Received January 22, 2019, accepted March 5, 2019, date of publication March 12, 2019, date of current version April 1, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2904254

Density Peaks Clustering Based on Weighted Local Density Sequence and Nearest Neighbor Assignment

DONGHUA YU[1], GUOJUN LIU[1], MAOZU GUO[1,2,3], XIAOYAN LIU[1], AND SHUANG YAO[4]
[1]School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China
[2]School of Electrical and Information Engineering, Beijing University of Civil Engineering and Architecture, Beijing 100044, China
[3]Beijing Key Laboratory of Intelligent Processing for Building Big Data, Beijing 100044, China
[4]College of Economics and Management, China Jiliang University, Hangzhou 310018, China

Corresponding authors: Guojun Liu (hitliu@hit.edu.cn) and Maozu Guo (guomaozu@bucea.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61571164, Grant 61871020, Grant 61532014, Grant 61571163, and Grant 61671189, in part by the Key Science and Technology Plan Project of Beijing Municipal Education Commission of China under Grant KZ201810016019, in part by the High Level Innovation Team Construction Project of Beijing Municipal Universities under Grant IDHT20190506, and in part by the National Key Research and Development Plan Task of China under Grant 2016YFC0901902-5.

ABSTRACT Density peaks clustering (DPC) is a density-based clustering algorithm with excellent clustering performance including accuracy, automatically detecting the number of clusters, and identifying center points. However, the local density of DPC strongly depends on the cutoff distance which must be prespecified; in addition, the strategy assigns each remaining point to the same cluster as its nearest neighbor of higher density in descending order of local density, which is likely to cause cluster label error propagation. To overcome these limitations, we propose an improved DPC by introducing weighted local density sequence and two-stage assignment strategies, called DPCSA. Many previous improved DPC algorithms neglect additional complexity, whereas DPCSA incorporates the nearest neighbor dynamic table to enhance clustering efficiency. The experimental results for 12 artificial and 11 real-world datasets, including Olivetti face, verify that the DPCSA clustering performance is significantly superior to DPC and DPC via heat diffusion (HDDPC), and slightly superior to fuzzy weighted k-nearest neighbors density peak clustering (FKNNDPC). In addition, the DPCSA is more computationally efficient than FKNNDPC and HDDPC, but less than DPC. The source code of DPCSA is available at https://github.com/Yu123456/DPCSA.

INDEX TERMS Cluster analysis, density peaks, K-nearest neighbors, local density, nearest neighbor dynamic table.

I. INTRODUCTION

Cluster analysis is the process of dividing a set of points into non-overlapping subsets. Each subset is a cluster, such that points in the same cluster are similar to one another (intra-similarity) and dissimilar to points in other clusters (inter-dissimilarity) [1]. Clustering algorithms are generally categorized into five groups: hierarchical [2], [3], partitional-based [4]–[6], density-based [7], grid-based [8], and model-based [9] algorithms. Various clustering ensemble algorithms have also been proposed [10]. Clustering is widely used in research and engineering applications.

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

Density-based clustering algorithms are an important subclass of cluster analysis algorithms, and are very popular since they can define clusters with arbitrary shape and handle outliers well. This class of algorithms include DBSCAN [7], DBCURE-MR [11], fast DBSCAN [12], ST-DBSCAN [13], OPTICS [14], DENCLUE [15], etc. Rodriguez and Laio recently proposed a clustering by fast search and finding density peaks (DPC) [16], based on the assumptions that cluster center points are surrounded by neighbors with lower local density and they are relatively distant from points with higher local density. However, local density and distance for each point are defined relative to the prespecified parameter cutoff distance d_c. Center points must be determined manually using decision graph with significant separation between center and non-center points. One particular DPC advantage is that the number of clusters need not be explicitly specified. Many previous studies with different datasets have shown that DPC has achieved a excellent clustering performance and been widely adopted in many fields. For example, initial social circle clustering considering user distance [17], dividing cluster configurations based on measured sample mutual distance [18], [19], clustering cells using Student's *t* distributed stochastic neighbor embedded two-dimensional (2D) representation [20], and sub-image clustering and capturing local information in the label formation phase [21].

However, DPC has two main and significant shortcomings.

1) Cutoff distance must be prespecified based on users' experience and prior knowledge of the dataset. For benchmark datasets (e.g. datasets from the UCI repository), users can optimize d_c by repeating experiments using the known actual class labels. However, generally real class labels are unknown in the application to real problems.
2) The DPC assignment strategy assigns remaining points (i.e., non-center points) in descending order of local density to the same cluster as its nearest neighbor of higher density. This is highly likely to cause cluster label error propagation. Once a point is assigned to an incorrect cluster, other lower local density nearest neighbor points will also be assigned to the incorrect cluster. This shortcoming has been highlighted and discussed in detail for the Spiral dataset (Reference [22], Fig. 1(c)) even for well-designed cutoff distance $d_c = 1$ using exponential kernel.

In this paper, our main motivations are to avoid using prespecified parameter(s), reduce assignment error propagation, and improve clustering efficiency. Therefore, we propose DPC based on weighted local density sequence and nearest neighbor assignment (DPCSA). The main innovations and contributions can be summarized as follows.

1) Local density is divided into two terms, the fixed k-nearest neighbors (KNN) term ($k = 5$) and the weighted sequence term, to avoid requiring a prespecified parameter.
2) The assignment strategy is split into two stages using a boundary condition to reduce assignment error propagation. All points that satisfy the boundary condition are assigned in the first stage. Remaining points are assigned in the second stage based on a nearest neighbor assignment strategy that disrupts the local density descending assignment order.
3) We propose nearest neighbor dynamic table (NNDT) to enhance clustering efficiency, i.e., to speed up the second stage assignment strategy. This aspect has been largely ignored in previous improved DPC algorithms.
4) Experimental results, including 12 artificial datasets, 11 real-world datasets, and Olivetti face dataset, verify that DPSCA clustering performance and computationally efficient are improved than some state-of-the-art algorithms.

The remainder of this paper is organized as follows. Section II describes the research progress related to DPC considered in the literature. Section III briefly introduces DPC. Section IV describes the proposed DPCSA algorithm in detail, and Section V considers time complexity. Section VI provides detailed experiment setup and discusses artificial, real-world, and Olivetti face datasets. Computational time is compared and discussed between the various algorithms. Finally, Section VII summarizes and concludes the paper.

II. RELATED WORKS

Scholars have shown great interest in DPC and many improved algorithms have been proposed to address its defects.

The first aspect is improving the local density. Du *et al.* [23] proposed DPC-KNN and DPC-KNN-PCA where the improved local density is calculated by mean KNN distance, where k is computed as the percentage, p, of the total number of points. Liu *et al.* [24] used KNN to compute local density and proposed adaptive density peak clustering ADPC-KNN. Xie *et al.* [22] proposed FKNNDPC, which improves local density based on KNN. In contrast to DPC-KNN, the improved local density is calculated by the distance sum to KNN. Geng *et al.* [25] proposed RECOME using a density measure based on relative KNN kernel density with a parameter. Du *et al.* [26] proposed FNDP using fuzzy neighborhood relationships to define local density. Liu *et al.* [27] proposed shared-nearest-neighbor-based clustering by fast search and find of density peaks (SNNDPC) algorithm, which gives some new definitions, such as SNN similarity, local density and distance from the nearest larger density point. Seyedi *et al.* [28] proposed DPC-DLP which employs the idea of KNN to compute the global cutoff parameter and the local density of each point. Tao *et al.* [29] introduced the data field theory to adaptively select the d_c to compute local density and proposed F-DPC. Zhang *et al.* [30] proposed the developed density peak clustering algorithm with support vector data description (SVDD), and the cutoff distance was optimized by adjusted silhouette coefficient. Similar to requiring prespecified d_c, DPC-KNN, DPC-KNN-PCA, FKNNDPC, SNNDPC and DPC-DLP also require prespecifying the number of nearest neighbors. Therefore, these improved algorithms just transfer the prespecified parameter rather than overcoming the problem. Mehmood *et al.* [31] presented a non-parametric algorithm for DPC via heat diffusion (HDDPC) using kernel density estimation (KDE). Zhou *et al.* [32] also used KDE to optimize local density and proposed density peaks clustering by identifying the veins (IVDPC). However, KDE inherent defects seriously affect HDDPC clustering performance. KDE is acceptably accurate in one-dimensional (1D) or 2D data, but becomes highly inaccurate for higher dimensional or sparse data. Hence HDDPC tends to achieve poor clustering performance for most real-world high dimensional datasets. KDE is also somewhat time consuming, causing extremely low HDDPC clustering efficiency, and the authors do not provide any suggestions to accelerate HDDPC clustering process. Another non-parameterized algorithm is to use fixed parameter values. Strictly speaking, this cannot be called non-parameter algorithm, but for users, there is no need to predefine parameters in practical applications, which is approximately equivalent to a non-parameter algorithm. Adaptive clustering algorithm based on KNN (fixed $k = 5$) and density (ACND) has been proposed [33]. The fixed number addresses nearest neighbor interference susceptibility from accidental meeting of two or three outliers, i.e., 2NN, 3NN, or 4NN may be close to normal data points, and it is not easy to distinguish them.

The second aspect is to improve the assignment strategy. Liu *et al.* [27] introduced two concepts, inevitably subordinate and possibly subordinate, to assign the label of those non-center points. Lu *et al.* [34] proposed a simple density-based framework (DCF) and two steps assignment rule for core and non-core points. The first step is based on a minimum spanning tree and the second on an ordering mechanism. Chen *et al.* [35] proposed CLUB (CLUstering based on Backbone) with assignment strategy carried out in three steps. Seyedi *et al.* [28] used a graph-based label propagation to assign labels to remaining points in DPC-DLP. Xie *et al.* [22] proposed fuzzy weighted KNN assignment strategy (FKNNDPC, strategy 2) claimed to reduce cluster label error propagation. However, it is not effective on all datasets (e.g. Fig. 5(b) in this paper, using strategy 2, the red point in cluster 3 is assigned to cluster 2 by mistake). The key aspect of Strategy 2 is to learn the probability, p_i^c, that point $\boldsymbol{x}_i$ belongs to cluster c, and then assign $\boldsymbol{x}_i$ to c with the largest p_i^c. This step is time consuming and reduces clustering efficiency. Another potential risk for FKNNDPC strategy 1 is that some clusters only have the center point itself (i.e., isolated cluster) on dense data (e.g., Fig. 2(b) on the S4 dataset in this paper).

The other aspects, such as selection of the cluster centers, application of DPC and new similarity or distance measure and so on, are also improved. Ding *et al.* [36] proposed an entropy-based density peaks clustering algorithm for mixed type data employing fuzzy neighborhood (DP-MD-FN). The new similarity measure of DP-MD-FN avoids feature transformation and parameter adjustment between numerical and categorical attributes. Xu *et al.* [37] proposed two prescreening strategies, grid-division and circle-division, to fast find cluster centers for large-scale dataset. Another density peaks clustering algorithm based on grid (DPCG) also proposed by Xu *et al.* [38], which improves the efficiency using CLIQUE clustering algorithm to calculate the local density. Du *et al.* [39] focused on the distance measurement and proposed density peaks clustering using geodesic distance (DPC-GD). Bai *et al.* [40] used concept approximation to propose an acceleration algorithm (CFSFDP+A) involving fewer distance calculations. Jiang *et al.* [41] developed a DPC enhanced algorithm, called GDPC, with an alternative decision graph-based on gravitation theory and nearby distance to identify centers and anomalies accurately. Moreover, they tried to overcome some weakness, such as varying densities and irregular shapes, and proposed DPC-LG algorithm [42] to improve GDPC based on logistic distribution and gravitation. Sun *et al.* [43] proposed a graph-based density peak clustering algorithm which is applied into patient hospital admission graph cluster analysis.

III. DENSITY PEAKS CLUSTERING ALGORITHM

Assume that point $\boldsymbol{x}_i \in \mathrm{R}^m, i = 1, \ldots, n$ belongs to dataset X with m attributes. Let d_{ij} represent the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$,

$$d_{ij} = \left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\|_2 . \tag{1}$$

DPC computes local density ρ_i and distance δ_i from points with higher local density. Local density can be defined by counting the number of points in its neighborhood,

$$\rho_i = \sum_{j=1}^{n} \chi\left(d_{ij} - d_c\right). \tag{2}$$

where $\chi(d) = 1$ if $d < 0$ and $\chi(d) = 0$ otherwise, and d_c is the prespecified cutoff distance; or as the Gaussian kernel local density,

$$\rho_i = \sum_{j=1}^{n} \exp\left(-\frac{d_{ij}^2}{d_c^2}\right). \tag{3}$$

Thus, δ_i is measured by computing the minimum distance between $\boldsymbol{x}_i$ and any other points with higher local density,

$$\delta_i = \begin{cases} \min\limits_j \left(d_{ij}\right), & \exists j \text{ s.t. } \rho_j > \rho_i \\ \max\limits_j \left(d_{ij}\right), & \text{otherwise} \end{cases} \tag{4}$$

These points are referred to as decision graph, plotting δ_i as a function of ρ_i in 2D space. Center points are significantly separated from non-center points and can be selected from the decision graph by choosing only points with high ρ and relatively large δ. The number of clusters is also determined to correspond the number of center points. Subsequently, each remaining point is assigned to the same cluster as its nearest neighbor point with higher local density.

IV. DPCSA CLUSTERING ALGORITHM

This section discusses local density improvement to avoid requiring a prespecified parameter, splitting the assignment strategy into two stages using a boundary condition to reduce assignment error propagation, and *NNDT* to improve clustering efficiency. Finally, the algorithm flow and complexity analysis are also provided.

A. LOCAL DENSITY BASED ON FIXED K-NEAREST NEIGHBORS AND WEIGHTED SEQUENCE

We employ KNN to improve local density and improve clustering performance. For example, FKNNDPC [22] and DPC-KNN [23] local densities can be expressed, respectively, as

$$\rho_i = \sum_{j \in kNN_i} \exp\left(-d_{ij}\right). \tag{5}$$

and

$$\rho_i = \exp\left(-\frac{1}{K}\sum_{j\in kNN_i} d_{ij}^2\right). \tag{6}$$

Although there are differences between (3), (5), and (6), all three local densities consider the sum of distances based on the exponential kernel with a prespecified parameter, d_c or k. The difference lies in that d_c in (3) controls the exponential kernel bandwidth with the sum of all points, whereas k in (5) and (6) controls the sum of KNN points with bandwidth = 1. Both cases (d_c or k) require prior knowledge and user experience. For 2D or three-dimensional (3D) datasets, users can obtain some prior knowledge from intuitive scatter plots, particularly point distributions. However, this cannot be achieved for higher dimension datasets.

Equations. (5) and (6) calculate local density based on KNN with bandwidth = 1, and achieve moderate clustering performance, i.e., the KNN points have an important impact on local density. Local density from (3) is computed for all points, including KNN points, with bandwidth = d_c and achieves reasonable clustering performance, i.e., points beyond KNN also impact local density. KNN is fixed ($k = 5$) for ACND, since 2NN, 3NN, or 4NN may be close to normal data points as discussed above. Therefore, we combined these advantages to propose an improved local density based on the weighted sequence with constant bandwidth and fixed KNN,

$$\rho_i = \sum_{j=1}^{K}\exp\left(-d'_{ij}\right) + \sum_{j=K+1}^{n-1}\frac{\exp\left(-d'_{ij}\right)}{(j-K)^2} \tag{7}$$

where, d'_{ij} ($j = 1, \ldots, n-1$) is the increasing order of d_{ij} ($j = 1, \ldots, n, j \neq i$), and $K = 5$. The first term of (7) maintains the KNN calculation from five nearest neighbor points, inheriting concepts from (5) and (6). The second term of (7) sums the weighted exponential kernel sequence, which inherits and improves the concept of (3). This second term is a supplement to the first term, and compensates for clustering performance reduction due to the fixed k. Thus, (7) incorporates advantages from (3), (5) and (6), and depends only on d_{ij} and n; i.e., DPCSA does not require prespecified parameters and inherits FKNNDPC, DPC-KNN, DPC, and ACND advantages.

B. TWO-STAGE ASSIGNMENT STRATEGIES

1) BOUNDARY CONDITION

For DPC, non-center points are assigned in descending order of local density to the same cluster as their nearest neighbor point with higher local density. Aside from the center points, two cases produce large δ_i.

1) Where $\boldsymbol{x}_i$ is a peak surrounded by other points with slightly higher local density, ρ_i, but insufficient to be considered a center point. Under the DPC assignment strategy, $\boldsymbol{x}_i$ will affect subsequent points with local densities $< \rho_i$.
2) Where $\boldsymbol{x}_i$ is an outlier with low local density and far from other points.

When these points are assigned incorrectly, the error may propagate, i.e., more points are assigned incorrectly. We do not want to assign these two cases' points to minimize assignment error. Therefore, we propose the boundary condition

$$\delta_i < \frac{1}{n}\sum_{l=1}^{n}\delta_l. \tag{8}$$

All points satisfying (8) are assigned a label in the first stage and other points in the second stage.

2) FIRST STAGE ASSIGNMENT STRATEGY

The first stage assignment strategy assigns a cluster label to $\boldsymbol{x}_i$ with distance δ_i less than the average, as in (8). The assignment method is the same as DPC, i.e., the point is assigned in descending order of local density to the same cluster as its nearest neighbor. If no nearest neighbor point with higher density is assigned, the point is also not assigned. All remaining unassigned points are assigned depending on the second stage assignment strategy.

The proposed first stage strategy assigns points around the center point that have relatively higher local density. Aside for the center points, large δ_i implies there are no points around it or that it is far from the point densely distribution region, e.g. an outlier.

3) SECOND STAGE ASSIGNMENT STRATEGY

Assuming the first stage assignment strategy has been performed, with n_1 points assigned to $S_1, \ldots, S_K$ clusters, where the ith cluster, S_i, contains m_i points, $m_1 + \cdots + m_K = n_1$, All remaining $n - n_1$ points comprise the set S_0. The second stage assignment strategy assigns $\boldsymbol{x}_i \in S_0$ to cluster S_k according to the minimum distance $\min_{i,j} d_{ij}, \boldsymbol{x}_i \in S_0, \boldsymbol{x}_j \in S_k, k = 1, \ldots, K$.

Let

$$VA_k = \min_{i,j}\left\{d_{ij}|\boldsymbol{x}_i \in S_0, \boldsymbol{x}_j \in S_k\right\}, \quad k = 1, \ldots, K. \tag{9}$$

and

$$VP_k = \arg\min_{i}\left\{d_{ij}|\boldsymbol{x}_i \in S_0, \boldsymbol{x}_j \in S_k\right\}, \quad k = 1, \ldots, K. \tag{10}$$

where VA_k is the minimum distance between S_0 and S_k; and VP_k is the point number corresponding to the minimum distance between S_0 and S_k. Let VA_l be the minimum value of VA, i.e.,

$$VA_l = \min\{VA_k|k = 1, \ldots K.\} \tag{11}$$

Thus, the VP_l point corresponding to the minimum distance VA_l is assigned to cluster S_l, and then we remove it from S_0 and recalculate (10) to select the next unassigned point, assign its cluster label, etc. until S_0 becomes empty.

However, it is very time consuming to repeatedly calculate (9) and (10) because each point $\boldsymbol{x}_i \in S_0$ is assigned a cluster label and we must update S_0 and S_l for the $\boldsymbol{x}_i$ is assigned to, and repeatedly calculate *VA* and *VP*. Clustering efficiency is very important and, would help to determine popularity, just as K-means is more popular than K-medoids [44], although the latter is more robust and accurate. Therefore, we propose the *NNDT* to improve clustering efficiency and hence allow wider DPCSA use.

C. NEAREST NEIGHBOR DYNAMIC TABLE

After performing the first stage assignment strategy, the cluster table, *CT*, can be constructed from the assigned points,

$$CT_{kl} = i, \quad \boldsymbol{x}_i \in S_k. \tag{12}$$

where CT_{kl} is the (k, l) element in the *CT* table and k represents the kth cluster. Hence the *NNDT* can be defined as

$$NNDT_{kl} = \arg\min_{j} \left\{ d_{ij} | i = CT_{kl}, \boldsymbol{x}_j \in S_0 \right\} \tag{13}$$

where $NNDT_{kl}$ is the (k, l) element in the *NNDT* table. Tables *CT* and *NNDT* store the point number rather than the point itself. For example, for $\boldsymbol{x}_i$, where i is its number in the dataset, the tables store i rather than $\boldsymbol{x}_i$. Since each cluster contains at least one center point, S_k, $k = 1, \ldots, K$ is not empty. Thus, *CT* and *NNDT* tables are not empty.

However, it is time consuming to use (13) to initialize and maintain *NNDT*. Assume DPCSA assigns $\boldsymbol{x}_p \in S_0$ to cluster S_k when performing the second stage assignment strategy, then *CT* and *NNDT* require updating. Maintaining *CT* is convenient and only adds point number, p, to *CT* at the end of the kth row. However, using (13) to directly update *NNDT* is inefficient because S_0 always changes. Running time can be significantly reduced by constructing an ascending order table, *ST*, whose element is point number and this point belongs to S_0,

$$ST_{il} = j \quad \text{s.t.} \begin{cases} d_{it} \le d_{ij} \le d_{iq}, \\ t = ST_{i,l-1}, \\ q = ST_{i,l+1}, \\ \boldsymbol{x}_t, \boldsymbol{x}_j, \boldsymbol{x}_q \in S_0. \end{cases} \tag{14}$$

where ST_{il} is the (i, l) element in the *ST* table, and i represents the ith point in dataset. $ST_{il} = j$ means that d_{ij} is the lth smallest element in the distance sequences between $\boldsymbol{x}_i$ and points in S_0. Once the table *ST* is constructed, there is no further updating and maintaining. Let cluster label variable $L_i = 0$ represent that $\boldsymbol{x}_i$ has not been assigned a cluster label, and other cases represent its cluster label. Thus, *NNDT* can be initialized efficiently using *ST*,

$$NNDT_{kl} = \begin{cases} ST_{i1}, & i = CT_{kl} \text{ and } L_i \neq 0 \\ ST_{i2}, & i = CT_{kl} \text{ and } L_i = 0 \end{cases} \tag{15}$$

where index k represents the kth cluster and index i represents the point number of $\boldsymbol{x}_i$.

It is also efficient to maintain *NNDT* using *ST*. Assuming current DPCSA state is $CT_{kl} = i$ and $NNDT_{kl} = ST_{iq}$, the next step would assign $\boldsymbol{x}_p \in S_0$ into S_k, i.e., $\boldsymbol{x}_p$ would be clustered into S_k. $\boldsymbol{x}_p$ should be added to both *CT* and *NNDT* with values p and ST_{pl}, respectively (l is the first element of the pth row in *ST* that satisfies $L_l = 0$), and then remove $\boldsymbol{x}_p$ from S_0 and set L_p as the true cluster label k and update *NNDT*,

$$NNDT_{kl} = \begin{cases} ST_{iq}, & NNDT_{kl} \neq p, \\ ST_{it}, & \text{otherwise}. \end{cases} \tag{16}$$

where the index, t, gradually increases from q until the first unassigned cluster label point ST_{it}, i.e., if $s = ST_{it}$, then $\boldsymbol{x}_s$ is the first point after $\boldsymbol{x}_q$ of the ith row in *ST* that satisfies $L_s = 0$. For the case $NNDT_{kl} \neq p$ in (16), $NNDT_{kl}$ keeps its original value. Meanwhile, without an extra operations, *VA* and *VP* can be updated in sync with *NNDT*. The next one to be assigned can then be chosen.

We provide an example to explain the above process. Figures 1 (a) and (b) show an example dataset and distance matrix, respectively. Figure 1 (c) shows that $\boldsymbol{x}_2, \boldsymbol{x}_3$ have been assigned to S_1 and $\boldsymbol{x}_4, \boldsymbol{x}_5$ to S_2. In the next step, point 1, i.e., $\boldsymbol{x}_1$, is selected and clustered into S_1, then tables *CT* and *NNDT* are updated (arrows).

D. ALGORITHM FLOW

Algorithm 1 shows DPCSA primary flow. Step 2 calculates the improved local density, steps 5-9 are the first stage assignment strategy, and steps 12-17 are the second stage assignment strategy, which includes NNDT updating (step 16). The flow diagram of DPCSA is given in Fig. 2.

V. COMPLEXITY ANALYSIS

All four algorithms: DPC, HDDPC, FKNNDPC, and DPCSA can be divided into four main parts: (a) calculate distance matrix, (b) calculate ρ_i, (c) calculate δ_i, (d) assign cluster label. These algorithms use the same method to calculate the distance matrix and δ_i with time complexity $= O(n^2)$. Thus, time complexity differences depend on the remaining two parts.

HDDPC estimates the kernel density function before computing local density, which is time consuming; whereas FKNNDPC searches KNN points of $\boldsymbol{x}_i$, with time complexity to calculate local density for all points $= O(Kn^2)$. FKNNDPC assignment process involves three strategies, and the strategy 2 fuzzy KNN assignment method is rather complicated. Time complexity of the whole assignment process $= O(n^3)$ [22], mainly due to updating the recognition matrix and calculating

$$p_i^c = \sum_{j \in kNN_i, y_j == c} \gamma_{ij} * w_{ij} \tag{17}$$

where

$$\gamma_{ij} = w_{ij} / \sum_{l \in kNN_j} w_{ij}$$

DPCSA improves local density and employs a two-stage assignment strategy based on the boundary condition. *NNDT* also speeds up clustering. Since incremental sorting time complexity $O(n \log n)$, time complexity for the

number	point	number	point
1	(0.8,1.1)	5	(3.0,3.0)
2	(1.0,1.0)	6	(3.3,2.8)
3	(1.1,0.9)	7	(1.5,4.0)
4	(3.1,3.1)		

(a)

0.0000	0.2236	0.3606	3.0480	2.9069	3.0232	2.9833
0.2236	0.0000	0.1414	2.9698	2.8284	2.9206	3.0414
0.3606	0.1414	0.0000	2.9732	2.8320	2.9069	3.1257
3.0480	2.9698	2.9732	0.0000	0.1414	0.3606	1.8358
2.9069	2.8284	2.8320	0.1414	0.0000	0.3606	1.8028
3.0232	2.9206	2.9069	0.3606	0.3606	0.0000	2.1633
2.9833	3.0414	3.1257	1.8358	1.8028	2.1633	0.0000

(b)

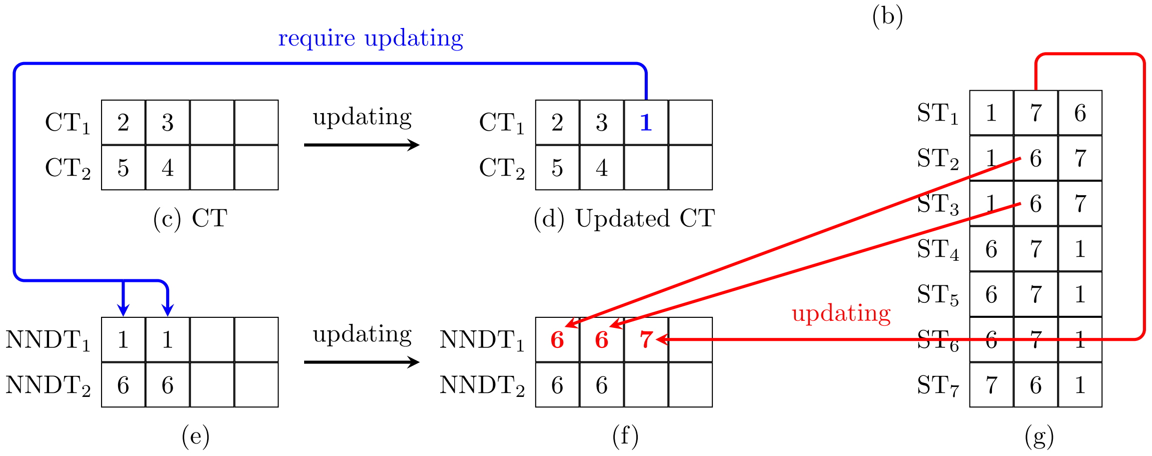


FIGURE 1. Proposed CT and NNDT tables example dataset. Point 1 (blue, (d)), i.e.,x_1, was clustered into S_1. Thus, *NNDT* must update its value with the nearest neighbor point corresponding to point 1 (blue arrow). Use *ST* to update *NNDT* (red arrow indicates update direction);(d) and (f) are the updated tables. (a) Dataset. (b) Distance matrix. (c) CT. (d) Updated CT. (e) NNDT. (f) Updated NNDT. (g) ST.

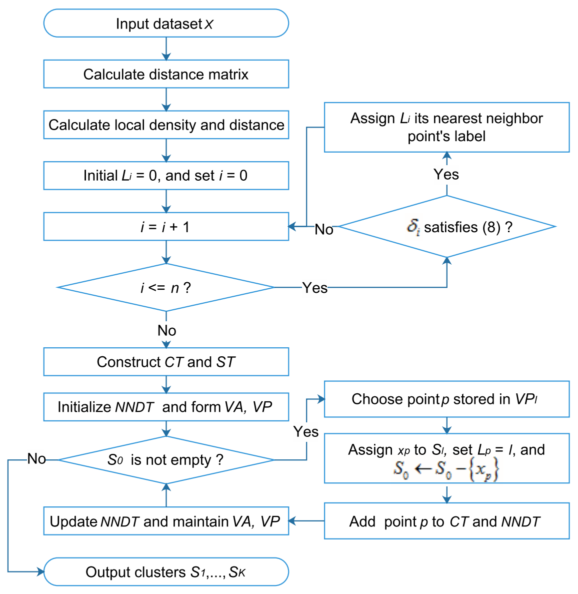

Input dataset X
Calculate distance matrix
Calculate local density and distance
Initial L_i = 0, and set i = 0
i = i + 1
i <= n ?
Yes
No
δ_i satisfies (8) ?
Yes
No
Assign L_i its nearest neighbor point's label
Construct CT and ST
Initialize NNDT and form VA, VP
S_0 is not empty ?
Yes
No
Choose point p stored in VP_l
Assign x_p to S_l, set L_p = l, and $S_0 \leftarrow S_0 - \{x_p\}$
Add point p to CT and NNDT
Update NNDT and maintain VA, VP
Output clusters $S_1,\ldots,S_K$

FIGURE 2. DPCSA flow diagram.

improved local density based on weighted sequence with constant bandwidth and fixed KNN $= O(n^2 \log n)$, somewhat larger than the other algorithm corresponding terms that have time complexity $= O(n^2)$. Time complexity for the first stage assignment strategy $= O(n)$, similar to DPC. Although each remaining point is assigned a cluster label for the second stage assignment strategy, which requires finding the nearest neighbor between assigned and unassigned cluster label points, *NNDT* improves running efficiency with overall time complexity $= O(n^2 \log n)$. Experimental results (see Section VI-F) confirm that DPCSA is more efficient than FKNNDPC and HDDPC, but less than DPC.

VI. EXPERIMENTAL SETUP AND ANALYSIS

This section discusses DPSCA testing and verification for clustering performance and efficiency compared with the well-known DPC [16], FKNNDPC [22], and HDDPC [31] algorithms, using artificial and real-word datasets. The Olivetti Face dataset is also employed to test algorithm clustering performance. Runtime comparisons are included and we discuss order sensitivity and the fixed number of DPCSA. We ensure fair comparison coding all algorithms in MATLAB on a PC with Intel® Core™ 3.5 GHz i3 CPU with 12 GB RAM. We implement FKNNDPC algorithm referring to [22] and ensure the consistent programming style of FKNNDPC and DPCSA. The source code of HDDPC is provided by the original authors.

A. DATASETS AND PREPROCESSING

Ten real-world datasets were acquired from the UCI repository [45] and the Olivetti face real-world dataset from previous literature [46]. Twelve classic artificial datasets

Algorithm 1 DPCSA

```
Input: Dataset $X$
Output: Clusters $S_1, \ldots, S_K$
1: Normalize the data and calculate the distance matrix between each pair of points.
2: Calculate local density $\rho_i$ and distance $\delta_i$ for each $\boldsymbol{x}_i$ using (7) and (4), respectively.
3: Generate the decision graph with significant separation between center and non-center points, then find the cluster center points and determine the number of clusters.
4: Initial class label $L_i = 0$ and assign the center points class label.
5: for $i = 1$ to $n$ do
6:     if $\delta_i$ satisfies (8) then
7:         Assign $L_i$ its nearest neighbor point's label.
8:     end if
9: end for
10: Construct $CT$ and $ST$ using (12) and (14), respectively.
11: Initialize $NNDT$ using (15) and form $VA$, $VP$ simultaneously.
12: while $S_0$ is not empty do
13:     Choose point $p$ stored in $VP_l$ with minimum distance in $VA$ using (11).
14:     Assign $\boldsymbol{x}_p$ to $S_l$, i.e., set $L_p = l$, and $S_0 \leftarrow S_0 - \{\boldsymbol{x}_p\}$.
15:     Add point $p$ to $CT$ and $NNDT$.
16:     Update $NNDT$ using (16) and maintain $VA$, $VP$, simultaneously.
17: end while
```

(Aggregation, D31, Dim-sets, Flame, Path-based, S sets, and SD) were acquired from various literatures [16], [47]–[51].

Tables 1 and 2 summarize the acquired datasets. Some artificial dataset point distributions had intentional shapes to challenge detecting the correct number of clusters, i.e., Flame, Path-based, and Spiral. All real-world and artificial datasets were commonly regarded as benchmarks to test various clustering algorithm performance and efficiency. Data size, number of attributes, and number of clusters varied for each dataset. Generally, artificial dataset higher class numbers compensated for inherent weaknesses in the real-world dataset lower class numbers. In contrast, real-world dataset higher attribute numbers compensate for weakness inherent in artificial dataset lower attribute numbers.

To avoid numerical range influence from the attribute values, each attribute was normalized before calculating distance,

$$z_{ij} = \frac{x_{ij} - \min\limits_t (x_{tj})}{\max\limits_t (x_{tj}) - \min\limits_t (x_{tj})}, \quad j = 1, \ldots, m. \; i = 1, \ldots, n. \tag{18}$$

where z_{ij} represents the normalized value of attribute j for $\boldsymbol{x}_i$, and t, l is point number.

TABLE 1. Artificial datasets.

Dataset	Size	Attributes	Classes
Aggregation	788	2	7
D31	3100	2	31
Dim512	1024	512	16
Dim1024	1024	1024	16
Flame	240	2	2
Path based	300	2	3
Spiral	312	2	3
S1	5000	2	15
S2	5000	2	15
S3	5000	2	15
S4	5000	2	15
SD	2000	2	5

TABLE 2. Real-world datasets.

Dataset	Size	Attributes	Classes
Dermatology	366	34	6
Iris	150	4	3
Ionosphere	351	34	2
Libras	360	91	15
Parkinsons	195	23	2
Pima	768	8	2
Seeds	210	7	3
Segmentation	2310	19	7
Wdbc	569	30	2
Wine	178	13	3
Olivetti face	400	$92 * 112$	40

We applied min-max normalization for DPCSA, FKNNDPC, and DPC [22], whereas HDDPC used the original attribute value [31].

B. CLUSTERING EVALUATION

An appropriate and uniform evaluation index is both required and meaningful to compare the different clustering algorithms. Therefore, we select three popular clustering evaluation indexes [52], [53]: clustering accuracy (ACC) in (19), adjusted mutual information (AMI) in (20) and adjusted rand index (ARI) in (21). Larger evaluation index values indicate improved clustering performance, and all index upper bounds = 1, representing perfectly correct clustering. Actual run times are also recorded to support time complexity analysis from Section V.

$$ACC = \frac{\sum_{i=1}^{n} \delta\left(s_i, map(r_i)\right)}{n} \tag{19}$$

$$AMI = \frac{I(\boldsymbol{U}, \boldsymbol{V}) - E\{I(\boldsymbol{M})|a, b\}}{\sqrt{H(\boldsymbol{U})H(\boldsymbol{V})} - E\{I(\boldsymbol{M})|a, b\}} \tag{20}$$

$$ARI = \frac{RI - E[RI]}{\max(RI) - E[RI]} \tag{21}$$

DPC and FKNNDPC require prespecified parameter that can significantly influence clustering performance. DPC requires the cutoff distance, d_c, to calculate the local density. Users can choose d_c relatively freely so the average number of neighbors is approximately $1 \sim 2\%$ of the total number of points in the dataset [16], and d_c, can also be adjusted using repeated experiments. The prespecified KNN parameter, k, for FKNNDPC is required to calculate local density, and assign point cluster labels in strategies 1 and 2. We use the generic symbol *Par* as the prespecified parameter when setting d_c for DPC and k for FKNNDPC.

All four clustering algorithms assign the remaining (non-center) points cluster label starting from the center point. If the algorithm finds incorrect center points or more than two center points within a single same cluster, this error may propagate, i.e., more points will be assigned incorrectly. Therefore, it is important that the algorithm finds the correct number of clusters and center points in different clusters. If two center points belong to the same cluster, the cluster would split and the true cluster(s) would disappear with no center point(s) found. In the evaluation, F refers to the number of cluster center points found and P to the number of clusters the cluster center points occupy.

In our experiments, *Par* (i.e., d_c or k) arise from the recommended optimal value used in FKNNDPC [22].

C. EXPERIMENTAL RESULTS ON ARTIFICIAL DATASETS

This section first shows DPCSA, FKNNDPC, HDDPC, and DPC clustering performance on eight artificial datasets with intuitive clustering scatter diagrams except for dim512, dim1024, S1, and S3 datasets. Small circles in the scatter diagram represent non-center points and large squares represent center points. All points in a given cluster have the same color. Since the SD dataset has no real class labels, ACC, AMI, and ARI are only calculated for the remaining 11 datasets.

The S4 dataset consists of 15 clusters with noise and heavy overlapping, which poses significant challenges to any clustering algorithm to detect the true number of clusters and correctly identify the center points. Figures 3 (a)-(d) show that all four clustering algorithms can recognize the true number of clusters, which is also confirmed by $F = 15$ (the algorithm found all 15 center points) and $P = 15$ (the algorithm found each center point in a distinct cluster) as shown in Table 3. However, finding the correct center points does not guarantee clustering accuracy. The different assignment strategies caused clustering results to differ significantly. Figure 3 (b) shows only the center point itself in cluster $C11$. We identified this failure for the FKNNDPC process was due to strategy 1. The center point of cluster $C5$ was selected before the center point of $C11$, and assigned its cluster label. Point proximity and heavy overlapping between $C5$ and $C11$ meant that points that should belong to $C11$ were assigned to $C5$, because they satisfied assignment strategy 1 conditions when the $C5$ center point was selected. This potential risk limits widespread FKNNDPC use. However, this error did not occur in the other algorithms (see Figs. 3 (a), (c), and (d)). All ACC, AMI, and ARI index values for DPCSA, HDDPC, and DPC were superior to FKNNDPC (Table 3). The clustering scatter diagram and evaluation indexes confirm that FKNNDPC clustering performance was significantly poorer than the other clustering algorithms. The DPCSA ACC, AMI, and ARI indexes are slightly lower than for DPC, but higher than for HDDPC.

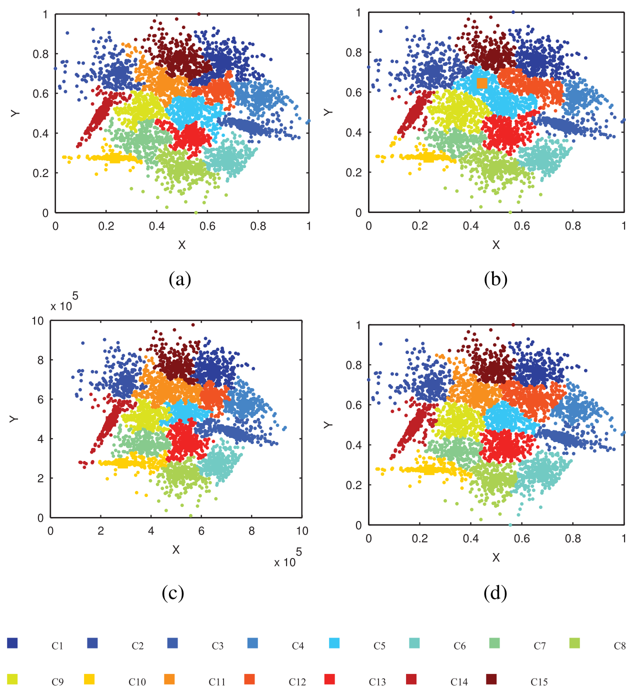


FIGURE 3. S4 clustering outcomes. (a) DPCSA. (b) FKNNDPC. (c) HDDPC. (d) DPC.

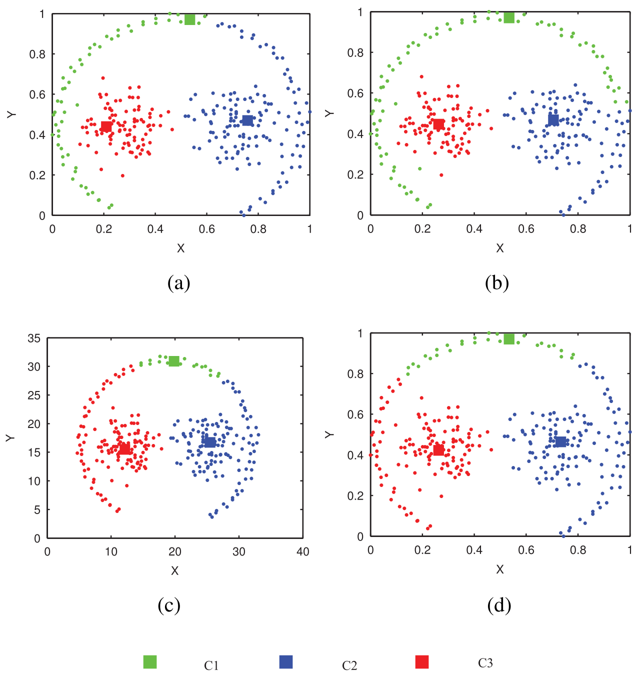


FIGURE 4. Path-based clustering. (a) DPCSA. (b) FKNNDPC. (c) HDDPC. (d) DPC.

The Path-based dataset points form a distributed face shape. The special aspect is that the points of one cluster surround the points of the remaining two clusters, hence many clustering algorithms fail, including DPC and HDDPC. Figures 4 (a)-(d) show that DPCSA, FKNNDPC, HDDPC, and DPC all detect three clusters and identify corresponding center points correctly, but split cluster $C1$ (face outline). HDDPC and DPC split $C1$ into three clusters, and DPCSA and FKNNDPC split it into two. Thus, DPCSA and FKNNDPC outperform HDDPC and DPC. Since $C1$ (Fig. 4 (b), green points) contains more points than in Fig. 4 (a), FKNNDPC clustering outperforms DPCSA.

We analyzed DPCSA and FKNNDPC clustering process to identify the underlying reason. Figure 4 (b) shows that $C2$ points (blue points) with proximity to $C1$ (green points) highest membership depended on their own K ($K = 8$) nearest neighbors in assignment strategy 2 belonging to cluster $C2$. Therefore, they were divided into cluster $C2$ erroneously. Figure 4 (a) (DPCSA) shows that since the distance of the nearest pair of points between $C1$ (green points) and $C2$ (blue points) is greater than each point with its nearest neighbor in $C2$. Thus, the nearest neighbor assignment strategy splits these face outline points into two clusters ($C1$ and $C2$). This excess distance in the Path-based dataset point distribution causes clustering failure. If this distance was reduced, clustering performance would be significantly improved.

We maintained the Path-based dataset distribution as much as possible by moving only two points (points 105 and 108, Fig. 5, large red circles) to new positions (Fig. 5, green asterisks) rather than deleting or adding new points. Blue arrows in Fig. 5 indicate the position and direction of movement. We referred to the modified dataset as New Path-based. Moving these data points mitigates the effects of the excess distance.

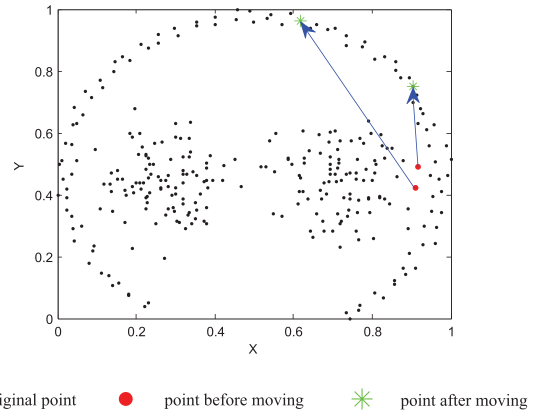


FIGURE 5. Moving two points in the Path-based dataset.

Figures 6 (a)-(d) for the New Path-based dataset show that all four algorithms detect three clusters and identify the corresponding center points correctly, which is consistent with the $F = 3$ and $P = 3$ (Table 3). Figures 4 (c) and 6 (c) are very similar, as are Figs. 4 (d) and 6 (d), confirming that moving the two points had almost no effect on HDDPC or DPC. Thus, HDDPC and DPC do not capture local information changes. Figures 4 (a) and 6 (a) (DPCSA), and 4 (b) and 6 (b) (FKNNDPC) exhibited the most significant differences with moving the two points. Figures 6 (a) (DPCSA) and 6 (b) (FKNNDPC) completely separated the three clusters,

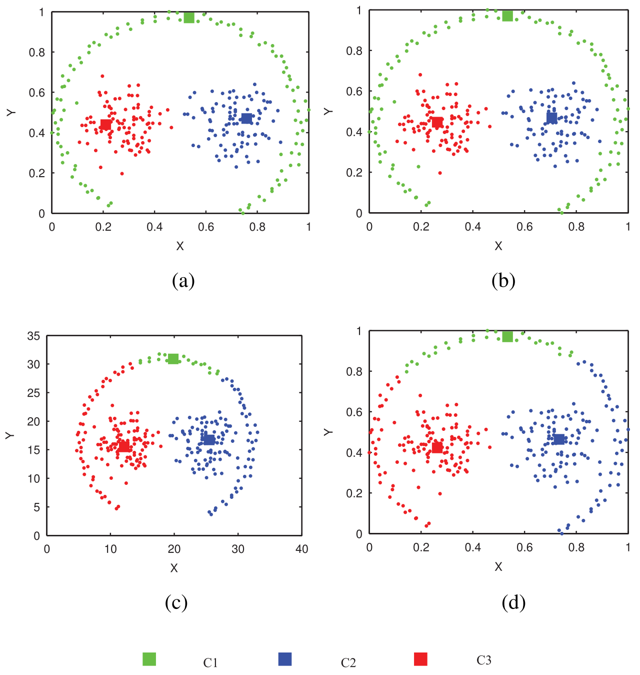


FIGURE 6. Clustering for the New Path-based dataset. (a) DPCSA. (b) FKNNDPC. (c) HDDPC. (d) DPC.

achieving $ACC = 0.9933$ and 0.9900, respectively (Table 3). The only difference between these two algorithms was point 204, which FKNNDPC wrongly assigned to $C1$.

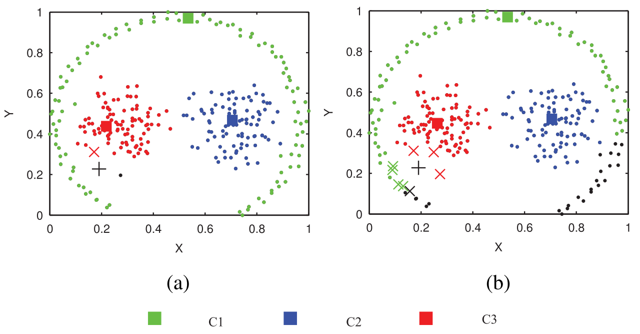


FIGURE 7. Clustering phase diagram before assigning point 204 (+ symbol). Black represents unassigned points, green is $C1$, blue is $C2$, and red is $C3$. The × symbol is (a) single nearest neighbor and (b) K($K = 8$) nearest neighbors for point 204.

Figure 7 shows clustering process before assigning point 204. FKNNDPC identified point 204 as an outlier that would be assigned a cluster label in strategy 2. Its membership $p_{204}^{1} = 0.43664$, $p_{204}^{2} = 0$, $p_{204}^{3} = 0.34256$ was calculated based on K ($K = 8$) nearest neighbors (Fig. 7 (b), × symbol). None of the cross points belong to $C2$, so membership $p_{204}^{2} = 0$. The "+" point has no contribution to calculating point 204 membership, because it is an unassigned cluster label. Thus, from the principle of maximum membership, point 204 would be assigned to $C1$, i.e., an error assignment. The DPCSA nearest neighbor to point 204 was the red cross point, which belongs to $C3$ (Fig. 7 (a)). Thus, the nearest neighbor assignment strategy would assign point 204 to $C3$, which is the correct result. Therefore, DPCSA outperforms FKNNDPC, particularly for this dataset with a shaped distribution.

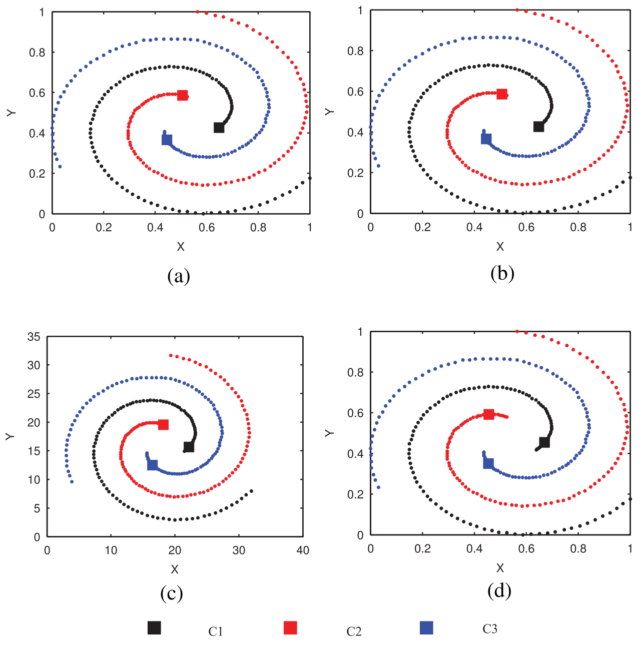


FIGURE 8. Clustering for the Spiral dataset. (a) DPCSA. (b) FKNNDPC. (c) HDDPC. (d) DPC.

Figure 8 shows that all four algorithms detect the three true clusters and correctly identify the center points for the Spiral dataset. The center points were completely identical, and there was no point cluster label assignment error. Thus, all algorithms obtained $ACC = 1$ and $AMI = 1$. Figure 8 shows completely correct clustering on the Flame dataset, except for FKNNDPC, which only assigns two erroneous point cluster labels (Fig. 9 (b)).

Figure 10 shows that all algorithms detect the Aggregation dataset seven true clusters and correctly identify seven center points. HDDPC was closest to the completely correct result with $ACC = 0.9987$, $AMI = 0.9955$, and $ARI = 0.9978$, followed by FKNNDPC and DPC with the same clustering accuracy $ACC = 0.9975$. Although DPCSA achieved the poorest result for this case, clustering accuracy, $ACC = 0.9505$, was still very high. Some of the points belonging to $C4$ were assigned to $C5$. The reason for the relatively low clustering accuracy is that unassigned points from strategy 1 are densely distributed and strategy 2 assigns to their nearest neighbor point's cluster label.

The SD dataset points were drawn from a probability distribution with non-spherical and strongly overlapping peaks. Figure 11 shows that all four algorithms detect the true five clusters and identify five center points correctly, even those points corresponding to different densities and non-spherical peaks. $C5$ points clustered by HDDPC and DPC

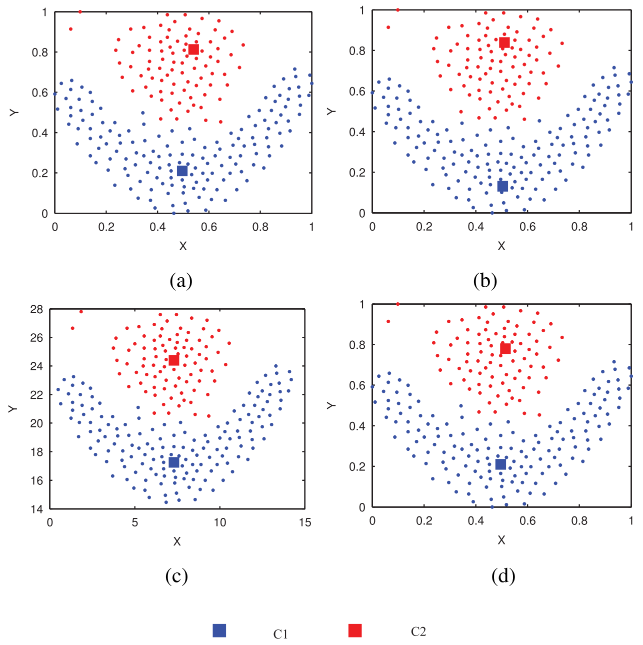


FIGURE 9. Clustering for the Flame dataset. (a) DPCSA. (b) FKNNDPC. (c) HDDPC. (d) DPC.

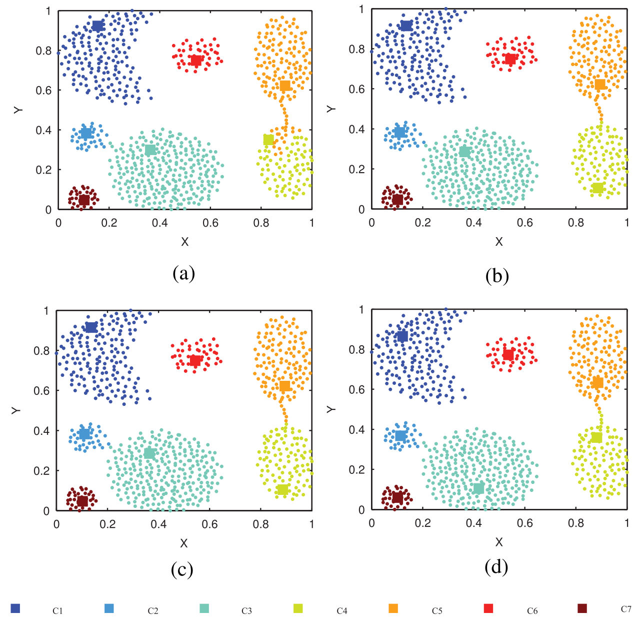


FIGURE 10. Clustering for the Aggregation dataset. (a) DPCSA. (b) FKNNDPC. (c) HDDPC. (d) DPC.

(Figs. 11 (c) and (d), black points) were distributed in a rectangular region that tend towards a circular region, similar to DPCSA clustering (Fig. 11 (a)). Therefore, DPCSA achieved better clustering performance than the other three algorithms on the SD dataset.

In contrast to the 2D datasets (two attributes) analyzed above, there is no corresponding intuitive or convenient plot for high dimensionality datasets (Dim512 and Dim1024 datasets). Table 3 shows the accuracy indices were

TABLE 3. ACC, AMI, and ARI indices for artificial datasets.

Algorithm	ACC	AMI	ARI	F/P	*Par*	ACC	AMI	ARI	F/P	*Par*
	Aggregation					D31				
DPC	0.9975	0.9922	0.9956	7/7	2	0.9668	0.9539	0.9332	31/31	2
HDDPC	**0.9987**	**0.9955**	**0.9978**	7/7	-	0.9677	0.9544	0.9352	31/31	-
FKNNDPC	0.9975	0.9907	0.9949	7/7	8	**0.9690**	**0.9566**	**0.9375**	31/31	9
DPCSA	0.9505	0.9306	0.9294	7/7	-	0.9677	0.9552	0.9353	31/31	-
	Dim512					Dim1024				
DPC	**1**	**1**	**1**	16/16	2	**1**	**1**	**1**	16/16	2
HDDPC	**1**	**1**	**1**	16/16	-	**1**	**1**	**1**	16/16	-
FKNNDPC	**1**	**1**	**1**	16/16	8	**1**	**1**	**1**	16/16	8
DPCSA	**1**	**1**	**1**	16/16	-	**1**	**1**	**1**	16/16	-
	Flame					Spiral				
DPC	**1**	**1**	**1**	2/2	5	**1**	**1**	**1**	3/3	2
HDDPC	**1**	**1**	**1**	2/2	-	**1**	**1**	**1**	3/3	-
FKNNDPC	0.9917	0.9267	0.9666	2/2	5	**1**	**1**	**1**	3/3	6
DPCSA	**1**	**1**	**1**	2/2	-	**1**	**1**	**1**	3/3	-
	Path-based					New-Path-based				
DPC	0.7333	0.4997	0.4530	3/3	2	0.7367	0.5032	0.4558	3/3	2
HDDPC	0.7000	0.4620	0.4244	3/3	-	0.7033	0.4654	0.4266	3/3	-
FKNNDPC	**0.8967**	**0.7744**	**0.7323**	3/3	8	0.9900	0.9524	0.9696	3/3	8
DPCSA	0.8233	0.7073	0.6133	3/3	-	**0.9933**	**0.9694**	**0.9798**	3/3	-
	S2					S4				
DPC	**0.9662**	**0.9409**	**0.9306**	15/15	2	**0.7950**	**0.7240**	**0.6313**	15/15	2
HDDPC	0.9638	0.9371	0.9257	15/15	-	0.7836	0.7086	0.6121	15/15	-
FKNNDPC	0.9654	0.9405	0.9289	15/15	13	0.7600	0.6994	0.5998	15/15	13
DPCSA	0.9580	0.9333	0.9152	15/15	-	0.7872	0.7142	0.6219	15/15	-

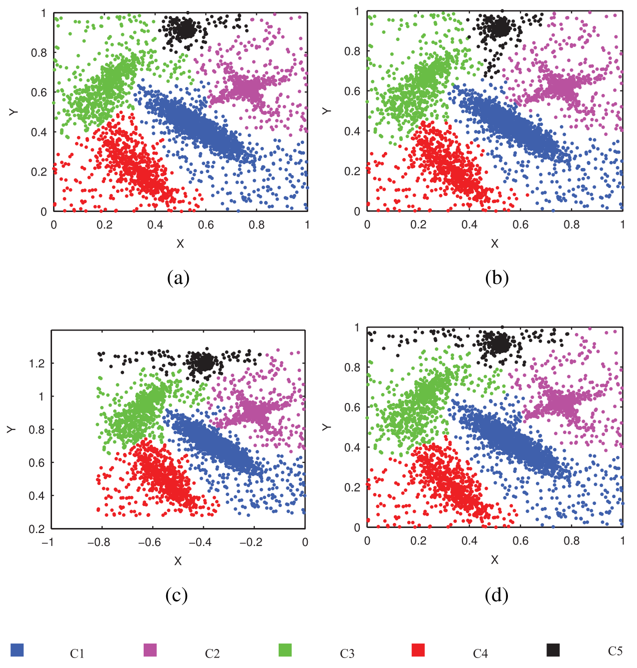


FIGURE 11. Clustering for the SD dataset. (a) DPCSA. (b) FKNNDPC. (c) HDDPC. (d) DPC.

all 1, hence the clustering results were completely correct for these datasets. The same clustering outcomes DPC, HDDPC, and DPCSA occurred for the Flame and Spiral datasets, with FKNNDPC being only slightly inferior ($ACC = 0.9917$) on the Flame dataset.

Table 3 shows evaluation index values for all datasets, where the best clustering performance are shown in bold. The symbol "-" in the sixth and eleventh columns (shown as *Par* in the table header) mean there are no corresponding value. Thus, HDDPC and DPCSA required no prespecified parameter and successfully clustered the dataset without prior knowledge.

We selected prespecified parameters for DPC and FKNNDPC by repeated experiments on artificial datasets with real cluster labels. Using these parameters, DPC achieved the best clustering performance for six of the 10 datasets and outperformed the other three algorithms (each algorithm achieved best clustering performance for five of the 10 datasets) with a weak advantage (only more than one dataset). Although FKNNDPC claims to be robust to the prespecified parameter, K, it selected five K values for the 10 datasets. Thus, DPC is more robust than FKNNDPC selecting two d_c values for the 10 datasets. The F/P columns (Table 3) show that all four algorithms find the correct number of clusters and identify all cluster centers that lie in different real clusters. FKNNDPC is also somewhat more noise sensitive than DPCSA. For example, consider the S1, S2, S3, and S4 datasets; S4 has more noise and overlap than S1-S3. FKNNDPC clustering accuracy reduces from $ACC = 0.9654$ to 0.7600 for the S2 and S4 datasets, respectively; whereas DPCSA accuracy reduces from $ACC = 0.9580$ to 0.7872, respectively.

D. EXPERIMENTAL RESULTS ON REAL-WORLD DATASETS

Prespecified DPC and FKNNDPC parameters were also selected by repeated experiments and recommended values, as shown in Table 4 for 10 real-world datasets. Values corresponding to best clustering performances were in bold.

From the *ACC* perspective, FKNNDPC achieved best clustering performance, with highest *ACC* for six of the 10 datasets. DPCSA achieved highest value for three of the 10 datasets, and DPC and HDDPC only achieved highest value for one, providing the poorest clustering performance.

Although DPCSA appears less competitive than FKNNDPC, DPCSA does not require the (optimized) prespecified parameter. Since there is no general method to find the optimal parameter selection, it can be a challenging problem, particularly for datasets without prior knowledge or true cluster labels. Although FKNNDPC had generally better accuracy, the difference to DPCSA was very small, e.g. the Iris dataset $ACC = 0.9733$ and 0.9667 for FKNNDPC and DPCSA, respectively, i.e., only 0.0066 difference. Ionosphere, Seeds, Wdbc, and Wine datasets exhibit similar outcomes. Thus, FKNNDPC and DPCSA clustering performance for the real-world datasets were very good, whereas DPC and HDDPC performance was poor.

For most datasets, FKNNDPC and DPCSA detected the true number of clusters and identified the correct center points. For example, Iris, Pima, Seeds, Wdbc, and Wine datasets (Table 4, F/P columns). For the Ionosphere dataset, FKNNDPC found three clusters for two true clusters with two center points in the same cluster, whereas DPCSA found the correct number of clusters and center points. The Libras dataset was troublesome for all four algorithms, because the number of points in each cluster was small. DPC detected eight clusters with center points scattered over seven real clusters. HDDPC detected six clusters with their center points scattered over six real clusters. Although FKNNDPC detected 12 more clusters than DPC, their center points were scattered over only seven real clusters, the same as DPC. DPCSA detected 14 clusters with center points scattered over 10 real clusters. Therefore, DPCSA was the best algorithms to detect the largest number of real clusters.

E. EXPERIMENTAL RESULTS FOR THE OLIVETTI FACE DATASET

Images (pixel matrix) are very special data types. Generally, pixel matrices are stretched into a vector and image clustering is converted into challenging high dimensional data clustering. The Olivetti face dataset is a widely used benchmark for machine learning algorithms to identify the number of subjects without previous training [46]. The pixel matrix is 92×112 and should be stretched into a 10,304 dimension vector. This dataset contains 40 subjects (clusters) where each subject has 10 different images. This dataset poses a serious challenge to DPC because the ideal number of clusters (of distinct subjects) is comparable to the number of elements (of different images, 10 for each subject).

Redundant features can negatively affect the result, particularly in high dimensional datasets, although the effects can be reduced using principal component analysis (PCA) [54]. Table 5 shows cumulative variance contribution rate corresponding to the number of principal components. We extracted the first 110 principal components based on the principle of preserving 90% cumulative variance contribution.

TABLE 5. Cumulative variance contribution from principal component.

Number	1	2	···	109	110	111
Rate (%)	17.6	30.5	···	89.9	90.0	90.1
Number	···	188	189	···	398	399
Rate (%)	···	94.9	95.0	···	99.9	100

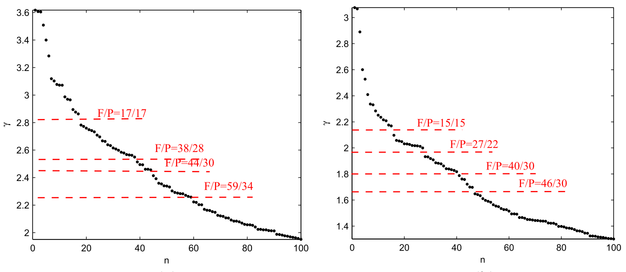


FIGURE 12. Decision graphs for the first 100 images in the Olivetti face dataset. Red dashed lines represent center point separation. (a) DPCSA. (b) FKNNDPC (*Par* = 4).

Image clustering requires computing the distance between two images. We used the Euclidean distance between two compressed images (110 dimension stretched vector). The decision graph consisted of $\gamma_i = \rho_i * \delta_i$ in decreasing order for images replacing ρ and δ. The maximal value, γ_1, is significantly larger than the second largest, γ_2, which would compress the distribution of the remaining points in the vertical axis. The user generally manually selects center points in the decision graph inconveniently. Therefore, let $\gamma_1 = \gamma_2 + 0.01$ and only the first 100 points are draw with no more than 100 clusters for Olivetti face dataset.

TABLE 6. Selection scheme performance for the Olivetti face dataset.

DPCSA				FKNNDPC			
F/P	ACC	AMI	ARI	F/P	ACC	AMI	ARI
17/17	0.345	0.392	0.151	15/15	0.315	0.375	0.142
38/28	0.535	0.602	0.362	27/22	0.473	0.528	0.311
44/30	0.550	0.623	0.387	40/30	0.558	0.605	0.405
59/34	0.580	0.636	0.427	46/30	0.540	0.609	0.393

Figure 12 shows DPCSA and FKNNDPC decision graphs, where the four red dashed lines represent the likely center point selection schemes. Table 6 shows the corresponding performance metrics. The first red dashed line in Fig. 12 (a) shows that DPCSA found 17 center points ($F = 17$) that lie in 17 clusters ($P = 17$), whereas Fig. 12 (b) shows that FKNNDPC found 15 correct center points ($F/P = 15/15$), two clusters fewer than DPCSA. The second and third red dashed lines in Fig. 12 (a) identify 28 and 30 true clusters, respectively, which is close to the true number of clusters. Therefore, the third red dashed line was selected as the optimal DPCSA result with $ACC = 0.5500$. Since the third and fourth red dashed lines for FKNNDPC (Fig. 12 (b)) detected the same true clusters ($P = 30$) and fewer center points ($F = 40$), the third red dashed line was also selected as the optimal result with $ACC = 0.5575$.

TABLE 4. ACC, AMI, and ARI indices for real-world datasets.

Algorithm	ACC	AMI	ARI	F/P	Par	ACC	AMI	ARI	F/P	Par
	Dermatology					Iris				
DPC	0.5978	0.4757	0.3617	4/3	2	0.8867	0.7668	0.7196	3/3	2
HDDPC	0.2989	0.0398	0.0421	2/1	-	0.6667	0.5768	0.5681	2/2	-
FKNNDPC	0.7737	**0.8645**	**0.7299**	6/5	7	**0.9733**	**0.9124**	**0.9222**	3/3	7
DPCSA	**0.7933**	0.7455	0.7116	6/5	-	0.9667	0.8831	0.9038	3/3	-
	Ionosphere					Libras				
DPC	0.6923	0.0897	0.1433	2/2	0.65	0.3111	0.3315	0.1756	8/7	0.5
HDDPC	0.6439	0.0016	0.0045	2/2	-	0.2222	0.1767	0.0752	6/6	-
FKNNDPC	**0.7520**	**0.2840**	**0.3550**	3/2	8	0.4139	**0.5095**	**0.3106**	12/7	9
DPCSA	0.7350	0.1335	0.2135	2/2	-	**0.4222**	0.4857	0.2637	14/10	-
	Parkinsons					Pima				
DPC	0.5897	0.1892	0.0036	2/1	5	0.6497	**0.0341**	**0.0780**	2/1	4
HDDPC	0.7231	-0.005	0.0000	2/2	-	**0.6563**	0.0045	0.0179	2/2	-
FKNNDPC	**0.8205**	**0.1772**	**0.2686**	2/1	5	0.6484	0.0012	0.0131	2/2	6
DPCSA	**0.8205**	**0.1772**	**0.2686**	2/1	-	0.6523	0.0008	0.0022	2/2	-
	Seeds					Segmentation				
DPC	0.9000	0.7172	0.7341	3/3	2	**0.6381**	**0.6226**	**0.5039**	7/6	1
HDDPC	0.8857	0.6919	0.7024	3/3	-	0.2905	0.2059	0.1006	4/2	-
FKNNDPC	**0.9286**	**0.7757**	**0.8024**	3/3	8	0.5762	0.5456	0.4128	7/6	7
DPCSA	0.8810	0.6609	0.6873	3/3	-	0.6286	0.5654	0.4532	7/6	-
	Wdbc					Wine				
DPC	0.6309	0.0035	0.0048	2/2	9	0.6180	0.4625	0.4446	3/3	3
HDDPC	0.6292	0.0009	0.0024	2/2	-	0.6629	0.3198	0.3829	3/2	-
FKNNDPC	**0.8401**	**0.3974**	**0.4502**	2/2	7	**0.9551**	**0.8550**	**0.8708**	3/3	7
DPCSA	0.8137	0.3361	0.3771	2/2	-	0.9101	0.7480	0.7414	3/3	-

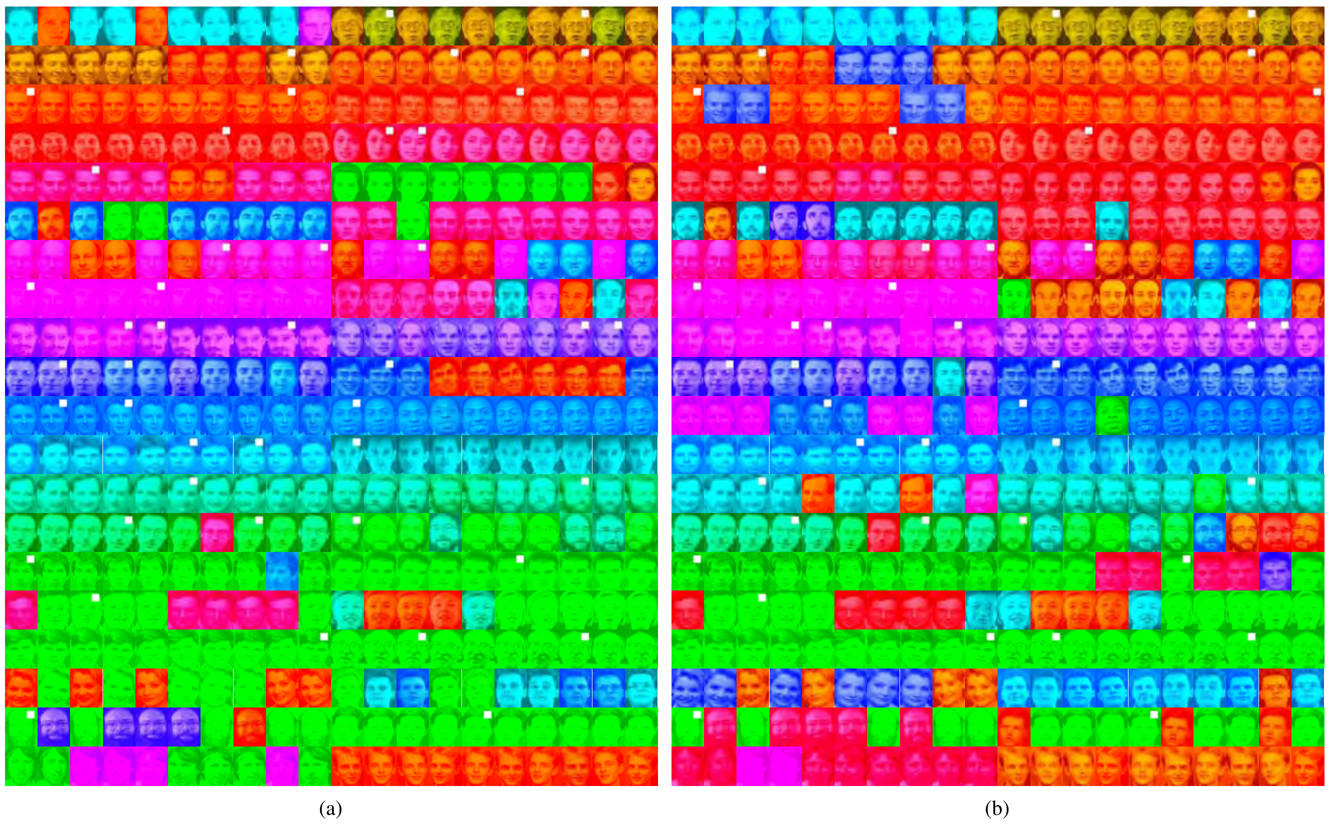


FIGURE 13. Clustering for the Olivetti face dataset. Images with the same color belong to the same cluster, center point is marked with a white square. (a) DPCSA (44 center points). (b) FKNNDPC ($Par = 4$, 40 center points).

Clustered images correspond to the 44 center points found by DPCSA (Fig. 13 (a)) and 40 center points found by FKNNDPC (Fig. 13 (b)). Images from different clusters are marked with different colors. The center point is marked with a white square at the upper right corner of the image. For convenience, a true class is described as a subject where S_i^j represents the jth image of subject i, and C_i represents cluster i clustered by algorithm. Figure. 13 (a) shows that image S_1^{10}

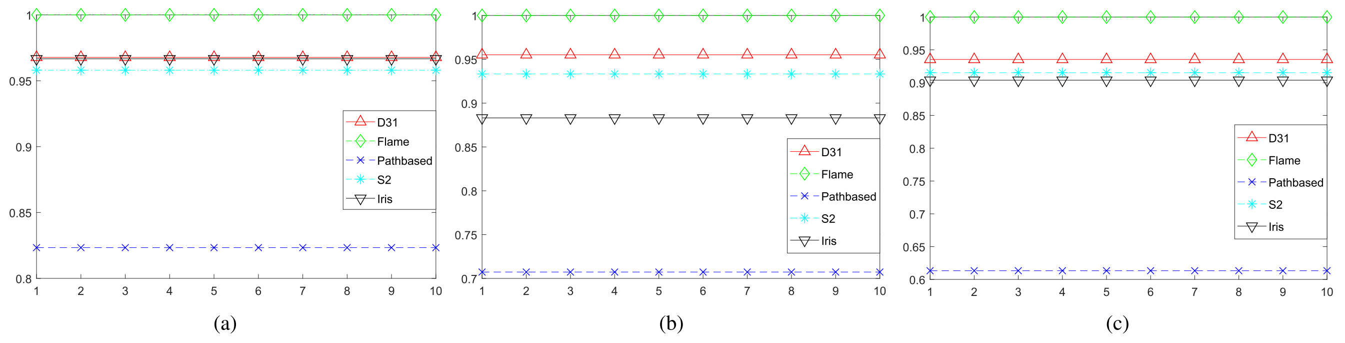


FIGURE 14. Results of different datasets in random order. (a) ACC. (b) AMI. (c) ARI.

is assigned to C_{19} (center point S_{17}^{5}) that differs from the rest of the images S_{1}^{j}, $(j = 1, \dots, 9)$. This error may be due to the stretched vector, which loses some information from the original image. Although FKNNDPC assigns image S_{1}^{10} to C_{28} (center point S_{25}^{3}) in Fig. 13 (b), this also includes images S_{1}^{2} and S_{1}^{5}, which may be the FKNNDPC local density definitions modifying the S_{1}^{2} and S_{1}^{5} cluster labels and assigning them to C_{28}. This is not always effective, because the rest of the images S_{1}^{j}, $(j \neq 2, 5, 10)$ (Fig. 13 (b)) are assigned to C_{27}. This indicates that both DPCSA and FKNNDPC can assign the same subject images to different clusters. By selecting a well-designed parameter, FKNNDPC can recognize the correct number of clusters similar to DPCSA which avoids prespecified parameters. They achieved very similar clustering accuracy ($ACC = 0.550$ and 0.5575 for DPCSA and FKNNDPC,respectively). Since it requires no prespecified parameters achieves comparable or superior clustering performance, the proposed DPCSA is a competitive algorithm.

F. RUNTIME COMPARISON

Section V discussed DPC, HDDPC, FKNNDPC, and DPCSA algorithm the time complexity. This section compares real runtime of these algorithms for different datasets, as shown in Table 7.

TABLE 7. Algorithm runtimes (unit: *s*).

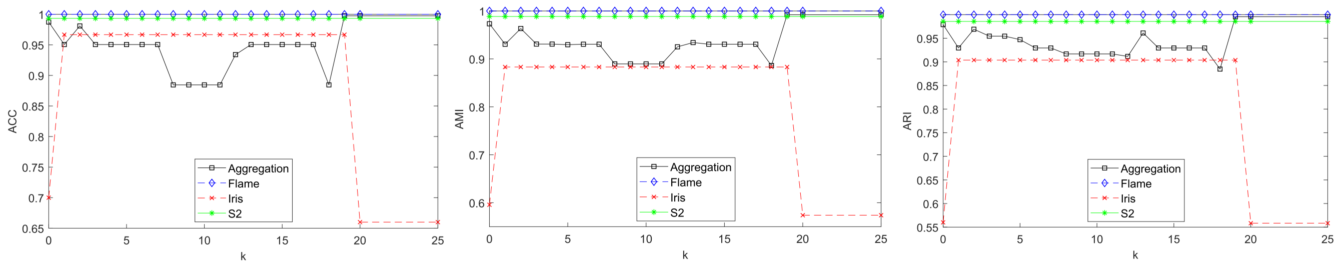
Dataset	DPC	HDDPC	FKNNDPC	DPCSA
Dermatology	0.1451	1.846	0.7433	0.3591
Iris	0.1344	0.8105	0.2683	0.1833
Ionosphere	0.7136	1.8292	1.4204	0.3374
Libras	0.1755	1.8599	0.7747	0.3669
Parkinsons	0.2110	0.9162	0.7183	0.2191
Pima	0.2129	6.6044	3.7022	1.2322
Seeds	0.1351	0.6938	0.2995	0.2277
Segmentation	0.1394	0.7039	0.4217	0.2186
Wdbc	0.1717	6.7802	1.3261	0.5642
Wine	0.1672	0.9524	0.2771	0.2093
Aggregation	0.2121	6.5305	3.1045	0.9067
D31	1.4188	107.7421	36.1629	15.6798
Flame	0.1545	0.7084	0.5099	0.2302
Path based	0.1555	1.7984	0.5270	0.2782
Spiral	0.1484	1.7704	0.6140	0.2625
S1	3.7751	3559.489	78.521	33.334
S2	3.6204	6107.639	81.539	40.539
S3	3.6072	4731.357	94.481	44.046
S4	3.6951	4728.152	103.667	48.284

HDDPC was the least efficient algorithm with runtimes an order of magnitude greater than the other three algorithms and increasing exponentially as the number of points increases. For example, runtime for the Iris dataset with 150 points $= 0.8105\ s$, but for the S1 dataset with 5,000 points $= 3559.489s$. There was less than one order of magnitude difference across the other three algorithm runtimes. However, the differences were significant between DPC, FKNNDPC, and DPCSA. DPC was the fastest, then DPCSA, approximately twice or longer than DPC, and FKNNDPC runtime was approximately twice or longer than DPCSA. FKNNDPC slow speed was mainly due to searching the fuzzy KNN for each point and assignment strategy 2 while computing point membership for each cluster, whereas DPCSA only searched nearest neighbor point and assigned cluster labels, and took advantage of NNDT to further reduce runtime. These runtimes were consistent with the additional computation of the improved algorithms. Thus, DPCSA is more efficient than FKNNDPC and HDDPC, but less than DPC.

G. ORDER SENSITIVITY

The order sensitivity is a measure of algorithm sensitivity to the order of points in dataset. The strong randomness result of the algorithm must be order sensitivity. Therefore, order sensitivity can test the stability of algorithm and avoids the erroneous evaluation of algorithm performance caused by the random result.

We used DPCSA to cluster D31, Flame, Path-based, S2 and Iris five datasets again, and randomized the order of points in them. For each dataset, we repeated clustering 10 times and computed ACC, AMI and ARI values to discuss the order sensitivity of DPCSA. If there are significant differences among the results of repeated experiments, we believe that DPCSA is order sensitivity; otherwise, it is not order sensitivity. Figure 14 shows that the ACC, AMI and ARI values are stable for the same dataset without any abrupt change or severe fluctuations. In the figure, each line represents a single dataset, and each point on the line is one experiment. In this regard, it is clear that DPCSA is not order sensitivity and the clustering result of DPCSA is not randomness.



(a) (b) (c)

FIGURE 15. Clustering evaluation for different *k* values on the Aggregation, Flame, Iris, S2 datasets. (a) ACC. (b) AMI. (c) ARI.

H. DISCUSSION REGARDING FIXED KNN PARAMETER

This paper addressed the prespecified parameter limitation by improving local density, based on both fixed KNN ($k = 5$) and weighted local density sequence. The parameter requiring pre-specification was converted to a constant while preserving or even improving clustering performance, which is convenient the user to directly cluster real-world datasets without real class labels.

We selected $k = 5$ after careful consideration. On one hand, nearest neighbor is susceptible to interference from accidental meeting of two or three outliers, hence 2NN, 3NN, or 4NN may be close to normal data points, and not easily distinguished. On the other hand, we investigated DPC-KNN and FKNNDPC for many datasets, finally settling on $k = 5$, except for the Olivetti face dataset, where $k = 4$ since each cluster had only 10 images. Thus, k remains an empirical parameter and may not be optimal. Therefore, we used the Aggregation, Flame, Iris, S2 datasets and investigated the effects of different k on ACC, AMI, ARI, as shown in Fig. 15. Clustering performance was robust and ACC, AMI, ARI were maximized for k = 2-19 on the Iris dataset. Except for Aggregation, Flame and S2 had similar conclusion. The best clustering performance of Aggregation corresponded to $k = 19, 20, 25$, and the $k = 5$ selected in this paper was less than them. Considering both clustering performance and avoiding predefined parameter, we regard the fixed $k = 5$ value as the first term of the improved local density. Hence, considering all the aspects we fixed $k = 5$.

VII. CONCLUSION

This paper proposes a modified DPC algorithm incorporating weighted local density sequence and nearest neighbor assignment (DPCSA) to improve DPC from three aspects: (a) local density is improved to avoid user prespecified parameter based on both weighted local density sequence and fixed KNN, (b) assignment strategy is split into two stages using a boundary condition to reduce assignment error propagation, (c) nearest neighbor dynamic table ($NNDT$) is proposed to improve clustering efficiency.

We compare the proposed DPCSA clustering performance with DPC, HDDPC and FKNNDPC algorithms using ACC, AMI, ARI and efficiency on artificial and real-world datasets, and the well-known Olivetti face dataset. All experiments verify that DPCSA clustering without requiring a prespecified parameter is significantly better than DPC and HDDPC, and slightly better than FKNNDPC. DCPSA is more efficient than FKNNDPC and HDDPC, and less than DPC.

Overall, DPCSA is a highly accurate and efficient clustering algorithm that does not require user prespecified parameter. Thus, it is suitable for a wide range of research and practical applications.

For future work, we want to solve the following problems. First, DPCSA with a fixed k value cannot be called non-parameter. We will continue to explore the non-parameters algorithm. Second, although DPCSA improves the clustering efficiency in assignment strategies, the high complexity of local density leads to the lower efficiency of DPCSA than DPC. We will continue to reduce the computational complexity of local density. Third, DPCSA clustering performance cannot exceed all other algorithms, so we will continue to study more reasonable definitions of local density and distance. Forth, we try to extend DPCSA to cluster big datasets.

REFERENCES

[1] J. Han, M. Kamber, and J. Pei, *Data Mining. Concepts and Techniques*. San Mateo, CA, USA: Morgan Kaufmann, 2006.

[2] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: An overview," *Wiley Interdiscipl. Rev. Data Mining Knowl. Discovery*, vol. 2, no. 1, pp. 86–97, 2012.

[3] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: An overview, II," *Wiley Interdiscipl. Rev. Data Mining Knowl. Discovery*, vol. 7, no. 6, p. e1219, 2017.

[4] A. K. Jain, "Data clustering: 50 Years beyond k-means," *Pattern Recognit. Lett.*, vol. 31, pp. 651–666, Jun. 2010.

[5] L. Kaufman and P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. Hoboken, NJ, USA: Wiley, 1990.

[6] M. Singh and M. Bansal, "A survey on various clustering techniques with k-means clustering algorithm," *Int. J. Comput. Sci. Netw. Secur.*, vol. 15, no. 6, pp. 60–65, 2015.

[7] 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*, Portland, Oregon, 1996, pp. 226–231.

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

[9] T. Chen, N. L. Zhang, T. Liu, K. M. Poon, and Y. Wang, "Model-based multidimensional clustering of categorical data," *Artif. Intell.*, vol. 176, no. 1, pp. 2246–2269, 2012.

[10] S. Vega-Pons and J. Ruiz-Shulcloper, "A survey of clustering ensemble algorithms," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 25, no. 3, pp. 337–372, 2011.

[11] Y. Kim, K. Shim, M.-S. Kim, and J. S. Lee, "DBCURE-MR: An efficient density-based clustering algorithm for large data using mapreduce," *Inf. Syst.*, vol. 42, pp. 15–35, Jun. 2014.

[12] K. M. Kumar and A. R. M. Reddy, "A fast DBSCAN clustering algorithm by accelerating neighbor searching using groups method," *Pattern Recognit.*, vol. 58, pp. 39–48, Oct. 2016.

[13] D. Birant and A. Kut, "ST-DBSCAN: An algorithm for clustering spatial-temporal data," *Data Knowl. Eng.*, vol. 60, no. 1, pp. 208–221, 2007.

[14] M. Ankerst, M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Philadelphia, PA, USA, 1999, pp. 49–60.

[15] A. Hinneburg and D. A. Keim, "An efficient approach to clustering in large multimedia databases with noise," in *Proc. 4th Int. Conf. Knowl. Discovery Data mining*, New York, NY, USA, 1998, pp. 58–65.

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

[17] M. Wang, W. Zuo, and Y. Wang, "An improved density peaks-based clustering method for social circle discovery in social networks," *Neurocomputing*, vol. 179, pp. 219–227, Feb. 2016.

[18] P. Contucci, R. Luzi, and C. Vernia, "Inverse problem for the mean-field monomer-dimer model with attractive interaction," *J. Phys. A, Math. Theor.*, vol. 50, no. 20, 2017, Art. no. 205002.

[19] M. Fedele and C. Vernia, "Inverse problem for multispecies ferromagneticlike mean-field models in phase space with many states," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 96, no. 4, 2017, Art. no. 042135.

[20] J. Cao *et al.*, "Comprehensive single-cell transcriptional profiling of a multicellular organism," *Science*, vol. 357, no. 6352, pp. 661–667, 2017.

[21] L. Dong, N. Feng, and Q. Zhang, "LSI: Latent semantic inference for natural image segmentation," *Pattern Recognit.*, vol. 59, pp. 282–291, Nov. 2016.

[22] 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.

[23] 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.

[24] 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.

[25] Y.-A. Geng, Q. Li, R. Zheng, F. Zhuang, R. He, and N. Xiong, "RECOME: A new density-based clustering algorithm using relative KNN kernel density," *Inf. Sci.*, vols. 436-437, pp. 13–30, Apr. 2018.

[26] M. Du, S. Ding, and Y. Xue, "A robust density peaks clustering algorithm using fuzzy neighborhood," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 7, pp. 1131–1140, 2018.

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

[28] S. A. Seyedi, A. Lotfi, P. Moradi, and N. N. Qader, "Dynamic graph-based label propagation for density peaks clustering," *Expert Syst. Appl.*, vol. 115, pp. 314–328, Jan. 2019.

[29] L. Tao, W. Li, and Y. Jin, "An optimal density peak algorithm based on data field and information entropy," in *Proc. Int. Conf. Data Mining, Commun. Inf. Technol.* New York, NY, USA: ACM, 2017, p. 4.

[30] C. Zhang, M. Ni, H. Yin, and K. Qiu, "Developed density peak clustering with support vector data description for access network intrusion detection," *IEEE Access*, vol. 6, pp. 46356–46362, 2018.

[31] 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.

[32] Z. Zhou, G. Si, Y. Zhang, and K. Zheng, "Robust clustering by identifying the veins of clusters based on kernel density estimation," *Knowl. -Based Syst.*, vol. 159, pp. 309–320, Nov. 2018.

[33] S. Bing, H. Lixin, and Y. Hong, "Adaptive clustering algorithm based on kNN and density," *Pattern Recognit. Lett.*, vol. 104, pp. 37–44, Mar. 2018.

[34] J. Lu and Q. Zhu, "An effective algorithm based on density clustering framework," *IEEE Access*, vol. 5, pp. 4991–5000, 2017.

[35] M. Chen, L. Li, B. Wang, J. Cheng, L. Pan, and X. Chen, "Effectively clustering by finding density backbone based-on kNN," *Pattern Recognit.*, vol. 60, pp. 486–498, Dec. 2016.

[36] 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.

[37] X. Xu, S. Ding, and Z. Shi, "An improved density peaks clustering algorithm with fast finding cluster centers," *Knowl.-Based Syst.*, vol. 158, pp. 65–74, Oct. 2018.

[38] X. Xu, S. Ding, M. Du, and Y. Xue, "DPCG: An efficient density peaks clustering algorithm based on grid," *Int. J. Mach. Learn. Cybern.*, vol. 9, no. 5, pp. 743–754, 2018.

[39] 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, 2018.

[40] L. Bai, X. Cheng, J. Liang, H. Shen, and Y. Guo, "Fast density clustering strategies based on the k-means algorithm," *Pattern Recognit.*, vol. 71, pp. 375–386, Nov. 2017.

[41] J. Jiang, D. Hao, Y. Chen, M. Parmar, and K. Li, "GDPC: Gravitation-based density peaks clustering algorithm," *Physica A, Statistical Mech. Appl.*, vol. 502, pp. 345–355, Jul. 2018.

[42] J. Jiang, Y. Chen, D. Hao, and K. Li, "DPC-LG: Density peaks clustering based on logistic distribution and gravitation," *Phys. A, Stat. Mech. Appl.*, vol. 514, pp. 25–35, Jan. 2019.

[43] C. Sun, Q. Li, H. Li, Y. Shi, S. Zhang, and W. Guo, "Patient cluster divergence based healthcare insurance fraudster detection," *IEEE Access*, vol. 7, pp. 14162–14170, 2018.

[44] X. Wu *et al.*, "Top 10 algorithms in data mining," *Knowl. Inf. Syst.*, vol. 14, no. 1, pp. 1–37, 2008.

[45] D. Dua and C. Graff, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California, Irvine CA, USA, Tech. Rep., 2017. [Online]. Available: http://archive.ics.uci.edu/ml/citation_policy.html

[46] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in *Proc. IEEE Workshop Appl. Comput. Vis.*, Dec. 1994, pp. 138–142.

[47] 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.

[48] P. Fränti and O. Virmajoki, "Iterative shrinking method for clustering problems," *Pattern Recognit.*, vol. 39, no. 5, pp. 761–775, 2006.

[49] P. Franti, O. Virmajoki, and V. Hautamaki, "Fast agglomerative clustering using a k-nearest neighbor graph," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 11, pp. 1875–1881, Nov. 2006.

[50] L. Fu and E. Medico, "FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data," *BMC Bioinf.*, vol. 8, no. 1, p. 3, 2007.

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

[52] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: is a correction for chance necessary?" in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, Montreal, Quebec, Canada, 2009, pp. 1073–1080.

[53] D. Cai, X. He, and J. Han, "Document clustering using locality preserving indexing," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 12, pp. 1624–1637, Dec. 2005.

[54] R. Vidal, Y. Ma, and S. Sastry, *Generalized Principal Component Analysis*. New York, NY, USA: Springer, 2016.

DONGHUA YU received the M.S. degree from the College of Science, Harbin Engineering University, Harbin, China, in 2015. He is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Harbin Institute of Technology.

His research interests include machine learning and bioinformatics.

GUOJUN LIU received the Ph.D. degree from the School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China, where he is currently an Associate Professor.

His current research interests include machine learning, computer vision, and image processing.

MAOZU GUO received the Ph.D. degree from the School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China.

He is currently a Professor with the School of Electrical and Information Engineering, Beijing University of Civil Engineering and Architecture, Beijing, China. His current research interests include machine learning, bioinformatics, and image processing.

XIAOYAN LIU received the Ph.D. degree from the School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China, where she is currently an Associate Professor.

Her current research interests include machine learning and bioinformatics.

SHUANG YAO received the Ph.D. degree from the School of Economics and Management, Harbin Engineering University.

She is currently a Lecturer with the College of Economics and Management, China Jiliang University. Her research interests include data mining and decision analysis.