Received June 25, 2019, accepted July 3, 2019, date of publication July 22, 2019, date of current version August 20, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2927757

Density Peaks Clustering Based on Local Minimal Spanning Tree

RENMIN WANG AND QINGSHENG ZHU
Chongqing Key Laboratory of Software Theory and Technology, College of Computer Science, Chongqing University, Chongqing 400044, China

Corresponding author: Qingsheng Zhu (qszhu@cqu.edu.cn)

This work was supported in part by the National Nature Science Foundation of China under Grant 61272194, Grant 61502060, and Grant 61702060, and in part by the Project of Chongqing Education Commission under Grant KJZH17104.

ABSTRACT The fake center is a common problem of density-based clustering algorithms, especially for datasets with clusters of different shapes and densities. Clustering by fast search and find of density peaks (DPC) and its improved versions often ignore the effect of fake centers on clustering quality. They usually have a poor performance even the actual number of centers are used. To solve this problem, we propose a density peaks clustering based on local minimal spanning tree (DPC-LMST), which generates initial clusters for each potential centers first and then introduce a sub-cluster merging factor (SCMF) to aggregate similar sub-clusters. Meanwhile, we introduce a new strategy of representative points to reduce the size of data and redefine local density ρ_i and distance δ_i of each representative point. Furthermore, the hint of γ is redesigned to highlight true centers for datasets with clusters of different densities. The proposed algorithm is benchmarked on both synthetic and real-world datasets, and we compare the results with K-means, DPC, and the three state-of-the-art improved DPC algorithms.

INDEX TERMS Clustering, density peaks, fake centers, local minimal spanning tree, representative points.

I. INTRODUCTION

Clustering is an important field of data mining, and it plays an important role in the fields of pattern recognition [1], [2], image processing [3], recommendation [4], and etc. Several different clustering strategies such as partitioning, hierarchical, distribution-based and density-based clustering have been proposed [5]–[7], and the density-based methods have drawn broad attention and research due to their advantages of intuition and detecting clusters with an arbitrary shape. DBSCAN [8] and mean-shift [9] method are typical representatives of density-based approaches, however, the former is sensitive to user-provided parameters and the latter is computationally costly. In 2014, a master piece of density-based clustering algorithm, named DPC, has been proposed based on the assumptions of cluster centers by Rodriguez and Laio [10]. In DPC, for each data point i, two quantities: local density ρ_i and its distance ρ_i from points of higher density are computed. And the only points of high δ and relatively high ρ are identified as cluster centers.

DPC is simple and effective, but it also has some drawbacks and many researchers focus on cut-off distance or label propagation. For example, Du *et al.* [11] introduced k-nearest neighbors (kNN) to local density computation. In [12] and [13], researchers also used the idea of kNN to define their local density. Li and Tang [14] defined their local density with mutual k-nearest-neighbor graph (mKNN graph). Liu *et al.* [15] designed their local density based on shared nearest neighbors (SNN). In [16], Seyedi *et al.* proposed a graph-based label propagation to assign labels to remaining points and form final clusters. In addition, some researchers applied the DPC method to various fields [17]–[19].

However, DPC and DPC-based algorithms often ignore the problem of fake center, which would lead to poor quality of clustering. The main reason of fake centers is that there are some points which have relatively large local density and distance within the same clusters, and finally leads to the γ values of them are close to or even greater than that of some true centers. That is the fake centers are the data points whose γ values are local maximum. In fact, fake center is a common issue of density-based algorithms especially if the datasets contain clusters with different shapes and densities. In addition, true centers would be covered up by fake centers, which results in incorrect results even using the correct number of centers.

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

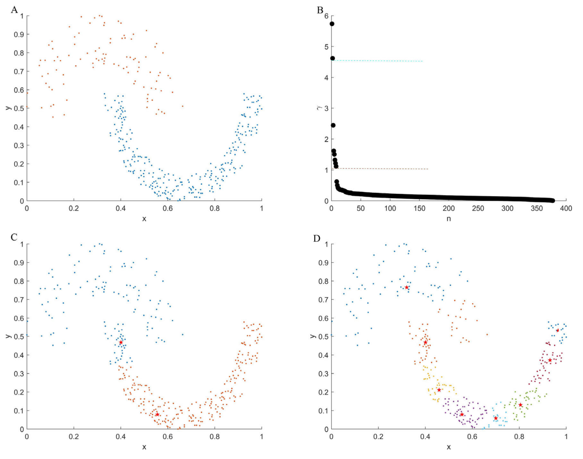


FIGURE 1. Example of DPC on *jain* dataset.

Fig.1 depicts the results of traditional DPC on the classic jain dataset. As is shown in Fig.1 (A), *jain* dataset contains two clusters of different densities. Fig.1 (B) shows its plot of $\gamma_i = \rho_i \times \delta_i$ sorted in decreasing order, and the local density is calculated by Gaussian kernel method. According to Fig.1 (B), the hint provided by it is not clear and it is reasonable to choose the top-2, or top-8 scored points as centers. Fig.1 (C) and (D) show the results for selecting two and eight centers respectively, and cluster centers are marked with red pentagram. True center of the lower-density cluster is lost and the other cluster contains a fake center in Fig.1 (C). The price to pay for finding true center of the lower-density cluster is that more fake centers have to be added, as is shown in Fig.1 (D). In Fig.1 (D), γ value of the true center of the lower-density cluster only ranks 7th, therefore, 5 fake centers are added. As can be seen from this example, fake centers lead to low quality of clustering result by using DPC algorithm even the actual number of centers are given. In addition, DPC method is hard to recognize clearly the number of centers sometimes, and results might vary from selection of centers.

Some algorithms have been proposed to solve the problem of fake centers through aggregating all potential centers or clusters. However, they usually have the following problems: (1) only one intuitive factor, such as distance or density, is taken into account; (2) merge method of potential centers or clusters often lack fault tolerance. For example, Xu *et al.* [20] developed a density-peak-based hierarchical clustering method (DenPEHC), which directly generates clusters on each possible cluster layer, and introduces a grid granulation framework to enable DenPEHC to handle larger-scale and high-dimension datasets. In DenPEHC, the gradient of γ determines the centers and stairs. Wang *et al.* [21] introduced a quantity affinity to handle multiple density peaks existing in one cluster and assign each point to its true cluster. However, the quantify affinity only consider the distance between candidate centers. In [22], Liu *et al.* presented a method to select initial centers automatically and aggregate clusters if they are density reachable. However, density is not the only factor in fake centers, and δ is another factor of fake center. In fact, the problem of fake center would still occur on data with non-spherical clusters, especially for manifold data.

To solve the above problems and improve clustering efficiency, a density peaks clustering algorithm based on local minimum spanning tree (DPC-LMST) is proposed in this paper. The DPC-LMST introduces a new idea of local minimal spanning tree (LMST) to generate representative points, redefines local density ρ_i and distance δ_i for each representatives, applies new strategy of γ to select potential centers, and defines a sub-cluster merging factor (SCMF) which integrates density and boundary distance between sub-clusters to aggregate sub-clusters. The proposed method has the following characteristics: (1) a strategy of representative points based on LMST significantly reduces the size of original data; (2) hint of γ is more adaptive to arbitrary datasets; and (3) the SCMF combines two factors of inner densities and boundary distance between different sub-clusters, and it has fault tolerance for choice of initial centers.

The DPC-LMST algorithm is performed both on synthetic and real-world datasets. And the results are compared with K-means [23], DPC [10], FkNN-DPC [12], DPC-DLP [16], and SNN-DPC [15] in terms of three popular metrics (Adjust Mutual Information, AMI [24]; Adjust Rand Index, ARI [25] and Fowlkes-Mallows Index, FMI [26]). Meanwhile, we compare the running time of our method with the competing algorithms. The paper is organized as follows: In section II, we introduce k-nearest neighbors (kNN) and DPC algorithm. The improved approach and analysis are described in Section III. Experiments results are presented in Section IVand Section V concludes the paper.

II. RELATED WORKS

In this section, we review some basic concepts concerning k-nearest neighbors and traditional DPC method.

A. K-NEAREST NEIGHBORS

k-nearest neighbors (kNN) is usually used to measure a local neighborhood of an instance in the fields of classification, clustering, local outlier detection, and etc. Given a data point i, Let kNN(i) be a set of nearest neighbors of a point i and it can be expressed as:

$$kNN(i) = \{j | d_{ij} \leq d_{i,kth(i)}\} \tag{1}$$

where $d(x_i, x_j)$ is the Euclidean distance between i and j, and $kth(i)$ is the k-th nearest neighbor of i. Local regions measured by kNN are often termed k-nearest neighborhood, which, in fact, is a circular or spherical area of radius $R = d_{i,kth(i)}$. Therefore, kNN-based method can not apply to handle datasets with clusters of non-spherical distributions. This is the major weakness of kNN measurement. In addition, kNN-based algorithms are often sensitive to k.

To overcome the drawbacks of cut-off distance of DPC, some researchers [13]–[15] introduce the idea of kNN to calculate the local densities and the k is set as a percentage of the size of a dataset. Liu *et al.* [15] use the idea of SNN to compute local density, and SNN is also defined based on kNN. Therefore, these methods usually have poor clustering results when handling datasets with non-spherical clusters.

B. DPC ALGORITHM

The main idea of DPC is that cluster centers are characterized by a higher density than their neighbors and by a relatively large distance from points with higher densities [8]. For each data point i, it computes two quantities: its local density ρ_i and its distance δ_i from points of higher density.

$$\rho_i = \sum_j \chi(d_{ij} - d_c) \tag{2}$$

where $d_{ij}$is the distances between data points and d_c is a cutoff distance, and $\chi(x) = 1$ if $x < 0$ else $\chi(x) = 0$.Or it can be defined via a Gaussian kernel:

$$\rho_i = \sum_j (-\frac{d_{ij}^2}{d_c^2}) \tag{3}$$

Meanwhile, a hint for choosing the number of centers is provided and it is defined as follow:

$$\gamma_i = \rho_i \times \delta_i \tag{4}$$

Traditional DPC algorithm consists of two steps: finding centers and then propagating labels of the centers to the remaining points. Although it is simple and effective, traditional DPC exists the problems of label propagation and local density with respect to cut-off distance, and many researchers devoted to improve these issues [11], [12], [27]. In addition, fake center is another important factor that influences the performance of DPC and its improved versions, but it is often overlooked. Because clustering quality of DPC and DPC-based algorithms depends on choices of centers. However, the exact number of centers sometimes is not clear according to decision graphs, and clustering results would vary from different choices, especially for datasets with clusters of different shapes and densities. In other words, fake center which is a common problem for datasets with clusters of complex distributions would lead to low quality of clustering results. To solve above problems and improve clustering efficiency, density peaks clustering based on local minimal spanning tree (DPC-LMST) is proposed.

III. PROPOSED ALGORITHM

In this section, we present a detailed description of our algorithm and take *jain* dataset as an example to show it. The proposed algorithm can be divided into two phases: initial clustering and sub-clusters merging. Obviously, pure sub-clusters, namely include only points of the same true cluster, are useful to improve the quality of final results. To improve the speed and quality of initial clustering, we introduce two new definitions of representative and local neighborhood. In the second phase, some adjacent and similar clusters will be merged depending on the comparison of density difference and boundary distance between initial clusters. Finally, The proposed algorithm restore the final labels of representatives back to original data according to correspondence between representatives and original data. And then we can get the final clustering result of the original data. The main processes of our algorithm are described as follows:

step 1: generate representatives of a dataset based on local minimal spanning tree, and calculate two quantities (ρ and δ) for them.
step 2: execute density peaks clustering for the representative points to get initial clusters.
step 3: calculate structure differences and border distances between initial clusters.
step 4: merge the initial clusters according to a sub-cluster merging factor (SCMF).
step 5: restore representatives back to original data and form the final clusters.

A. REPRESENTATIVES BSAED ON LMST

To overcome the limitations of kNN, we present a new local structure, named local minimal spanning tree (LMST for short). Compared with kNN, it is data-independent and ensures that all neighbors of a point are from the same clusters. A local minimal spanning tree is composed of two fundamental units: node set N and edge set E. Given a point i and parameter ϵ, a recursive definition of LMST can be written blow:

$$T_n^i = \{< i, j > | j \in 1th(i)\}, \quad n = 1 \tag{5}$$
$$T_n^i = \{< i, j > | i \in N_{n-1} \wedge j \in 1th(N_{n-1}(i))\}, \quad n \in [2, \epsilon] \tag{6}$$

where $N_{\text{n}-1}(i)$ denotes the nodes of $T_{\text{n}-1}(i)$, and $1th(i)$ and $1th(N_{\text{n}-1}(i))$ are the 1-th neighbor or neighbors corresponding to i and $N_{\text{n}-1}(i)$. From this definition, nodes of a LMST of a data point can be regarded as a set of direct and indirect neighbors of it. In the process of constructing a LMST of an instance, the reference point of the LMST is dynamic instead of a fixed reference point such as kNN.

Let $d_{\text{n}}(n = 1, 2, \ldots, \varepsilon)$ be distance of the n-th edge of a $T(i)$, then its inner average distance (IAD) is defined as the mean of the distances of its edges.

$$IAD_i = \frac{1}{\epsilon} \sum_1^{\epsilon} d_n \tag{7}$$

Like kNN, LMST also need to provide an integer to determine the size of it. However, our method is robust over a wide range of ε values. The details of generating LMST for a dataset are described in Algorithm 1. Unlike traditional methods, it does not construct a LMST structure for each point (see line 2.3 in Algorithm 1). Because neighboring points may have similar even the same LMST structures and they can share them. Additionally, it is worth noting that different LMSTs can share one or more neighbors.

Techniques of representative-based clustering which use representative points to perform clustering analysis can

Algorithm 1 Generating LMST Structures

```
Input: dataset $X = \{x_1, x_2, \ldots x_n\}$ , parameter $\varepsilon$
Output: $Tmatrix$
1. calculate k-nearest neighbors for each data point;
2. while $X \neq 0$
2.1    $p \leftarrow X[1]$; // choose first point of $X$ as the starting point
2.2    $Tmatrix \leftarrow Tmatrix + \{T(p)\}$;// construct and save a LMST structure
2.3    $X \leftarrow X - N(p)$; // remove points of the $T(p)$ from $X$
3. end while
4. output $Tmatrix$
```

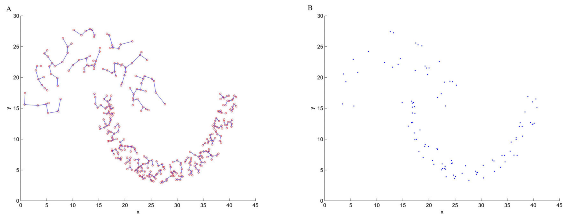


FIGURE 2. Example of representatives on *jain* dataset for $\epsilon = 5$. (A) LMST graph. (B) Representative points.

improve clustering efficiency [28]–[31] . Inspired by similarity of neighbors of LMST, we use centroid of each LMST as its corresponding representative point in a dataset, and then we only need to handle the representative points instead of all points of the dataset. Fig.2 (A) and 2(B) show the LMST graph and representative points of *jain* dataset for $\epsilon = 5$, respectively. We observe that the representatives highly remain local and global features of original data, and the number of data is reduced from 377 to 101.

B. INITIAL CLUSTERING

In this section, we handle the representative points to get initial clusters based on basic idea of DPC. To adapt new data context (representative points), we redefine local density ρ and distance from higher density δ. Meanwhile, we redesign the hint of γ to highlight true centers of datasets with different densities.

According to the idea of DPC, two quantities need to be calculated for each representative point r_i: its local density ρ_i and its distance δ_i from representative points of higher density.

$$\rho_i = \exp(-IAD_i) \tag{8}$$

$$\delta_i = \begin{cases} \min\{d(r_i, r_j)\}, & \text{if } \exists\, r_j \text{ s.t. } \rho_i < \rho_j \\ \max\{d(r_i, d_j)\}, & \text{otherwise} \end{cases} \tag{9}$$

where $d(r_\mathrm{i}, r_\mathrm{j})$ is the distance between r_i and r_j. Like some improved DPC algorithms, the new local density equation also utilizes the environment which the points are located. In addition, our definition of local density is more adaptive and robust than definitions of kNN-based due to the advantages of LMST.

DPC and its improved versions choose centers by using the decision graph or a hint of ordered γ. However, it is easy to miss true centers of low-density clusters of a dataset. To solve this problem, a new strategy for choosing centers is provided by the plot of $\gamma_\mathrm{i} = \delta_\mathrm{i}$ sorted in decreasing order. Compared to traditional strategy of $\gamma_\mathrm{i} = \rho_\mathrm{i} \times \delta_\mathrm{i}$, our method is more appropriate for arbitrary datasets. In fact, δ_i not ρ_i is the core of center choice when datasets do not contain noise data or outliers. Note that we do not handle the datasets which contains noise data or outliers in this article.

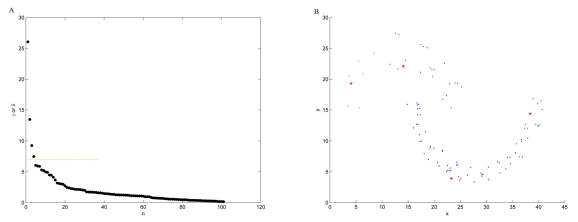


FIGURE 3. Initial clustering for the representatives in Fig.2 (B). (A) Plot of $\gamma_i = \delta_i$ in decreasing order. (B) Initial clustering result, and centers are marked with red pentagrams.

Fig.3 (A) shows plot of $\gamma_i = \delta_i$ sorted in decreasing order for data in Fig.2 (B). Clearly, we intuitively chose the top-4 scored representative points as centers according to the plot, and Fig.3 (B) shows the clustering result. We can observe that four initial clusters are generated, and each true cluster contains two pure sub-clusters. Therefore, our new strategy of γ can highlight the true centers for low-density clusters and reduce the number of fake centers. Compared to tradition DPC, our approach is more likely to get true centers and form pure clusters (see Fig.1).

C. SUB-CLUSTERS COMBLINATION

To combine the similar sub-clusters automatically, a sub-cluster merging factor (SCMF) is presented. The basic idea of SCMF is that sub-cluster A and B can be merged, if and only if they have similar cluster density and relatively small boundary distance. In addition, our SCMF has good fault tolerant to centers choice. That is we often obtain consistent and correct results for different choices of initial centers.

For any two sub-clusters A and B, we compute two quantities: their density difference $DD(A, B)$ and their boundary distance $BD(A, B)$ between A and B. Note that DD and BD are symmetric. For a sub-cluster A, its density $d(A)$ is defined as:

$$d(A) = \frac{\sum_{r_i \in A} IAD_i}{|A|} \tag{10}$$

where $|A|$ denotes the number of representative points of A. In fact, $d(A)$ is computed by the average distance of edges of internal LMST structures of A. Thus, the density difference between two sub-clusters $DD(A, B)$ can be expressed as follow:

$$DD(A, B) = \begin{cases} \dfrac{d(A)}{d(B)}, & \text{if } d(A) \geq d(B) \\ \dfrac{d(B)}{d(A)}, & \text{otherwise} \end{cases} \tag{11}$$

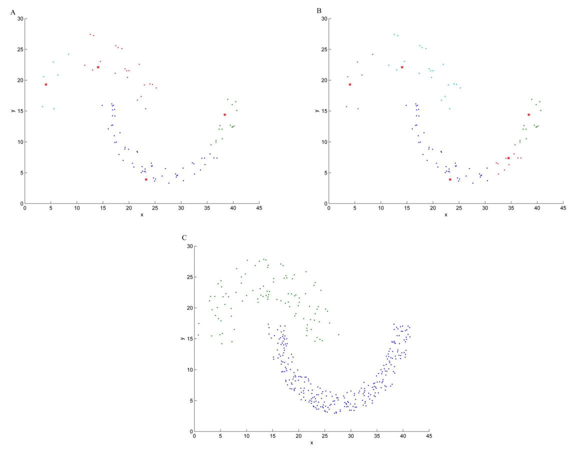


FIGURE 4. An example of fault tolerant of SCMF on *jain*. (A) Initial result for 4 centers.(B) Initial result for 5 centers. (C) the end result is the same by using SCMF.

Equation 11 is symmetric and its value should be at least 1. Intuitively, it is designed to indicate the comparison of the sparsity between two sub-clusters. In general, densities of different portions of a cluster should be similar.

The boundary distance between two clusters $BD(A, B)$ is measured by computing minimal distance between the representative point r_i and r_j, which are belong to different sub-clusters A and B, respectively.

$$BD(A, B) = min\big(d(r_i, r_j)\big), \quad r_i \in A, \; r_j \in B \tag{12}$$

where $d(r_i, r_j)$ is the distance between two representative points. Combining the effects of DD and BD, a sub-cluster merging factor (SCMF) can be expressed as follow:

$$SCMF(A, B) = BD(A, B) \times DD(A, B)^2 \tag{13}$$

A higher value of $SCMF(A, B)$ indicates a larger divergence between A and B. Because no consensus has been reached on quantitative standard of a cluster, it is hard to set a threshold of SCMF. Intuitively, the SCMF between similar sub-clusters should be much smaller than that between real clusters. There, we assume that A and B can be emerged if $SCMF(A, B) \leq Avg(SCMF)/5$, and

$$Avg(SCMF) = mean\big(SCMF(U, V)\big), \quad U \neq V \tag{14}$$

where $mean(\cdot)$ denotes the mean function, both U and V are sub-clusters. In Equation 13, the weight of DD is higher than BD because density is more important than boundary distance when distinguishing clusters intuitively.

SCMF is a mechanism which combines similar sub-clusters automatically. To illustrate the fault tolerant of the SCMF, we take jain dataset as an example. Fig.4(A) and 4(B) show results of representative points for respectively choosing 4 and 5 centers, and 4(C) shows the same final result. Moreover, we can obtain the same and correct result for different choice of centers and these results also can be observed for other datasets. Thus, our method has good fault tolerant to choice of centers compared to DPC and other relative algorithms.

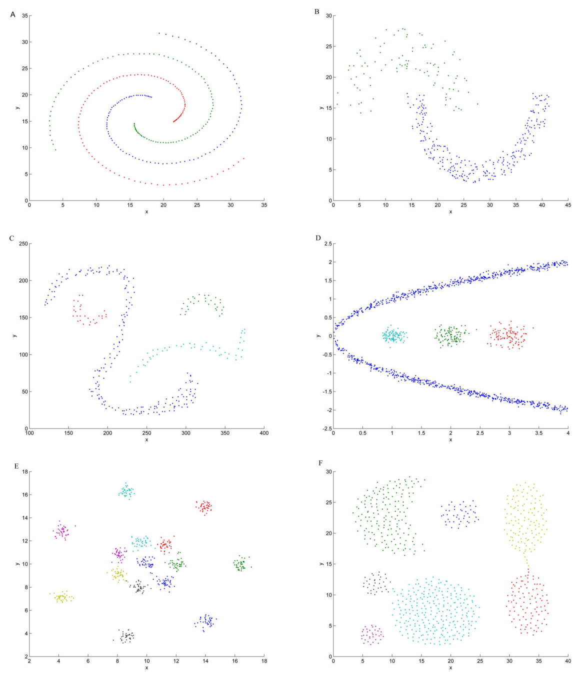


FIGURE 5. Ground truth of synthetic datasets. (A) Spiral: 312 points and 3 clusters.(B) Jain:377 points and 2 clusters.(C) Db2: 315 points and 4 clusters.(D) Mix: 1300 points and 4 clusters.(E) R15: 600 points and 15 clusters.(F) Aggregation:788 points and 7 clusters.

IV. EXPERIMENTS

To evaluate our algorithm, we made a comparison among DPC, FkNN-DPC, DPC-NLP, SNN-DPC, and K-means both on synthetic and real-world datasets. The K-means algorithm is implemented in the toolbox of MATLAB, and the other competing algorithms are based on the source code provided by the authors. DPC, FkNN-DPC, SNN-DPC and K-means are performed with use of the actual number of centers. DPC-NLP automatically set its centers, and DPC-LMST get our initial centers based on plots of δ sorted in decreasing order. Parameter k of FkNN-DPC and SNN-DPC and parameter ϵ of our method are equal to 5 for synthetic and real-world dataset, respectively. Three metrics, including adjusted mutual information (AMI), adjust rand index (ARI) and Fowlkes-Mallows index (FMI) were employed to examine the clustering accuracy. The upper bounds of the metrics are 1, and larger values indicate better clustering quality.

A. SYNTHETIC DATASETS

First, we benchmark our algorithm on synthetic datasets which are different in terms of the overall distribution, densities and number of clusters, as is shown in Fig.5. For these datasets, fake centers are easily occurred by using DPC and DPC-related methods. In addition, the clustering results would be not good even if they are performed with use of the actual number of centers.

Fig.6 (A)-(F) depict distributions of representative points for 6 synthetic datasets for $\epsilon = 5$, respectively. We can see that the new data contexts inherit the main features from the

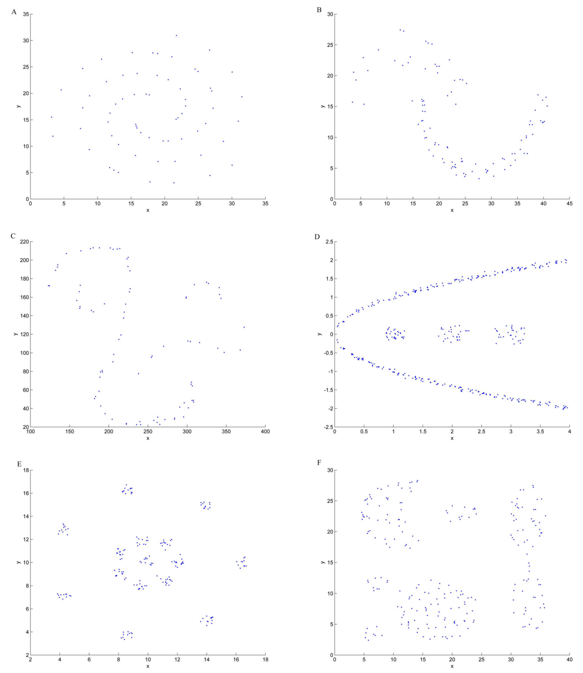


FIGURE 6. Representative points of synthetic datasets corresponding to Fig.4 for $\epsilon = 5$. There are 65, 101, 86, 383, 184, and 232 representative points in (A)-(F) corresponding to (A)-(F) in Fig.5, respectively.

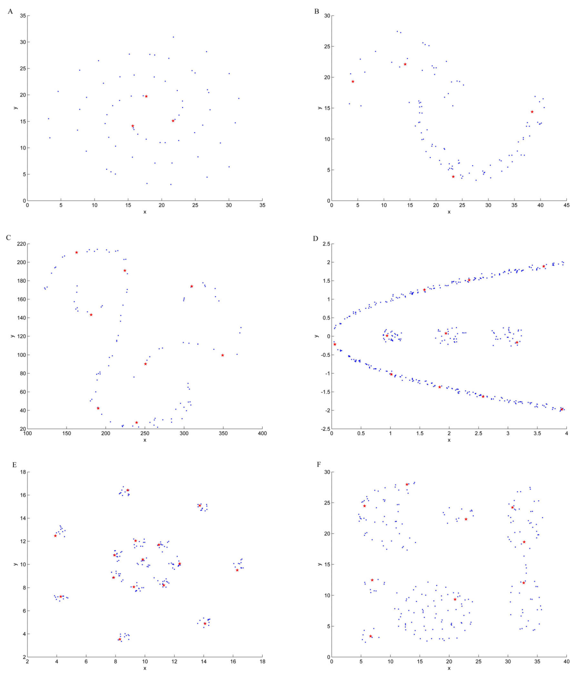


FIGURE 7. Initial centers for representative datasets in Fig.6 by using DPC-LMST. 3, 4, 8, 11, 15, and 9 initial centers are chosen, respectively.

original datasets as a whole and maintain similar local features, through comparing figures in Fig.5 with that in Fig.6. Moreover, the numbers of datasets of representative points are significantly reduced. Therefore, the design of representative points has potential to improve the efficiency of the proposed algorithm. Fig.7 shows the results of initial choice of centers

TABLE 1. AMI, ARI and FMI scores on synthetic datasets.

Alg.	AMI	ARI	FMI	AMI	ARI	FMI
		spiral			*mix*	
DPC	**1.0000**	**1.0000**	**1.0000**	0.2145	0.0279	0.4424
FkNN-DPC	**1.0000**	**1.0000**	**1.0000**	0.2639	0.5868	0.4029
DPC-DLP	0.3325	0.5538	0.3721	0.2463	0.5707	0.3965
SNN-DPC	**1.0000**	**1.0000**	**1.0000**	0.3960	0.2598	0.5882
K-means	-0.0060	0.3277	-0.0055	0.2241	0.5495	0.3884
DPC-LMST	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**1.0000**
		jain			*R15*	
DPC	0.5470	0.6229	0.8385	**0.9938**	**0.9928**	**0.9933**
FkNN-DPC	0.0485	0.5920	0.2224	0.9858	0.9868	0.9897
DPC-DLP	0.3325	0.5538	0.3721	0.6534	0.6893	0.8325
SNN-DPC	0.0363	-0.0646	0.7083	0.9896	0.9828	0.9839
K-means	0.3295	0.7003	0.3437	0.9068	0.9141	0.9552
DPC-LMST	**1.0000**	**1.0000**	**1.0000**	0.9162	0.9240	0.9517
		db2			*Aggregation*	
DPC	0.4796	0.2394	0.5527	0.8596	0.7548	0.8072
FkNN-DPC	0.2032	0.5233	0.4530	**0.9855**	**0.9886**	**0.9775**
DPC-DLP	0.2406	0.5217	0.4012	0.6746	0.7427	0.8235
SNN-DPC	0.4925	0.3018	0.5659	0.8817	0.8252	0.8634
K-means	0.2188	0.4996	0.4022	0.7746	0.8258	0.8449
DPC-LMST	**1.0000**	**1.0000**	**1.0000**	0.8824	0.9138	0.8794

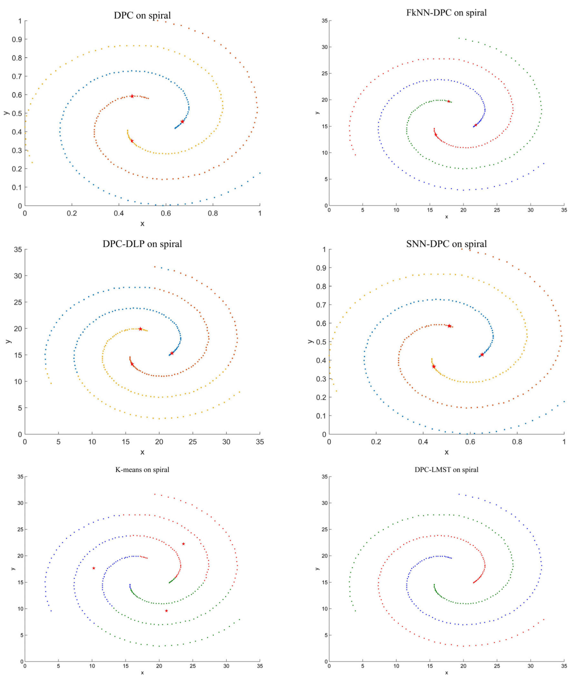


FIGURE 8. Clustering results on *spiral* by 6 algorithms.

for the representative data in Fig.6 by using DPC-LMST. We can see that 3, 4, 8, 11, 15, and 9 initial centers are chosen for the 6 synthetic datasets, respectively.

Table 1 shows the AMI, ARI and FMI scores on all synthetic datasets shown in Fig.5, and where the best scores are highlighted as bold. From Table 1, we can see that the proposed algorithm performs quite well on most synthetic datasets. DPC-LMST performs best on 4 datasets and well on $R15$ and *aggregation* datasets, and it is the only method performed quite well on most datasets. Except spiral dataset, DPC only perform best on $R15$. The improved versions of DPC, including FkNN-DPC, DPC-DLP, and SNN-DPC, obtain low quality on *jain*, *db2*, and *mix* datasets. Fig.8-13 intuitively show the comparative results of the 6 approaches

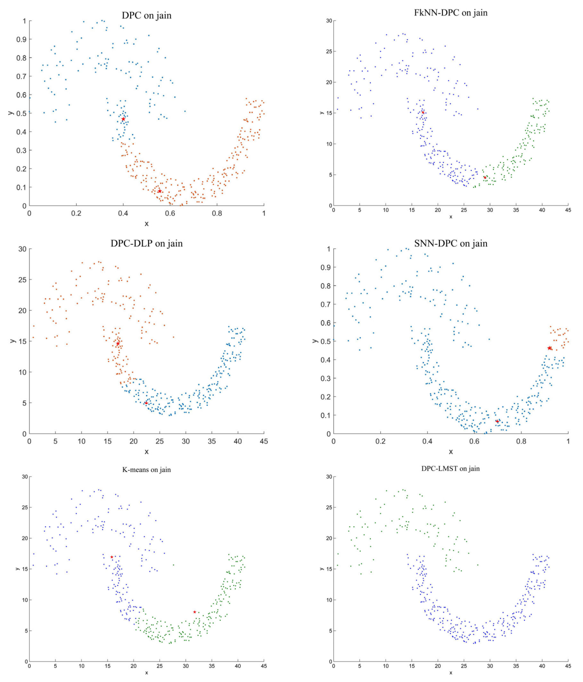


FIGURE 9. Clustering results on *jain* by 6 algorithms.

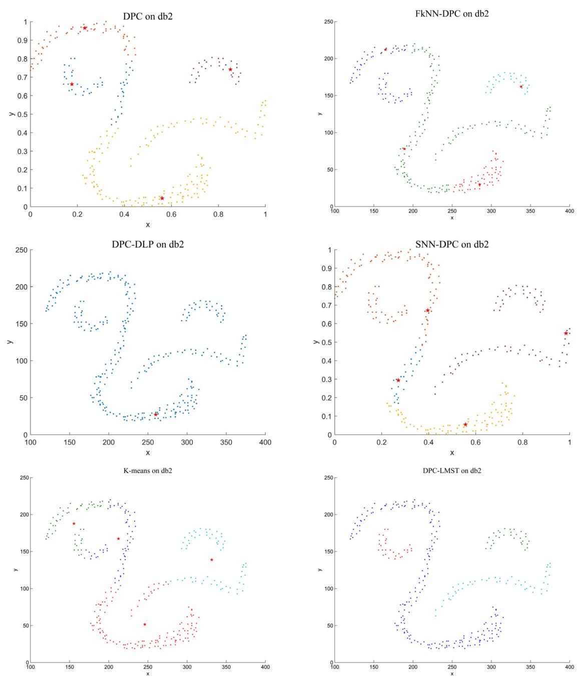


FIGURE 10. Clustering results on *db*2 by 6 algorithms.

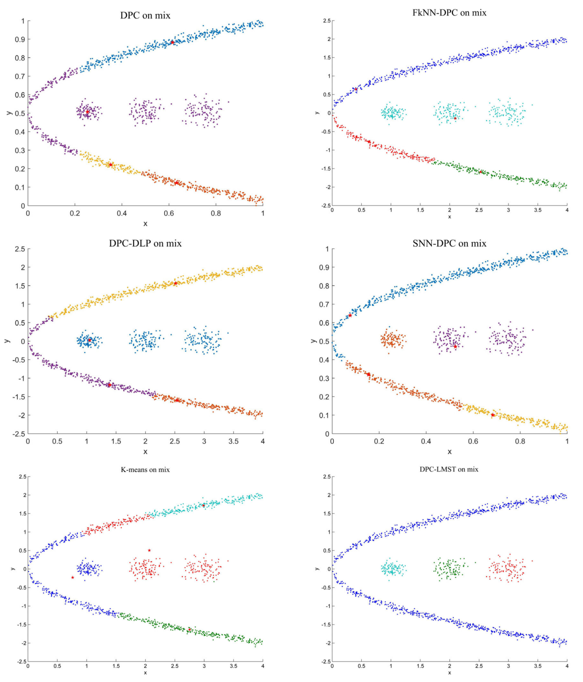


FIGURE 11. Clustering results on *mix* by 6 algorithms.

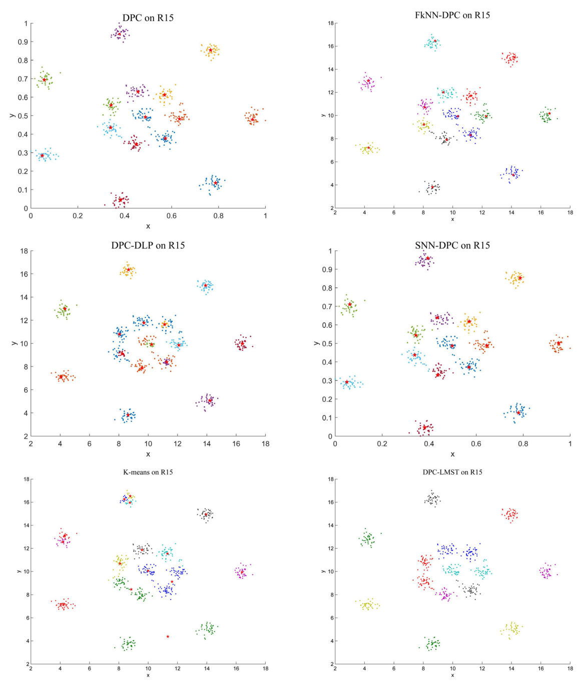


FIGURE 12. Clustering results on *R*15 by 6 algorithms.

on synthetic datasets, and the centers are marked with red pentagrams. Note that there will be no more centers in final results of DPC-LMST, because the proposed algorithm merges adjacent and similar sub-clusters not initial centers.

To demonstrate the robustness of our method, Fig.14 shows the results of DPC-LMST on 4 different datasets over a wide range of ϵ. We observe that the metric values of DPC-LMST remain high and similar within a relatively wide range of ϵ. For example, three metric values of our method are 1 for $5 \leq \epsilon \leq 16$ on jain and for $5 \leq \epsilon \leq 28$ on *mix*, respectively. Similarly, DPC-LMST remain high quality for $5 \leq \epsilon \leq 25$ on both *R*15 and *aggregation*. Although we can see downward trends when ϵ reaches relatively large values, these ranges of ϵ are large relative to the size of the datasets. Therefore, DPC-LMST is insensitive to parameter ϵ.

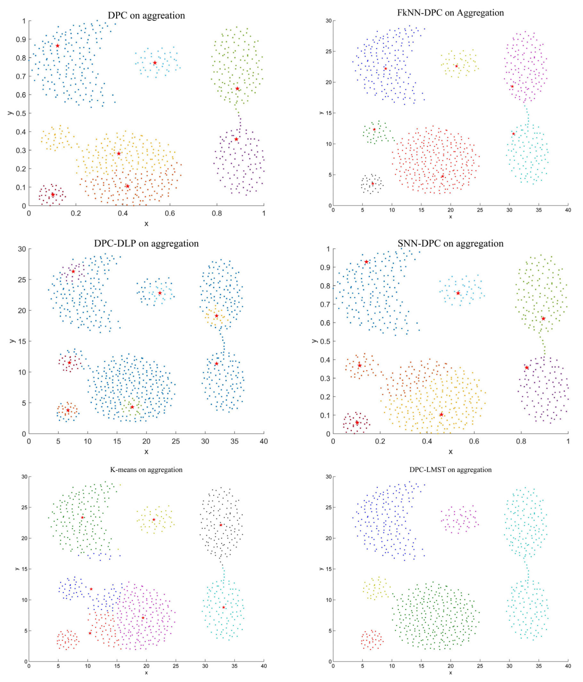


FIGURE 13. Clustering results on *aggregation* by 6 algorithms.

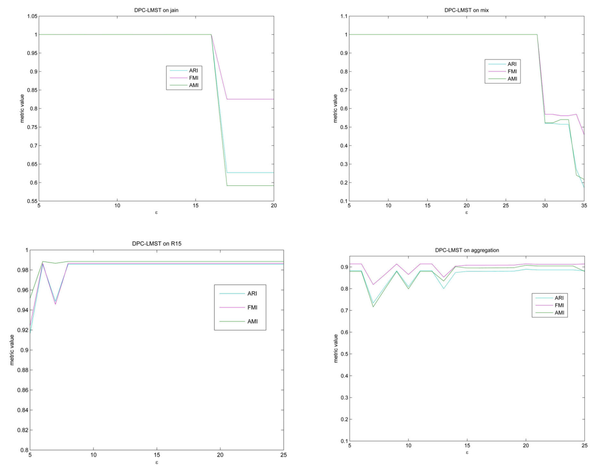


FIGURE 14. Results on 4 synthetic datasets for a wide range of ϵ by DPC-LMST.

TABLE 2. Real-world datasets.

Datasets	Size	Attributes	clusters
Iris	150	4	3
Wine	178	13	3
Seeds	210	7	3
Ecoli	336	8	8
Ionoshere	351	34	2
Libras	360	90	15
segmentation	2310	19	7
waveform	5000	21	3

B. REAL-WORLD DATASETS

We conduct extensive real-world experiments to demonstrate the performance of DPC-LMST. These datasets vary from scale, attribute and task, as shown in Table 2. Table 3 shows the result of the six algorithms, and DPC-LMST outperforms the 5 competing methods on most test cases (the best scores are highlighted in bold). SNN-DPC has good performance on *ionospere*, *libras*, and *segmentation*, however, it worth noting that all the competing algorithms are performed with use of the correct numbers of centers or clusters of datasets. Clustering quality of the competing methods will get worse once incorrect values of centers or clusters are provided by their decision graphs. On the contrary, DPC-LMST chooses centers according to plot of γ, but it has good fault tolerant to choice of centers. Moreover, our algorithm is sensitive to a wide range of ϵ.

We also evaluate the running time of 5 density peak clustering algorithms on the real-world datasets, as is shown in Table 4. We perform experiments on the same computer with MATLAB 2017a. For each approach, the number of centers is preset or chosen automatically. The results show that the running time of DPC-LMST nearly equal to even worse than DPC, FkNN-DPC, and DPC-DLP when the amount and dimension of data is relatively small, such as *Iris*, *Wine*, *Seeds*, *Ecoli*, *Ionospere* and *Libras* datasets. However, for the

TABLE 3. Results of different algorithms on real-world datasets.

Alg.	AMI	ARI	FMI	AMI	ARI	FMI
		Iris			*Ionospere*	
DPC	0.8606	0.8857	0.9233	0.1504	0.2357	0.6491
FkNN-DPC	0.8831	0.9038	0.9335	0.1314	0.1321	0.5841
DPC-DLP	0.8052	0.7881	0.8276	0.0052	0.1957	0.5705
SNN-DPC	0.9001	0.9124	0.9322	**0.3624**	**0.3358**	**0.7029**
K-means	0.7331	0.7163	0.8112	0.1294	0.1776	0.6053
DPC-LMST	**0.9125**	**0.9208**	**0.9459**	0.1334	0.3221	0.6217
		Wine			*Libras*	
DPC	0.7065	0.6724	0.7835	0.5358	0.3193	0.3717
FkNN-DPC	0.8038	0.7990	0.8667	0.4754	0.3184	0.3976
DPC-DLP	0.0573	0.5446	0.1330	0.1548	0.4327	0.3321
SNN-DPC	0.8125	0.7892	0.8853	0.5837	**0.6234**	**0.4572**
K-means	0.8473	0.8685	0.9126	0.5232	0.3094	0.3612
DPC-LMST	**0.8624**	**0.8856**	**0.9013**	**0.6002**	0.5847	0.3359
		Seeds			*Segmentation*	
DPC	0.7299	0.7890	0.8589	0.6927	0.6004	0.6730
FkNN-DPC	0.6971	0.7422	0.8276	0.5830	0.4367	0.5581
DPC-DLP	0.1254	0.5846	0.6233	0.4879	0.3651	0.4268
SNN-DPC	0.7435	0.7646	0.8165	**0.7346**	**0.7505**	**0.7753**
K-means	0.6705	0.7049	0.8026	0.6102	0.5049	0.5758
DPC-LMST	**0.8253**	**0.8837**	**0.8660**	0.6675	0.5837	0.6273
		Ecoli			*Waveform*	
DPC	0.4978	0.4465	0.5775	0.3261	0.2698	0.5292
FkNN-DPC	0.4755	0.5535	0.6919	0.0774	0.0086	0.5050
DPC-DLP	0.0560	0.2784	0.2973	0.1802	0.2257	0.4759
SNN-DPC	0.5791	**0.6221**	0.7129	0.3174	0.2984	0.6057
K-means	0.5051	0.4190	0.5542	0.3630	0.2536	0.5037
DPC-LMST	**0.6158**	0.5124	**0.7252**	**0.5024**	**0.6173**	**0.6558**

TABLE 4. Running time of 5 density peak clustering algorithms on real-world datasets (unit: second).

Name	DPC	FkNN-DPC	DPC-DLP	SNN-DPC	DPC-LMST
Iris	0.0073	0.0171	0.0083	0.0425	0.0340
Wine	0.0078	0.0185	0.0084	0.0562	0.0370
Seeds	0.0089	0.0254	0.0137	0.0604	0.0432
Ecoli	0.0152	0.0438	0.0257	0.1833	0.0868
Ionospere	0.0148	0.0525	0.0294	0.1246	0.1134
Libras	0.0324	0.0611	0.0326	0.1912	0.1072
Segmentation	1.2352	1.4762	8.1147	10.3729	0.6895
Waveform	4.4891	5.6215	48.4805	52.6652	3.3800

larger datasets, the proposed method performs better than the competing methods, especially the running times of DPC-DLP and SNN-DPC are more than 10 times that of DPC-LMST on *Segmentation* and *Waveform* datasets. That means our approach also offers efficiency advantage when the size of data is relatively large.

V. CONCLUSION

In this paper, we proposed an improved density peaks clustering based on local minimal spanning tree, named DPC-LMST. First, a new strategy of representative is presented to reduce the size of original data, and local and global characteristics of original data are well maintained. Meanwhile, new definitions of local density ρ, distance δ and hint of γ are redesigned to highlight true centers and obtain pure sub-clusters. Finally, a sub-cluster merging factor (SCMF) which has good fault tolerance is defined to aggregate initial sub-clusters automatically. Experiments both on synthetic and real-world datasets have demonstrated that the proposed algorithm is effective, efficient, and robust. However, the major disadvantage of the proposed algorithm is it would merge clusters with blurred boundaries together (see DPC-LMST on $R15$ and *aggregation* in Fig.12 and 13, respectively). In future work, we will explore better solutions to improve this issue.

REFERENCES

[1] J. Lu, Y.-P. Tan, G. Wang, and G. Yang, "Image-to-set face recognition using locality repulsion projections and sparse reconstruction-based similarity measure," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 6, pp. 1070–1080, Jun. 2013.

[2] J. Lu, G. Wang, W. Deng, and K. Jia, "Reconstruction-based metric learning for unconstrained face verification," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 1, pp. 79–89, Jan. 2015.

[3] J. Lu, V. E. Liong, X. Zhou, and J. Zhou, "Learning compact binary face descriptor for face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 10, pp. 2041–2056, Oct. 2015.

[4] H. Zhang, T. W. S. Chow, and Q. M. J. Wu, "Organizing books and authors by multilayer SOM," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 12, pp. 2537–2550, Dec. 2016.

[5] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, May 2005.

[6] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surveys*, vol. 31, no. 3, pp. 264–323, Sep. 1999.

[7] P. Berkhin, *Grouping Multidimensional Data: Recent Advances in Clustering*. New York, NY, USA: Springer-Verlag, 2005, pp. 127–160.

[8] J. MacQueen, in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, vol. 1, L. M. Le Cam, J. Neyman, Eds. Berkeley, CA, USA: Univ. California Press, 1967, pp. 281–297.

[9] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 8, pp. 790–799, Aug. 1995.

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

[11] 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.

[12] 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.

[13] B. Shi, L. Han, and H. Yan, "Adaptive clustering algorithm based on kNN and density," *Pattern Recognit. Lett.*, vol. 104, pp. 37–44, Mar. 2018.

[14] Z. Li and Y. Tang, "Comparative density peaks clustering," *Expert Syst. Appl.*, vol. 95, Apr. 2018, pp. 236–247.

[15] 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.

[16] 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.

[17] Y.-W. Chen, D.-H. Lai, H. Qi, J.-L. Wang, and J.-X. Du, "A new method to estimate ages of facial image for large database," *Multimed. Tools Appl.*, vol. 75, no. 5, pp. 2877–2895, Feb. 2016.

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

[19] Y. Zhang, Y. Xia, Y. Liu, and W. Wang, "Clustering sentences with density peaks for multi-document summarization," in *Proc. HLT-NAACL*, May/Jun. 2015, pp. 1262–1267.

[20] J. Xu, G. Wang, and W. Deng, "DenPEHC: Density peak based efficient hierarchical clustering," *Inf. Sci.*, vol. 373, pp. 200–218, Dec. 2016.

[21] G. Wang, Y. Wei, and P. Tse, "Clustering by defining and merging candidates of cluster centers via independence and affinity," *Neurocomputing*, vol. 315, pp. 486–495, Nov. 2018.

[22] 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.

[23] J. MacQueen, in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, vol. 1, L. M. Le Cam, J. Neyman, Eds. Univ. Berkeley, CA, USA: Univ. California Press, 1967, pp. 281–297.

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

[25] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mach. Learn. Res.*, vol. 11, pp. 2837–2854, Oct. 2010.

[26] E. B. Fowlkes and C. L. Mallows, "A method for comparing two hierarchical clusterings," *J. Amer. Stat. Assoc.*, vol. 78, no. 383, pp. 553–569, 1983.

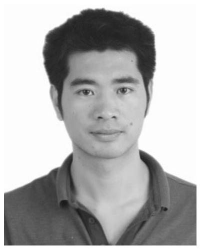
[27] J. Hou and H. Cui, "Experimental evaluation of a density kernel in clustering," in *Proc. 7th Int. Conf. Intell. Control Inf. Process. (ICICIP)*, Dec. 2017, pp. 55–59.

[28] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-means clustering algorithm," *J. Roy. Stat. Soc. C, (Appl. Statist.)*, vol. 28, no. 1, pp. 100–108, 1979.

[29] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms*, Jan. 2007, pp. 1027–1035.

[30] M. E. Celebi, H. A. Kingravi, and P. A. Vela, "A comparative study of efficient initialization methods for the k-means clustering algorithm," *Expert Syst. Appl.*, vol. 40, no. 1, pp. 200–210, Jan. 2013.

[31] Z. Liu, Q. Zheng, Z. Ji, and W. Zhao, "Sparse self-represented network map: A fast representative-based clustering method for large dataset and data stream," *Eng. Appl. Artif. Intell.*, vol. 68, pp. 121–130, Feb. 2013.



RENMIN WANG is currently pursuing the Ph.D. degree with Chongqing University. He devotes himself to the research of machine learning and data mining.

QINGSHENG ZHU received the B.S., M.S., and Ph.D. degrees in computer science from Chongqing University, in 1983, 1986, and 1990, respectively. He is currently a Professor with the College of Computer Science, Chongqing University, and also the Director of the Chongqing Key Laboratory of Software Theory and Technology. His main research interests include Ecommerce, data mining, and service oriented computing.