Arabian Journal for Science and Engineering (2018) 43:7177–7187
https://doi.org/10.1007/s13369-017-3060-7

RESEARCH ARTICLE - COMPUTER ENGINEERING AND COMPUTER SCIENCE

Efficient and Intelligent Density and Delta-Distance Clustering Algorithm

Xuejuan Liu[1] · Jiabin Yuan[1] · Hanchi Zhao[2]

Received: 14 June 2017 / Accepted: 21 December 2017 / Published online: 8 January 2018

Abstract
Density and delta-distance clustering (DDC) is an ideal clustering method that computes the density and delta distance of data. When data derived from the two indicators are large, these areas can be defined as cluster centers. DDC has good clustering performance compared with some other clustering algorithms. However, DDC has a high time complexity and requires manual identification of cluster centers. To fill these gaps, an efficient and intelligent DDC (EIDDC) algorithm is proposed in this study. EIDDC begins from using a sampling method based on locality-sensitive hashing (LSH) to obtain a small-scale dataset. The density and delta distance of each data point are calculated from this dataset to reduce time complexity. Cluster centers are intelligently recognized by utilizing density-based spatial clustering of applications with noise-based outlier detection technology. Experiment results show that LSH can obtain good representatives of the original dataset and that the proposed outlier detection method can recognize the cluster centers of a given dataset. The results also reveal the efficiency of EIDDC.

Keywords LSH · Outlier detection · Density · Delta distance · Clustering

1 Introduction

Clustering is a method of unsupervised learning that partitions a dataset into clusters, such that intra-cluster similarity is maximized and inter-cluster similarity is minimized [1]. Its applications range from image processing to pattern recognition and social networks [2–4]. Many clustering algorithms have been developed, such as K-means [5], fuzzy C-means [6], and self-organizing map [7]. In these algorithms, the number of clusters is required to be preset and the local optimal solution is often obtained. Moreover, cluster results rely on data distribution [8]. Rodriguez and Laio presented a new clustering method based on the metrics of density and delta-distance clustering (DDC), which can identify the number of clusters and shows good performance for data with arbitrary shape [9].

DDC has been adopted in numerous applications because of its good performance. This method has been enhanced for hyperspectral band selection. First, the ranking score of each band was computed by weighing the normalized density and delta distance and an exponential-based learning rule was then employed to adjust the cutoff threshold for a different number of selected bands [10]. A density-ordered tree (DOT) was constructed using DDC to represent the original data in hypernetwork, and community detection was converted to a DOT partition problem [11]. The age of facial image was estimated using DDC in [12]. The estimation process mainly includes the following two steps: (1) estimating the density peaks for each age group and (2) obtaining the possible estimated ages according to the distances of the facial image to the peaks. In [13], DDC was used in calculating the density and delta distance of each user to find social circles in social networks. Other applications include anomalous cell detection [14], fault diagnosis in cloud computing [15], and image processing [16].

DDC begins with computing the density and delta distance of data, and the data with anomalously large indicators are defined as cluster centers. The performance of EIDDC, as an ideal clustering method, should still be improved. Wang [17] introduced the concept of entropy to determine the cutoff distance value d_c, which is a parameter used in calculating

✉ Xuejuan Liu
liu_juanjuan80@126.com

[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China

[2] Pujiang Institute, Nanjing Tech University, Nanjing 210016, China

density. Without a quantitative criterion to determine whether a data point is a cluster center in DDC, Zhong [18] proposed that the product of density and delta distance could be used as quantitative criteria to recognize cluster centers, whereas other parameters must be preset. Other improvements in DDC include using kernel density estimation to obtain the density [13]. Nevertheless, work has seldom been performed to resolve the high computation complexity and the manual recognition of cluster center problem in DDC.

An efficient and intelligent DDC (EIDDC) is developed in this study. The novelty of EIDDC is due to introducing a sampling method based on locality-sensitive hashing (LSH) function to improve clustering speed, and embedding a density-based spatial clustering of applications with noise (DBSCAN)-based outlier detection approach to determine cluster centers intelligently.

The rest of this paper is organized as follows: Sect. 2 briefly introduces the original DDC algorithm. Section 3 presents the EIDDC algorithm. Details and results of the experiment on synthetic and UCI dataset are presented in Sect. 4. Finally, conclusions are presented in Sect. 5.

2 Original DDC Approach

For a given dataset $X = \{x_1, x_2, \ldots, x_N\}$, N is the size of dataset X, and clustering divides X into k clusters $C = \{c_1, c_2, \ldots, c_k\}$. For each cluster C_i,$C_i \subseteq X$ and $C_1 \cup C_2 \ldots C_k = X$.

The clustering process of DDC consists of two steps. The first step is to compute the density and delta distance of each data point. The data with two large parameters are then defined as cluster centers. As used in [13], the Gaussian kernel function is adopted in this study in computing density to avoid large statistical errors. Density ρ_i is defined as follows:

$$\rho_i = \sum_j \exp\left(-\frac{\| d_{i,j} - d_c \|^2}{2\sigma^2}\right), \tag{1}$$

where σ is a smoothing parameter, $\| d_{i,j} - d_c \|$ represents the distance between $d_{i,j}$ and d_c, $d_{i,j}$ represents the Euclidean distance between data i and j, and d_c denotes cutoff distance. Delta distance δ_i is defined as the minimum distance from x_i to points of higher density, and δ_i is calculated as

$$\delta_i = \begin{cases} \max_j(d_{ij}) & \text{if datum i has the biggest density} \\ \min_{j:\rho_j > \rho_i}(d_{ij}) & \text{otherwise.} \end{cases} \tag{2}$$

In the first step, the time complexity of computing these two parameters is $O(N^2)$. The second step is assigning each data point to the same clusters as those of its nearest neighbor in the cluster centers, and the corresponding complexity is $O(N)$.

The time complexity in DDC is $O(N^2)$, and such clustering speed is considered slow in the current big data era. Moreover, manual recognition of cluster centers may be difficult. Some improvements are proposed in this study to solve these issues.

3 EIDDC Approach

In this study, the EIDDC approach divides clustering into three steps. The first step is using LSH to sample dataset X to obtain a small Y. The second step is calculating the density and delta distance of each data point in Y. The cluster centers are then recognized by utilizing outlier detection technology. The last step is assigning a label for each data point of dataset X.

3.1 LSH Sampling

On the basis of the principle of DDC, the key process is to obtain the cluster centers, which are the data points with relatively large density and are far away from the others in the dataset. However, the time complexity $O(N^2)$ in DDC is high. If the size of dataset X is reduced, and data that are more likely to be cluster centers are kept, clustering may perform poorly or be the same as the original approach, but with high clustering speed.

LSH is a sensitive distance hash function [19] that can map data close to each other into a same bucket and those distant from each other into a different one. If LSH is used as a sampling method for a dataset, then densely distributed data can be mapped into the same bucket and sparsely into a different one. During sampling, a large amount of data can be extracted from a bucket with many data, and only few data can be extracted from a bucket with small amount of data. Thus, data located in or around the center of a densely distributed data region can be extracted as the sample dataset. These data are preferred cluster centers in DDC algorithm. Therefore, determining the better cluster centers among those from the small-scale dataset sampled by a LSH-based sampling method will exert little or no impact.

Several distance-sensitive functions can be used in LSH [20,24,25]. Due to being the most best-known distance measure and useful in many cases, the Euclidean distance is usually used as a similarity metric in many clustering algorithms. In the present study, the Euclidean distance is also selected as the hash function [20] in the sampling process. The function is defined as follows:

$$h_{a,b}(v) = \left\lfloor \frac{a \cdot v + b}{w} \right\rfloor, \tag{3}$$

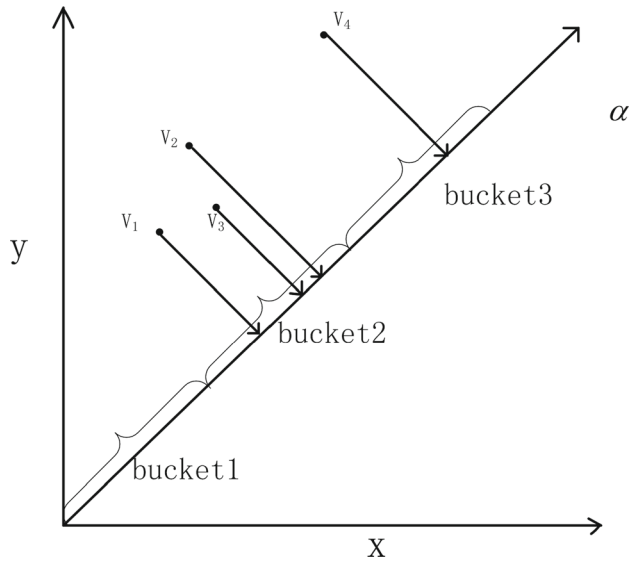

Fig. 1 The principle of LSH

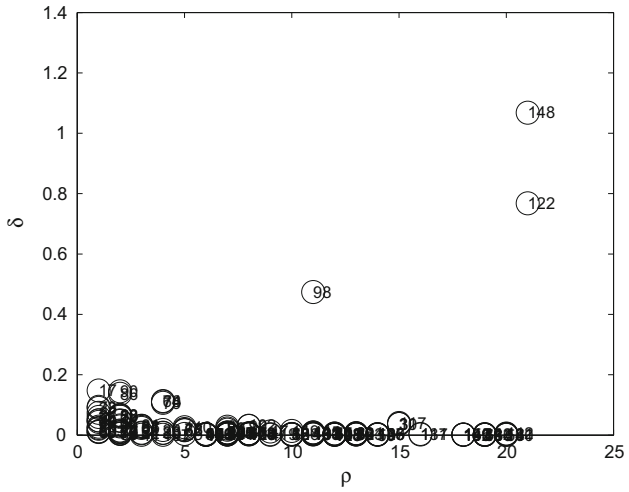

Fig. 2 An example of cluster centers

where v is the data to be mapped, a is a random vector of the same dimension to v, w is the bandwidth of the bucket, b is a random variable between $[0, w]$, and $a \cdot v$ is an operation of v projecting to a. The result $h_{a,b}(v)$ is an integer and the label of the bucket. Figure 1 shows the mapping of data into buckets using the Euclidean distance-based hash function, and the data have two dimensions x and y. In Fig. 1, v_1, v_2, and v_3 are mapped into the same bucket (bucket 2) because they are relatively close to each other, whereas v_4 is mapped into another bucket (bucket 3) because of its distance from the others.

The sampling process for dataset X using the Euclidean distance-based hash function is described in Algorithm 1. a, b, and w are the parameters of the Euclidean distance-based hash function, and r is the sampling ratio. The first step is to define the initial sampled dataset Y. The second is to map each data point into its corresponding bucket. Third, data points are extracted at the ratio r from each bucket. At last, sampled dataset Y is returned, that is $|Y| = r|X|$.

Algorithm 1 LSH sampling algorithm

$Input : X, a, b, w, r$
$Output : Y$
$Procedure :$
1: $Define\ Y = []$
2: **for** $each\ x_i \in X$ **do**
3: $\quad h_{a,b}(v) = \lfloor \frac{a \cdot v + b}{w} \rfloor$
4: **end for**
5: **for** $each\ h_{a,b}(x_i)$ **do**
6: $\quad select\ r * |h_{a,b}(x_i)|\ number\ of\ data\ to\ obtain\ Y$
7: **end for**
8: $return\ Y$

3.2 Intelligent Recognition of Clusters Centers

The manual recognition of cluster centers is difficult and troublesome when the number of cluster centers is considerably large. Thus, an outlier detection technology is proposed to identify the cluster centers intelligently. Only the density and delta distance of the sampled dataset Y should be computed. In this study, cluster centers are also recognized from this small-scale dataset.

Given that their density ρ and delta distance δ are relatively large, cluster centers are distant from most of other data in a 2D coordinate map, which consists of two parameters. Figure 2 shows an example of the DDC approach, in which the 98th, 122nd, and 148th data points are the cluster centers of a given dataset. These cluster centers are distant from the other non-cluster centers. Figure 2 also illustrates that these non-cluster centers are connected as a respective cluster. In this study, a new dataset Z, which comprises density and delta distance, can be determined. The cluster centers are the outliers of this dataset because they are distant from most other data points. In this manner, the cluster centers, which are also the outliers in dataset Z, can be determined using the outlier detection technology.

Several commonly used outlier detection methods include those that are based on statistical distribution, density, or clustering. In dataset Z, non-outliers connect together to form a density-connected region; however, outliers are far from the density-connected region. In comparison with other methods, DBSCAN has better performance for clustering density-connected regions [21], and it is developed in this study to find the outliers of dataset Z; these outliers are also the cluster centers of dataset Y.

The process of intelligently recognizing cluster centers using DBSCAN is described in Algorithm 2. For each data point z_i in dataset Z, if z_i is core data, then cluster C is gen-

erated and expanded by finding all the data that are densely connected to it. After the expansion, cluster C is removed from dataset Z. Otherwise, z_i, being not core data, can be joined in dataset O. Finally, O is defined as the dataset of all the outliers in Z; these outliers are also the cluster centers of dataset Y.

Algorithm 2 Intelligent recognition of cluster centers

Input : Z
Output : O
Procedure :
1: *Define* $O = []$
2: **for** *each* $z_i \subseteq Z$ **do**
3: **if** z_i *is a core data* **then**
4: *create a cluster* C *included* z_i *and its nearest neighbors*;
5: *find* z_i *which is density connected to* C *to expand* C;
6: *remove* C *from* Z;
7: **else**
8: z_i *is an outlier*
9: $O = O \cup z_i$
10: **end if**
11: **end for**
12: *return* O

3.3 EIDDC Approach

To increase the clustering speed and obtain data with high probability to be cluster centers, a LSH-based sampling method is used to extract small dataset Y from X in EIDDC. After computing the density ρ and delta distance δ of each data point in dataset Y, a DBSCAN-based outlier detection method is utilized to recognize cluster centers $O = \{o_1, o_2, \ldots, o_k\}$. Finally, each data point in dataset X is labeled according to its distance to each cluster center in dataset O.

In EIDDC, the size of the sampled dataset Y is assumed as n, and $n < N$. The time complexity of the LSH sampling process is $O(N)$. The required time for the computation of the density and delta distance of each data point in dataset Y is $O(n^2)$. The time complexity of recognizing cluster centers using the DBSCAN-based method is $O(n*log(n))$ [21], and the final step is $O(N)$. Thus, the time complexity of EIDDC is $O(n^2)$. Such clustering speed will be faster than that of DDC, whose time complexity is $O(N^2)$.

4 Experimental Evaluation

Experiments were conducted to assess the efficiency and intelligence of the proposed method. The datasets used in the experiments are shown in Table 1, among which datasets test and test1 are synthetic datasets, whereas the other five are UCI datasets. All the UCI datasets are normalized before the experiments.

Table 1 Dataset used in experiments

Dataset	Size	Dimension	Clusters
Test	2580	2	3
Test1	150	2	3
Wine	178	13	3
Seeds	210	7	3
Banknote	1372	4	2
Spambase	4601	58	2
Imageseg	2100	19	7

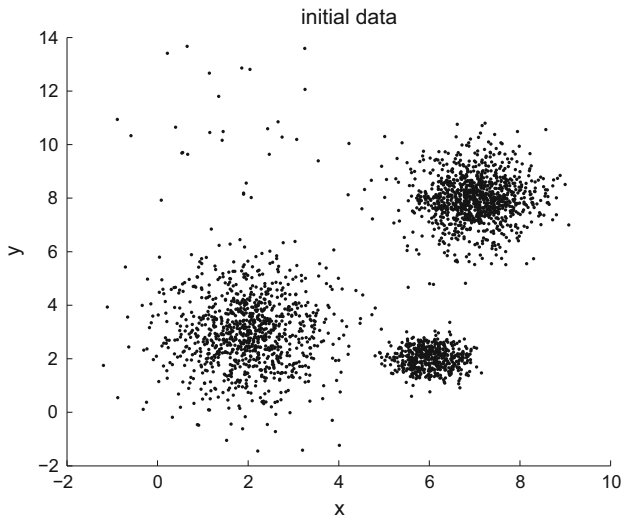

Fig. 3 The distribution of the dataset test

The experiments aim to answer the following questions:

1. Is the dataset extracted by LSH method a good representation?
2. Can the outlier detection technology, which is based on DBSCAN, identify clustering centers intelligently?
3. In comparison with DDC, can EIDDC cluster efficiently?
4. In comparison with the other clustering methods, how well does EIDDC perform?

4.1 LSH Sampling Experiment

The following experiments were validated via the dataset test. Different parameters and sampling ratio were introduced to verify the effects of LSH sampling method. Figure 3 shows the distribution of the initial dataset test, and the distributions of sampled dataset are exhibited in Fig. 4. The left and right column show the distribution of sampled dataset obtained by LSH sampling method when ratio $r = 0.2$ and $r = 0.1$,

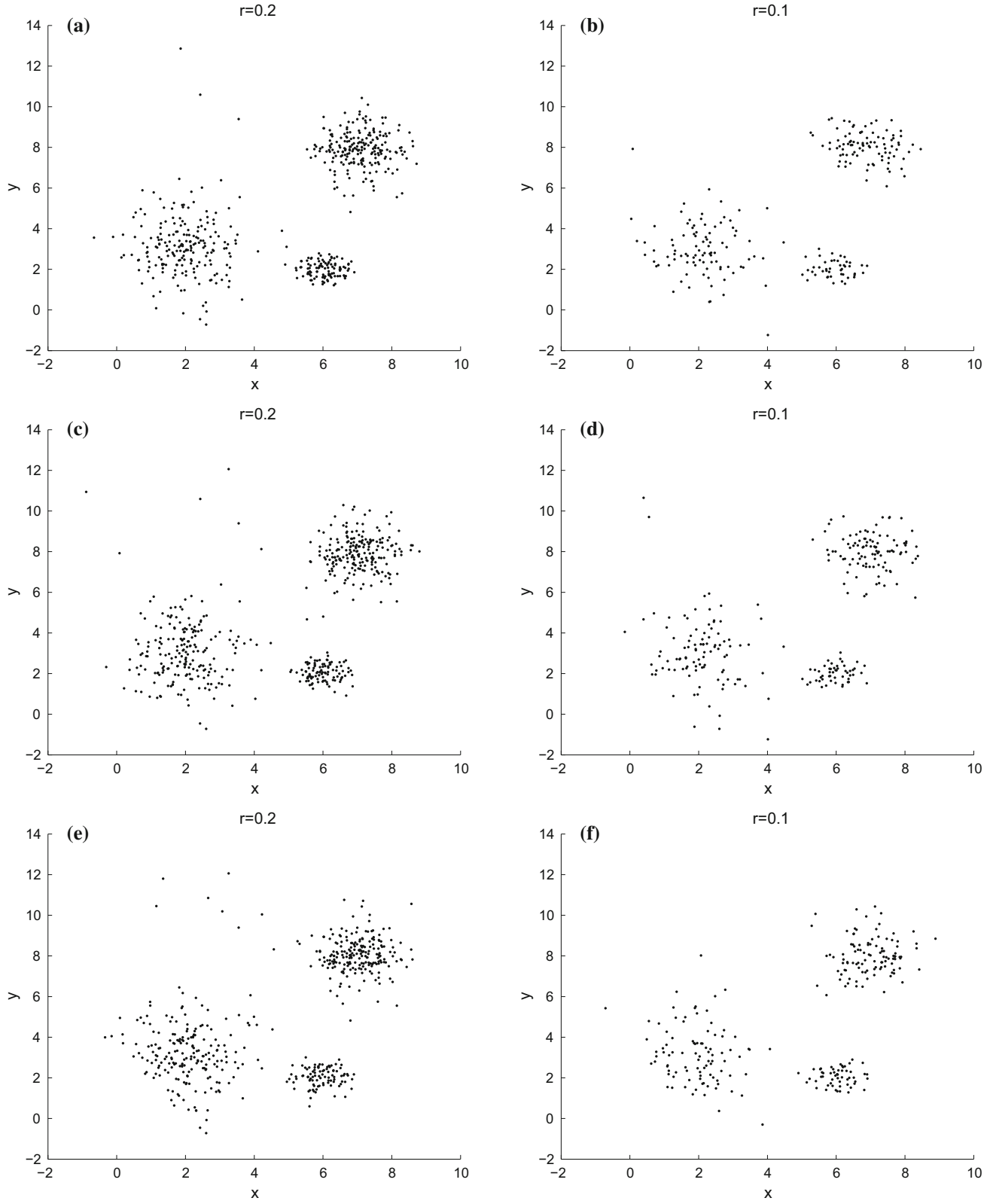

Fig. 4 The results of LSH sampling

respectively. The difference among these three rows lies in the setting of different sampling parameters.

First, the parameters of LSH sampling are as follows: $a = [6, 6]$, $w = 0.3$, and $b = 0.2$. The results are shown in Fig. 4. Figure 4a, b represents the subdata when the sampling ratios are $r = 0.2$ and $r = 0.1$, respectively. On the basis of the description of the DDC approach, a cluster center should be located in or around the center of densely distributed data region. From Fig. 4a, b, the points that have higher probability to be cluster centers can be extracted exactly whether $r = 0.2$ or $r = 0.1$. Thus, the reduction in the dataset size using the proposed method will not affect the process of finding the cluster centers. Then, different parameters of LSH are also set as $a = [4, 3]$, $w = 0.5$, $b = 0.2$ and $a = [0.5, 0.3]$, $w = 3$, $b = 2$, respectively. The results are shown in Fig. 4c–f. In these two cases, good cluster centers can also be obtained whether $r = 0.2$ or $r = 0.1$, and the sampled subdata are a good representation of the original dataset.

Hence, regardless of the parameters and sampling ratio, data that have high probability to be cluster centers can remain in the subdata using LSH sampling method.

4.2 Intelligent Recognition of Cluster Centers

In dataset test1 and some UCI datasets, experiments verified whether the DBSCAN-based method (Algorithm 2) could intelligently find the cluster centers, and Fig. 5 shows the results in 2D coordinate maps, where the x-axis stands for density and the y-axis stands for delta distance. The left column demonstrates the result of the original approach, namely DDC in a different instance. The right column provides the result of the intelligent recognition of cluster centers corresponding to that in the left column. In each right subfigure, each digit encircled in red stands for the index in the dataset, and it is recognized as a cluster center. Non-clustered centers are not marked with digits and not encircled in red in the right subfigure. To effectively identify cluster centers using the DBSCAN method, the density ρ and delta distance δ are normalized. This normalization is also embodied in the axes of the right column.

The result of the DDC experiment on test1 is shown in Fig. 5a, in which the 95th, 145th, and 150th data points are the cluster centers of test1. Figure 5b shows the result of Algorithm 2, in which the data encircled in red are the cluster centers. By comparing Fig. 5a, b, Algorithm 2 can accurately identify the cluster centers. If the cutoff distance d_c is changed, then the 106th, 113th, and 133th data points are the cluster centers, as shown in Fig. 5c. In this situation, Algorithm 2 also recognizes these cluster centers, as shown in Fig. 5d. The same contrasts are deployed in the UCI datasets of wine and seeds. The comparison of wine is shown in Fig. 5e, f, and that of seeds is shown in Fig. 5g, h. The comparison of these two datasets reveals that the DBSCAN-based method can intelligently identify the cluster centers.

This study also verifies through experiments whether Algorithm 2 works when the cluster centers are close to each other. Figure 6 shows the third run of experiment on EIDDC on imageseg dataset ($w = 0.4, b = 0.1, r = 0.2$). From Fig. 6a, finding the cluster centers is difficult, and the 184th and 20th data points are extremely close neighbors. After utilizing Algorithm 2, the data encircled in red are recognized as the cluster centers (Fig. 6b). The performance of EIDDC is relatively good, as shown in Table 4. That is, if cluster centers are close neighbors, Algorithm 2 can also recognize them. However, EIDDC may not be suitable when the cluster centers exactly form a cluster in the 2D coordinate map, which consists of density and delta distance. In this situation, two conditions must be satisfied: The number of clusters is particularly large and the cluster centers should connect densely together; however, these conditions are rarely encountered in normal circumstances.

Overall, the results of experiments show the DBSCAN-based outlier detection method can intelligently recognize the cluster centers.

4.3 Experiment on the Efficiency of EIDDC Approach

The following experiments utilized DDC and EIDDC algorithm on the UCI dataset to demonstrate the efficiency of EIDDC in terms of cluster accuracy (CA) and cluster time (CT).

For a given dataset $X = \{x_1, x_2, \ldots, x_N\}$, the actual label of x_i is $L(x_i)$ and the resulting label in the experiments is $L'(x_i)$. CA compares the resulting label of every data point with its actual label and is defined as follows:

$$\mathrm{CA} = \frac{1}{N}\sum_{i=1}^{N} f(L'(x_i) - L(x_i)), \tag{4}$$

where

$$\begin{cases} f(x) = 1 & \text{if } x = 0 \\ f(x) = 0 & \text{otherwise.} \end{cases} \tag{5}$$

CT is the time for clustering. Two sampling ratios were selected as $r = 0.2$ and $r = 0.1$ in EIDDC. For every dataset, the DDC approach is performed, followed by the EIDDC approach for five times for every subdataset to avoid randomness from the process of sampling. The parameters of LSH in EIDDC were reset as $w = 0.3$ and $b = 0.12$, and vector a was generated by a random function. The results of CA and CT are shown in Tables 2 and 3, and the maximum of CA values and minimum of CT are rendered in bold in the corresponding table.

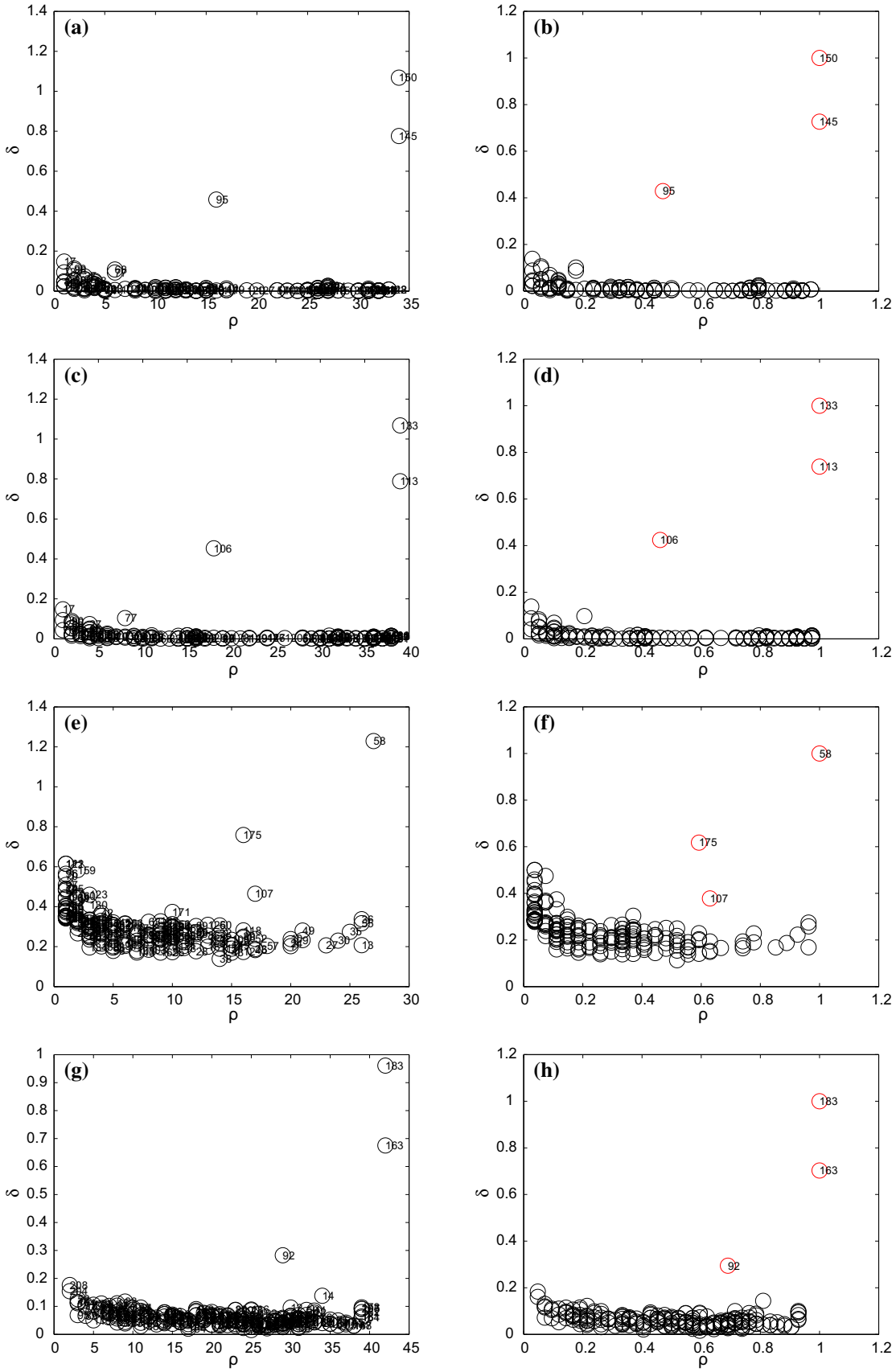

Fig. 5 The results of intelligent recognition of cluster centers

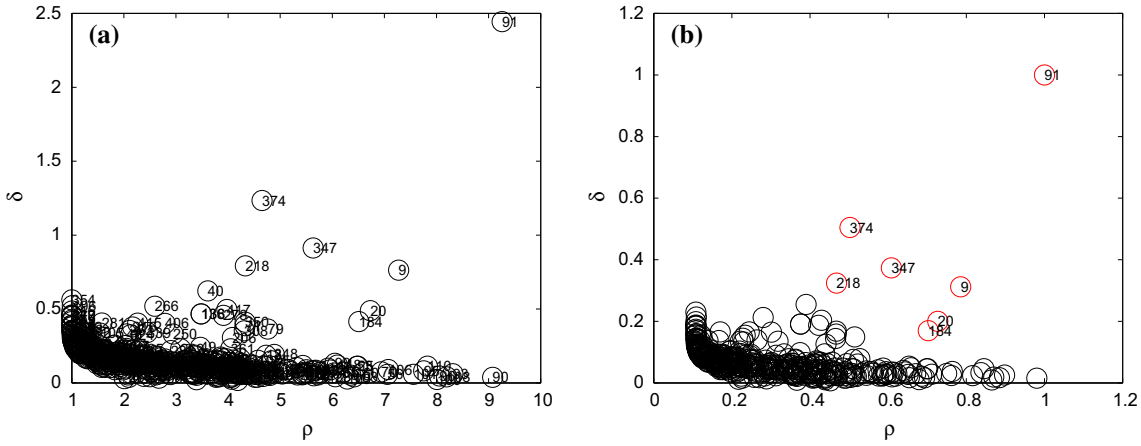

Fig. 6 Intelligent recognition of cluster centers near each other

Table 2 The comparison of CAs between DDC and EIDDC (%): $w = 0.3, b = 0.12$

Dataset	DDC		EIDDC					
			1	2	3	4	5	Average
Seeds	**86.7**	$r = 0.2$	90.0	66.7	66.7	85.7	85.2	77.2
		$r = 0.1$	88.6	33.3	66.7	66.7	66.7	64.4
Wine	66.7	$r = 0.2$	46.6	53.9	86.0	89.9	70.8	**69.4**
		$r = 0.1$	33.3	66.7	91.0	54.5	33.3	55.8
Banknote	**88.9**	$r = 0.2$	74.1	76.2	87.5	83.9	50.0	74.3
		$r = 0.1$	59.1	82.8	81.9	70.0	61.5	71.1
Spambase	**57.9**	$r = 0.2$	33.3	34.7	45.2	40.2	60.6	42.8
		$r = 0.1$	57.9	56.8	36.4	50.0	50.0	50.0
Imageseg	**67.8**	$r = 0.2$	70.4	67.4	66.4	55.2	67.6	65.1
		$r = 0.1$	63.1	67.4	67.3	67.1	60.3	65.1

Table 3 The comparison of CTs between DDC and EIDDC (s): $w = 0.3, b = 0.12$

Dataset	DDC		EIDDC					
			1	2	3	4	5	Average
Seeds	5.14	$r = 0.2$	2.08	1.63	1.63	1.67	1.73	**1.75**
		$r = 0.1$	1.12	1.19	1.24	1.16	1.28	**1.2**
Wine	4.53	$r = 0.2$	1.5	1.48	1.46	1.54	1.63	**1.52**
		$r = 0.1$	1.36	1.33	1.41	1.41	1.42	**1.39**
Banknote	164.9	$r = 0.2$	11.1	10.5	10.9	10.2	9.86	**10.5**
		$r = 0.1$	4.94	4.56	4.84	4.63	4.42	**4.68**
Spambase	2240	$r = 0.2$	58.8	56.2	49.9	56.3	53.8	**55.0**
		$r = 0.1$	18.4	19.3	18.6	20.1	20.5	**19.4**
Imageseg	403	$r = 0.2$	21.6	20.1	20.6	20.6	20.7	**20.7**
		$r = 0.1$	8.63	8.83	8.25	8.62	8.58	**8.58**

As given in Table 2, most CA values in EIDDC, including the average, are lower than those in DDC because the reduced dataset size may have caused the improved option for cluster centers to be left out. Nevertheless, in other cases, the CAs of EIDDC are higher than those of DDC because the most remarkable option may have appeared, while other data are removed during sampling. Although the overall clustering effect decreases, the magnitude of this decrease is within the acceptable range. As shown in Table 3, each CT in EIDDC is less than that in DDC, and CT is less when the sampling ratio is smaller. Evidently, the CT value is lesser when $r = 0.1$ than that when $r = 0.2$ in dataset. The same trend is observed for other datasets in Table 3. In addition, the size of a dataset is large and this acceleration effect is considerably

Table 4 The comparison of CAs between DDC and EIDDC (%): $w = 0.4$, $b = 0.1$

Dataset	DDC		EIDDC					
			1	2	3	4	5	Average
Seeds	**86.7**	$r = 0.2$	86.7	86.2	50.5	85.2	87.6	79.2
		$r = 0.1$	65.0	55.8	68.1	53.2	66.7	61.8
Wine	66.7	$r = 0.2$	55.1	92.7	91.6	89.3	53.2	**76.4**
		$r = 0.1$	46.1	52.2	88.8	45.3	46.6	55.8
Banknote	**88.9**	$r = 0.2$	83.8	80.8	84.6	66.9	86.4	80.5
		$r = 0.1$	56.8	78.6	50.0	84.4	67.6	67.5
Spambase	57.9	$r = 0.2$	60.6	59.8	60.6	60.6	62.2	**60.7**
		$r = 0.1$	59.8	57.9	57.9	57.7	50.0	56.6
Imageseg	**67.8**	$r = 0.2$	57.0	64.5	64.1	59.4	66.3	62.3
		$r = 0.1$	67.4	64.0	64.4	55.2	63.7	62.9

Table 5 The comparison of CTs between DDC and EIDDC (s): $w = 0.4$, $b = 0.1$

Dataset	DDC		EIDDC					
			1	2	3	4	5	Average
Seeds	5.14	$r = 0.2$	1.82	1.7	1.83	1.66	1.68	**1.74**
		$r = 0.1$	1.26	1.28	1.36	1.33	1.23	**1.29**
Wine	4.53	$r = 0.2$	1.6	1.55	1.51	1.51	1.61	**1.56**
		$r = 0.1$	1.13	1.11	1.15	1.2	1.16	**1.15**
Banknote	164.9	$r = 0.2$	14.6	10.9	10.8	10.7	11.8	**11.8**
		$r = 0.1$	4.51	4.32	4.78	4.6	4.75	**4.59**
Spambase	2240	$r = 0.2$	54.5	57.7	55.5	53.5	52.2	**54.7**
		$r = 0.1$	18.1	19.3	20.1	18.5	19.2	**19.0**
Imageseg	403	$r = 0.2$	21.1	21.3	21.2	0.9	20.9	**21.1**
		$r = 0.1$	8.54	8.79	8.8	8.75	9.07	**8.79**

excellent. Taking dataset images as an example, when $r = 0.2$, the acceleration rate equals $20.7/403 = 0.05$. This rate is only slightly higher than $(r = 0.2)^2 = 0.04$. The ideal acceleration effect is not achieved because the time for LSH sampling and intelligent recognition of clustering centers is also included in CT in the EIDDC method.

In addition, other experiments were conducted to observe whether the same effect can be achieved if the parameters of LSH function were changed as $w = 0.4$, $b = 0.1$, and α was also generated by a random function. The results are shown in Tables 4 and 5. The comparison of CAs in Tables 2 and 4 and CTs in Tables 3 and 5 reveals that EIDDC can achieve the same effect even when the LSH parameters have been changed.

Based on the above analysis, EIDDC can obtain the effect of efficient clustering compared with DDC.

4.4 Comparison of Experiments with Other Clustering Methods

Other experiments were also conducted to compare EIDDC with other clustering methods, such as KSOM [22], K-means [5], and spectral clustering methods.

To eliminate the random effects of the presetting of initial value, KSOM and K-means methods were run five times to obtain the average clustering performance. Spectral clustering method was only run once. In KSOM, the number of iterations was limited to 200, the weight matrix was generated by a random function in [0, 1], and the learning rate decreased with the number of iterations, whose initial value was preset at 0.5. In K-means, the initial cluster centers were randomly selected from the dataset and the iteration times were also preset to 200. For the spectral clustering method, the algorithm in [23] was utilized to cluster all the datasets. Owing to clustering being an unsupervised machine learning method, the number of clusters k is usually unknown and is only preset as the actual value of every dataset in the three clustering methods. Tables 6 and 7 summarize the comparison of CA and CT results between EIDDC and the other clustering methods, respectively. For the convenience of comparison, the maximal CA and minimal CT are also, respectively, shown in bold in Tables 6 and 7. Table 6 shows that CAs of EIDDC are nearly the same values with other methods in datasets banknote, spambase, and imageseg and smaller values in datasets seeds and wine. In Table 7, the CTs of EIDDC are less than those of the others in all the

Table 6 The comparison of CAs between EIDDC and other clustering methods (%)

Algorithm	Seeds	Wine	Banknote	Spambase	Imageseg
KSOM	60.9	52.0	58.6	56.9	40.9
K-means	**88.5**	92.3	59.2	**62.2**	61.3
Spectral	84.8	**94.9**	62.6	57.3	**65.7**
EIDDC					
$r = 0.2$	79.2	76.4	**80.5**	60.7	62.3
$r = 0.1$	61.8	55.8	67.5	56.6	62.9

Table 7 The comparison of CTs between EIDDC and other clustering methods (s)

Algorithm	Seeds	Wine	Banknote	Spambase	Imageseg
KSOM	3.51	2.79	8.85	34.5	25.5
K-means	2.11	2.14	4.47	20.6	9.63
Spectral	4.47	4.0	68.9	5131	458.0
EIDDC					
$r = 0.2$	1.74	1.56	11.8	54.7	21.1
$r = 0.1$	**1.29**	**1.15**	**4.59**	**19.0**	**8.79**

Table 8 The comparison of CTs between EIDDC and other clustering methods (s)

Algorithm	Seeds	Wine	Banknote	Spambase	Imageseg
KSOM	52.5	47.5	45.0	52.2	37.9
K-means	73.4	79.2	57.7	60.2	59.6
Spectral	**80.0**	**84.8**	58.7	53.4	61.0
EIDDC					
$r = 0.2$	79.2	76.4	**80.5**	**60.7**	62.3
$r = 0.1$	61.8	55.8	67.5	56.6	**62.9**

given datasets when the sampling ratio $r = 0.1$ and only larger than that of KSOM and K-means in dataset Banknote and Spambase when $r = 0.2$. Comprehensively, considering the two terms of CA and CT, EIDDC has nearly the same performance with the other clustering methods.

The results shown in Tables 6 and 7 were obtained when the number of clusters k is exactly equal to the real number of clusters in every dataset. However, in the usual case, obtaining the value of k is difficult without any prior knowledge. Nevertheless, the value of k can be intelligently achieved in EIDDC. How will the other clustering methods perform if the k value is incorrectly set? Therefore, additional contrast experiments, in which the other parameters remain unchanged, except for k slightly larger than its real value, were also conducted. KSOM and K-means also ran five times, while the spectral clustering ran only once. The comparison of CAs is shown in Table 8. In this case, the CAs of the other clustering methods decrease due to the incorrect setting of k value. The performance of the other clustering methods is reasonably believed to be worse if the difference between the actual and hypothetical k value is larger. By contrast, confusion from presetting the k value rarely appears in the EIDDC method.

5 Conclusion

In this study, EIDDC algorithm was proposed to solve the high time complexity and manual identification of cluster centers of the original approach. In the proposed method, the size of the original dataset was initially reduced using the LSH method. Density and delta distance for the reduced dataset were then computed, from which the cluster centers were intelligently recognized using DBSCAN-based outlier technology. The time required for the two additional steps in EIDDC is minimal and can be ignored compared with the entire clustering time in DDC. The clustering accuracy of the proposed method is a little lower than that of DDC. However, when considering the magnitude of decrease in clustering time, this limitation can also be insignificant. Overall, EIDDC can achieve more efficient and intelligent clustering effects compared with those of the original approach. Although it is performed more commonly compared with the other methods, EIDDC can be a good option as a clustering method due to its ability to recognize the numbers of clusters and cluster centers automatically.

Acknowledgements I appreciate my co-authors for their valuable help and support in accomplishing the manuscript. This research work was supported by the National Natural Science Foundation of China (No. 61571226).

References

1. Jain, A.K.; Dubes, R.C.: Algorithms for Clustering Data, pp. 45–46. Prentice-Hall, Englewood Cliffs (1988)
2. Gracia, C.D.; Sudha, S.: Adaptive clustering of embedded multiple web objects for efficient group prefetching. Arab. J. Sci. Eng. **42**(2), 715–724 (2017)
3. Tagarelli, A.; Karypis, G.: A segment-based approach to clustering multi-topic documents. Knowl. Inf. Syst. **34**(3), 563–595 (2013)
4. Wang, Q.; Chen, G.: Fuzzy soft subspace clustering method for gene co-expression network analysis. Int. J. Mach. Learn. Cybern. **8**(4), 1157–1165 (2017)
5. Wu, X.; Kumar, V.; Quinlan, J.R.; et al.: Top 10 algorithms in data mining. Knowl. Inf. Syst. **14**(1), 1–37 (2008)
6. Salgado, P.; Garrido: fuzzy clustering of fuzzy systems. In: IEEE International Conference on Systems Man and Cybernetics, pp. 2368–2373 (2004)
7. Masahiro, E.; Masahiro, U.; Takaya, T.: A clustering me tho d using hierarchical self-organizing maps. J. VLSI Signal Process. Syst. Signal Image Video Technol. **32**(1/ 2), 105–118 (2002)
8. Xu, D.; Tian, Y.: A comprehensive survey of clustering algorithms. Ann. Data Sci. **2**(2), 165–193 (2015)

9. Rodriguez, A.; Laio, A.: Clustering by fast search and find of density peaks. Science **344**(6191), 1492–1496 (2014)
10. Jia, S.; Tang, G.; Zhu, J.; et al.: A novel ranking-based clustering approach for hyperspectral band selection. IEEE Trans. Geosci. Remote Sens. **54**(1), 88–102 (2016)
11. Cheng, Q.; Liu, Z.; Huang, J.; et al.: Community detection in hypernetwork via density-ordered tree partition. Appl. Math. Comput. **276**, 384–393 (2016)
12. Chen, Y.W.; Lai, D.H.; Qi, H.; et al.: A new method to estimate ages of facial image for large database. Multimed. Tools Appl. **75**(5), 2877–2895 (2016)
13. Wang, M.; Zuo, W.; Wang, Y.: An improved density peaks-based clustering method for social circle discovery in social networks. Neurocomputing **179**, 219–227 (2016)
14. Dandan, M.; Xiaowei, Q.; Weidong, W.: Anomalous cell detection with kernel density-based local outlier factor. China Commun. **12**(9), 64–75 (2015)
15. Wang, T.; Zhang, W.; Ye, C.; et al.: Fd4c: automatic fault diagnosis framework for web applications in cloud computing. IEEE Trans. Syst. Man Cybern. Syst. **46**(1), 61–75 (2016)
16. Lu, J.; Wang, G.; Deng, W.; et al.: Reconstruction-based metric learning for unconstrained face verification. IEEE Trans. Inf. Forensics Secur. **10**(1), 79–89 (2015)
17. Wang, S.; Wang, D.; Li, C.; et al.: Comment on "Clustering by fast search and find of density peaks". arXiv preprint arXiv:1501.04267 (2015)
18. Zhong, J.; Peter, W.T.; Wei, Y.: An intelligent and improved density and distance-based clustering approach for industrial survey data classification. Expert Syst. Appl. **68**, 21–28 (2017)
19. Gionis, A.; Indyk, P.; Motwani, R.: Similarity search in high dimensions via hashing. VLDB **99**(6), 518–529 (1999)
20. Datar, M.; Immorlica, N.; Indyk, P.; et al.: Locality-sensitive hashing scheme based on p-stable distributions. In: Proceedings of the Twentieth Annual Symposium on Computational Geometry. ACM, pp. 253–262 (2004)
21. Ester, M.; Kriegel, H.P.; Sander, J.; et al.: A density-based algorithm for discovering clusters in large spatial databases with noise. Kdd **96**(34), 226–231 (1996)
22. Yun, X.; Chong-zhao, H.; Huan-hong, W.; et al.: Kernel-based self-organizing map clustering. J. Xi'an J. Univ. **39**(12), 1307–1310 (2005)
23. Ng, A.Y.; Jordan, M.I.; Weiss, Y.: On spectral clustering: analysis and an algorithm. In: Dietterich, T., Becker, S., Ghahramani, Z. (eds.) Advances in Neural Information Processing Systems, pp. 849–856. MIT Press, Cambridge (2002)
24. Indyk, P.; Motwani, R.: Approximate nearest neighbors: towards removing the curse of dimensionality. In: Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing. ACM, pp. 604–613 (1998)
25. Charikar, M.S.: Similarity estimation techniques from rounding algorithms. In: Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing. ACM, pp. 380–388 (2002)