IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 9, 2021, accepted April 23, 2021, date of publication May 10, 2021, date of current version May 18, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3078584

Clustering by Constructing Hyper-Planes

LUHONG DIAO, MANMAN DENG, AND JINYING GAO
Beijing Institute for Scientific and Engineering Computing, Beijing University of Technology, Beijing 100022, China
Faculty of Science, College of Mathematics, Beijing University of Technology, Beijing 100022, China

Corresponding author: Luhong Diao (diaoluhong@bjut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 11772013, and in part by the general research projects of Beijing Educations Committee in China under Grant KM201910005013.

ABSTRACT As a ubiquitous method in the field of machine learning, clustering algorithm attracts a lot attention. Because only some basic information can be utilized, clustering data points into correct categories is a critical task especially when the cluster number is unknown. This paper presents an algorithm which can find the cluster number automatically. It firstly constructs hyper-planes based on the marginal of sample points. Then an adjacent relationship between data points is defined. Based on it, connective components are derived. According to a validity index proposed in this paper, the high-qualified connective components are selected as cluster centers. Meanwhile, the clusters' number is also determined. Another contribution of this paper is that all the parameters in this algorithm can be set automatically. To evaluate its robustness, experiments on different kinds of benchmark datasets are carried out. They show that the performances are even better than some other methods' best results which are selected manually.

INDEX TERMS Clustering algorithm, hyper-planes, support vector machine, validity index.

I. INTRODUCTION

Cluster is a basic operation among the community of machine learning. Based on the similarity of the data points, clustering algorithms put them into categories so that the points in the same category are as similar as possible and as dissimilar as possible with points within other categories. Because they always handle the datasets without pre-existing labels, only some basic information like distances between points, density of points or points' distribution can be used to derive the final results. Therefore they reveal the intrinsic pattern of data.

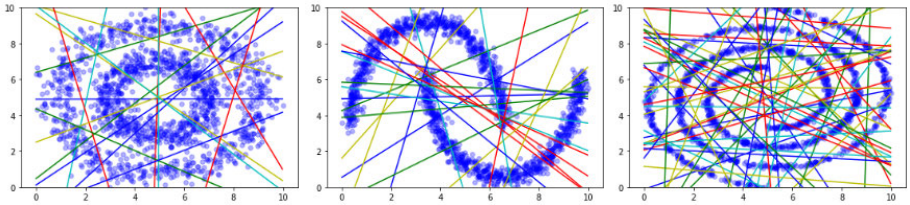
The cluster number is an important parameter for the algorithms. Many of the clustering algorithms require that it must be known such as K-means [1], K-medoids [2] and hierarchical clustering [3]. When it is not given in advance, to get the right clusters is obviously more difficult. One common way to determine the number of clusters is trying the different ones of clusters to choose the best number, which is done by optimizing the objective functions [4]–[7].

A few methods can determine it directly. One kind of these methods are based on the density of points like two famous methods density-based spatial clustering of applications with noise (DBSCAN) [8] and clustering by fast search and find of density peaks (CFSFDP) [9]. Recently another kind of methods named I-nice [10] are proposed. Instead of finding the regions of high density directly, they derive the number of cluster centers on the idea of simulating the process of man's viewing peaks of mountains. The advantage of these two kinds of methods is that they are not sensible to the distribution of data points [11]. Their main problem is their sensitiveness to parameters such as the thresholds for determining the adjacent relationship or the quantity of neighbor points. Some works are proposed in order to overcome it [12]–[15].

The Affinity Propagation algorithm [16] derives the clusters' center and number by exchanging the information between data points, which is a rather novel idea. However, the main problem is also the selection of some parameters' values. Whereas these values are related to the number of clusters usually, it is difficult to be determined in advance. Therefore the voting idea is also adopted for algorithms in which the cluster number is known [17].

Some algorithms cluster points based on distributions [18]. The performance relies on the distributions' capability to represent the data points. So this kind of methods cannot always performance well because the real data always distributes on some complex nonlinear manifolds. Recently some works has turned to deal with the data on special distributions [19], [20], which derive rather good results. However, they cannot be applied to other kinds of manifolds flexibly.

The associate editor coordinating the review of this manuscript and approving it for publication was Wentao Fan.



(a) (b) (c)

FIGURE 1. Results of $f_\delta : S \to L$. (a) Distribution of S is two concentric circles. (b) Distribution of S is two interleaving moons. (c) Distribution of S is swiss-roll.

To overcome these problems, a clustering algorithm by constructing hyper-planes is presented in this paper. There already exist some clustering algorithms which form groups of data by finding large margin hyper-planes [21], [22] or hyper-spheres [23]. The differences between the proposed method and the existed methods based on hyper-planes [21], [22] mainly lie in two points. One is that ways to find the hyper-planes are different. The hyper-planes constructed in the existed methods are based on some constraints. And these constraints usually include that the hyper-planes void to crossing through the high density regions. In our algorithm, the hyper-planes depend on the marginal space between global or local points in different clusters, it may crossing through some high density regions. So it can be approached for the data points which take the distribution on more general manifolds. The other is the existed methods acquire the clusters' number to be preset whereas our algorithm can determine it automatically. The method based on the hyper-sphere [23] also can find the clusters' number. However, it is an supervised learning method.

The main contribution of our algorithm is that we present a robust scheme to determine the main parameter's value, which overcomes an inherent difficulty for the clustering algorithm. The experimental results indicate that its performances are even better than some other famous methods' results which are selected manually.

The rest of this paper is organized as follows: Section 2 presents the clustering algorithm; Section 3 carries out some experiments; Section 4 contains some concluding remarks and directions for future research.

II. CLUSTERING BASED ON HYPER-PLANES

This section proposes the clustering algorithm. It firstly constructs some hyper-planes based on the marginal space between subgroups. By combining these hyper-planes, it derives some connective components. Then it merges some connective components to generate the final clusters. The last part gives the complexity analysis of this algorithm.

A. CONSTRUCTING HYPER-PLANES

Suppose a data set S which contains a group of points. The algorithm groups S into two sets S_1 and S_2 by using K-means method. Then it derives a hyper-plane $l : (w, b)$ based on Support Vector Machine (SVM) [24] to segment S_1 and S_2. The points in two sets S_1 and S_2 are defined as affiliated to l. The same operation is done both on S_1 and S_2 recursively until data sets are smaller than a threshold δ. After that, a group of hyper-planes, denoted as L, are derived. The map from S to L is defined as $f_\delta : S \to L$.

Fig. 1 is the examples of the all the hyper-planes derived on three nonlinear manifolds. For selecting the proper hyper-planes, L is grouped into two categories according to the hyper-planes' values of $\|w\|$. It is done by using the K-means method. The category with smaller $\|w\|$, denoted as L', is the hyper-planes which prefer to distinguishing the points well as shown in Fig. 2a. The map from L to L' is defined as $\varphi : L \to L'$.

However, φ doesn't work well for more complex manifolds as shown in Fig. 2b and Fig. 2c. For dealing with this problem, S is divided into some subgroups $S_{L',1}$, $S_{L',2}$..., $S_{L',m}$ such that points in $S_{L',i}$, $1 \le i \le m$ lie on same side of each hyper-plane in L', denoted as $\gamma_{L'} : S \to \{S_i\}$. Then a new hyper-plane set $L' \cup \varphi\left(\bigcup_{1\le i\le m} f_\delta(S_{L',i})\right)$ can be derived. If L' is not equal to $L' \cup \varphi\left(\bigcup_{1\le i\le m} f_\delta(S_{L',i})\right)$, let $L' = L' \cup \varphi\left(\bigcup_{1\le i\le m} f_\delta(S_{L',i})\right)$. Repeat this operation until L' unvaried.

Then all the generated hyper-planes in this process are put into the set TL. Based on TL, the proper hyper-plane set can be derived. For example, Fig. 3 is the hyper-planes set $\varphi(TL)$ which segments the data points well. Pseudocode for this process is presented in Algorithm 1.

B. DERIVING THE CONNECTIVE COMPONENTS

After the hyper-plane set H is found, the adjacency between data points can be determined. We define that if a data point x is affiliated to $l : (w, b)$ or $|w \cdot x + b| > 1$, x is isolated to $l : (w, b)$. If two points x_i and x_j lie opposite side of any hyper-plane l in H and there is at least one isolated to l, x_i and x_j are not adjacent. Otherwise, they are adjacent. Then it can derive some connective components based on this adjacent relationship. We define this map is $\tau_{H,A} : S \to \{C_1, C_2, \ldots, C_k\}$, where C_i, $1 \le i \le k$ is a connective component and A is the affiliated relation matrix derived in Algorithm 1. Pseudocode for this process is presented in Algorithm 2.

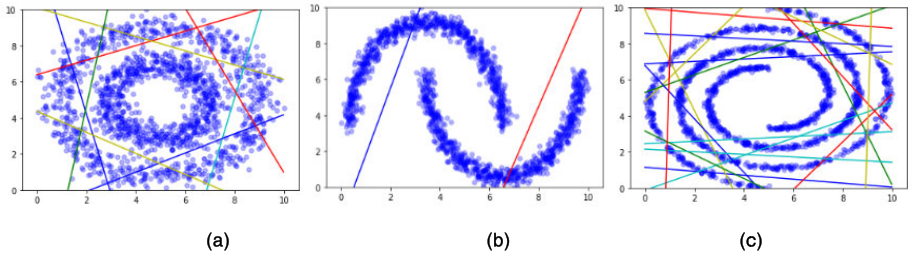


FIGURE 2. Results of $\varphi(f_\delta(S))$. (a) Distribution of S is two concentric circles. (b) Distribution of S is two interleaving moons. (c) Distribution of S is swiss-roll.

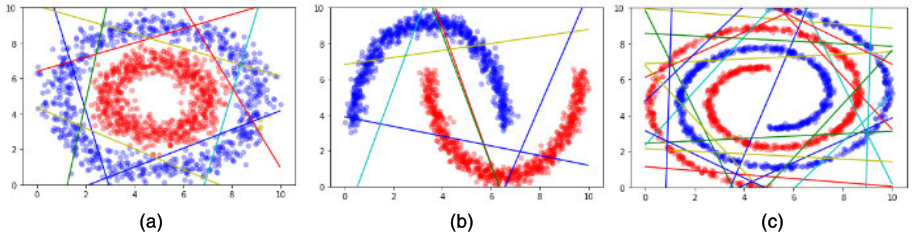


FIGURE 3. Results of connective components for synthetic datasets. Different colors represent different connective components. The colored lines are the hyper-planes. (a) Distribution of two concentric circles. (b) Distribution of two interleaving moons. (c) Distribution of swiss-roll.

Algorithm 1 Constructin Hyper-Planes

Input: The dataset S. The threshold $\delta \in \Re$.
Output: The hyper-plane set TL. The affiliated matrix A.($A(i,j) = 1$ means point x_i is affiliated to the hyper-plane l_j,Otherwise, $A\ (i,j) = 0$.)
Initialization: $P \leftarrow \{S\}$. $TL \leftarrow \emptyset$. $L' \leftarrow \emptyset$. $A \leftarrow \emptyset$.
1: Repeat
2: $curL' \leftarrow L', tempL \leftarrow \emptyset$
3: **for** each set $S_i \in P$ **do**
4 : $TL \leftarrow TL \cup f_\delta\ (S_i)$
5 : $tempL \leftarrow tempL \cup f_\delta\ (S_i)$
6 : Let $curA$ be a zero matrix, its size is $|S| \times |f_\delta\ (S_i)|$
7 : **for** each $l_j \in f_\delta\ (S_i)$ **do**
8 : **if** $x_k \in S$ is affiliated to l_j **do**
9 : $curA\ (k,j) = 1$
10 : **end if**
11 : **end for**
12 : $A \leftarrow (A, curA)$
13 : **end for**
14 : $L' \leftarrow L' \cup \varphi\ (tempL)$
15 : $P \leftarrow \gamma_{L'}\ (S)$
16 : **Until** $curL' = L'$

C. DETERMINING THE CLUSTER NUMBER AND CENTERS

Before finding the clusters' center, we need compute the validity index for the connective components. One of the difficulties for computing the validity index is to choose distance measures. Because they are related to data points' distribution closely [25], [26], it is uneasy to computer the proper distance when the distribution is unknown. Lots of them have been proposed from seventies in last century to now [3]–[5], [7], [27]–[31].

Because the adjacency between data points is defined based on the hyper-plane set, we can also use this adjacency relation to compute the distance more accurately. A new measure of mean intra-connective-component distance and nearest-connective-component distance for each connective component is proposed as following:

All the point x is firstly transformed into $x^{(H)} = (d^L(x, l_1), \ldots, d^L(x, l_{|H|}))$ where $l_i \in H\ (1 \le i \le |H|)$ and $d^L(x, l : (w, b)) = (wx + b) / \|w\|$. Then the mean intra-cluster distance of C_i is computed as:

$$IntraDis(i) = \sum_{x_p, x_q \in C_i} \frac{2 d_g(x_p^{(H)}, x_q^{(H)})}{|C_i|\,(|C_i| - 1)} \tag{1}$$

where $d_g(x_i, x_j)$ is the geodesic distance between x_i and x_j based on the adjacent relationship.

The nearest-connective-component distance of C_i is computed as:

$$InterDis(i) = \frac{1}{|C_i|} \sum_{x_p \in C_i} \min_{x_q \notin C_i} d(x_p^{(H)}, x_q^{(H)}) \tag{2}$$

where $d(x_i, x_j)$ is the Euclidean distance between x_i and x_j. Based on them, the measure M of C_i is:

$$M_i = \begin{cases} \dfrac{InterDis(i)}{IntraDis(i)}, & |C_i| > \max(1, \dfrac{\delta}{4}) \\ 0, & |C_i| \le \max(1, \dfrac{\delta}{4}) \end{cases} \tag{3}$$

Algorithm 2 Deriving the Connective Components

Input: The dataset S. The affiliated matrix A. The hyper-plane set H.
Output: The connective component set C.
Initialization: Define the adjacent matrix $adjM$ with size $|S|\times|S|$. All its elements are one. Let $islM$ be a zero matrix, its size is $|S| \times |H|$.
1: for each $x_i \in S$ and $l_j\,(w, b) \in H$ **do**
2: **if** $|wx_i + b| > 1$ or $A\,(i, j) == 1$ **do**
3: $islM(i, j) = 1$
4: **end if**
5: end for
6: for $x_i, x_j \in S$ **do**
7: **for** each $l_k \in H$ **do**
8: **if** x_i, x_j lie opposite side of l_k and $islM\ (i, k) + islM\ (j, k) > 0$ **do**
9: $adjM\ (i, j) = 0$
10: break
11: **end if**
12: **end for**
13: end for
14: derive the connective component $C_1, C_2, ...$ based on $adjM$
15 : $C \leftarrow \{C_1, C_2, ...\}$

The threshold $\delta/4$ is set to filter small size connective components off. Points in connective component with the larger M prefer to be in the same cluster with larger probability.

If $x_p \in C_i$, $x_q \in C_j$ and x_q is the nearest inter-class neighbor of x_p such that $d\left(x_p^{(H)}, x_q^{(H)}\right) = \min\limits_{x \notin C_i} d\left(x_p^{(H)}, x^{(H)}\right)$, we define that C_j is a neighbor of C_i. If M_i is larger than the values of all its neighbors', C_i is a peak connective components (PCC). If only one neighbor's M is larger than M_i, C_i is called hillside connective component (HCC).

PCC is the best connective component among all its neighbors. To make the result more accurate, we adopt the PCC whose M is larger than half of the connective components, denoted as FPCC, as the seed of cluster centers. Pseudocode for this process is presented in Algorithm 3.

D. DERIVING THE CLUSTERS

After the cluster centers are found, the clusters are extended further by adding the HCCs gradually. Put a FPCC C_i into R_i. Then repeat the following operation until no connective component in R_i is unvisited. Pick an unvisited connective component C_p from R_i. If C_q is a neighbor of C_p, $M_p > M_q$ and C_q is a HCC, put C_q into R_i. After all such C_q are done, mark C_p is visited. R_i is one cluster. Pseudocode for this process is presented in Algorithm 4.

E. COMPLEXITY ANALYSIS OF THE ALGORITHM

The flowchart of the algorithm is illustrated in Fig. 4. If the dataset S contains n points with d dimensions, the complexity analysis of each steps are followings:

Algorithm 3 Determining the Cluster Number and Centers

Input: Dataset S. The connective component set $C = \{C_i\}$.
Output: The FPCC set F. The HCC set HC. The neighbor relationship matrix CR. (If C_i and C_j are neighbor, $CR\,(i, j) = 1$. Otherwise, $CR\,(i, j) = 0$.)
Initialization: $F \leftarrow \emptyset$. $HC \leftarrow \emptyset$. Let CR be a zero matrix, its size is $|C| \times |C|$.
1: for each $C_i \in C$ **do**
2: Compute M_i according to (3)
3: end for
4: for $C_i, C_j \in C$ **do**
5: **if** there exist two points $x_p \in C_i, x_q \in C_j$ and $d\left(x_p^{(H)}, x_q^{(H)}\right) = \min\limits_{x \notin C_i} d\left(x_p^{(H)}, x^{(H)}\right)$ **do**
6 : $CR\,(i, j) = 1$
7: **end if**
8: end for
9: for each $C_i \in C$ **do**
10: **if** there doesn't exist any connective component $C_j \in C$ satisfies $CR\,(i, j) = 1$ and $M_j > M_i$ **do**
11: **if** M_i is bigger than half of the connective components at least **do**
12 : $F \leftarrow C_i$
13: **end if**
14: **end if**
15: **if** there only exists one connective component $C_j \in C$ satisfies $CR\,(i, j) = 1$ and $M_j > M_i$ **do**
16: $HC \leftarrow C_i$
17: end if
18: end for

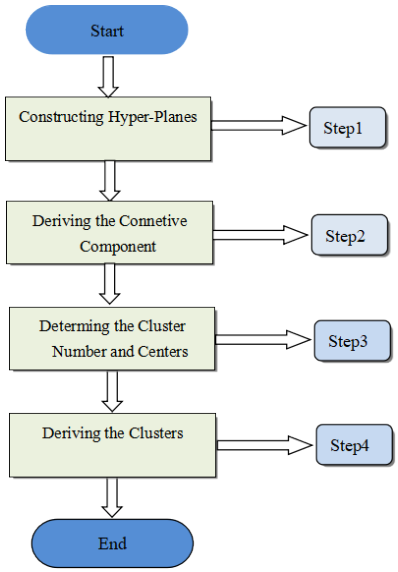


FIGURE 4. The Flow-Chart of the Algorithm.

Step I. Constructing hyper-planes. We proved that the average computation complexity of map $f_\delta : S \rightarrow L$ is $O(dn^3)$. The detailed proof is following:

Let $T(n)$ denotes the average computation complexity of map. Because the computational complexity of SVM and

Algorithm 4 Deriving the Clusters

Input: The connective component set $C = \{C_i\}$. The FPCC set F. The HCC set HC. The neighbor relationship matrix CR.
Output: The cluster result R.
Initialization: $R \leftarrow \emptyset$. $TR \leftarrow \emptyset$.
1: for each $C_i \in P$ **do**
2 : $R_i \leftarrow C_i$
3 : $Q \leftarrow \{C_i\}$
4: **while** Q is not empty **do**
5: select a connective component $C_j \in Q$
6 : $Q \leftarrow Q/\{C_j\}$
7: **for** each C_k which satisfies $C_k \in HC$ and $CR(k, j) = 1$ **do**
8 : $Q \leftarrow Q \cup \{C_k\}$
9 : $R_i \leftarrow R_i \cup C_k$
10 : **end for**
11 : **end while**
12 : $R \leftarrow R \cup \{R_i\}$
13: end for

K-means are $O(n^3)$ and $O(dn)$, we have

$$T(n) \leq \frac{1}{n}\sum_{i=1}^{n-1}(T(n-i)+T(i-1)) + c_1 n^3 = \frac{2}{n}\sum_{i=1}^{n-1} T(i) + c_1 n^3 \quad (4)$$

where c_1 is a constant number. Let

$$B(n) = \frac{2}{n}\sum_{i=1}^{n-1} T(i) + c_1 n^3 \quad (5)$$

we have

$$nB(n) - (n-1)B(n) = 2B(n-1) + c_1(n^4 - (n-1)^4) \quad (6)$$

Then

$$nB(n) - (n+1)B(n-1) = 4c_1 n^3 - 6c_1 n^2 + 4c_1 n - c_1 \quad (7)$$

Therefore

$$\frac{B(n)}{n+1} - \frac{B(n-1)}{n} \leq c_2 n \quad (8)$$

where c_2 is a constant number. Let

$$F(n) = \frac{B(n)}{n+1} \quad (9)$$

we can get $F(n) \leq c_2 n(n-1)/2$. Then we have

$$T(n) \leq c_3 n^3 \quad (10)$$

where c_3 is a constant number.

Therefore, the computational complexity of the Algorithm 1 is $O(dIn^3)$ where I is the number of the algorithm's iteration. Because $I \ll n$ in general, the average computational complexity of step 1 is $O(dn^3)$.

Step II. Deriving the connective components. As shown in Algorithm 2, the most time consuming process is computing the $adjM$. So the computational complexity of step 2 is $O(d|H|n^2)$. Because $|H| < n$, the computation complexity is less than $O(dn^3)$.

Step III. Determining the cluster number and centers. The computational complexity of computing (1) is $d\,|C_i|^3$. Because $\sum |C_i| = n$, the computational complexity of step 3 is no more than $O(dn^3)$.

Step IV. Deriving the clusters. As shown in Algorithm 4, the computational complexity of step 4 is $O(d|C|^2)$, which is less than $O(dn^2)$.

Based on the analysis of each step, the algorithm's computational complexity is $O(dn^3)$.

III. EXPERIMENTAL RESULTS

For benchmarking our algorithm, we applied the proposed algorithm on some synthetic data and some real-world datasets.

A. EXPERIMENTS WITH PRESET δ

Three synthetic datasets are generated for evaluating the algorithm's representable capability. Their distributions are two concentric circles, two interleaving moons, and swiss-roll as shown in Fig. 1. These manifolds are also the most commonly used ones for testing clustering algorithms. Each dataset contains 1600 data points involving with the Gaussian noises. The hyper-plane set $H = \varphi(TL)$. And the values of δ are set to be 110, 110, and 45, respectively. Experimental results are presented in Fig. 3 where it shows that the connective components are the ideal clusters. This method also can achieve ideal result on real-world datasets.

As pointed out by Rodriguez and Laio, the Olivetti Face Dataset (OFD) poses a serious challenge for algorithms to find the number of clusters automatically because the "ideal" number of clusters is comparable with the number of elements in the data set [9]. Our algorithm can also get ideal result on these data as shown in Fig. 5. It is derived by approaching the algorithm on the first 100 image in OFD where $H = TL$ and $\delta = 15$.

Meanwhile, these experimental results indicate that the parameter δ varies greatly as to the data distribution, the quantity of points and the clusters' number. In Sec. 3.B.2, we will discuss how to find its proper value in detail.

B. EXPERIMENTS ON REAL BENCHMARK DATASETS

1) THE DATASETS

This part introduces the real-world datasets. Among them, 20 benchmark datasets are obtained from different repositories [32]–[36]. Because different features of these data points have different meanings, the range of all features should be normalized to be same so that each feature contributes approximately proportionately to the final similarity measure. Instead of being standardized to standard deviation 1, values of each feature are normalized by using min-max

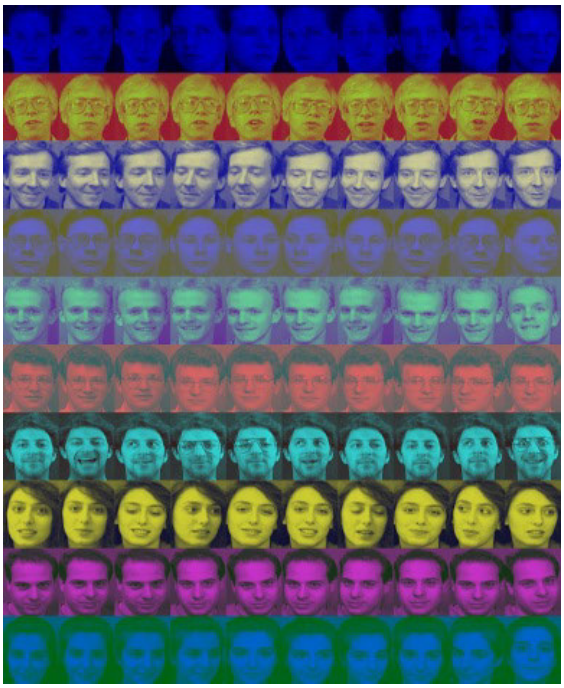


FIGURE 5. Visualization results on the first 100 images in OFD. Images in the same connective components are with same colors.

normalization except for those image datasets. Because of the reason mentioned in Sec. 3.1, the OFD is also included in the datasets. Table 1 contains a summary of these datasets. All the data are centralized by shifting the origin to their mean before being processed.

2) THE PARAMETERS' SETTING

There are two parameters H and δ in the algorithm. The hyper-plane dataset H is easily to be set. In all these experiments they all are TL. As shown in Sec. 3.A, δ is important and difficult to be set. Here, we present a scheme to determine it automatically. It includes two parts. The first step gets its approximate range. The second one determines the value.

Step I. set δ to be big enough firstly. Then we can get the connective components and calculate the average values of FPCCs' M_f. Let $\delta = \delta/2$. Repeat the operation until δ is small enough. Fig. 6 is the bar charts results. The value of X-axis and Y-axis are the divided time and the average of FPCCs' M_f, respectively. The minimum and maximum of the range are set to be the two neighbors (blue spots in Fig. 6) of the last valley point (red spot in Fig. 6). When δ's proper value is small, there may be no valley point in the line chart as shown in the Fig.6.s and Fig.6.t. Under this circumstance, the minimum and maximum values are set to be the last two points.

Step II. After the range of δ is gotten, its value can be found based the quantity of points assigned into clusters. Fig. 7 is the line charts by adaptively sampling in the range. The value of X-axis and Y-axis are δ and quantity of the points in clusters, respectively. The last valley points are set to be the value of δ, as the red spots shown in Fig. 7. It indicates that this method is effective although the proposed algorithm doesn't obtain the best results always. After δ is determined, the algorithm is repeated 5 times to derive the best result.

TABLE 1. Summary of datasets.

Datasets	Num of Instances	Dimensions	Classes
BreastTissue	106	9	6
Wine	178	13	3
Parkinson	195	22	2
Seeds	210	7	3
Glass	213	9	7
Vertebral	310	6	3
Leaf	340	14	36
Dermatology	358	34	6
Synthetic	600	60	6
R15	600	2	15
S1_001S1	10000	2	30
AI4I 2020 MP	10000	11	3
Imm	120	600×800	12
Jaffe	213	256×256	10
Jaffe with noise	213	256×256	10
Coil20	1440	128×128	20
Coil20 with noise	1440	128×128	20
USPS	1854	16×16	10
OFD	400	112×92	40
OFD with noise	400	112×92	40

3) THE EXPERIMENTAL RESULTS AND ANALYSIS

This part presents the experimental results. Before it is given, the compared methods and evaluation measures are briefly introduced.

Our algorithm only assigns part of the points into the different categories. Other points are viewed as outliers. So it is compared with the related algorithms CFSFDP and DBSCAN which also can find outliers. Whereas CFSFDP and DBSCAN are both sensible to their main parameters' values, their results are derived by adaptively sampling their main parameters' space. These parameters are CFSFDP's dc and DBSCAN's eps and min_samples. CFSFDP's dc and DBSCAN's eps are the thresholds for determining the adjacent relationship. Only when the distance between two points is smaller than them, these two points are considered to be neighbors. Min_samples is the number of samples (or total weight) in a neighborhood for a point to be considered as a core point. These three parameters are the most important ones for their algorithms. All values of the other parameters like the methods for distance computing or the nearest neighbor search are set to be default ones.

Fig.8 is the visualization of the results on OFD. The result of DBSCAN is derived by adjusting two main parameters eps and min_samples to generate a best result which contains the points nearly as many as ours. The result of CFSFDP is selected in the work [9], which is the best result. Both CFSFDP and our algorithm contained no single cluster included images of two different categories while DBSCAN does. CFSFDP and our algorithm got 22 and 26 correct clusters respectively.

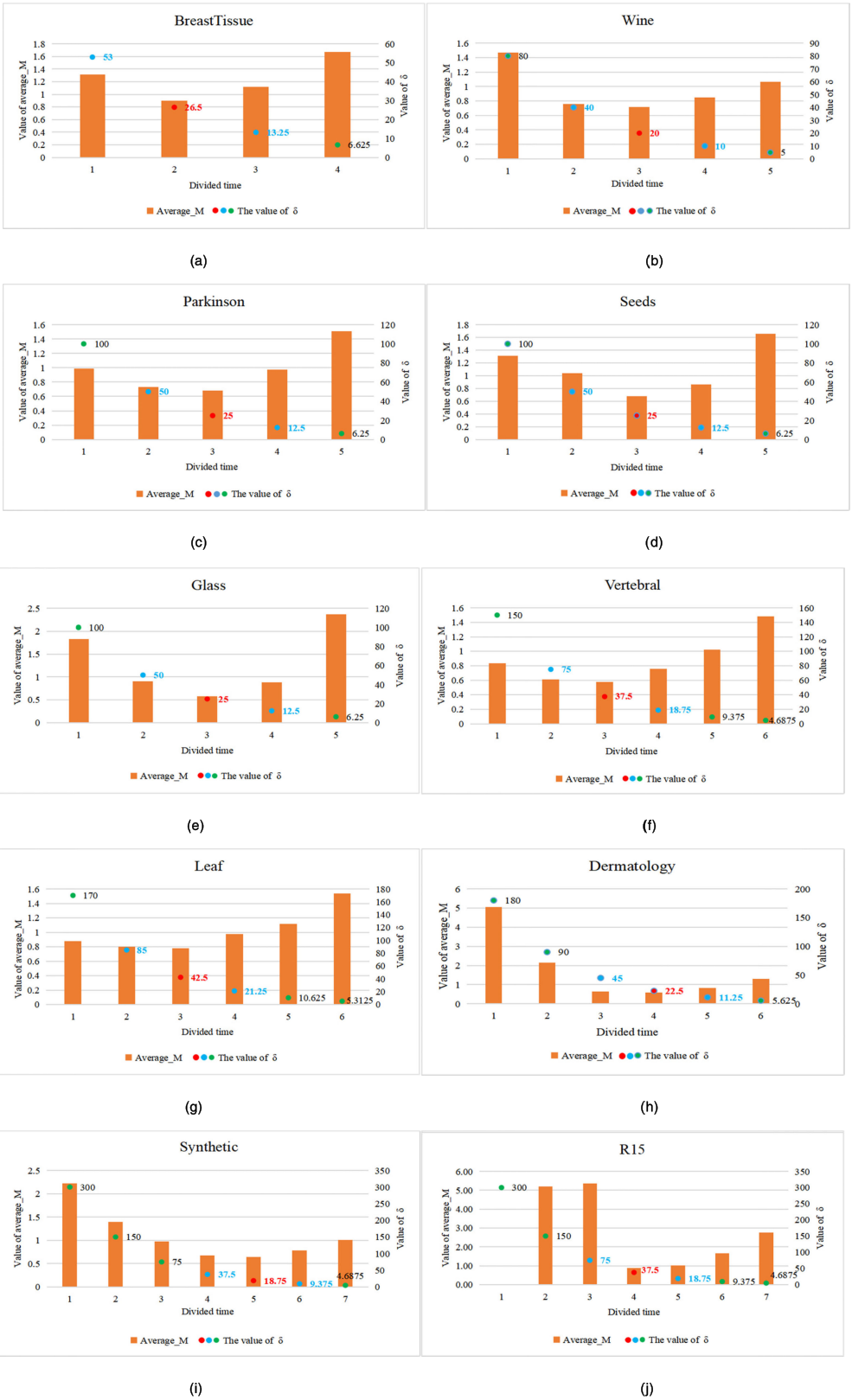


FIGURE 6. The relationship between the average values of FPCCs and δ. The orange bar is the average value of the FPCCs' M. The green, blue and red spots are the value of the δ. (a) Breast Tissue. (b) Wine. (c) Parkison. (d) Seeds. (e) Glass. (f) Vertabral. (g) Leaf. (h) Dermatology. (i) Synthetic. (j) R15. (k) S1_001S1. (l) AI4I 2020 Mp. (m) Imm. (n) Jaffe. (o) Jaffe with noise. (p) Coil20. (q) Coil20 with noise. (r) USPS. (s) OFD. (t) OFD with noise.

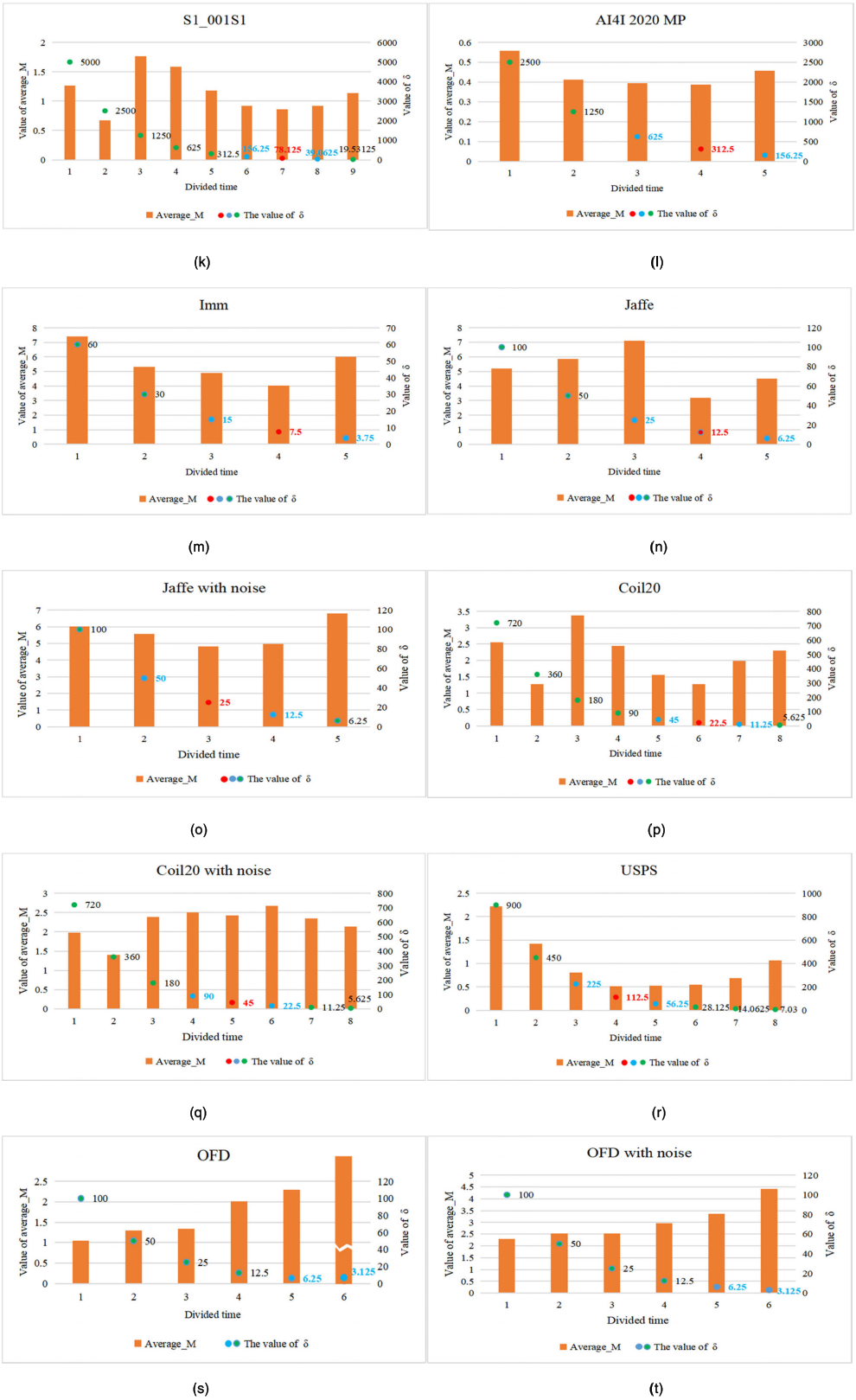


FIGURE 6. *(Continued.)* **The relationship between the average values of FPCCs and δ. The orange bar is the average value of the FPCCs' M. The green, blue and red spots are the value of the δ. (a) Breast Tissue. (b) Wine. (c) Parkison. (d) Seeds. (e) Glass. (f) Vertabral. (g) Leaf. (h) Dermatology. (i) Synthetic. (j) R15. (k) S1_001S1. (l) AI4I 2020 Mp. (m) Imm. (n) Jaffe. (o) Jaffe with noise. (p) Coil20. (q) Coil20 with noise. (r) USPS. (s) OFD. (t) OFD with noise.**

FIGURE 7. The Performance of the algorithm as a function of δ: ARI (orange bar), NMI (gray bar) and quantity of points in clusters (blue line). The red spot is the value of δ which is selected for clustering algorithm. (a) Breast Tissue. (b) Wine. (c) Parkison. (d) Seeds. (e) Glass. (f) Vertabral. (g) Leaf. (h) Dermatology. (i) Synthetic. (j) R15. (k) S1_001S1. (l) AI4I 2020 Mp. (m) Imm. (n) Jaffe. (o) Jaffe with noise. (p) Coil20. (q) Coil20 with noise. (r) USPS. (s) OFD. (t) OFD with noise.

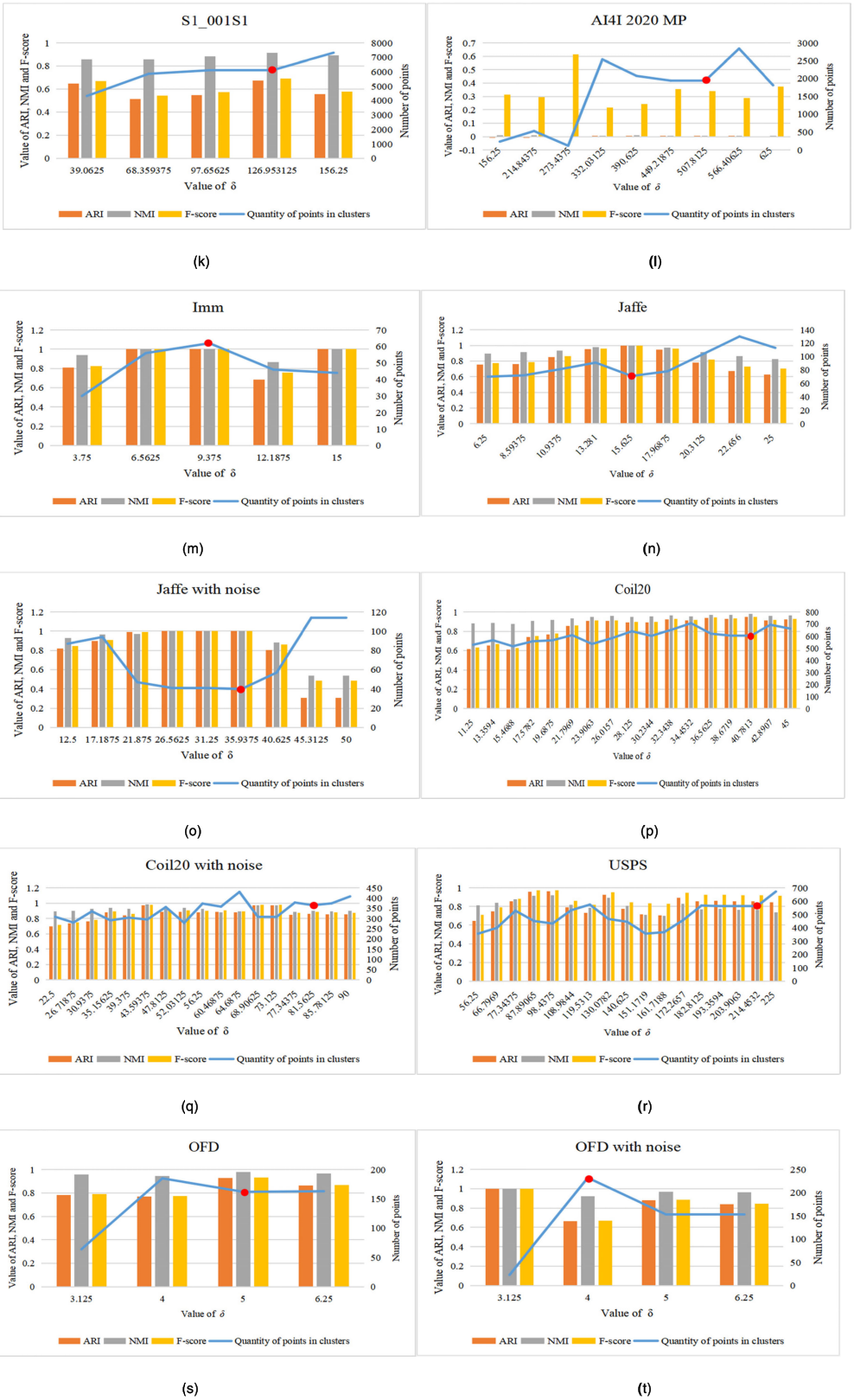
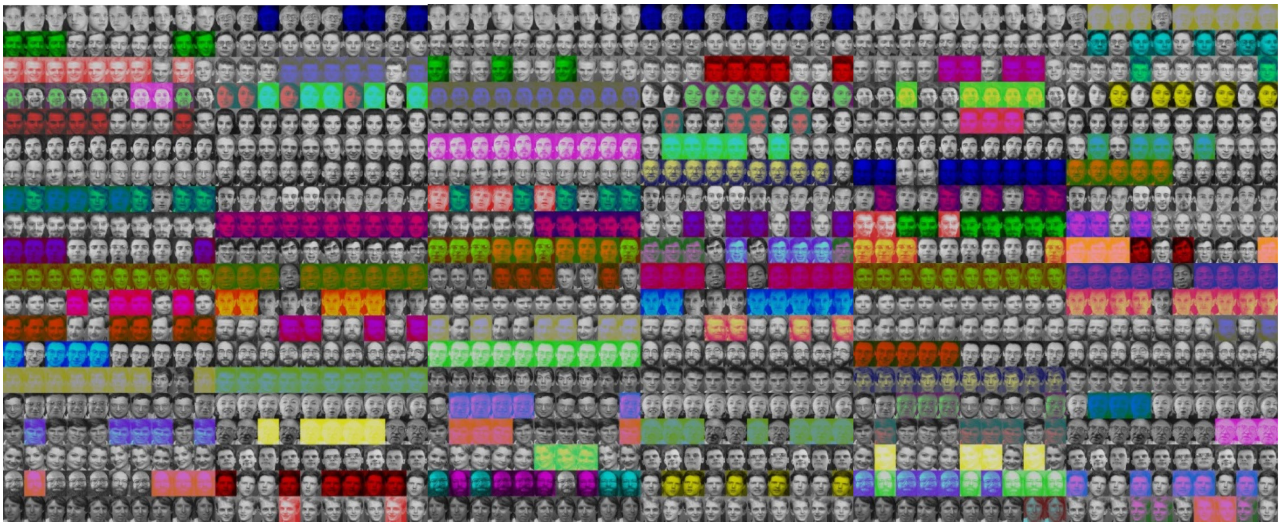


FIGURE 7. *(Continued.)* **The Performance of the algorithm as a function of δ: ARI (orange bar), NMI (gray bar) and quantity of points in clusters (blue line). The red spot is the value of δ which is selected for clustering algorithm. (a) Breast Tissue. (b) Wine. (c) Parkison. (d) Seeds. (e) Glass. (f) Vertabral. (g) Leaf. (h) Dermatology. (i) Synthetic. (j) R15. (k) S1_001S1. (l) AI4I 2020 Mp. (m) Imm. (n) Jaffe. (o) Jaffe with noise. (p) Coil20. (q) Coil20 with noise. (r) USPS. (s) OFD. (t) OFD with noise.**



(a) (b) (c)

FIGURE 8. Pictorial clusters' results on OFD. (a) DBSCAN. (b) CFSFDP. (c) Our Algorithm. Faces with same color belong to the same cluster while the gray images are not in any cluster.

TABLE 2. ARI on Real-world datasets.

Datasets	DBSCAN	CFSFD	Our algorithm
BreastTissue	0.15	0	**0.22**
Wine	0.96	0.69	**0.97**
Parkinson	0.38	0.36	**0.43**
Seeds	**0.94**	0.89	0.86
Glass	0.24	0.19	**0.38**
Vertebral	0.2	**0.64**	0.38
Leaf	0.25	**0.32**	0.15
Dermatology	0.74	0.40	**0.98**
Synthetic	0.93	**0.94**	0.80
R15	**1**	**1**	0.995
S1_001S1	0.17	0.05	**0.67**
AI4I 2020 MP	0	0	**0.01**
Imm	**1**	0.92	**1**
Jaffe	0.98	0.95	**1**
Jaffe with noise	0.33	0.76	**1**
Coil20	**0.95**	0.63	**0.95**
Coil20 with noise	0.70	0.27	**0.86**
USPS	**0.98**	0.76	0.86
OFD	0.92	0.91	**0.93**
OFD with noise	0.49	0.46	**0.66**

TABLE 3. NMI on Real-world datasets.

Datasets	DBSCAN	CFSFD	Our algorithm
BreastTissue	0.40	0	**0.44**
Wine	0.95	0.68	**0.96**
Parkinson	0.34	0.34	**0.41**
Seeds	**0.91**	0.86	0.84
Glass	**0.51**	0.38	0.47
Vertebral	0.32	**0.63**	0.30
Leaf	**0.73**	0.69	0.50
Dermatology	0.89	0.60	**0.96**
Synthetic	**0.92**	**0.92**	0.86
R15	**1**	**1**	0.996
S1_001S1	0.75	0.40	**0.91**
AI4I 2020 MP	0	0	0
Imm	**1**	0.96	**1**
Jaffe	0.98	0.98	**1**
Jaffe with noise	0.54	0.90	**1**
Coil20	0.96	0.84	**0.98**
Coil20 with noise	0.85	0.59	**0.90**
USPS	**0.95**	0.63	0.76
OFD	**0.98**	**0.98**	**0.98**
OFD with noise	0.89	0.81	**0.92**

For presenting the quantitative results, the performances are measured by three benchmark measures Adjusted Rand Index (ARI) [37], Normalized Mutual information (NMI) [38] and F-Score. ARI and NMI range from $[-1,1]$ and $[0,1]$, respectively. The lager the values of ARI and NMI, the better the clustering results. 1 means the clustering results are identical. 0 or -1 means that the two set of clusters are independent. F-Score is a way of combining the precision and recall of the cluster, and it is defined as the harmonic mean of the model's precision and recall as following:

$$F - Score = 2 \times \frac{precesion \times recall}{precesion + recall}. \tag{11}$$

Table 2, Table 3 and Table 4 are the ARI, NMI and F-Score results which are derived by carrying the three algorithms on the datasets.

The performances of CFSFDP and DBSCAN are related to the quantity of these points put into clusters. The fewer the quantity of points, the better the clustering results usually. Therefore, the quantity of points in CFSFDP's and DBSCAN's results is set to be more than 3/4 of those in our algorithm's results. The parameters' ranges and sampling steps are shown in Table 5, Table 6, and Table 7. Although our algorithm did not always get a better ARI,

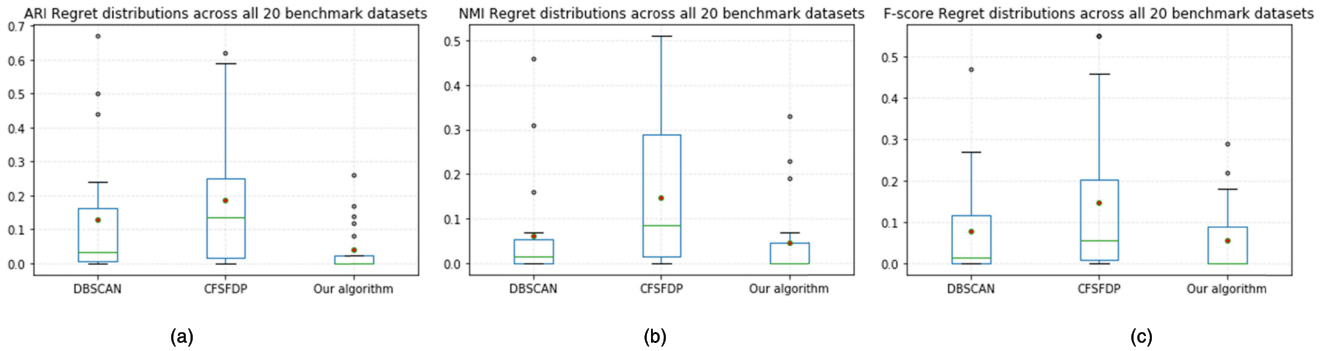


FIGURE 9. Results of Regret across all 20 datasets. The red spots represent the mean. The green lines represent the median. (a) ARI Regret. (b) NMI Regret. (c) F-Score Regret.

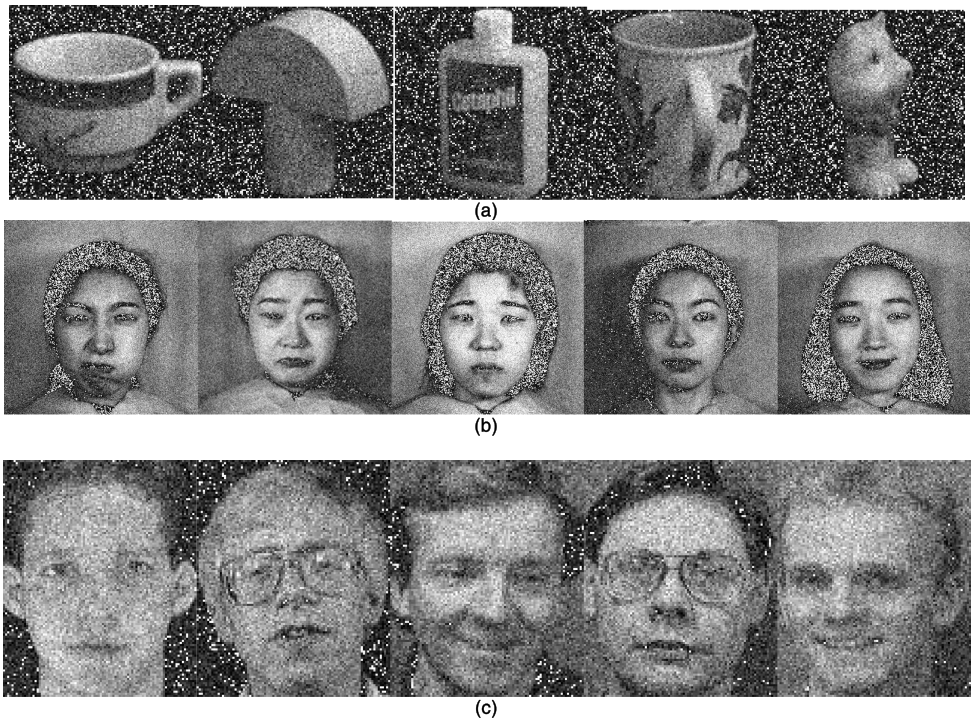


FIGURE 10. Noise images in three image datasets. (a) Coil20. (b) Jaffe. (c) OFD.

NMI and F-Score, it obtained the best performance than the others.

It shows that the proposed algorithm is more robust because all its parameters are set automatically. Meanwhile, CFSFDP's and DBSCAN's results are the best ones which are selected manually. And Table 5, Table 6, and Table 7 show that the main parameters of CFSFDP and DBSCAN vary greatly under different circumstances. The experiments also reveal that the method for deriving the approximated range is robust. As shown in Fig. 7, the performance in the selected range of δ is usually high in more than half of the examples.

Meanwhile, our results are more stable, which means that the algorithm can deal with the data distributed on more complex manifolds. For demonstrating its stableness, the measure *regret* is adopted. When an algorithm is carried on s specific dataset, the regret is the difference between this algorithm's index and the highest index from among all algorithms which are approached to the same dataset [21]. Fig. 9 contains the boxplots of the three algorithms carried on all the datasets. The mean and median of regret for ARI, NMI and F-Score of the proposed algorithm are both best.

The most important step of the algorithm is constructing hyper-planes. If the hyper-planes segment the points well, we may get the ideal results as shown in Fig. 3 and Fig. 5. However, it is not easy to find all the proper hyper-planes because the marginal between points may be not clear. Three reasons may lead to a narrow marginal. They are noise, dense data points and high dimensions of data. So we tested our algorithm on the datasets such as two large scale datasets

TABLE 4. F-Score on Real-world datasets.

Datasets	DBSCAN	CFSFD	Our algorithm
BreastTissue	0.36	0.28	**0.45**
Wine	**0.98**	0.80	**0.98**
Parkinson	**0.82**	0.80	0.60
Seeds	**0.96**	0.93	0.91
Glass	0.57	0.54	**0.58**
Vertebral	**0.62**	0.79	0.61
Leaf	0.31	**0.36**	0.21
Dermatology	0.80	0.52	**0.98**
Synthetic	0.95	**0.97**	0.83
R15	**1**	**1**	0.995
S1_001S1	0.22	0.14	**0.69**
AI4I 2020 MP	**0.63**	**0.63**	0.34
Imm	**1**	0.93	**1**
Jaffe	0.98	0.96	**1**
Jaffe with noise	0.73	0.79	**1**
Coil20	**0.96**	0.67	0.95
Coil20 with noise	0.77	0.33	**0.88**
USPS	**0.99**	0.85	0.92
OFD	**0.93**	0.92	**0.93**
OFD with noise	0.53	0.47	**0.67**

TABLE 5. CFSFDP's dc's space and sampling step.

Datasets	Min d_c	Max d_c	Step size	Best d_c
BreastTissue	10	9000	10	10
Wine	10	80	1	59
Parkinson	1	32	1	27
Seeds	0.1	2	0.1	0.7
Glass	0.1	2.5	0.1	1.7
Vertebral	1.5	20	0.5	4
Leaf	0.03	0.33	0.01	0.18
Dermatology	1	5	0.1	4.1
Synthetic	5	50	1	46
R15	0.01	0.5	0.01	0.43
S1_001S1	10000	540000	10000	130000
AI4I 2020 MP	100	1500	100	500
Imm	0.02	0.20	0.01	0.19
Jaffe	0.01	0.20	0.01	0.14
Jaffe with noise	0.05	0.25	0.01	0.18
Coil20	0.001	0.1	0.001	0.027
Coil20 with noise	0.05	0.23	0.01	0.19
USPS	0.1	0.3	0.01	0.25
OFD with noise	0.05	0.14	0.01	0.13

TABLE 6. DBSCAN's Min_samples' space and sampling step.

Datasets	Min min_sample	Max min_sample	Step size	Best min_sample
BreastTissue	3	30	1	3
Wine	3	25	1	21
Parkinson	3	15	1	10
Seeds	3	25	1	18
Glass	3	15	1	4
Vertebral	3	15	1	11
Leaf	3	10	1	3
Dermatology	3	30	1	21
Synthetic	3	120	1	10
R15	3	30	1	6
S1_001S1	30	200	5	150
AI4I 2020 MP	10	500	10	240
Imm	3	15	1	5
Jaffe	3	15	1	4
Jaffe with noise	3	30	1	3
Coil20	3	20	1	5
Coil20 with	3	30	1	4
USPS	3	20	1	5
OFD	3	10	1	4
OFD with noise	3	20	1	3

TABLE 7. DBSCAN's Eps's space and sampling step.

Datasets	Min eps	Max eps	Step size	Best eps
BreastTissue	0.1	10	0.1	2.6
Wine	1	10	1	5
Parkinson	1	4.7	0.1	4.5
Seeds	1.5	3.5	0.1	2.5
Glass	0.1	3.5	0.1	1.6
Vertebral	0.5	1.5	0.1	1.1
Leaf	0.1	2.0	0.1	1.3
Dermatology	5	18	0.1	12
Synthetic	5	15	1	9
R15	0.1	5	0.1	0.2
S1_001S1	0.1	2.5	0.1	0.6
AI4I 2020 MP	0.5	10	0.5	1.5
Imm	15000	20000	200	18000
Jaffe	4500	6500	100	6400
Jaffe with noise	15000	18000	50	16150
Coil20	1800	4500	100	1900
Coil20 with noise	8000	13000	100	9600
USPS	0.5	3.2	0.1	2.9
OFD	2500	3500	50	2850
OFD with noise	4500	6500	50	5100

S1_001S1 and AI2020 and two high dimensional dataset Imm and Jaffe. We also construct three datasets by adding heavy Gaussian noise to the three image datasets Jaffe, Coil20 and OFD as shown in the Fig 10. The experimental results indicate that our algorithm is still valid for these dataset. The reason is that the SVM in our algorithm is based on the soft margin formulation that is a most commonly used formulation for SVM. It allows SVM to make a certain number of mistakes and keep margin as wide as possible. So other points can still be classified correctly.

The proposed algorithm's main problem is that the time complexity is high. When it is approached to the large scale dataset, the algorithm's consuming time increases greatly as shown in Table 8. Because the connective components are segmented by linear structures, we can use the Euclidean distance to approximate geodesic distance instead of the multi-view methods [39]. So in the experiments, we use Euclidean distance to replace $d_g\left(x_i, x_j\right)$ in (1) when it is approached on the two large scale datasets S1_001S1 and AI4I 2020 MP. Besides, the running time of our algorithm is less than CFSFDP when it is approached on the image datasets. The

TABLE 8. The Algorithms' running time (S).

Datasets	DBSCAN	CFSFD	Our algorithm
BreastTissue	0.03	0.63	2.53
Wine	0.02	0.98	4.94
Parkinson	0.02	1.28	6.98
Seeds	$\leq$0.01	1.12	4.37
Glass	$\leq$0.01	1.18	5.42
Vertebral	$\leq$0.01	1.93	9.02
Leaf	0.02	2.48	11.78
Dermatology	0.02	3.47	19.47
Synthetic	0.09	11.93	74.21
R15	$\leq$0.01	4.23	89.00
S1_001S1	0.22	1032.37	48969.71
AI4I 2020 MP	1.65	1177.86	10949.34
Imm	26.36	20861.6	1149.64
Jaffe	13.88	5419.38	290.12
Jaffe with noise	13.25	4930.44	244.70
Coil20	110.72	93678.8	1495.45
Coil20 with noise	161.88	103734.	1380.13
USPS	3.00	9959.95	371.87
OFD	6.85	5416.81	260.44
OFD with noise	7.70	4199.67	608.05

reason is that it measures the similarity between images with complex wavelet structural similarity (CW-SSIM) index [40] as suggested by Rodriguez and Laio [9]. Its time complexity is high.

IV. CONCLUSION AND FUTURE WORK

To find data's intrinsic patterns, one common used idea is to approximate nonlinear models by using linear ones. Our algorithm is based on this idea. We give a new method for constructing hyper-planes. The experimental results and analysis indicate that it can find the proper hyper-planes to segment points. By combining the linear structures, the final results can approximate the nonlinear manifolds more flexibly and accurately. We also present a scheme to determine the parameters' values. The experiments shows that it is robust to be applied.

The main problem is that our algorithm's computational complexity is high. It limits the approach for large scale data. Based on some existed methods [41], the dense region of points may be found more efficiently. It is our next direction in the future work.

Another problem is that our algorithm only put part of points into categories. How to derive the clusters which contain all the data points will also be investigated in the future work.

REFERENCES

[1] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, 1967, pp. 281–297.
[2] L. Kaufman and P. J. Rousseeuw, "Clustering by means of medoids," in *Statistical Data Analysis Based on the L1 Norm and Related Methods*, Y. Dodge, Ed. Basel, Switzerland: Birkhäuser Verlag, 1987, pp. 405–416.
[3] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, Sep. 1967.
[4] I. Gurrutxaga, I. Albisua, O. Arbelaitz, J. I. Martín, J. Muguerza, J. M. Pérez, and I. Perona, "SEP/COP: An efficient method to find the best partition in hierarchical clustering based on a new cluster validity index," *Pattern Recognit.*, vol. 43, no. 10, pp. 3364–3373, Oct. 2010.
[5] K. Mali and S. Mitra, "Clustering and its validation in a symbolic framework," *Pattern Recognit. Lett.*, vol. 24, no. 14, pp. 2367–2376, Oct. 2003.
[6] S. Yue, J.-S. Wang, T. Wu, and H. Wang, "A new separation measure for improving the effectiveness of validity indices," *Inf. Sci.*, vol. 180, no. 5, pp. 748–764, Mar. 2010.
[7] S. Zhou, Z. Xu, and F. Liu, "Method for determining the optimal number of clusters based on agglomerative hierarchical clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 12, pp. 3007–3017, Dec. 2017.
[8] 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. 2nd Int. Conf. Knowl. Discovery Data Mining*, 1996, pp. 323–333.
[9] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, Jun. 2014.
[10] Y. He, Y. Wu, H. Qin, J. Z. Huang, and Y. Jin, "Improved I-nice clustering algorithm based on density peaks mechanism," *Inf. Sci.*, vol. 548, pp. 177–190, Feb. 2021.
[11] B. Tu, X. Yang, N. Li, C. Zhou, and D. He, "Hyperspectral anomaly detection via density peak clustering," *Pattern Recognit. Lett.*, vol. 129, pp. 144–149, Jan. 2020.
[12] M. A. Masud, J. Z. Huang, C. Wei, J. Wang, I. Khan, and M. Zhong, "I-nice: A new approach for identifying the number of clusters and initial cluster centres," *Inf. Sci.*, vol. 466, pp. 129–151, Oct. 2018.
[13] 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.
[14] 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.
[15] L. Wang, W. Zhou, H. Wang, M. Parmar, and X. Han, "A novel density peaks clustering halo node assignment method based on K-nearest neighbor theory," *IEEE Access*, vol. 7, pp. 174380–174390, 2019.
[16] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, Feb. 2007.
[17] C. Panagiotakis, "Point clustering via voting maximization," *J. Classification*, vol. 32, no. 2, pp. 212–240, Jul. 2015.
[18] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Trans. Inf. Theory*, vol. 21, no. 1, pp. 32–40, Jan. 1975.
[19] W. Fan and N. Bouguila, "Spherical data clustering and feature selection through nonparametric Bayesian mixture models with von Mises distributions," *Eng. Appl. Artif. Intell.*, vol. 94, Art. no. 103781, Sep. 2020.
[20] W. Fan, N. Bouguila, J.-X. Du, and X. Liu, "Axially symmetric data clustering through Dirichlet process mixture models of Watson distributions," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 6, pp. 1683–1694, Jun. 2019.
[21] D. P. Hofmeyr, "Clustering by minimum cut hyperplanes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1547–1559, Aug. 2017.
[22] K. Zhang, I. W. Tsang, and J. T. Kwok, "Maximum margin clustering made practical," *IEEE Trans. Neural Netw.*, vol. 20, no. 4, pp. 583–596, Apr. 2009.
[23] T. Li, G. Kou, Y. Peng, and Y. Shi, "Classifying with adaptive hyperspheres: An incremental classifier based on competitive learning," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 4, pp. 1218–1229, Apr. 2020.
[24] B. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annu. Workshop Comput. Learn. Theory*, Pittsburgh, PA, USA, 1992, pp. 144–152.
[25] M. C. Thrun and A. Ultsch, "Using projection-based clustering to find distance- and density-based clusters in high-dimensional data," *J. Classification*, 2020, doi: 10.1007/s00357-020-09373-2.
[26] D. Tan, W. Zhong, C. Jiang, X. Peng, and W. He, "High-order fuzzy clustering algorithm based on multikernel mean shift," *Neurocomputing*, vol. 385, pp. 63–79, Apr. 2020.
[27] R. C. de Amorim and C. Hennig, "Recovering the number of clusters in data sets with noise features using feature rescaling factors," *Inf. Sci.*, vol. 324, pp. 126–145, Dec. 2015.
[28] T. Calinski and J. Harabasz, "A dendrite method for cluster analysis," *Commun. Statist.-Theory Methods*, vol. 3, no. 1, pp. 1–27, 1974.

[29] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, no. 2, pp. 224–227, Apr. 1979.
[30] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, Nov. 1987.
[31] E. Zhu and R. Ma, "An effective partitional clustering algorithm based on new clustering validity index," *Appl. Soft Comput.*, vol. 71, pp. 608–621, Oct. 2018.
[32] National Science Foundation. (2018). *University of California Irvine Machine Learning Repository*. [Online]. Available: https://archive.ics.uci.edu/ml/index.php
[33] M. J. Lyons, M. Kamachi, and J. Gyoba, "Coding facial expressions with Gabor wavelets (IVC special issue)," 2020, *arXiv:2009.05938*. [Online]. Available: http://arxiv.org/abs/2009.05938
[34] T. Qiu and Y. Li. (2020). *Clustering-by-InTree-Ensemble-PR2020*. [Online]. Available: https://github.com/Teng-Qiu-Clustering
[35] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (COIL-100)," Dept. Comput. Sci., Center Res. Intell. Syst., Columbia Univ., New York, NY, USA, Tech. Rep. CUCS-006-96, 1996.
[36] J. Fagertun and M. B. Stegmann. (2005). *The IMM Frontal Fa e Database*. [Online]. Available: http://www2.imm.dtu.dk/pubdb/edoc/imm3943.pdf
[37] L. Hubert and P. Arabie, "Comparing partitions," *J. Classification*, vol. 2, no. 1, pp. 193–218, Dec. 1985.
[38] M. Meilă, "Comparing clusterings—An information based distance," *J. Multivariate Anal.*, vol. 98, no. 5, pp. 873–895, May 2007.
[39] G. Zhang, Y. Zhou, X. He, and C. Wang, "Huang Dong. One-step kernel multi-view subspace clustering," *Knowl.-Based Syst.*, vol. .189, no. 15, pp. 1–14, 2020.
[40] M. P. Sampat, Z. Wang, S. Gupta, A. C. Bovik, and M. K. Markey, "Complex wavelet structural similarity: A new image similarity index," *IEEE Trans. Image Process.*, vol. 18, no. 11, pp. 2385–2401, Nov. 2009.
[41] Y. Chen, X. Hu, W. Fan, L. Shen, Z. Zhang, X. Liu, J. Du, H. Li, Y. Chen, and H. Li, "Fast density peak clustering for large scale data based on kNN," *Knowl.-Based Syst.*, vol. 187, Jan. 2020, Art. no. 104824.

LUHONG DIAO was born in Zibo, Shandong, China, in 1978. He received the B.S. and M.S. degrees in computer science from Shandong University, Jinan, in 2000 and 2003, respectively, and the Ph.D. degree in computer science and technology from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, in 2007.

From 2007 to 2010, he was a Lecturer with the College of Applied Sciences, Beijing University of Technology. Since 2010, he has been an Associate Professor with the College of Mathematics, Beijing University of Technology. He is the author of more than 30 articles. His research interests include machine learning and computer vision. He is currently a member of the Chinese Computer of Federation, the Chine Graphics Society, the China Society of Industrial and Applied Mathematics, and ACM.

MANMAN DENG was born in Zhoukou, Henan, China, in 1996. She received the bachelor's degree in mathematics from the North University of China, in 2018. She is currently pursuing the master's degree with the Faulty of Sciences, Beijing Institute for Scientific and Engineering Computing, Beijing University of Technology.

She is interested in problems in computer vision and machine learning. She is also a Student Member of the China Society of Industrial and Applied Mathematics.

JINYING GAO was born in Qiqihar, Heilongjiang, China, in 1994. She received the bachelor's degree from Qufu Normal University, in 2017, and the master's degree from the Beijing University of Technology, in 2020.

Since 2020, she has been a Research Assistant with the Institute of Automation, Chinese Academy of Sciences. She is interested in problems in computer vision and machine learning.