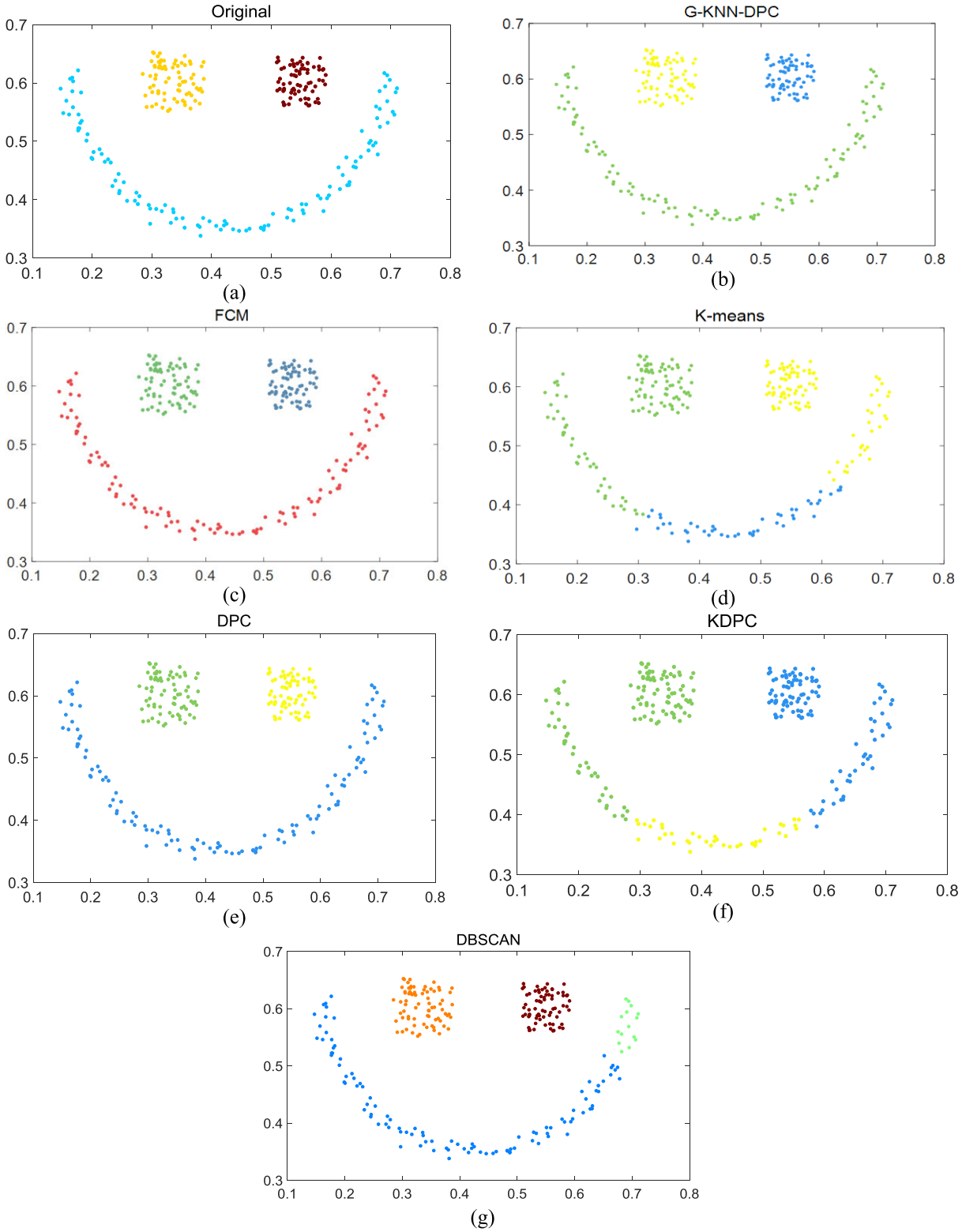

FIGURE 10. Clustering results of A3 data set on different algorithms.

NMI, RI shown in Table 2, Table 3, and Table 4. And the best results are shown in boldface. *Parameter* in three tables is the parameters specified.

Banknotes is composed of 2 clusters and 1372 data points. From the results shown in these tables, we can see that the G-KNN-DPC, improved DPC, FCM, K-means and KDPC

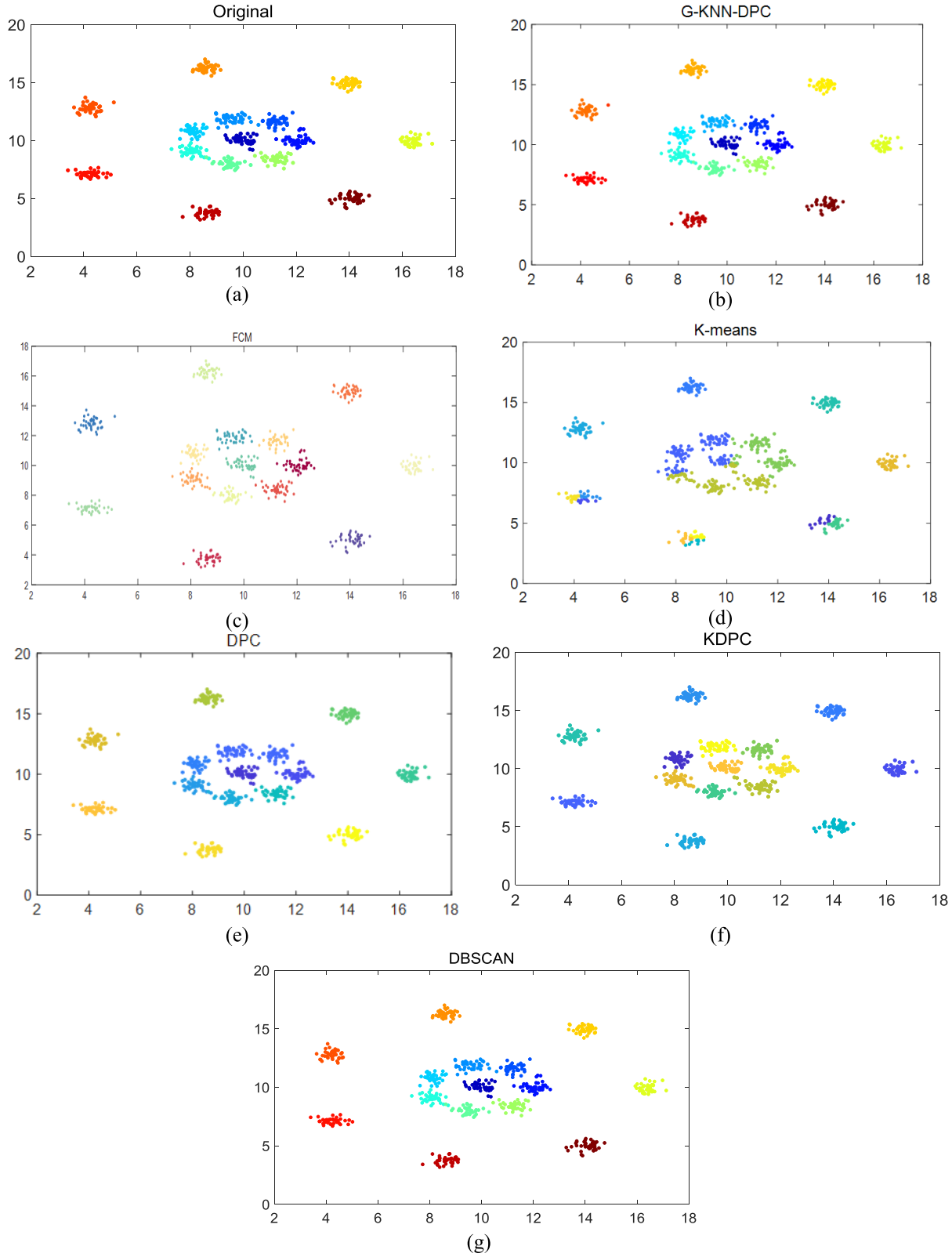

FIGURE 11. Clustering results of R15 data set on different algorithms.

can all find the cluster centers, but the benchmark data of these five algorithms are very different. The benchmark results for G-KNN-DPC are much better than that for improved DPC, FCM, K-means and KDPC. The clustering accuracy (ACC) of G-KNN-DPC reaches 0.732, but other algorithms did not do as well. This algorithm, however, has

TABLE 3. NMI for four algorithms on thirteen data sets.

Data sets	Proposed algorithm	DPC	FCM	K-means	KDPC	DBSCAN
Four-lines	**1.000**	**1.000**	**1.000**	0.602	0.635	**1.000**
parameter	k=7; n=4	dc=0.033	n=4	n=4	n=4	k=6; Eps=0.033
Jain	**1.000**	**1.000**	0.190	0.193	0.994	**1.000**
parameter	k=7; n=2	dc=13.2	n=2	n=2	n=2	k=10; Eps=0.08
Spiral	0.971	**1.000**	0.220	0.515	1.000	**1.000**
parameter	k=7; n=3	dc=13.6	n=3	n=3	n=3	k=4; Eps=2
Aggregation	**1.000**	**1.000**	0.764	0.813	0.874	0.956853
parameter	k=7; n=7	dc=3.11	n=7	n=7	n=7	k=6; Eps=3
A3	**1.000**	0.777	0.711	0.721	0.507	0.669173
parameter	k=7; n=3	dc=0.026	n=3	n=3	n=3	k=6; Eps=0.026
R15	**1.000**	**1.000**	0.540	0.531	0.989	**1.000**
parameter	k=7; n=15	dc=0.58	n=15	n=15	n=15	k=6; Eps=0.58
banknotes	0.214	0.108	0.206	**0.255**	0.240	0.056
parameter	k=10; n=2	dc=2.559	n=2	n=2	n=2	k=20; Eps=2.559
bcw	**0.355**	0.292	0.303	0.157	0.027	0.036
parameter	k=4; n=2	dc=1.414	n=2	n=2	n=2	k=20; Eps=1.414
seeds	**0.443**	0.125	0.075	0.441	0.151	0.132
parameter	k=17; n=3	dc=0.896	n=3	n=3	n=3	k=20; Eps=0.896;
vertebral	**0.922**	0.803	0.902	0.891	0.227	0.026
parameter	k=10; n=2	dc=16.511	n=2	n=2	n=2	k=20; Eps=16.511
Iris	**0.467**	0.443	0.369	0.401	0.316	0.292
parameter	k=10; n=3	dc=0.2	n=3	n=3	n=3	k=20; Eps=2.1;
Wholesale	**0.021**	0.014	0.001	0.002	0.004	0.017
parameter	k=7; n=3	dc=4356.541	n=3	n=3	n=3	k=20; Eps=4356.541
Wifilocalization	**0.00330**	0.00322	0.00326	0.00322	NaN	NaN
parameter	k=10; n=4	dc= 9.274	n=4	n=4	n=4	k=20; Eps=9.274

a heavy computational load in searching for the optimal parameter of a model. As NMI shown in Table 3, K-means gets the best results but still poor, G-KNN-DPC did not do well. As for RI, it shows that the result of G-KNN-DPC is still better than others. However, DBSCAN gets the poorest result and it cannot find these two clusters.

Bcw has two clusters. From the results in Tables 2-4, we can see the clustering results obtained by G-KNN-DPC are superior to those obtained by other methods. The ACC of the algorithm proposed is about 10% higher than those of other methods. The ACC value and the NMI value of the G-KNN-DPC are better than those of other methods as shown in and Table 3 and Table 4.

Seeds has 310 points of three clusters. Three evaluation indicators of this algorithm indicate its remarkable. Furthermore, from the results of each algorithm on Seeds. We can see that G-KNN-DPC is best among the five clustering algorithms. G-KNN-DPC obtained the ACC, NMI, RI values of 0.866, 0.443, and 0.767 respectively, which is higher than those obtained by other algorithms. And the result of K-means is a little better than that of FCM.

Vertebral has 310 points, and it is divided into two clusters. We gain the best result on this UCI data set. Table 2, Table 3, and Table 4 show the results of G-KNN-DPC are significantly better than those obtained by other methods. DBSCAN has the worst performance on this data set.

Iris has 150 points, which divided into 3 clusters. From Tables 3, we found that the clustering accuracy G-KNN-DPC is highest, the ACC reaches 0.727. The same situation happens in Tables 3 and Tables 4, the value of NMI and RI for G-KNN-DPC are 0.467 and 0.820. The improved DPC is followed by G-KNN-DPC. The performance of improved DPC is better than others. DBSCAN still get worse results.

The Wholesale data set consists of 440 data points and 3 clusters. From Tables 2-4, it is obvious that G-KNN-DPC got the best values of ACC and NMI among all six clustering algorithms. The values of ACC, NMI and RI obtained by G-KNN-DPC are 0.561,0.021 and 0.553 respectively. Larger values of these benchmarks indicate that the experimental results obtained by G-KNN-DPC are closer to the true results than those obtained by the other clustering algorithms.

The Wifilocalization data set consists of 2000 points and 4 clusters. G-KNN-DPC obtained the ACC, NMI and RI values of 0.970, 0.00330 and 0.960 respectively, which are higher than those obtained by other algorithms. The results also show that KDPC and DBSCAN has the worst performance on this dataset, even it did not get experimental results on NMI. And DBSCAN cannot identify all clusters.

TABLE 4. RI for four algorithms on thirteen data sets.

Data sets	Proposed algorithm	DPC	FCM	K-means	KDPC	DBSCAN
Four-lines	**1.000**	**1.000**	0.792	0.779	0.743	**1.000**
parameter	k=7; n=4	dc=0.033	n=4	n=4	n=4	k=6; Eps=0.033
Jain	**1.000**	**1.000**	0.626	0.627	0.999	**1.000**
parameter	k=7; n=2	dc=13.2	n=2	n=2	n=2	k=10; Eps=0.08
Spiral	0.991	**1.000**	0.554	0.554	**1.000**	**1.000**
parameter	k=7; n=3	dc=13.6	n=3	n=3	n=3	k=4; Eps=2
Aggregation	**1.000**	**1.000**	0.883	0.900	0.901	0.957
parameter	k=7; n=7	dc=3.11	n=7	n=7	n=7	k=6; Eps=3
A3	0.837	0.707	**0.843**	0.828	0.707	0.667
parameter	k=7; n=3	dc=0.026	n=3	n=3	n=3	k=6; Eps=0.026
R15	**1.000**	**1.000**	0.731	0.724	0.986	**1.000**
parameter	k=7; n=15	dc=0.58	n=15	n=15	n=15	k=6; Eps=0.58
banknotes	**0.607**	0.559	0.562	0.557	0.530	0.511
parameter	k=10; n=2	dc=2.559	n=2	n=2	n=2	k=20; Eps=2.559
bcw	0.589	**0.599**	0.524	0.525	0.503	0.732
parameter	k=4; n=2	dc=1.414	n=2	n=2	n=2	k=20; Eps=1.414
seeds	**0.767**	0.559	0.502	0.519	0.697	0.720
parameter	k=17; n=3	dc=0.896	n=3	n=3	n=3	k=20; Eps=0.896;
vertebral	**0.979**	0.866	0.964	0.959	0.566	0.491
parameter	k=10; n=2	dc=16.511	n=2	n=2	n=2	k=20; Eps=16.511
Iris	**0.820**	0.783	0.780	0.717	0.780	0.329
parameter	k=10; n=3	dc=0.2	n=3	n=3	n=3	k=20; Eps=2.1;
Wholesale	**0.553**	0.489	0.498	0.538	0.439	0.507
parameter	k=7; n=3	dc=4356.541	n=3	n=3	n=3	k=20; Eps=4356.541
Wifilocalization	**0.960**	0.905	0.489	0.498	0.837	0.282
parameter	k=10; n=4	dc= 9.274	n=4	n=4	n=4	k=20; Eps=9.274

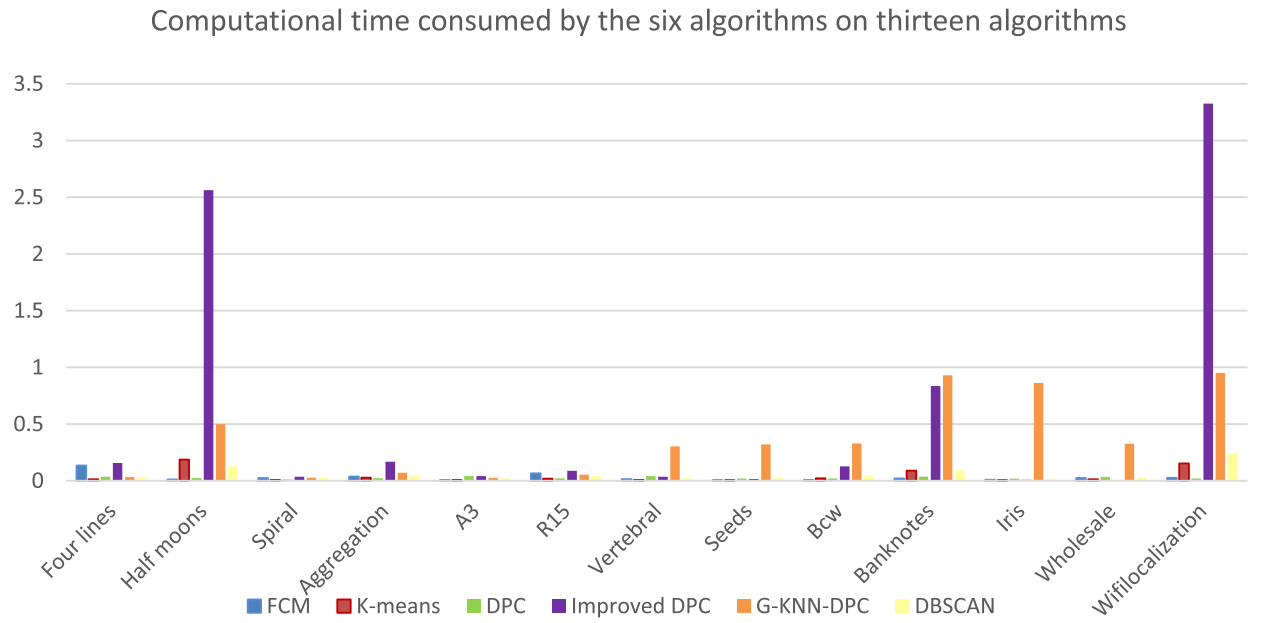

FIGURE 12. Computational time consumed by the six algorithms on thirteen algorithms.

F. COMPUTATIONAL TIME CONSUMED RESULTS ANALYSIS

Figure 12 shows the average computational time consumed by each of the algorithms to complete the process of clustering. For time graphs, G-KNN-DPC is represented in oranges bars, FCM by blue bars, K-means by red bars, DPC by green bars, improved DPC by purple bars, and DBSCAN by yellow bars. The time consumed is plotted against corresponding algorithms and datasets. From Figure 12, it is observed that

TABLE 5. The number of data sets in which each of the six algorithms showed top clustering performance of the different evaluation indexes.

Algorithms	ACC	NMI	RI
G-KNN-DPC	12	11	10
DPC	5	5	6
FCM	2	1	1
K-means	0	1	0
KDPC	2	0	1
DBSCAN	4	4	4

improved DPC has the highest execution run time across the thirteen datasets, which means it gets the worst result in the running time of the algorithm. This is followed by G-KNN-DPC. Other algorithms have a shorter running time across all the thirteen datasets. It however consumed a long time in all its process, G-KNN-DPC achieved the best solutions as earlier discussed.

Table 5 will make the results of the comparison experiment clearer. Table 5 gives the number of data sets in which each of the five algorithms showed the top clustering performance for the different evaluation indexes when using synthetic data sets and real data sets. For ACC, the G-KNN-DPC algorithm gained the best clustering performance by achieving the highest value on twelve of the thirteen data sets. NMI and RI both showed similar results. In all cases, the G-KNN-DPC demonstrated the best clustering performance. In each evaluation index, the G-KNN-DPC showed the best clustering performance. In conclusion, these results demonstrate that the G-KNN-DPC algorithm is effective and better than the other clustering algorithms.

V. CONCLUSION

The new algorithm proposed in this paper first calculate optional d_c, then redefines the distance with K-nearest neighbors of every data point. The algorithm recognizes the core point and constructs the cluster around the core point, then attempts to detect the outliers. It makes full use of K-nearest neighbors. It produces very good clustering results on the two-dimensional data. The results are superior to other existing algorithms on the two-dimensional and high-dimensional data set. We use the Gini coefficient to determine the thresholds adaptively. The parameter d_c of the algorithm is automatically determined by the data themselves, and it has application value. G-KNN-DPC recognizes clusters of arbitrary shapes, of different sizes, of different densities, and reduces the effect of outliers. The experiments indicate that the algorithm runs very effectively. The effectiveness of the algorithm on high dimensional data is tested with UCI data sets. In the future, the accuracy of the algorithm needs to be further improved for high-dimensional data.

REFERENCES

[1] H. V. Jagadish, J. Gehrke, A. Labrinidis, Y. Papakonstantinou, J. M. Patel, R. Ramakrishnan, and C. Shahabi, "Big data and its technical challenges," *Commun. ACM*, vol. 57, no. 7, pp. 86–94, Jul. 2014.

[2] I. Guyon, U. Von Luxburg, and R. C. Williamson, "Clustering: Science or art?" in *Proc. ICML Workshop Unsupervised Transf. Learn.* Jun. 2012, pp. 65–79.

[3] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[4] J. Han and K. Micheline, *Data Mining: Concepts and Techniques*, vol. 5, no. 4. San Francisco, CA, USA: Morgan Kaufmann, 2006, pp. 1–18.

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

[6] J. Hou and E. Xu, "An improved density peak clustering algorithm," *Intelligent Data Engineering and Automated Learning—IDEAL 2017* (Lecture Notes in Computer Science), vol. 10585. Berlin, Germany: Springer, 2017.

[7] 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," *Inf. Sci.*, vol. 354, pp. 19–40, Aug. 2016.

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

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

[10] S. Wang, D. Wang, C. Li, and Y. Li, "Comment on 'clustering by fast search and find of density peaks,'" 2015, *arXiv:1501.04267*. [Online]. Available: http://arxiv.org/abs/1501.04267

[11] 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, Oct. 2016.

[12] J. Gao, L. Zhao, Z. Chen, P. Li, H. Xu, and Y. Hu, "ICFS: An improved fast search and find of density peaks clustering algorithm," in *Proc. IEEE 14th Int. Conf. Dependable, Autonomic Secure Comput., 14th Int. Conf. Pervasive Intell. Comput., 2nd Int. Conf. Big Data Intell. Comput. Cyber Sci. Technol. Congr. (DASC/PiCom/DataCom/CyberSciTech)*, Aug. 2016, pp. 537–543.

[13] X. Xu, Y. Ju, Y. Liang, and P. He, "Manifold density peaks clustering algorithm," in *Proc. 3rd Int. Conf. Adv. Cloud Big Data*, Oct. 2015, pp. 311–318.

[14] Y. Zhang, S. Chen, and G. Yu, "Efficient distributed density peaks for clustering large data sets in MapReduce," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 12, pp. 3218–3230, Dec. 2016.

[15] L. Yaohui, M. Zhengming, and Y. Fang, "Adaptive density peak clustering based on K-nearest neighbors with aggregating strategy," *Knowl.-Based Syst.*, vol. 133, pp. 208–220, Oct. 2017.

[16] 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, May 2016.

[17] J. Huang, Q. Zhu, L. Yang, D. Cheng, and Q. Wu, "QCC: A novel clustering algorithm based on quasi-cluster centers," *Mach. Learn.*, vol. 106, no. 3, pp. 337–357, Mar. 2017.

[18] D. Yu, G. Liu, M. Guo, X. Liu, and S. Yao, "Density peaks clustering based on weighted local density sequence and nearest neighbor assignment," *IEEE Access*, vol. 7, pp. 34301–34317, 2019.

[19] L. Xu, J. Zhao, Z. Yao, A. Shi, and Z. Chen, "Density peak clustering based on cumulative nearest neighbors degree and micro cluster merging," *J. Signal Process. Syst.*, vol. 91, no. 10, pp. 1219–1236, Oct. 2019.

[20] X. Xu, S. Ding, H. Xu, H. Liao, and Y. Xue, "A feasible density peaks clustering algorithm with a merging strategy," *Soft Comput.*, vol. 23, no. 13, pp. 5171–5183, Jul. 2019.

[21] M. Du, S. Ding, Y. Xue, and Z. Shi, "A novel density peaks clustering with sensitivity of local density and density-adaptive metric," *Knowl. Inf. Syst.*, vol. 59, no. 2, pp. 285–309, May 2019.

[22] J. Jiang, Y. Chen, X. Meng, L. Wang, and K. Li, "A novel density peaks clustering algorithm based on k nearest neighbors for improving assignment process," *Phys. A, Stat. Mech. Appl.*, vol. 523, pp. 702–713, Jun. 2019.

[23] M. D. Parmar, W. Pang, D. Hao, J. Jiang, W. Liupu, L. Wang, and Y. Zhou, "FREDPC: A feasible residual error-based density peak clustering algorithm with the fragment merging strategy," *IEEE Access*, vol. 7, pp. 89789–89804, 2019.

[24] M. Parmar, D. Wang, X. Zhang, A.-H. Tan, C. Miao, J. Jiang, and Y. Zhou, "REDPC: A residual error-based density peak clustering algorithm," *Neurocomputing*, vol. 348, pp. 82–96, Jul. 2019.

[25] T. Liu, H. Li, and X. Zhao, "Clustering by search in descending order and automatic find of density peaks," *IEEE Access*, vol. 7, pp. 133772–133780, 2019.

[26] C. Wu, J. Lee, T. Isokawa, J. Yao, and Y. Xia, "Efficient clustering method based on density peaks with symmetric neighborhood relationship," *IEEE Access*, vol. 7, pp. 60684–60696, 2019.
[27] L. Duan, L. Xu, Y. Liu, and J. Lee, "Cluster-based outlier detection," *Microelectron. Comput.*, vol. 168, no. 1, pp. 151–168, 2009.
[28] F. Angiulli and C. Pizzuti, "Fast outlier detection in high dimensional spaces," in *Proc. Eur. Conf. Princ. Data Mining Knowl. Discovery*. Berlin, Germany: Springer, 2002, pp. 15–17.
[29] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. Kdd*, 1996, vol. 96, no. 34, pp. 226–231.
[30] S. Yue, M. Wei, Y. Li, and X. Wang, "Ordering grids to identify the clustering structure," in *Proc. Int. Symp. Adv. Neural Netw.* Berlin, Germany: Springer, 2007, pp. 612–619.
[31] R. Mehmood, R. Bie, L. Jiao, H. Dawood, and Y. Sun, "Adaptive cutoff distance: Clustering by fast search and find of density peaks," *J. Intell. Fuzzy Syst.*, vol. 31, no. 5, pp. 2619–2628, Oct. 2016.
[32] 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, Oct. 2016.
[33] Y. Li, Y. Chen, and Q. Li, "A clustering algorithm for fuzzy numbers based on fast search and find of density peaks," *Intell. Data Anal.*, vol. 23, pp. 25–52, Jun. 2019.
[34] S. Wang, C. Li, G. Ding, D. Wang, and Y. Li, "Clustering by fast search and find of density peaks with data field," *Chin. J. Electron.*, vol. 25, no. 3, pp. 397–402, May 2016.
[35] S. Ding, M. Du, T. Sun, X. Xu, and Y. Xue, "An entropy-based density peaks clustering algorithm for mixed type data employing fuzzy neighborhood," *Knowl.-Based Syst.*, vol. 133, pp. 294–313, Oct. 2017.
[36] M. Du, S. Ding, X. Xu, and Y. Xue, "Density peaks clustering using geodesic distances," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 8, pp. 1335–1349, Aug. 2018.
[37] W. Zang, L. Ren, W. Zhang, and X. Liu, "Automatic density peaks clustering using DNA genetic algorithm optimized data field and Gaussian process," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 31, no. 8, Aug. 2017, Art. no. 1750023.
[38] S. J. Peter, "Local density-based hierarchical clustering using minimum spanning tree," *J. Discrete Math. Sci. Cryptography*, vol. 16, nos. 2–3, pp. 125–137, Jun. 2013.
[39] D. Cheng, Q. Zhu, J. Huang, Q. Wu, and L. Yang, "A local cores-based hierarchical clustering algorithm for data sets with complex structures," *Neural Comput. Appl.*, vol. 31, no. 11, pp. 8051–8068, 2019.
[40] M. Parmar, D. Wang, A.-H. Tan, C. Miao, J. Jiang, and Y. Zhou, "A novel density peak clustering algorithm based on squared residual error," in *Proc. Int. Conf. Secur., Pattern Anal., Cybern. (SPAC)*, Dec. 2017, pp. 43–48.
[41] A. Gionis, H. Mannila, and P. Tsaparas, "Clustering Aggregation," *ACM Trans. Knowl. Discovery Data*, vol. 1, no. 1, p. 4, 2007.
[42] H. Chang and D.-Y. Yeung, "Robust path-based spectral clustering," *Pattern Recognit.*, vol. 41, no. 1, pp. 191–203, Jan. 2008.
[43] C. J. Veenman, M. J. T. Reinders, and E. Backer, "A maximum variance cluster algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 9, pp. 1273–1280, Sep. 2002.
[44] J. Hou and H. Cui, "Experimental evaluation of a density kernel in clustering," in *Proc. 7th Int. Conf. Intell. Control Inf. Process. (ICICIP)*, Dec. 2016, pp. 55–59.
[45] R. Shang, W. Zhang, F. Li, L. Jiao, and R. Stolkin, "Multi-objective artificial immune algorithm for fuzzy clustering based on multiple kernels," *Swarm Evol. Comput.*, vol. 50, Nov. 2019, Art. no. 100485.
[46] Z. He, X. Xu, and S. Deng, "K-ANMI: A mutual information based clustering algorithm for categorical data," *Inf. Fusion*, vol. 9, no. 2, pp. 223–233, Apr. 2008.
[47] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mach. Learn. Res.*, vol. 11, pp. 2837–2854, Jan. 2010.

DONG JIANG is currently pursuing the master's degree with the School of Business, Shandong Normal University, China. Her research interests include machine learning, genetic algorithm, data mining, and artificial intelligence.

WENKE ZANG received the M.S. and Ph.D. degrees from Shandong Normal University, China, in 2005 and 2018, respectively. He is currently an Associate Professor with Shandong Normal University. His research interests include machine learning, data mining, and service science.

RUI SUN is currently pursuing the master's degree with the School of Business, Shandong Normal University, China. Her research interests include data mining, machine learning, genetic algorithm, and artificial intelligence.

ZEHUA WANG is currently pursuing the master's degree with the School of Business, Shandong Normal University, China. Her research interests include artificial intelligence, genetic algorithm, data mining, and machine learning.

XIYU LIU (Member, IEEE) received the Ph.D. degree in mathematical sciences from Shandong University, in 1990. He is currently a Professor, a Ph.D. Supervisor, and the Dean of the School of Management Science and Engineering, Shandong Normal University, China. His current research interests include membrane computing and data mining.