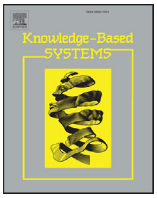
Knowledge-Based Systems 133 (2017) 208–220

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys



Adaptive density peak clustering based on K-nearest neighbors with aggregating strategy

Liu Yaohui[a,b], Ma Zhengming[a,*], Yu Fang[b]

[a] School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, Guangdong, 510006 China
[b] School of Software and Communication Engineering, Xiangnan University, Chenzhou, Hunan, 423000 China

ARTICLE INFO

Article history:
Received 5 September 2016
Revised 30 June 2017
Accepted 8 July 2017
Available online 12 July 2017

Keywords:
Clustering algorithm
Density peaks
K-nearest neighbors
Aggregating

ABSTRACT

Recently a density peaks based clustering algorithm (dubbed as DPC) was proposed to group data by setting up a decision graph and finding out cluster centers from the graph fast. It is simple but efficient since it is noniterative and needs few parameters. However, the improper selection of its parameter cutoff distance d_c will lead to the wrong selection of initial cluster centers, but the DPC cannot correct it in the subsequent assignment process. Furthermore, in some cases, even the proper value of dc was set, initial cluster centers are still difficult to be selected from the decision graph. To overcome these defects, an adaptive clustering algorithm (named as ADPC-KNN) is proposed in this paper. We introduce the idea of K-nearest neighbors to compute the global parameter d_c and the local density ρ_i of each point, apply a new approach to select initial cluster centers automatically, and finally aggregate clusters if they are density reachable. The ADPC-KNN requires only one parameter and the clustering is automatic. Experiments on synthetic and real-world data show that the proposed clustering algorithm can often outperform DBSCAN, DPC, K-Means++, Expectation Maximization (EM) and single-link.

1. Introduction

Clustering is the task to find a set of groups that similar objects are in the same group but different objects are separated into different groups. Since clustering can uncover the inherent, potential and unknown knowledge, principles or rules in the real-world, it has been widely used in many fields, including data mining, pattern recognition, machine learning, information retrieval, image analysis and computer graphics [1,8,13,16,24,32,33]. Several different clustering strategies such as the partitioning, the hierarchical, the density-based, the distribution-based have been proposed [13,21,24,33], but no consensus has been reached even on the definition of a cluster [24].

The K-means clustering algorithm is the popular one of the partitioning methods. It starts with K initial cluster centers and then assigns each object iteratively to the “closest” cluster by optimizing an objective function [8,15,21]. However, assigning each object to its nearest center makes the K-means algorithm fail to detect nonspherical clusters [15]. K-means++ [2] provides a method to select initial cluster centers and improves the accuracy of K-means.

Density-based clustering is a nonparametric approach where the clusters are considered to be high-density areas and separated from each other by contiguous regions with low density of objects [1,7,8,19,24,25,32]. In density-based spatial clustering of applications with noise (DBSCAN) [8], points are classified as core objects or outliers with the density thresholds and the core objects are assigned to a cluster if they are closely packed together. However, choosing an appropriate threshold can be nontrivial [8,19].

Rodriguez and Laio proposed a clustering by fast search and find of density peaks (DPC) algorithm [24], like DBSCAN and the mean-shift [31] method, which is able to detect arbitrary clusters and needn’ t specify the number of clusters as the partitioning algorithms do. The core of the DPC is setting up a decision graph using two quantities of each point i: the local density ρ_i and the distance δ_i from points of higher density. The clustering centers are selected through the decision graph, and then each of the rest points is assigned to the cluster its nearest neighbor of higher density belongs to. Though DPC is simple and effective, it has some drawbacks. First, the decision graph will be set up incorrectly if the parameter cutoff distance d_c is not proper. Second, errors will be propagated in the subsequent assignment process but no way was taken in the DPC to correct it. Third, initial cluster centers are selected manually but not automatically, whereas it is very difficult to get correct selection on some datasets.

[*] Corresponding author.
E-mail addresses: liuyh28@mail2.sysu.edu.cn (L. Yaohui), issmzm@mail.sysu.edu.cn (M. Zhengming), yfjammy@qq.com (Y. Fang).

http://dx.doi.org/10.1016/j.knosys.2017.07.010
0950-7051/© 2017 Elsevier B.V. All rights reserved.

Recently, some successors of DPC were proposed trying to overcome these defects. 3DC [19] clustering selects the-highest-confidence objects recursively, differ from DPC which selects all the possible "anomalous" objects by setting a threshold. However, in the dividing step, points whose densities are lower than the threshold will be looked as outliers and not be processed in succedent procedures, this may omit those clusters with lower density. DPC-KNN [7] introduces the ideas of the k nearest neighbors to compute the local density of a point and uses the principal component analysis (PCA) [27] to reduce the dimensions of datasets; this makes the clustering algorithm get better results. Nevertheless, the clustering process is the same as DPC. So the drawbacks of DPC still exist in it. Fuzzy weighted K-Nearest Neighbors Density Peak Clustering (FKNN-DPC) [32] proposes a new local density metric based on the K-nearest neighbors too, applies two new strategies for objects assignment, in which the fuzzy method is introduced. It is more robust than DPC, but the model becomes more complex and the algorithm is not automatic too.

In this paper, we proposed an adaptive density peak clustering based on K-nearest neighbors with aggregating strategy, dubbed as ADPC-KNN for simplicity. The ADPC-KNN also introduces the idea of K-nearest neighbors to compute the local density, proposes a new approach to select initial cluster centers, and applies an aggregation strategy to merge clusters if they are density reachable. The ADPC-KNN algorithm has the following new features: (1) A new local density metric is proposed based on the K-nearest neighbors. It widens the gap of the density between core objects and border objects (outliers) making the density peaks to be found efficiently and correctly; (2) A new way for initial cluster centers selection is adopted which ensures that all cluster centers can be found correctly even on unbalanced datasets whose distribution of classes present in a data is not uniform. (3) A new idea of cluster density reachable is proposed. Clusters which satisfy the density reachable conditions will be aggregated together.

The proposed algorithm is performed on synthetic and real-world datasets, which are widely used for the performance tests of clustering algorithms. The results of ADPC-KNN are compared with DBSCAN, K–means and DPC in terms of three very popular benchmarks: F-measure (F1) [23], Adjusted Mutual Information (AMI) and Adjusted Rand Index (ARI) [30]. The rest of the paper is organized as follows: Section 2 briefly describes the principle of DPC and does a comparative analysis of local density metrics used in algorithms presented before. Section 3 makes a detailed description of our adaptive clustering algorithm. Section 4 gives our experiment results. Section 5 draws some conclusions.

2. Related works

In this section, we will review DPC briefly and give a short analysis to the local density metrics by taking a synthetic dataset as an example.

2.1. Density peaks clustering

The DPC algorithm bases on the assumption that a cluster center has higher local density than those of its neighbors and a relatively large distance from the other centers. In order to set up a decision graph and then find the ideal cluster centers, DPC computes two quantities of each point i: the local density ρ_i defined by (1) and the distance δ_i from points of higher density defined by (2) [24].

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

$$\delta_i = min_{j:\rho_i<\rho_j} d_{ij} \tag{2}$$

Where d_{ij} is the distance between points i and j, and d_c is a cutoff distance inputted by users. $\chi(t) = 1$ if $t < 0$ and $\chi(t) = 0$ otherwise. For the point with highest density, its delta is taken as $\delta_i = max_j(d_{ij})$. After the density and the delta values of all points are calculated, DPC plots the decision graph, which consists of the collection of points (ρ_i, δ_i). One can find out cluster centers in the upper-right region of the decision graph, which are points with high δ and relatively high ρ. With the cluster centers, DPC assigns the remaining points to the same cluster as its nearest neighbor of higher density in a single step. As a result, the execution of DPC is efficient. Specifically, for "small" datasets (e.g., for the Sonar dataset), it is difficult to make a reliable estimate of the densities. So, DPC adopts another density metric given by (3) to calculate the local densities [24].

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

However, it has no objective metric to decide whether the dataset is small or large and clustering by using the two density metrics will produce very different results. In addition, for small datasets, the clustering results of DPC can be greatly affected by the cutoff distance d_c even using (3) to calculate the local density [32]. To eliminate the influence from the cutoff distance d_c and give an uniform density metric for datasets with any size, DPC-KNN and FKNN-DPC introduce the idea of the K-nearest neighbor into the local density calculation.

The local density proposed by DPC-KNN [7] is:

$$\rho_i = exp\left(-\frac{1}{K}\sum_{j\in KNN_i} d_{ij}^2\right) \tag{4}$$

where K is the input parameter and KNN_i is the set of K-nearest neighbors of point i.

The local density proposed by FKNN-DPC [32] is:

$$\rho_i = \sum_{j\in KNN_i} exp(-d_{ij}) \tag{5}$$

Comparing (4) and (5) with (1) and (3), we can see that to calculate the local density for point i in DPC-KNN and FKNN-DPC only needs K points in KNN_i, while it needs the whole dataset in DPC. If the KNN of each point is known in advance, the complexity of computing density of a point by using the former two equations is in general much less than that by the latter two [32].

2.2. Density metrics analyses

The density metrics (3–5) use Gaussian kernels to calculate the local density values. We demonstrate the differences between these metrics in Fig. 1:

Fig. 1(a) visualizes the Gaussian functions $exp(-t)$ and $exp(-\frac{t^2}{\sigma^2})$ with different σ. The green dash line and the blue dash dot line are curves of $exp(-\frac{t^2}{\sigma^2})$ with $\sigma = \sqrt{2}$ and $\sigma = 1$, respectively. The curve with small σ goes down more quickly than the curve with a large one. Comparing the red solid line curve which represents $exp(-t)$ with the curves of $exp(-\frac{t^2}{\sigma^2})$, it is obviously that values of $exp(-\frac{t^2}{\sigma^2})$ are greater than those of $exp(-t)$ when $t < \sigma^2$ but decay faster on the contrary.

Fig. 1(b) shows a synthetic data containing 83 points, in which seven points with their indexes are marked by red circles.

Fig. 1(c) shows the normalized density values of points in (b), which calculated by five different metrics with $K = 5$ and $d_c = 0.0557$. ρ_i^1 (1) and ρ_i^2 (3) are used by DPC, ρ_i^3 (4) and ρ_i^4 (5) are used by FKNN-DPC and DPC-KNN, respectively. The density metric ρ_i^5 (8) is proposed in this paper, we will give a detailed description in Section 3.

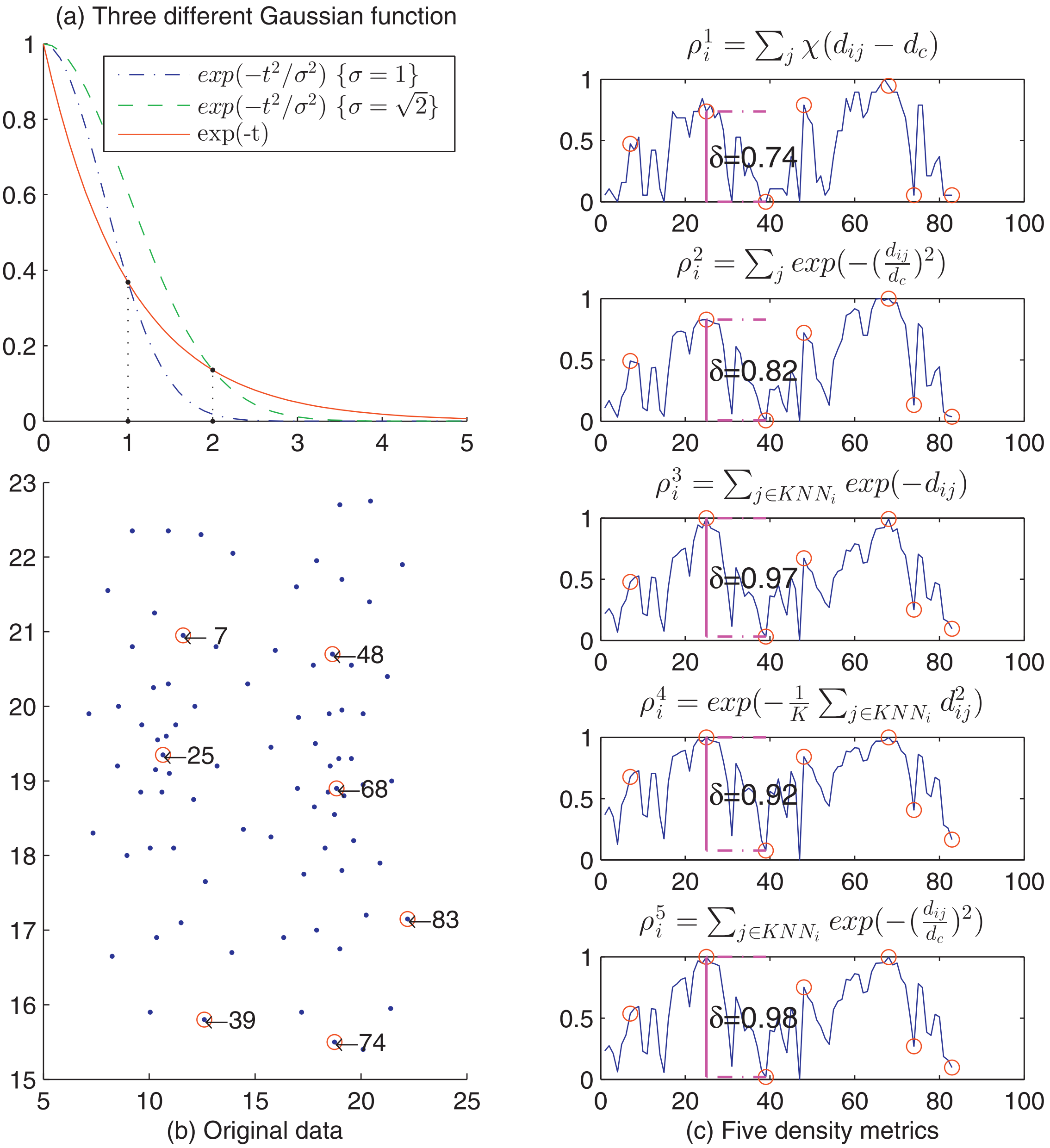


Fig. 1. Demonstration of the local density metrics. (a) Gaussian kernel function adopted by metrics; (b) original data used to calculate the local densities; (c) the local densities of points in (b), the number of nearest neighbor (*KNN*) is set to 5 and with this value, the cutoff distance d_c is computed by (6).

In Fig. 1(b), point {25} is in the core area of the left cluster and {68} is another core point in the right clusters, whereas points {39, 74, 83} are in their border area, points {7, 48} are intermediates. As Fig. 1(c) shows, their local densities calculated by the five metrics are all coincide with their spacial distributions. Most important of all, the larger density gap between a point in core area and a point in another area is gotten, the easier these points can be discriminated and then the more accurate result will the clustering has. Fig. 1(c) shows the density gaps δ between point {25} and {39} gotten by using the five different density metrics. Obviously, the gaps got by the metrics ρ_i^3 (4) and ρ_i^5 (8) are larger than those got by the others while the largest one is gotten by our density metrics ρ_i^5 (8).

3. Adaptive density peak clustering based on K-nearest neighbors with aggregating strategy

There are still some defects in DPC and its successors. To solve the problems in DPC and its successors, we will improve DPC in three aspects by defining a new local density metric based on the K-nearest neighbors, adopting a new way to select initial cluster centers and aggregating clusters if they are density reachable. In this section, we will give the essential details of the ADPC-KNN algorithm and analyze its complexity theoretically.

3.1. The local density metric of ADPC-KNN

As we analysed in Section 2, values of the Gaussian function $exp(-\frac{t^2}{\sigma^2})$ is larger than $exp(-t)$ when $t < \sigma^2$ but decay more quickly than the latter. As for the local density calculation based on the idea of K-nearest neighbor, the density metric using $exp(-\frac{t^2}{\sigma^2})$

makes points in core areas more discriminable to points in other areas. This will help the clustering to get more accurate results. So, we propose a new density metric based on the idea of K-nearest neighbor by using Gaussian kernel $exp(-\frac{t^2}{\sigma^2})$. For simplifying the clustering algorithm and making it more adaptable, the parameter σ (or d_c) is computed by the input parameter K as follows:

$$d_c = \mu^K + \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(\delta_i^K - \mu^K\right)^2} \tag{6}$$

where N is the number of points in the dataset, δ_i^K is the distance between points i and its Kth nearest neighbor, which defined by $\delta_i^K = max_{j \in KNN_i}(d_{ij})$, and μ^K is the mean value of δ_i^K of all points which defined by:

$$\mu^K = \frac{1}{N}\sum_{i=1}^{N}\delta_i^K \tag{7}$$

In (6), the second item in the right of the equation is the standard deviation of distance between each point and its corresponding Kth neighbor. It makes the clustering more adaptable and robust on corrupt datasets, which contain any data that cannot be understood and interpreted correctly by machines.

Our density metric is defined as:

$$\rho_i = \sum_{j \in KNN_i} exp\left(-\frac{d_{ij}^2}{d_c^2}\right) \tag{8}$$

This definition uses the distribution information of K nearest neighbors of point i and the parameter d_c to calculate its local density ρ_i. It has advantages as the density metrics of FKNN-DPC and DPC-KNN have, but can make points in the core areas more discriminable to points in other areas.

3.2. Terminology used

Some of the concepts used in DBSCAN and OPTICS are redefined here in terms of our requirements. We extend these concepts defined for objects to clusters.

Definition 1. (**Core-distance of a cluster**): The core-distance of a cluster C_u, denoted by σ_u, is defined by:

$$\sigma_u = \frac{1}{|C_u|}\sum_{i \in C_u} d_{cp,i} \tag{9}$$

The symbol $|\cdot|$denotes the cardinal of a set, $d_{cp,\ i}$ denotes the distance between the center point of the cluster and point i. The core-distance of a cluster C_u is the mean value of distances between the center point of the cluster and all points belong to the uth cluster.

Definition 2. (**Border-points-pair set between two clusters**): The border-points-pair set between two clusters C_u and C_v, denoted by B_u^v, is defined by:

$$B_u^v = \left\{(i, j) | d_{ij} < min(\sigma_u, \sigma_v), i \in C_u, j \in C_v\right\} \tag{10}$$

The set B_u^v contains all border points between these two clusters. Obviously, B_u^v is the same as B_v^u.

Definition 3. (**Border-density of a cluster**): The border-density of a cluster C_u, denoted by ρ_u^B, is defined by:

$$\rho_u^B = max_{(i,j) \in B_u}\left(\frac{\rho_i + \rho_j}{2}\right) \tag{11}$$

where B_u is the union of all border-points-pairs sets between C_u and other clusters in the cluster set, which defined by $B_u = \bigcup_{v \neq u} B_u^v$.

Definition 4. (**Density directly-reachable**): A cluster C_u is density directly-reachable from a cluster C_v with respect to border-density if

1) $B_u^v \neq \phi$;
2) $\exists (i, j) \in B_u^v,\ \rho_i < \rho_u^B \& \rho_j < \rho_v^B$.

It is easy to verify that density directly-reachable is symmetric.

Definition 5. (**Density reachable**): C_u is density reachable to C_v if there exist a clusters path $C_1 = C_u, C_2, \cdots, C_n = C_v$, where each C_{i+1} is directly-reachable to C_i.

The density reachable is symmetric and transitive. One can use mathematical induction to prove its veracity.

3.3. The major steps of ADPC-KNN

Inputs: dataset X, parameter K.

Output: the cluster C.

Step 1: Preprocess data like normalizing and reducing dimensions;

Step 2: Compute the Euclidean distance matrix and calculate the cutoff distance d_c using (6);

Step 3: Calculate ρ_i and δ_i for point i using (8) and (2), respectively;

Step 4: Select all points whose δ larger than the cutoff distance d_c as initial cluster centers;

Step 5: Assign each remaining point to the nearest cluster center;

Step 6: Calculate the core distance σ and border density ρ^B of each cluster using (9) and (11);

Step 7: Aggregate all density reachable clusters;

Step 8: Return the clustering C.

3.4. Complexity analyses of ADPC-KNN

Suppose the dataset has N points and let $|C|$ denote the number of clusters. In processing of ADPC-KNN, there are three objects need storing spaces: First, the matrix storing the distance from each point to its K-nearest neighbors has KN entries. Second, each point has two attributes as ρ and δ, which needs $2N$ spaces. Third, clusters need spaces to store their border-points-pair sets, core-distances ($|C|$), and border-densities ($|C|$). Although N points can produce N^2 points-pairs, since the number of border points are far less than N generally, spaces required by these objects do not exceed $O(N^2)$. So the space complexity of ADPC-KNN is of the same order as DPC in [24].

The time complexity of ADPC-KNN depends on the following aspects: (a) computing the distance between points ($O(N^2)$); (b) sorting the distance vector of each point ($O(N^2)$), of course one can use quick sort method and then the time complexity of sorting will be down to ($O(NlogN)$); (c) calculating the cutoff distance ($O(N)$); (d) calculating the local densityρ with K-nearest neighbors ($O(KN)$) but K is not great than N; (e) calculating the distance δ for each point ($O(N^2)$); (f) Selecting initial cluster centers and assign each remaining point to the nearest cluster center ($O(N^2)$); (g) calculating the core distance σ of each cluster ($O(N)$); Because each subcluster has no intersection, from (9) we can see the distance between the center point of the subcluster and other points in this subcluster will be used only once. So the time complexity of calculating the core distances of all subcluster is ($O(N)$); (h) getting boder-points-pair sets between all clusters ($O(N^2)$); The time complexity will reach to top when the dataset is partitioned into subclusters with same size, in which the time to get a boder-points-pair set between two clusters is $(\frac{N}{|C|})^2$ and $|C|$ is greater than 1, so the time complexity of getting boder-points-pair sets between all clusters is$O(N^2)$; (i) calculating the border density ρ^B of each cluster ($O(N)$).

Table 1
Synthetic datasets.

Dataset	Instances	Dimensions	Clusters	Sources
Flame	240	2	2	[11]
Aggregation	788	2	7	[12]
Spiral	312	2	3	[4]
D31	3100	2	31	[29]
R15	600	2	15	[29]
Unbalance	6500	2	8	[18]
A3	7500	2	50	[17]
Dim-set	1024	<=1024	16	[10]
S-set	5000	2	15	[9]
Birch	100,000	2	100	[34]

The above analysis demonstrates the overall time complexity of ADPC-KNN is $O(N^2)$, which is same to DPC.

4. Experiments and results

In this section, we conducted experiments on synthetic and real-world datasets, which are commonly used to test the performances of clustering algorithms. The performance of ADPC-KNN was compared with DPC [24], DBSCAN [8], K-means++ [2], EM [6] and single-link [22]. Three popular criteria F1 measure (F1) [23], adjusted mutual information (AMI) and adjusted rand index (ARI) [30] were used to evaluate the performances of the above clustering algorithms. Each benchmark value ranged from −1.0 to 1.0, and the larger it is the better is the clustering. The codes of DBSCAN and DPC were provided by their authors. The code of K-means++ was provided by Laurent Sorber in [28]. EM and single-link were implemented by using the functions “fitgmdist” and “clusterdata” defined in Matlab2014a, respectively.

The synthetic datasets we used in experiments are listed in Table 1, which were all downloaded from the website “Clustering datasets” [18] and come from research published in [4,9–12,17,29,34]. Table 2 describes the real-world datasets from [20,26].

There are various parameters the six clustering algorithms needed setting. ADPC-KNN needs only one parameter K, the number of nearest neighbors of a point, to be pre-specified. DBSCAN has two input parameters, the maximum radius *Eps* and the minimum points *MinPts*. For DPC, the cutoff distance d_c is required and initial cluster centers are selected manually on the decision graph which composed of the density ρ and the distance δ. It must be noted that we only adopted the Gaussian kernel metric in (3) to calculate local densities since the two density metrics (1) and (3) have little different efficiency on large-size datasets, but the metric (3) gets better results on small size datasets. K-means++, EM and single-link were accepted the true clusters number K as their input parameters. We implemented the algorithms on each dataset for a number of times and listed the best result of each method out. The parameters of ADPC-KNN, DPC, and DBSCAN were carefully chosen for every implementation.

Tables 3 and 4 show parameters settings for the six clustering algorithms and their results in terms of the number of clusters (Cl) be found, the values of benchmarks as F1, AMI and ARI on datasets listed in Tables 1 and 2. The column “Par” of each algorithm is its parameter we set: For ADPC-KNN, it is parameter K that refers the number of neighbors of a point; For DPC, it is d_c that refers the cutoff distance. It must be noted that the value of d_c we set is the real cutoff distance but not the percent value as other algorithms do; For DBSCAN, “Par” have two values separate by “/”, which represent the parameters *Eps* and *MinPts*; For K-means++, EM and single-link, it refers the true number of clusters. The bold font indicates the best of the results and the bar ’-’ represents there are no corresponding values. Moreover, the result of each algorithm for the synthetic datasets is displayed embedded in two-dimensional space as different marked and colored plots.

For real world datasets, it should be noted that we did a few data preprocessing on some of them or selected the subset from them to do experiments, which are all listed below:

- All samples with null or uncertain values or duplicates in the datasets were removed. Such datasets are Water-t, Breast-wpbc, Echocardiogram, Internet-a and Pima.
- The Multiple-f dataset consists of features of handwritten numerals (‘0’–‘9’) extracted from a collection of Dutch utility maps and includes six feature sets (files), we only use the subset mfeat-fou to test. In this set, each image is represented by 76 Fourier coefficients of the character shapes.
- The Japanese-v dataset has 2 subsets, we take its ’size_ae.train’ to test here.
- There are 2 datasets named Waveform listed in Table 2. The second one tagged with ’n’ has additional 19 noise features with mean 0 and variance 1 compared to the first one.
- The Olivetti-f dataset has 40 subjects and each subject has 10 different images. We took s31 to s40 subjects to do the experiments and vectorized all images.
- The SPECT-h datasets were divided into two subsets and we took the SPECT.train subset to test the algorithms.
- Each data in Wholesale has 8 attributes. We did clustering process according to values of the first six attributes and selected values of the region attribute as clustering label.
- All text attributes in the Cylinder-b were removed.
- Each attribute in Internet-a, Cylinder-b and Spambase was normalized.
- All text values in Chess were replaced by numbers, such as ’f’ was replaced by 0 and ’t’ by 1 and so on.
- The attributes no. 1, 10–13 were removed from Echocardiogram and the second attribute (’still-alive’) was selected as clustering label.
- Some preprocessing steps were applied to remove noises, the artifacts, and the pectoral muscle in a mammogram. Then by biniarizing the image, the pixels represent potential micro calcifications were obtained. The 12 feature vector was constructed for each of these pixels, which forms the sample space we handle.
- Noise with a uniform distribution was added to each feature of the data sets Iris and Pima, respectively, which generate the two noisy data sets: Iris (noise) and Pima (noise).

4.1. Experiments on synthetic datasets and results analysis

In this subsection, we show the performance of ADPC-KNN, DPC, DBSCAN, K-means++, EM and single-link on 10 synthetic datasets listed in Table 1. The result of each algorithm on 8 of these synthetic datasets is displayed embedded in two-dimensional space as different marked and colored shapes, just as Figs. 2–7 show. Each cluster center achieved by ADPC-KNN or DPC has been marked out by opposite color to that of points in the same cluster. In DBSCAN, cluster centers have no meaning because they are chosen randomly among those points satisfying the core point condition, moreover, some points labeled as outlier are not displayed. In K-means++, EM and single-link, no cluster center was displayed.

The Aggregation set has 7 clusters of different size and shapes, and two pairs of clusters are connected each other. Fig. 2 shows ADPC-KNN and DPC can find both cluster centers and correct clusters while DPC labels one point wrongly in the adjacent region of the right two clusters. In Table 3, the benchmarks data of ADPC-KNN are all 1.00 exactly and their values of DPC are also showed as 1.00 for data rounding. DBSCAN finds out all apart clusters but cannot partition two different clusters connected each other. K-

Table 2
Real-world datasets.

Dataset	Instances	Attributes	Clusters	Dataset	Instances	Attributes	Clusters
Iris	150	4	3	Breast_wpbc	699	10	2
Seeds	210	7	3	Multiple-f	2000	76	10
Olivetti-f	400	92*112	40	Japanese-v	4274	12	9
Zoo	101	18	7	Sonar	208	60	2
Page-b	5473	10	5	Libras-m	360	90	15
Internet-a	3279	1558	2	Pima	768	8	2
Pen-based	10,992	16	10	Cylinder-b	512	39	2
Heart-Cleveland	303	14	5	Liver-d	345	7	2
Waveform (n)	5000	40	3	Waveform	5000	21	3
Wine	178	13	3	Musk (v1)	476	168	2
Ecoli	336	8	8	Echocardiogram	132	13	2
Spambase	4601	57	2	Wholesale	440	8	3
Monk-3	432	8	2	SPECT-h	267	22	2
Chess	3196	36	2	Semeion	1593	256	10
Iris (noise)	150	4	3	Pima (noise)	768	8	2

Table 3
Comparison of three benchmarks for 6 clustering algorithms on synthetic datasets.

Algorithm	Par	Cl	F1	AMI	ARI	Algorithm	Par	Cl	F1	AMI	ARI
Aggregation						A3					
ADPC-KNN	40	7	**1.00**	**1.00**	**1.00**	ADPC-KNN	75	50	**0.98**	**0.98**	**0.97**
DPC	0.05	7	1.00	1.00	1.00	DPC	0.05	50	0.83	0.90	0.75
DBSCAN	0.15/9	5	0.85	0.80	0.81	DBSCAN	0.04/8	40	0.82	0.84	0.63
K-Means++	7	7	0.84	0.83	0.76	K-Means++	50	50	0.83	0.91	0.85
EM	7	7	0.81	0.80	0.68	EM	50	–	–	–	–
Single-link	7	7	0.85	0.80	0.80	Single-link	50	50	0.39	0.61	0.32
D31						R15					
ADPC-KNN	25	31	**0.97**	**0.96**	**0.94**	ADPC-KNN	20	15	**1.00**	**0.99**	**0.99**
DPC	0.03	31	**0.97**	0.95	0.93	DPC	0.04	15	**1.00**	**0.99**	**0.99**
DBSCAN	0.07/3	23	0.71	0.77	0.56	DBSCAN	0.17/3	10	0.73	0.76	0.53
K-Means++	31	31	0.90	0.93	0.86	K-Means++	15	15	0.92	0.94	0.89
EM	31	31	0.81	0.86	0.73	EM	15	15	0.85	0.88	0.78
Single-link	31	31	0.31	0.43	0.15	Single-link	15	15	0.77	0.79	0.54
Flame						Unbalance					
ADPC-KNN	25	2	**1.00**	**1.00**	**1.00**	ADPC-KNN	120	8	**1.00**	**1.00**	**1.00**
DPC	0.10	2	**1.00**	**1.00**	**1.00**	DPC	0.01	8	**1.00**	**1.00**	**1.00**
DBSCAN	0.2/5	2	0.79	0.68	0.27	DBSCAN	0.13/8	8	0.99	0.98	**1.00**
K-Means++	2	2	0.84	0.39	0.45	K-Means++	8	8	0.77	0.85	0.74
EM	2	2	0.79	0.40	0.32	EM	8	8	0.97	0.95	0.96
Single-link	2	2	0.69	0.01	0.01	Single-link	8	8	0.78	0.83	0.61
Spiral						Dim1024					
ADPC-KNN	15	3	**1.00**	**1.00**	**1.00**	ADPC-KNN	63	16	**1.00**	**1.00**	**1.00**
DPC	0.06	3	**1.00**	**1.00**	**1.00**	DPC	0.01	16	**1.00**	**1.00**	**1.00**
DBSCAN	0.29/5	3	**1.00**	**1.00**	**1.00**	DBSCAN	10/8	16	**1.00**	**1.00**	**1.00**
K-means++	3	3	0.35	−0.01	−0.01	K-Means++	16	16	0.96	0.97	0.94
EM	3	3	0.35	−0.01	−0.01	EM	16	–	–	–	–
Single-link	3	3	**1.00**	**1.00**	**1.00**	Single-link	16	16	**1.00**	**1.00**	**1.00**
S4						Birch					
ADPC-KNN	160	15	**0.80**	0.72	**0.64**	ADPC-KNN	110	100	**0.71**	**0.86**	**0.61**
DPC	0.04	15	**0.80**	**0.73**	**0.64**	DPC	–	–	–	–	–
DBSCAN	0.04/8	20	0.53	0.40	0.09	DBSCAN	0.02/9	40	0.37	0.59	0.20
K-Means++	15	15	**0.80**	0.72	0.63	K-Means++	100	100	0.68	0.84	0.58
EM	15	15	0.64	0.62	0.47	EM	100	–	–	–	–
Single-link	15	15	0.13	0.04	0.00	Single-link	100	–	–	–	–

means++, EM and single-link cannot recognize all clusters even some of them are departed.

A3 has 50 clusters with 7500 points and D31 has 31 clusters with 3100 points. Their clusters distribute randomly on 2-d space and have mild overlapping. R15 has 600 points partitioned into 15 clusters, in which one cluster laying in the center of space is surrounded by seven other clusters closely. Table 3 shows clustering results of the 6 algorithms on the 3 sets, but only the result on A3 is display in Fig. 3. The results show ADPC-KNN can find all cluster centers out correctly and assigns almost all points to their corresponding clusters on these sets. DPC can get similar results to our algorithm on D31 and R15, whereas it fails to pick all cluster centers out on A3. It must be pointed out that the initial cluster centers are selected manually from the decision graph when processing the DPC algorithm, but sometimes the complexity of dataset like A3 makes the selection very difficult, so the number of clusters will not be able to find out correctly. In all sets we experimented on, single-link often recognizes clusters close to each other as whole while outliers as independent clusters too. DBSCAN cannot find all clusters out. EM gets no result because the process was aborted for creating ill-conditioned covariance at iteration.

Flame has two clusters with different shapes. Points are distributed homogeneously, so most of them have same densities. Unbalance has 8 clusters with different number of points and densities. Eah of the left 3 clusters has 2000 points but the right 5 objects have 100 points each one. As Table 3 show, ADPC-KNN and DPC get the precise clustering results on the two sets. On Unbalance, DBSCAN gets a nearly precise results and EM also works well.

Table 4
Comparison of three benchmarks for 6 clustering algorithms on real-world datasets.

Algorithm	Par	Cl	F1	AMI	ARI	Algorithm	Par	Cl	F1	AMI	ARI
Iris						Iris (noise)					
ADPC-KNN	17	3	**0.90**	**0.79**	**0.76**	ADPC-KNN	14	3	**0.74**	**0.47**	**0.45**
DPC	0.04	3	**0.90**	**0.79**	**0.76**	DPC	0.04	3	**0.74**	**0.47**	**0.45**
DBSCAN	0.7/10	2	0.77	0.56	0.55	DBSCAN	0.6/5	2	0.68	0.43	0.36
K-means++	3	3	0.89	0.75	0.73	K-means++	3	3	0.70	0.36	0.35
EM	3	3	0.73	0.57	0.48	EM	3	3	0.71	0.40	0.38
Single-link	3	3	0.77	0.58	0.56	Single-link	3	3	0.49	0.01	0.00
Seeds						Breast-wpbc					
ADPC-KNN	900	3	**0.92**	0.71	**0.77**	ADPC-KNN	15	2	**0.75**	**0.04**	**0.37**
DPC	0.06	3	0.91	0.72	0.76	DPC	2.0	2	0.58	0.01	−0.02
DBSCAN	0.85/2	5	0.71	0.44	0.41	DBSCAN	3.8/2	2	0.61	−0.01	−0.01
K-means++	5	3	0.91	**0.73**	**0.77**	K-means++	2	2	0.66	0.01	0.05
EM	3	3	0.89	0.72	0.72	EM	2	–	–	–	–
Single-link	3	3	0.50	0.00	0.00	Single-link	2	2	**0.75**	−0.00	−0.01
Olivetti-f						Multiple-f					
ADPC-KNN	17	10	**0.90**	**0.92**	**0.86**	ADPC-KNN	26	10	**0.74**	**0.68**	0.56
DPC	0.33	10	0.80	0.81	0.71	DPC	0.34	10	0.56	0.55	0.41
DBSCAN	–	–	–	–	–	DBSCAN	5.2/4	9	0.39	0.25	0.07
K-means++	10	10	0.87	0.87	0.79	K-means++	10	10	0.73	**0.68**	**0.58**
EM	10	–	–	–	–	EM	3	–	–	–	–
Single-link	10	10	0.72	0.76	0.60	Single-link	3	3	0.18	0.00	0.00
Zoo						Japanese-v					
ADPC-KNN	5	7	**0.87**	**0.86**	**0.87**	ADPC-KNN	46	9	**0.65**	**0.63**	**0.50**
DPC	0.1	7	0.69	0.69	0.55	DPC	0.14	9	0.58	0.52	0.37
DBSCAN	2.3/3	7	0.70	0.52	0.32	DBSCAN	0.9/6	10	0.42	0.39	0.07
K-means++	7	7	0.82	0.78	0.79	K-means++	9	9	0.48	0.45	0.30
EM	7	–	–	–	–	EM	9	9	0.49	0.51	0.34
Single-link	7	7	0.65	0.48	0.44	Single-link	9	9	0.20	0.05	0.00
Page-b						Sonar					
ADPC-KNN	900	**5**	**0.86**	0.14	0.06	ADPC-KNN	15	2	**0.66**	**0.02**	−0.00
DPC	0.01	5	**0.86**	0.09	0.03	DPC	0.23	2	0.56	−0.00	−0.00
DBSCAN	1.0/8	6	0.87	0.16	**0.28**	DBSCAN	9.2/4	2	0.65	**0.02**	−0.01
K-means++	5	5	0.77	0.05	−0.00	K-means++	2	2	0.55	0.01	**0.01**
EM	5	5	0.52	**0.22**	0.08	EM	2	–	–	–	–
Single-link	5	5	**0.86**	0.09	0.03	Single-link	2	2	**0.67**	0.00	0.00
Internet-a						Libras-m					
ADPC-KNN	800	2	**0.81**	**0.07**	**0.04**	ADPC-KNN	6	15	**0.50**	0.50	0.25
DPC	0.05	2	0.73	0.00	0.01	DPC	0.23	15	0.47	0.50	0.29
DBSCAN	–	–	–	–	–	DBSCAN	5.7/4	13	0.38	0.29	0.11
K-means++	2	2	0.72	0.03	−0.08	K-means++	15	15	**0.50**	**0.51**	**0.30**
EM	2	–	–	–	–	EM	15	–	–	–	–
Single-link	2	2	**0.81**	0.00	-0.00	Single-link	15	15	0.16	0.02	0.00
Pen-based						Heart-Cleveland					
ADPC-KNN	150	10	**0.74**	**0.72**	**0.58**	ADPC-KNN	5	5	**0.61**	**0.20**	**0.34**
DPC	0.16	10	0.73	0.70	0.56	DPC	0.05	5	0.46	0.16	0.08
DBSCAN	1.3/10	10	0.48	0.44	0.21	DBSCAN	1.2/2	5	0.47	0.01	−0.05
K-means++	10	10	0.67	0.65	0.49	K-means++	5	5	0.44	0.18	0.15
EM	10	–	–	–	–	EM	5	–	–	–	–
Single-link	10	10	0.18	0.01	0.00	Single-link	5	5	0.49	0.01	0.02
Liver-d						Cylinder-b					
ADPC-KNN	10	2	**0.67**	−0.00	−0.00	ADPC-KNN	38	2	0.68	0.00	−0.00
DPC	0.04	2	0.60	0.00	**0.01**	DPC	0.01	2	0.58	0.00	0.00
DBSCAN	1.2/2	2	0.61	0.00	−0.01	DBSCAN	3.2/4	3	0.60	0.00	**0.01**
K-means++	2	2	0.65	−0.00	−0.01	K-means++	2	2	0.58	0.00	**0.01**
EM	2	–	–	–	–	EM	2	–	–	–	–
Single-link	2	2	**0.67**	−0.00	−0.00	Single-link	2	2	**0.69**	−0.00	−0.00
Pima						Pima (noise)					
ADPC-KNN	22	2	**0.69**	0.01	0.02	ADPC-KNN	29	2	**0.69**	−0.00	−0.00
DPC	0.03	2	0.53	−0.00	−0.00	DPC	0.03	2	0.60	0.00	0.01
DBSCAN	1.4/4	2	0.66	**0.04**	**0.10**	DBSCAN	1.0/2	2	0.54	**0.02**	0/00
K-means++	2	2	0.64	0.03	0.07	K-means++	2	2	0.65	**0.02**	**0.07**
EM	2	–	–	–	–	EM	2	2	0.65	−0.00	0.01
Single-link	2	2	**0.69**	0.00	0.00	Single-link	2	2	**0.69**	0.00	0.00
Waveform(n)						Waveform					
ADPC-KNN	8	3	**0.57**	0.25	0.21	ADPC-KNN	30	3	0.60	0.31	0.25
DPC	0.33	3	0.56	0.13	0.10	DPC	0.24	3	0.61	0.36	0.30
DBSCAN	4.5/3	4	0.50	0.01	0.00	DBSCAN	2.8/3	2	0.45	0.00	0.00
K-means++	3	3	0.54	**0.36**	**0.25**	K-means++	3	3	0.54	0.36	0.25
EM	3	3	0.54	0.18	0.12	EM	3	3	**0.84**	**0.51**	**0.58**
Single-link	3	3	0.50	0.01	-0.00	Single-link	3	3	0.50	0.01	0.00

(*continued on next page*)

Table 4 (*continued*)

Algorithm	Par	Cl	F1	AMI	ARI	Algorithm	Par	Cl	F1	AMI	ARI
Wine						Musk (v1)					
ADPC-KNN	22	3	**0.72**	0.41	0.37	ADPC-KNN	142	2	**0.67**	0.00	-0.00
DPC	0.01	3	**0.72**	0.41	0.37	DPC	0.26	2	0.59	0.00	−0.00
DBSCAN	2.0/7	3	0.67	**0.51**	**0.38**	DBSCAN	7.5/8	3	0.63	**0.06**	**0.12**
K-means++	3	3	0.71	0.42	0.37	K-means++	2	2	0.57	0.02	0.01
EM	3	3	0.62	0.24	0.23	EM	2	2	0.55	0.00	−0.00
Single-link	3	3	0.50	0.02	0.01	Single-link	2	2	**0.67**	0.00	−0.00
Ecoli						Echocardiogram					
ADPC-KNN	5	8	**0.64**	**0.58**	**0.49**	ADPC-KNN	5	2	0.70	0.01	**0.06**
DPC	0.1	8	0.56	0.45	0.31	DPC	0.06	2	0.70	0.01	**0.06**
DBSCAN	1.3/2	3	0.43	0.06	0.05	DBSCAN	–	–	–	–	–
K-means++	8	8	**0.64**	0.46	0.41	K-means++	2	2	0.64	0.01	0.05
EM	8	–	–	–	–	EM	2	–	–	–	–
Single-link	8	8	0.43	0.06	0.04	Single-link	2	2	**0.71**	0.01	0.02
Spambase						Wholesale					
ADPC-KNN	1000	2	**0.68**	0.00	−0.00	ADPC-KNN	6	3	0.66	0.00	−0.03
DPC	0.01	2	0.67	**0.08**	**0.05**	DPC	0.03	3	0.62	−0.00	0.00
DBSCAN	17.3/1	2	**0.68**	0.00	0.00	DBSCAN	3.0/1	2	0.66	0.00	−0.01
K-means++	2	2	**0.68**	0.00	−0.00	K-means++	3	3	0.60	−0.00	0.01
EM	2	–	–	–	–	EM	3	3	0.52	0.00	**0.02**
Single-link	2	2	**0.68**	0.00	0.00	Single-link	3	3	**0.67**	−0.00	−0.01
Monk-3						SPECT-h					
ADPC-KNN	15	2	0.65	**0.08**	0.02	ADPC-KNN	5	2	**0.66**	0.03	0.00
DPC	0.29	2	0.64	0.06	**0.07**	DPC	0.22	2	0.64	-0.01	−0.00
DBSCAN	–	–	–	–	–	DBSCAN	1.3/2	2	0.61	**0.08**	**0.07**
K-means++	2	2	0.57	0.01	0.01	K-means++	2	2	0.64	0.03	0.04
EM	2	–	–	–	–	EM	2	–	–	–	–
Single-link	2	2	**0.66**	0.00	0.00	Single-link	2	2	**0.66**	−0.00	0.00
Semeion						Chess					
ADPC-KNN	16	10	**0.21**	**0.07**	**0.03**	ADPC-KNN	25	2	0.65	0.00	−0.00
DPC	0.60	10	0.19	0.03	0.01	DPC	0.38	2	0.65	0.00	−0.00
DBSCAN	8.5/1	12	0.18	0.00	0.00	DBSCAN	2.4/10	2	0.61	0.00	0.00
K-means++	10	10	0.18	0.06	**0.03**	K-means++	2	2	0.51	0.00	0.00
EM	10	–	–	–	–	EM	2	–	–	–	–
Single-link	10	10	0.18	−0.00	0.00	Single-link	2	2	**0.67**	0.00	0.00

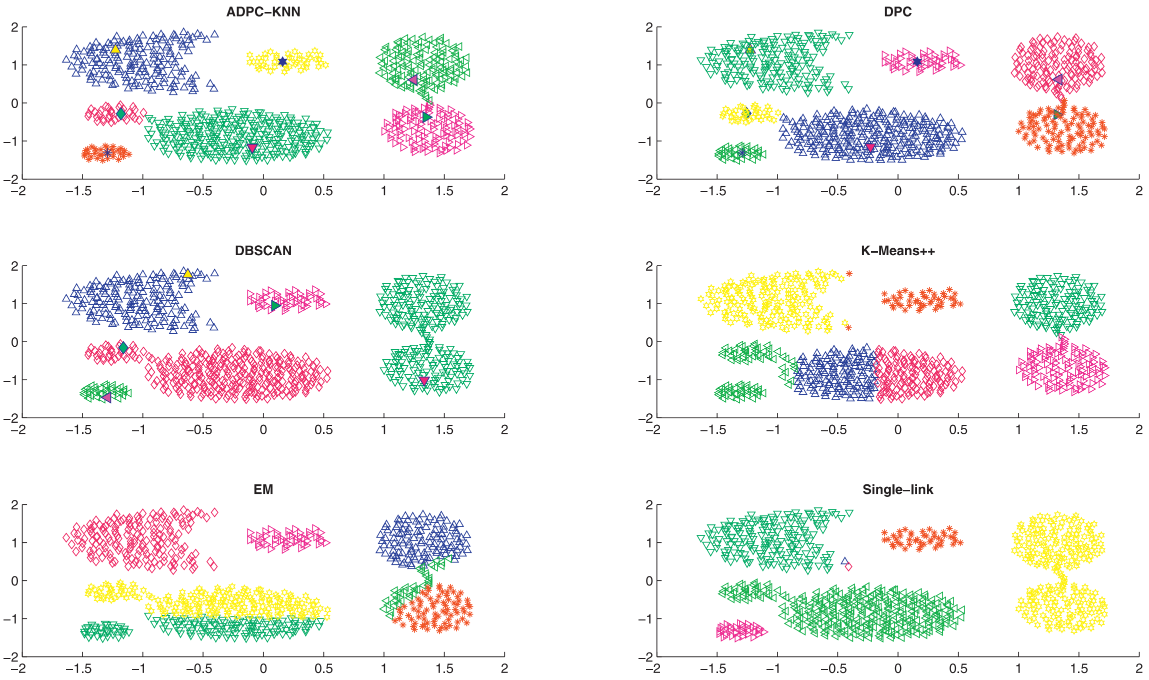


Fig. 2. Aggregation set.

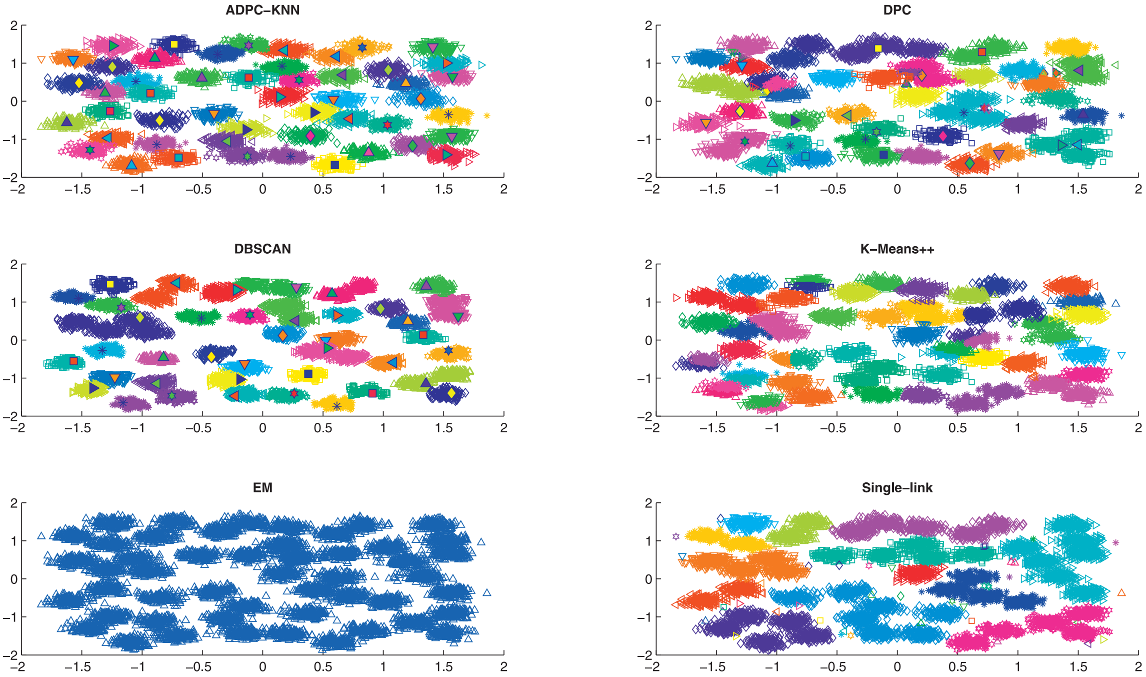


Fig. 3. A3 set.

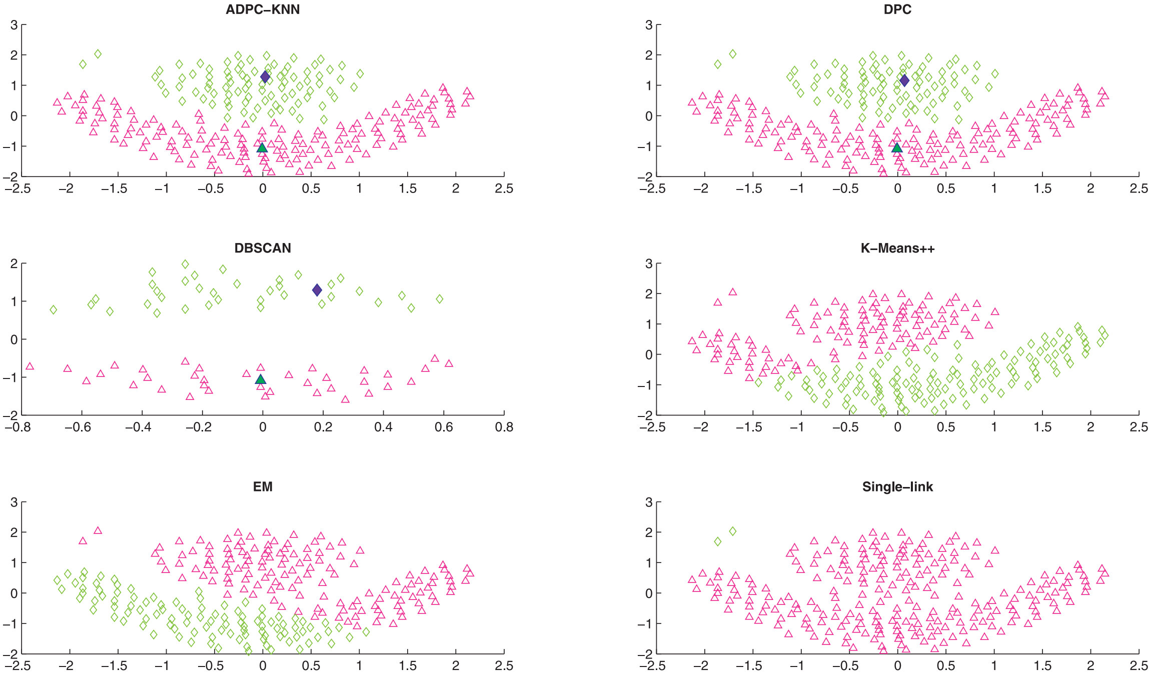


Fig. 4. Flame set.

On Flame, as Fig. 4 shows, DBSCAN can find all cluster correctly, but it labels some points as outliers which do not belong to any cluster and these points are not plotted in the graph. K-means++ and EM take some points belong to the bottom cluster to the upper one. Single-link takes the up-left two points, which are far away from other points, as one of the two clusters.

Spiral has 3 clusters which embrace each other and Dim1024 is a high-dimensional dataset and has 16 Gaussian clusters with 1024 points. From the results, we can see the clustering algorithms based on density and single-link get correct results while K-means++ and EM are powerless. Because points in Dim sets are distributed sparsely, EM cannot work. Here again we take Fig. 5 as an example to clarify this conclusion.

S4 has 15 Gaussian clusters with heavily overlapping and noise. From Table 3 and Fig. 6, we can see DPC, ADPC-KNN and K-means++ get very similar results, whereas DPC is the best one in terms of AMI. DBSCAN cannot find all clusters out. EM does a little better than DBSCAN. Single-link mixes up all connected clusters and takes outliers as separated cluster again.

The Birch2 set is 2-d data with 100,000 points and 100 clusters which are distributed along a sin curve. As Table 3 and Fig. 7 show, compared the results get by ADPC-KNN and the other 5 algorithms,

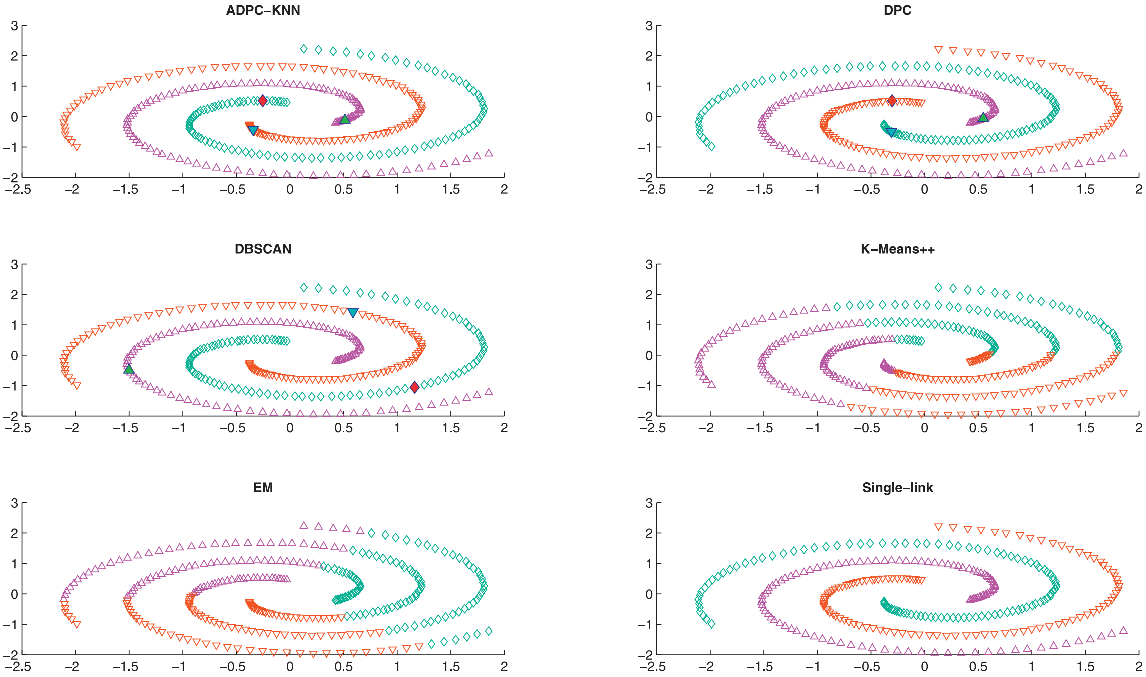


Fig. 5. Spiral set.

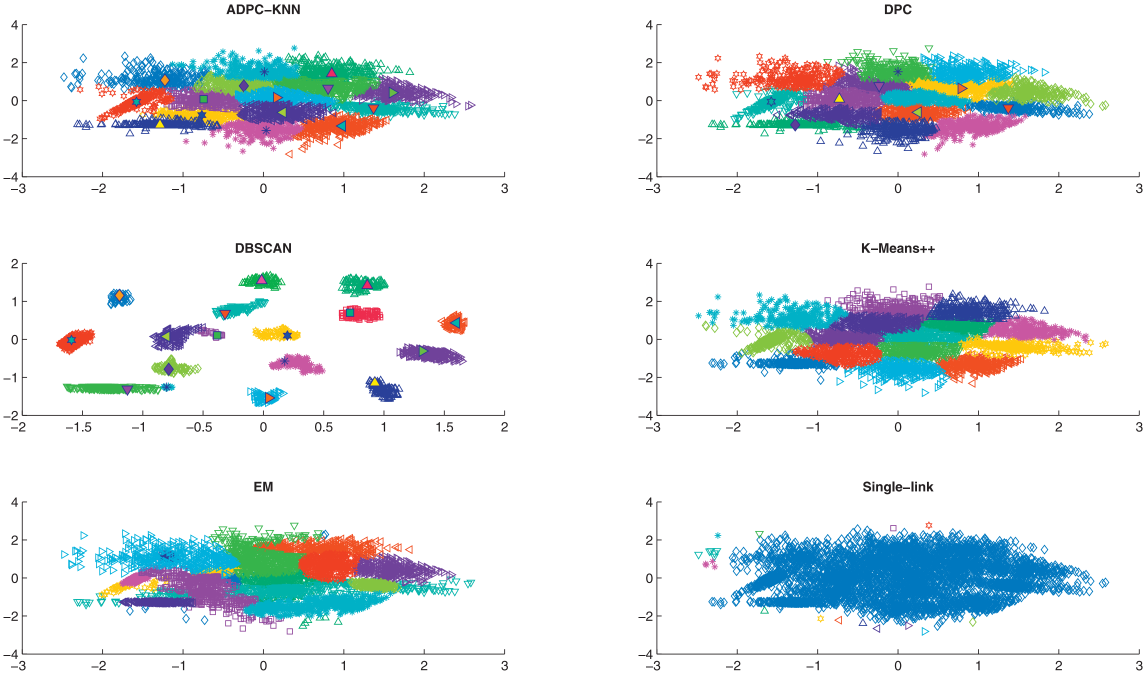


Fig. 6. S4 set.

ADPC-KNN finds out all clusters correctly and gets the best outcome. K-means++ works very closely to ours. DBSCAN finds only 40 clusters only. Because DPC and single-link need to create matrices to store distances between all pare of points, the memory they need exceed the capacity of the system we do experiments. EM do not work again for the same reason as on A3.

4.2. Experiments on real-world datasets and results analysis

In this subsection, the performance of each algorithm is benchmarked in terms of F1, AMI and ARI and shown in Tables 4. Thirty real-world datasets were chosen to test the power of ADPC-KNN to recognize the clusters on varied data., which are commonly used in clustering or classification and all listed in Table 2. In these datasets, twenty-seven come from the UCI machine learning repository [20] and one is Olivetti face dataset [26] dubbed as 'Olivetti-f'. For investigating the performances of the six algorithms on imbalance data sets, we chose mammograms from the MIAS database [14] to detect micro calcifications, which is a typical example for data imbalance problem [3]. Some preprocessing steps were applied to remove noises, the artifacts, and the pectoral muscle in a mammogram. Fig. 8(a) shows the result of pre-

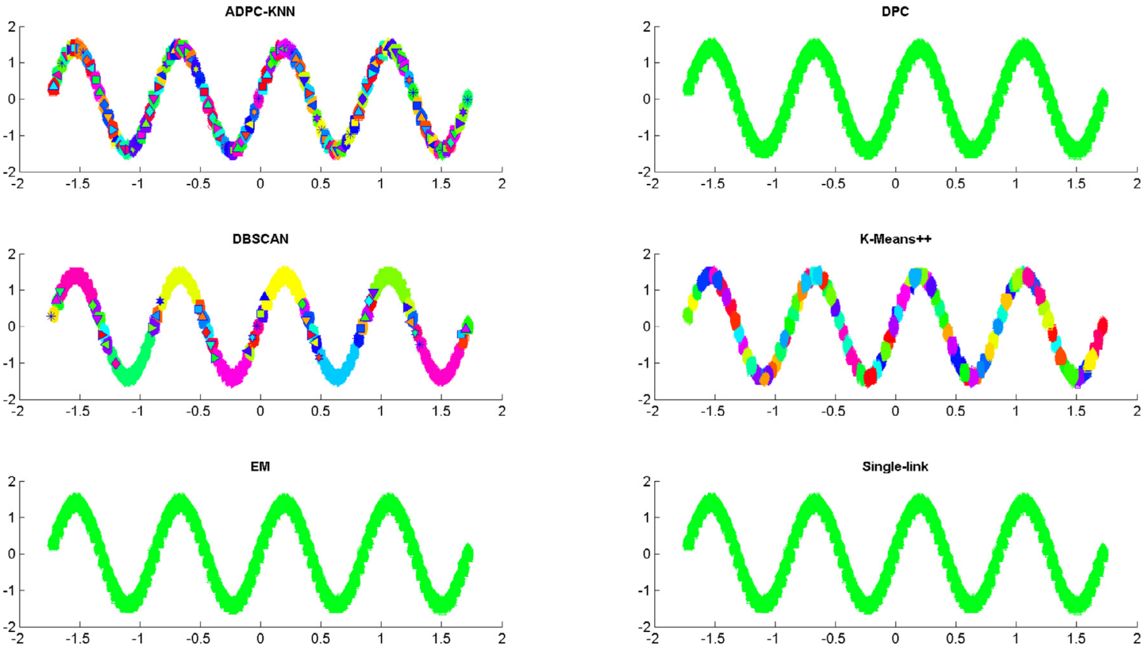
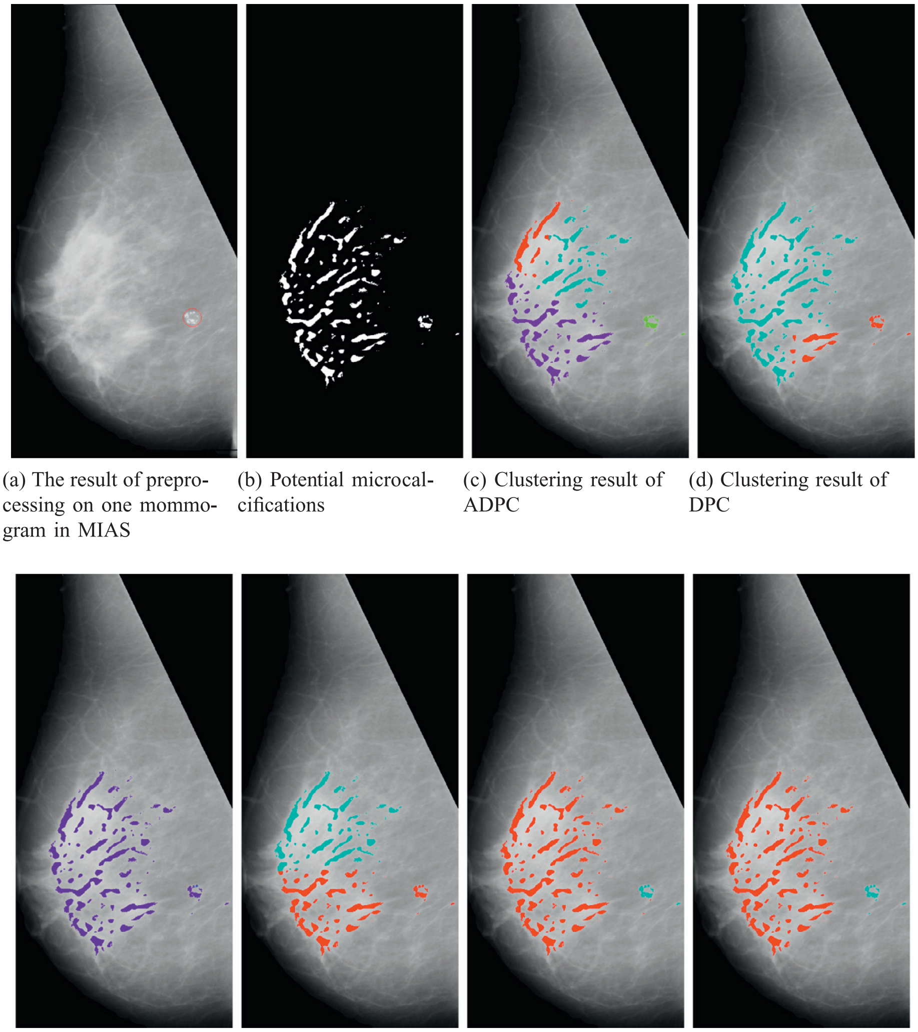


Fig. 7. Birch2.



(a) The result of preprocessing on one mommogram in MIAS

(b) Potential microcalcifications

(c) Clustering result of ADPC

(d) Clustering result of DPC

(e) Clustering result of DBSCAN

(f) Clustering result of KMeans++

(g) Clustering result of EM

(h) Clustering result of single-link

Fig. 8. Clustering results on an imbalance data set.

processing on one mommogram in MIAS [14]. The red circle indicates the cluster of micro calcifications. The potential micro calcifications were obtained by a biniarization threshold, as Fig. 8(b) demnostrated. Then we applied difference of Gaussian (DoG) and scaled symmetric difference of offset Gaussian (DOOG) filters on the image got in the first step, respectively, which got 9 features of each pixel. By appending the intensity and the location (x, y) of each pixel, we construct a feature vector of each pixel with 12 elements, which formed a sample the algorithms processing. Fig. 8(c)–(h) show the results the six methods got. EM and single linkage outperform other four methods on this data set. The ADPC also can departed the micro calcifications from other tissues of the breast, but it over-segmented the image. K-means++, DPC and DBSCAN cluster micro calcifications incorrectly.

As shown in Table 4, in terms of benchmark F1, AMI and ARI, ADPC-KNN outperforms all other 5 algorithms on Breast-wpbc, Olivetti-f, Japanese-v, Zoo, Internet-a, Ecoli and Semeion datasets. We also can see that ADPC-KNN outperforms DPC and DBSCAN on most of datasets. On Iris, Wine, Echocardiogram and Chess datasets, ADPC-KNN gets same results as DPC but does slightly worse than DPC on Waveform. It is also on Waveform that EM gets the best results. While EM cannot get anything on 21 datasets. Most of these datasets are composed of data with logical or discrete numerical attributes, which always have finite number of values. Breast-wpbc is a typical case. Pima is a representative of the others, the standard deviation of its fifth feature is more than 110 but the sixth one is less than 1. When EM run on these datasets, iill-conditioned covariance was created at iteration. On Libras-m K-means++ does best and ADPC-KNN gets same result at F1 but is little worse at AMI and ARI. Single-link performs better than others on Chess in terms F1 while values of AMI and ARI got by all methods are nearly 0. On Olivetti-f, Internet-a, Echocardiogram and Water-t datasets, DBSCAN can only find out 1 cluster butt gets nothing on Monk-3. It must be pointed out again that the initial cluster centers are very difficultly selected on some datasets when run DPC. In these cases, the centers and other points are so closely to each other on the decision graph, inspite of which was constructed by ρ and δ attributes of points or by $\gamma = \rho * \delta$ and indexes of points. It is also very difficult to choose an appropriate combination of input parameters *Eps* and *MinPts* before implementing DBSCAN.

We used the method in [5] to add noise with a uniform distribution to each feature of the datasets Iris and Pima for testing the performance of the six algorithms. The number of random noise added to each attribute is 10% of all instances and their range is between the minimum and the maximum value of the attribute. More details can be found in section 4.3.1 in [5]. As Table 4 shows, the performances of the six algorithms on Iris (noise) are all degraded significantly, but on Pima (noise), the degradation is slight.

5. Conclusions and future work

In this paper, we proposed an adaptive clustering algorithm. An uniform local density metric is defined by using Gaussian kernel and limiting the calculation to the K nearest neighbors of a point. This makes density values of points in the core area have large differences from those on border. By defining the cutoff distance as a function of the parameter *K*, ADPC-KNN needs only one input and become simpler than DPC and DBSCAN. The way finding initial cluster centers assures the true centers not be left out. It may pick out false centers but this problem was solved in succedent steps. A new concept as cluster density reachable was introduced in this paper. Lastly, ADPC-KNN aggregates those clusters meet the reachable conditions. Experiments on several synthetic datasets and real-world datasets show ADPC-KNN outperforms five other algorithms referenced in this paper.

However, the parameter *K* of our ADPC-KNN is pre-specified by hand and there is no hint to set its value now. More research are needed on how to choose the *K*.

Acknowledgments

This work is supported by Key Research Project of Hunan provincial education department (No. 17A200) and Scientific Research Fund of Chenzhou Municipal Science and Technology Bureau, Hunan(No. cz2015037).

References

[1] M. Ankerst, M.M. Breunig, H.P. Kriegel, J. Sander, OPTICS: ordering points to identify the clustering structure, SIGMOD Record 28 (2) (1999) 49–60.

[2] D. Arthur, S. Vassilvitskii, K-means++: the advantages of careful seeding, in: Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms, 07-09-January-2007, Stanford University, 2007, pp. 1027–1035.

[3] A. Kumar M.N, H.S. Sheshadri, On the classification of imbalanced datasets, Int. J. Comput. Appl. 44 (8) (2012).

[4] H. Chang, D.-Y. Yeung, Robust path-based spectral clustering, Pattern Recognit. 41 (1) (2008) 191–203, doi:10.1016/j.patcog.2007.04.010.

[5] D.F. Nettleton, A. Orriols-Puig, A. Fornells, A study of the effect of different types of noise on the precision of supervised learning techniques, Artif. Intell. Rev. 33 (4) (2010) 275–306.

[6] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm., J. R. Stat. Soc., Series B (Methodological) (1) (1977) 1–38.

[7] M.J. Du, S.F. Ding, H.J. Jia, Study on density peaks clustering based on k-nearest neighbors and principal component analysis, Knowl. Based Syst. 99 (2016) 135–145, doi:10.1016/j.knosys.2016.02.001.

[8] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Proceedings of Second International Conference on Knowledge Discovery and Data Mining 96, AAAI Press, 1996, pp. 226–231.

[9] P. Franti, I. Virmajoki, Iterative shrinking method for clustering problems, Pattern Recognit. 39 (5) (2006) 761–775, doi:10.1016/j.patcog.2005.09.012.

[10] P. Franti, O. Virmajoki, V. Hautamaki, Fast agglomerative clustering using a k-nearest neighbor graph, IEEE Trans. Pattern Anal. Mach. Intell. 28 (11) (2006) 1875–1881, doi:10.1109/TPAMI.2006.227.

[11] L. Fu, E. Medico, FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data, BMC Bioinf. 8 (2007) 3, doi:10.1186/1471-2105-8-3.

[12] A. Gionis, H. Mannila, P. Tsaparas, Clustering aggregation, in: ACM Transactions on Knowledge Discovery from Data, 1, International Conference on Data Engineering, 2007, pp. 1–30.

[13] J. Han, M. Kamber, J. Pei, Data Mining: Concepts and Techniques, 3rd ed., Morgan Kaufmane, 2011.

[14] J. Suckling, et al., The mammographic image analysis society digital mammogram database exerpta medica, in: International Congress Series, 1069, 1994, pp. 375–378.

[15] A. Jain, Data clustering: 50 years beyond k-means., Pattern Recognit. Lett. 31 (8) (2010) 651–666.

[16] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Comput. Surv. 31 (3) (1999) 264–323, doi:10.1145/331499.331504.

[17] I. Kärkkäinen, P. Fränti, Dynamic Local Search Algorithm for the Clustering Problem, Technical Report, Department of Computer Science, University of Joensuu, Joensuu, Finland, 2002.

[18] M. Learning, Clustering datasets, 2016. http://cs.joensuu.fi/sipu/datasets/.

[19] Z. Liang, P. Chen, Delta-density based clustering with a divide-and-conquer strategy: 3DC clustering, Pattern Recognit. Lett. 73 (2016) 52–59, doi:10.1016/j.patrec.2016.01.009.

[20] M. Lichman, UCI machine learning repository, 2016. http://archive.ics.uci.edu/ml.

[21] J. Macqueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, 1, 1967, pp. 281–297.

[22] F. Murtagh, P. Contreras, Algorithms for hierarchical clustering: an overview., in: Wiley Interdisciplinary Reviews-Data Mining And Knowledge Discovery, 2, 2012, pp. 86–97.

[23] D.M.W. Powers, Evaluation: from precision, recall and F-Measure to ROC, informedness, markedness & correlation, J. Mach. Learn. 2 (1) (2011) 37–63.

[24] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (6191) (2014) 1492–1496.

[25] S. Roy, D.K. Bhattacharyya, An approach to find embedded clusters using density based techniques, in: Distributed Computing and Internet Technology, Proceedings, 3816, 2005, pp. 523–535.

[26] F.S. Samaria, A.C. Harter, Parameterisation of a stochastic model for human face identification, in: Proceedings of the Second IEEE Workshop on Applications of Computer Vision (Cat. No.94TH06742), 1994, pp. 138–142, doi:10.1109/ACV.1994.341300.

[27] J. Shlens, A tutorial on principal component analysis, CoRR abs/1404.1100 (2014). http://arxiv.org/abs/1404.1100.

[28] L. Sorber, K-means++, 2016. http://cn.mathworks.com/matlabcentral/fileexchange/28804-k-means++.

[29] C.J. Veenman, M.J.T. Reinders, E. Backer, A maximum variance cluster algorithm, IEEE Trans. Pattern Anal. Mach. Intell. 24 (9) (2002) 1273–1280, doi:10.1109/TPAMI.2002.1033218.

[30] N.X. Vinh, J. Epps, J. Bailey, Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance, J. Mach. Learn. Res. 11 (2010) 2837–2854.

[31] K. Wu, M. Yang, Mean shift-based clustering, Pattern Recognit. 40 (11) (2007) 3035–3052.

[32] J.Y. Xie, H.C. Gao, W.X. Xie, X.H. Liu, P.W. Grant, Robust clustering by detecting density peaks and assigning points based on fuzzy weighted k-nearest neighbors, Inf. Sci. 354 (2016) 19–40, doi:10.1016/j.ins.2016.03.011.

[33] R. Xu, D. Wunsch, Survey of clustering algorithms, IEEE Trans. Neural Netw. 16 (3) (2005) 645–678, doi:10.1109/TNN.2005.845141.

[34] T. Zhang, R. Ramakrishnan, M. Livny, BIRCH: a new data clustering algorithm and its applications., Data Min. Knowl. Discov. (2) (1997) 141–182.