

A Novel Cluster Validity Index Based on Local Cores

Dongdong Cheng[✉], Qingsheng Zhu, *Member, IEEE*, Jinlong Huang, Quanwang Wu[✉], and Lijun Yang

Abstract—It is critical to evaluate the quality of clusters for most cluster analysis. A number of cluster validity indexes have been proposed, such as the Silhouette and Davies–Bouldin indexes. However, these validity indexes cannot be used to process clusters with arbitrary shapes. Some researchers employ graph-based distance to cluster nonspherical data sets, but the computation of graph-based distances between all pairs of points in a data set is time-consuming. A potential solution is to select some representative points. Inspired by this idea, we propose a novel Local Cores-based Cluster Validity (LCCV) index to improve the performance of Silhouette index. Local cores, with local maximum density, are selected as representative points. Since graph-based distance is used to evaluate the dissimilarity between local cores, the LCCV index is effective for obtaining the optimal cluster number for data sets containing clusters with arbitrary shapes. Moreover, a hierarchical clustering algorithm based on the LCCV index is proposed. The experimental results on synthetic and real data sets indicate that the new index outperforms existing ones.

Index Terms—Clustering analysis, clustering validity index, hierarchical clustering, local cores.

I. INTRODUCTION

CLUSTER analysis aims to divide objects into different clusters on the basis of some similar criteria so that objects in the same cluster are as similar as possible and objects in different clusters are as distinct as possible. As an important unsupervised learning method, cluster analysis has been widely applied to learning systems, pattern recognition, image processing, and statistics [1]. Clustering algorithms can be roughly classified into partitioning clustering, density-based clustering, and hierarchical clustering. The basic idea of a partitioning method is to partition the data set into clusters, such as K-means [2] and K-centers [3]. To avoid initializing cluster centers, some novel center-based clustering algorithms [4]–[8] are proposed. Density-based clustering (e.g., DBSCAN [9]) assumes that clusters are dense regions

separated by sparse regions so that it can discover clusters with arbitrary shapes. In contrast, hierarchical clustering builds a cluster hierarchy for the data set and obtains various clusters via agglomerative or divisive approaches [10]. Although a cluster hierarchy offers more information than that of a single partition, analyzing the hierarchies and obtaining the best partition from a number of clustering results is a difficult task. A novel approach [11] was proposed to solve the problem of multiple solutions in hierarchical clustering.

Clustering validation, which is used to evaluate the quality of clustering results [12], plays an important role in cluster analysis. Therefore, the research on clustering validation is important for learning systems. There are two main categories of clustering validation: external clustering validation and internal clustering validation. The main difference is whether the external information is used. Since external validation measures must know the true class labels in advance, they are mainly used for choosing an optimal clustering algorithm for a specific data set, whereas internal validation measures can be used to choose the best clustering algorithm as well as the optimal clustering results without any additional information. In the literature, numerous internal clustering validation measures have been proposed, such as the Calinski–Harabasz (CH) index [13], the Davies–Bouldin (DB) index [14], the standard deviation (SD) index [15], and the Silhouette index [16]. However, these indexes are only effective for data sets containing spherical clusters. The synchronization-based clustering algorithm Sync [17] employed the minimum description length (MDL) principle to select the best clustering results, but MDL is not effective for evaluating clusters with multiple patterns. To handle data sets containing clusters with arbitrary shapes, some novel internal cluster validity indexes have been proposed. A grid-based distance was proposed in [18] to improve the effectiveness of existing validity indexes. Liu *et al.* [19] presented a new clustering validation index based on nearest neighbors (CVNN index). The compact-separated proportion (CSP index) based on the agglomerative hierarchical algorithm was proposed in [20]. However, these measures are only effective for well-separated clusters.

To evaluate the quality of arbitrary-shaped clusters, we employ graph-based distance to measure the dissimilarity between objects. However, computing the graph-based distances between all pairs of points in a data set is time-consuming. A potential way is to select representatives and only compute the graph-based distance between them. Inspired by this idea, we propose a new internal clustering validation index in this paper, named the Local Cores-based Cluster Validity (LCCV) index, which is effective for evaluating arbitrary-shaped clusters. Local cores, with local maximum

Manuscript received April 10, 2017; revised July 20, 2017, November 15, 2017, January 29, 2018, and April 19, 2018; accepted July 3, 2018. Date of publication August 2, 2018; date of current version March 18, 2019. This work was supported in part by the National Nature Science Foundation of China under Grant 61502060, Grant 61702060, and Grant 61601060 and in part by the Project of Chongqing Education Commission under Grant KJZH17104. (Corresponding author: Qingsheng Zhu.)

D. Cheng, Q. Zhu, and Q. Wu are with the Department of Computer Science, Chongqing University, Chongqing 400044, China (e-mail: qszhu@cqu.edu.cn).

J. Huang is with the College of Computer Engineering, Yangtze Normal University, Chongqing 408000, China.

L. Yang is with the School of Computer Science and Technology, Southwest Minzu University, Chengdu 610041, China.

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TNNLS.2018.2853710

density, are selected as representatives. The LCCV index employs graph-based distance to evaluate the dissimilarity between local cores and improves the performance of the Silhouette index. Moreover, the new index can evaluate the clustering results of data sets with multiple structures. Applying the new index, a new hierarchical clustering algorithm based on LCCV (HC-LCCV) is proposed. Experimental results show that the LCCV performs well when evaluating clusters with arbitrary shapes.

The rest of this paper is organized as follows. Section II presents the related work. Section III introduces a new neighbor concept called natural neighbor and Section IV proposes a new validation index. A novel hierarchical clustering algorithm based on the new validation index is described in Section V. Experimental results on synthetic and real data sets are presented in Section VI. A discussion of LCCV is included in Section VII. Finally, the conclusions and future work are provided in Section VIII.

II. RELATED WORK

In this section, we review the literature on hierarchical clustering algorithms and internal cluster validity indexes in turn.

A. Hierarchical Clustering Algorithms

In accordance with bottom-up and top-down strategies, hierarchical clustering algorithms are divided into agglomerative and divisive methods. Agglomerative methods [21], [22] start with each point forming its own cluster and obtain a hierarchy by successively merging clusters, whereas divisive methods [23] begin with a single cluster containing all the points and proceed by iteratively splitting the clusters.

There are also algorithms that integrate partitioning methods with hierarchical methods [24]–[27]. In general, these algorithms first divide the data set into several clusters using partitioning methods, and then construct a hierarchical structure by merging these clusters. Balanced iterative reducing and clustering using hierarchies [24] partitions the data set into many small clusters with clustering feature-tree and then applies a global clustering algorithm on those clusters to achieve the final results. Cohesion-based self-merging [25] first uses K-means to partition the data set into several clusters and then continuously merges them based on cohesion in a hierarchical manner. K-means and agglomerative [26] applies K-means to the data set and each cluster is represented by its centroid; then, agglomerative approaches are applied on those centroids to build the final clustering hierarchy. Chameleon [27] produces clusters through two steps. First, it uses a graph structure to represent objects and employs a high-quality graph-partitioning algorithm, hMetis, to partition this graph into clusters. Then, Chameleon repeatedly merges the initial clusters showing the highest similarities by comparing clusters.

B. Internal Clustering Validation Indexes

The internal cluster validity indexes aim at quantitatively assessing a partition based solely on the data set and the cluster labels. Since the goal of clustering is to make objects in the same cluster similar and objects in different clusters

distinct, most measures assume that the clusters should be as compact and separated as possible, such as the CH [13], DB [14], SD [15], and Dunn [28]. The CH index evaluates clustering results based on the average between- and within-cluster sum of squares. The DB index first computes the similarity of each cluster by maximizing the similarities between a cluster and all others, and then obtains the average cluster of similarities as the index. The smaller the index is, the better the clustering result. The idea of the SD index is on the basis of the average scattering and the total separation of clusters. The Dunn index uses the minimum pairwise distance between objects in different clusters as the intercluster separation and the maximum diameter among all clusters as the intracluster compactness. These indexes take the means of the objects as the cluster centers, and they are not suitable for evaluating nonspherical clusters. The Silhouette index [16] evaluates the clustering results by averaging the cluster validity of a single sample, which may ignore the common characteristics of all objects.

Some research focuses on determining the optimal number of clusters in hierarchical clustering by defining new clustering validity indexes [10], [29], [30]. Gurrutxaga *et al.* [10] proposed a new clustering validity index called context-independent optimality and partiality to find the best partition in hierarchical clustering.

Recently, some new indexes have been proposed. Guo *et al.* [31] proposed a novel cluster validity index for categorical sequences. Marian *et al.* [32] employed generalized self-organizing maps to automatically determine the number of clusters and their multiprototypes.

Some indexes are proposed to address nonspherical clusters. MDL [17] exploits the regularities in data described by a clustering model for effective data compression. The clustering model with MDL is selected as the best clustering result. A density-based measure [33] is done by minimizing the SD of the minimum spanning tree distances of a cluster as the homogeneity measure, and minimizing the number of neighborhoods that mix patterns from different clusters as the separateness measure. CVNN [19] introduces a new separation based on the concept of nearest neighbors. This measure can dynamically select multiple objects as representatives for different clusters in different situations. CSP [20] employs the average weight of a minimum spanning tree in one cluster as the intracluster compactness cd and the minimum distance between the objects in one cluster and the objects in other clusters as the intercluster separation sd . Then, CSP defines the clustering dispersion degree as the difference of sd and cd and the clustering synthesis degree as the sum of sd and cd . Finally, the ratio of the clustering dispersion and the clustering synthesis degrees is the CSP index. This index evaluates the clustering results and determines the optimal number of clusters with arbitrary shapes. Nevertheless, these measures are only effective for well-separated clusters.

III. NATURAL NEIGHBOR

Let $\text{dist}(x, y)$ be the Euclidean distance between points x and y . In the Data Set X , point o is the k th nearest

neighbor of point p . The definitions of k nearest neighbors and reverse neighbors can be given as follows.

Definition 1 (k Nearest Neighbors): The k nearest neighbors of point p are a set of points x in X with $\text{dist}(p, x) \leq \text{dist}(p, o)$, i.e., $NN_k(p) = \{x \in X | \text{dist}(p, x) \leq \text{dist}(p, o)\}$.

Definition 2 (Reverse Neighbors): The reverse neighbors of point p are a set of points x that consider p as its k nearest neighbor, i.e., $RNN(p) = \{x \in X | p \in NN_k(x)\}$.

Natural neighbor [34] is a new neighbor concept. Compared with k nearest neighbors, this approach does not need to set any parameters and is a self-adaptive neighborhood method. The natural neighbor is inspired by friendship in human society: the friendship between two entities should be mutual. The key idea of natural neighbor is that points lying in sparse regions should possess a small number of neighbors, whereas points lying in dense regions should possess a large number of neighbors.

The formation of natural neighbor is achieved as follows: continuously expand the neighbor searching range r , and every time, compute the number of reverse neighbors for each point. When one of the following two conditions are satisfied: 1) all points have reverse neighbors and 2) the number of points without reverse neighbors does not change, we say that it reaches a natural stable state. The searching range r at this moment is the natural characteristic value λ . Formally, λ is obtained by

$$\lambda = \min \left\{ r \mid \left(\sum_{i=1}^n f(nb_r(i)) = 0 \text{ or } \sum_{i=1}^n f(nb_r(i)) = \sum_{i=1}^n f(nb_{r-1}(i)) \right) \right\} \quad (1)$$

where r is initialized with 1. $nb_r(i)$ is the number of point i 's reverse neighbors in the r th iteration (note that $nb_r(i) \geq 0$) and $f(x)$ is defined as follows:

$$f(x) = \begin{cases} 1, & \text{if } x = 0 \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

The detailed algorithm is shown in Algorithm 1. In each iteration, we search the r -nearest neighbors for each point, and count the number $nb_r(p)$ of p 's reverse neighbors. Then, we determine whether the terminal condition is met: 1) all points have reverse neighbors, i.e., the number of points without reverse neighbors (i.e., $nb_r(p) = 0$) is zero and 2) the number of points without reverse neighbors (i.e., $nb_r(p) = 0$) is the same in two successive iterations.

The value of $nb_\lambda(i)$ is larger for points in dense regions than for those in sparse regions, which is a good reflection of local point characteristics. Therefore, we regard $nb_\lambda(i)$ as the number of point i 's neighbors.

Definition 3 (Undirected Saturated Neighbor Graph): If point p is among the λ nearest neighbors of point q , then there will be an edge between p and q , and vice versa.

Definition 4 (Directed Saturated Neighbor Graph): If point p is among the λ nearest neighbors of point q , then there will be an edge from q to p .

Algorithm 1 NaN-Searching

Input: X (the data set)

Output: λ, LN

Initializing: $r=1$, $nb(i)=0$, $count(1) = N$, $NN_0(i) = \phi$, $RNN_0(i) = \phi$;

while true do

for each data point x in X do

 Find the r -th neighbor y of x ;

$nb(y) = nb(y) + 1$;

$NN_r(x) = NN_{r-1}(x) \cup \{y\}$;

$RNN_r(y) = RNN_{r-1}(y) \cup \{x\}$;

end

$nb_r = nb$;

$count(r) = \text{length}(\text{find}(nb_r=0))$;

if $count(r) = 0$ **or** $count(r) == count(r-1)$ **then**

 Break;

end

$r = r + 1$;

end

$\lambda = r$;

for each data point x in X do

$LN(x) = NN_{nb_\lambda}(x)$;

end

IV. LOCAL CORES-BASED CLUSTERING VALIDITY INDEX

The Silhouette index does well in evaluating spherical clusters, but it is unable to address nonspherical clusters. To solve the problem faced by the Silhouette index, we introduce graph-based distance to calculate the distance between points. However, it is time-consuming to calculate the graph-based distances between all pairs of points in a data set. A potential way is to select representatives from the data set and then calculate the graph-based distance between the representative points. Inspired by this idea, we propose a local cores-based cluster validity (LCCV) index, which calculates the local density and selects local cores from the data set. After that, it calculates the graph-based distance between local cores, and finally, it employs the new distance to improve the Silhouette index.

A. Local Density

The density of points in a dense region is usually larger than that in a sparse region. That is, the sum of the distances between a point and its neighbors in a dense region is usually smaller than that in a sparse region. STclu algorithm [35] employs k nearest neighbors to compute the local density k -density and proves that the k -density performs better in cluster center detection than ε -density [5], [9] does. In this paper, natural neighbor-based local density is defined in a similar way. The natural neighbor-based local density of point i is defined as

$$\rho(i) = \frac{\mu}{\sum_{j \in NN_\mu(i)} \text{dist}(i, j)} \quad (3)$$

where μ is the maximum value of nb_λ . $NN_\mu(i)$ represents the μ nearest neighbors of point i , and $\text{dist}(i, j)$ is the

Euclidean distance between points i and j . Different from the STclu algorithm, natural neighbor-based local density does not need to set parameter k , and the formation of natural neighbor helps find the appropriate parameter k .

B. Local Cores

In [7], local representatives were proposed to represent the data sets. Local representatives are points with local maximum density among the λ nearest neighbors. Since each point takes the same number of neighbors into consideration, local representatives cannot well represent the local features of the data set. In this section, local cores are introduced to represent the data set. The detailed definitions are as follows.

Definition 5 (Local Neighbors): For each point p , the $nb_\lambda(p)$ nearest neighbors are defined as its local neighbors (LNs).

Different from k nearest neighbors, LNs mean that different points have different number of neighbors. Points in dense regions have more neighbors than those in sparse regions.

Definition 6 (Representative): If the density of point q is the maximum in the LNs of point p , then q is the representative of p and its LNs, and we denote it as $Rep(p) = q$.

According to the definition, a point may have two or more representatives. In this case, they compete for being the representative of the point, and the representative competition rule (RCR) is defined as follows.

Representative Competition Rule: For point p , there are two candidate representatives R_1 and R_2 , then $Rep(p) = \arg \min_{R \in \{R_1, R_2\}} \{\text{dist}(p, R)\}$, which means that the representative closest to point p will be the final representative of p .

If the representative of point p is q , and the representative of q is r , then the representative of point p will be changed to r . We call it the representative transfer rule (RTR) which is defined as follows.

Representative Transfer Rule (RTR): If $Rep(p) = q$ and $Rep(q) = r$, then $Rep(p) = r$.

Definition 7 (Local Core): A point p is a local core if $Rep(p) = p$.

Local cores are points with local maximum density. Each local core becomes the initial cluster center. The rest of the points is assigned to the cluster that their representatives belong to. The local cores searching algorithm (LORE) is detailed in Algorithm 2, where $Rep(p)$ is the representative of point p , and $localCores(n_l)$ is the n_l th local core. Fig. 1 shows the LORE results for a data set. The red star points are local cores found by LORE, and then the data set is divided into two clusters according to the local cores.

C. Graph-Based Distance Between Local Cores

To capture the intrinsic geometric features of a data set, we use the geodesic distance as a dissimilarity measure between two points. However, in clustering settings, the exact geodesic distance between two points usually cannot be obtained directly, because we have no prior information about the underlying manifolds [36]. Nevertheless, as pointed out in [37], if a data set has sufficient points sampled from the

Algorithm 2 LORE

Input: LN (the local neighbors of each point), ρ (the natural neighbor-based density of each point)
Output: $localCores$ (the local cores), Rep (the representative of each point)
 Initializing: $Rep = \phi$, $localCores = \phi$, $cl = \phi$;
for each point x in the data set do
 Find the point y with the maximum density in $LN(x)$;
 for each point p in $LN(x)$ do
 if $Rep(p) == \phi$ then
 $Rep(p) = y$;
 end
 if $Rep(p) \neq \phi$ and $Rep(p) \neq y$ then
 Determine $Rep(p)$ according to RCR;
 end
 for each point z in the data set do
 if $Rep(z) == p$ then
 Determine $Rep(z)$ according to RTR;
 end
 end
 end
end
 $n_l = 0$; //The number of localCores;
for each point x in the data set do
 if $Rep(x) == x$ then
 $n_l = n_l + 1$;
 $localCores(n_l) = x$;
 end
end

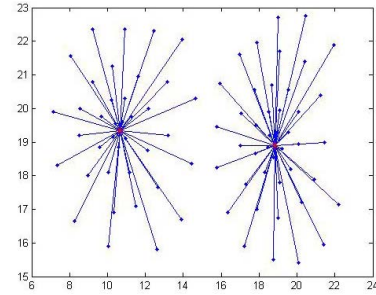


Fig. 1. Points are assigned to their representatives, and the red star points are local cores found by the LORE algorithm.

manifold, then the graph-based distance will be a good approximation of the geodesic distance. Given a graph, the graph-based distance between two points is defined as the shortest path connecting them. Yang *et al.* [38] and Jia *et al.* [39] construct the k -nearest neighbor graph and weight the edges by density sensitive distance instead of the Euclidean distance to calculate the shortest path between two nodes. Tu *et al.* [36] also exploit the k -nearest neighbor graph and the edges are weighted by a Gaussian kernel function. In this paper, we also construct a graph on the original data set and weight the edges by the Euclidean distance. The Dijkstra algorithm is employed to compute the shortest path. To avoid setting parameters and the influence of noise points, the graph we used is the directed

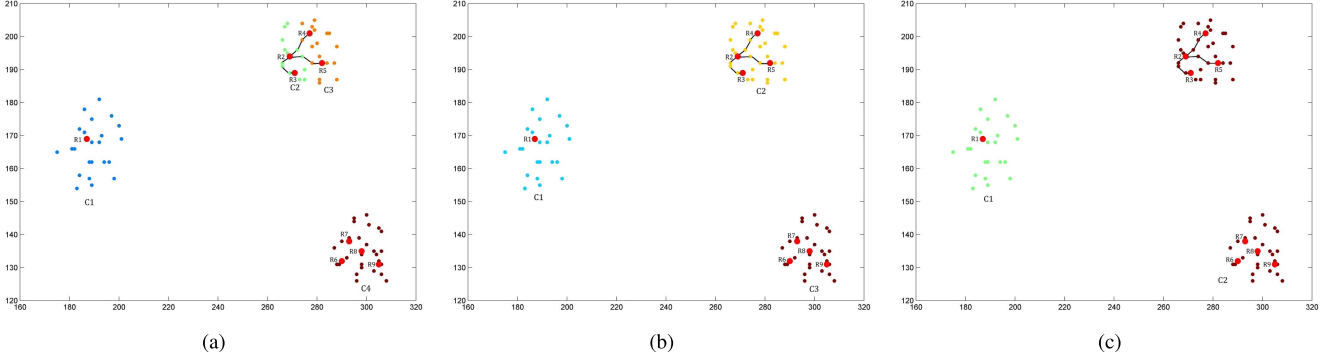


Fig. 2. Analysis of the LCCV index. LCCV index with (a) four clusters, (b) three clusters, and (c) two clusters.

saturated neighbor graph (DSNG) (refer to Definition 4). The graph-based distance between local cores is defined as follows.

Given a graph $G = (V, E)$, let $P = \{p_1, p_2, \dots, p_m\}$ represent the shortest path from $p_1 (v_i)$ to $p_m (v_j)$, $(p_k, p_{k+1}) \in E$, $1 \leq k < m$. Then, the graph-based distance between points v_i and v_j is computed as

$$D(i, j) = \sum_{k=1}^{m-1} \text{dist}(p_k, p_{k+1}) \quad (4)$$

where p_k and p_{k+1} are the points along the shortest path P . When there is no path between two points, their graph-based distance is set as the maximum value of all shortest paths.

D. LCCV Index

Given a Data Set X containing n objects, and a local core $i \in X$, let A denotes the cluster to which it has been assigned, and let $n_l(A)$ denote the number of local cores in cluster A . The average graph-based distance between local core i and other local cores in A is $a(i)$, i.e., $a(i) = (1/n_l(A) - 1) \sum_{j \in A, j \neq i} D(i, j)$. The average graph-based distance between local core i and the local cores in another cluster C is $d(i, C)$, i.e., $d(i, C) = (1/n_l(C)) \sum_{j \in C} D(i, j)$. After computing $d(i, C)$ for all clusters C ($C \neq A$), we select the smallest one (the cluster with the smallest value of $d(i, C)$ is the second best choice for i) and denote it by $b(i) = \min_{C \neq A} \{d(i, C)\}$. Thus, the LCCV value of local core i is defined as

$$\text{LCCV}(i) = \frac{b(i) - a(i)}{\max\{b(i), a(i)\}}. \quad (5)$$

When cluster A contains only a single local core i , we simply set $\text{LCCV}(i)$ to zero referring to [16]. From the above-mentioned definition, we can see that the range of $\text{LCCV}(i)$ is $(-1, 1)$. It is obvious that a larger value of $\text{LCCV}(i)$ indicates a higher correctness of assigning local core i to cluster A . The average LCCV value $\hat{\text{LCCV}}$ is defined as

$$\hat{\text{LCCV}} = \frac{1}{n} \sum_{i=1}^{n_l} (\text{LCCV}(i) \times n_i) \quad (6)$$

where n_l is the number of local cores and n_i is the number of points whose representative is local core i . A larger value of $\hat{\text{LCCV}}$ indicates a better clustering result.

TABLE I
GRAPH-BASED DISTANCE FROM R2 TO OTHER LOCAL CORES

	R1	R2	R3	R4	R5
R2	23.25	0.00	10.21	10.82	14.91
	R6	R7	R8	R9	
R2	23.25	23.25	23.25	23.25	

E. Analysis of the LCCV Index

When a local core, i , is well classified, $b(i)$ will be much greater than $a(i)$, leading to a larger value of $\text{LCCV}(i)$. The range of $\text{LCCV}(i)$ is $(-1, 1)$. When the value of $\text{LCCV}(i)$ is close to 1, it means that the intracluster graph-based distance $a(i)$ is much smaller than the smallest intercluster graph-based distance $b(i)$. Therefore, we can say that i is well clustered. When $\text{LCCV}(i)$ is approximately zero, it means that $a(i)$ and $b(i)$ are approximately equal, and hence, it is not clear at all which cluster should i be assigned to. The worst situation takes place when $\text{LCCV}(i)$ is close to -1 , which tells us that $a(i)$ is much larger than $b(i)$. It seems much more reasonable to assign i to the second-best cluster, so we can say that i has been misclassified. To conclude, $\text{LCCV}(i)$ measures how well local core i has been classified.

To better explain the LCCV index, we give a simple example, as shown in Fig. 2. The red points are local cores. The shortest paths from $R2$ to other local cores are shown in the figures. The graph-based distances between $R2$ and other local cores are shown in Table I. Since there is no path from $R2$ to $R1$, $R6$, $R7$, $R8$, and $R9$, their graph-based distances are set as the maximum value of the computed graph-based distance. In Fig. 2(a), there are four clusters and clusters $C2$ and $C3$ are close to each other. According to the definition, the average graph-based distance between $R2$ and other local cores in the same cluster $C2$ is $a(R2) = D(R2, R3) = 10.21$, and the smallest average graph-based distance between $R2$ and the local cores in another cluster (i.e., cluster $C3$) is $b(R2) = (D(R2, R4) + D(R2, R5))/2 = 12.86$. Thus, we have $\text{LCCV}(R2) = 0.21$. In Fig. 2(b), clusters $C2$ and $C3$ are merged. The average graph-based distance between $R2$ and other local cores in the same cluster $C2$ is $a(R2) = (D(R2, R3) + D(R2, R4) + D(R2, R5))/3 = 11.98$, and the smallest average graph-based distance

between $R2$ and the local cores in another cluster is $b(R2) = 23.25$; thus, $LCCV(R2) = 0.48$. In Fig. 2(c), the right two clusters are merged. The average intracluster graph-based distance of $R2$ is $a(R2) = (D(R2, R3) + D(R2, R4) + D(R2, R5) + D(R2, R6) + D(R2, R7) + D(R2, R8) + D(R2, R9))/7 = 18.42$, and the smallest average intercluster graph-based distance of $R2$ is $b(R2) = 23.25$. Therefore, we have $LCCV(R2) = 0.21$. Then, we can conclude that for $R2$, three clusters is the best choice.

To obtain the LCCV index, three main steps are required: searching local cores, computing the graph-based distances between them and calculating the LCCV value. There are three substeps in searching local cores: choosing the representative for each point, selecting local cores and assigning each point to its local core. Thus, the time complexity of searching local cores is $O(n)$. The time complexity of the Dijkstra algorithm is $O((n + e)\log(n))$, where e is the number of edges in the graph. The DSNG is sparse, and the number of its edges is less than $n \times \lambda/2$ (λ is a constant). Since the number of local cores is n_l ($n_l \ll n$), the time complexity of computing the graph-based distance between them is $O(n_l \times n \log(n))$. The LCCV calculation in each iteration is $O(n_l^2)$ when merging clusters. Therefore, the overall time complexity of computing the LCCV index is $O(n_l \times n \log(n))$.

V. HIERARCHICAL CLUSTERING ALGORITHM BASED ON LCCV

In this section, we introduce a similarity metric between clusters, and then present a novel hierarchical clustering algorithm based on local cores.

A. Similarity Between Clusters

Complete linkage [22] and single linkage [21] are commonly used methods to evaluate the similarity between clusters. The complete-linkage method uses the distance between the two farthest points of two clusters, whereas the single-linkage method uses the distance between the closest points of two clusters. However, the complete linkage has problems dealing with particular shapes, such as circles, and the single linkage suffers from the so-called chaining effect. To tackle data sets containing clusters with arbitrary shapes, we define similarity on the basis of the connectivity and closeness between clusters [40].

We construct an undirected saturated neighbor graph (USNG) on the original data sets. Given the cluster label of each object, the cut edges are the ones connecting two different clusters. For example, Fig. 3(a) shows the graph of a data set and points with different colors belonging to different clusters. Thus, the red edges in Fig. 3(b) are the cut edges.

Given clusters C_i and C_j , there exist points v_i and v_j , $v_i \in C_i$ and $v_j \in C_j$. Let $CE(C_i, C_j)$ be the set of cut edges between clusters C_i and C_j ; then, the similarity between clusters C_i and C_j is defined as follows.

Definition 8 (Connectivity): The connectivity between clusters C_i and C_j is the weight sum of the cut edges straddling the two clusters computed as follows:

$$\text{Conn}(C_i, C_j) = \sum_{e(v_i, v_j) \in CE(C_i, C_j)} w(v_i, v_j) \quad (7)$$

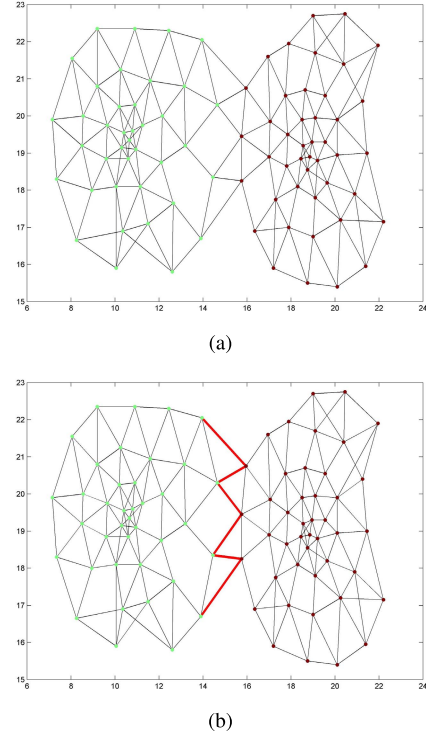


Fig. 3. (a) USNG. (b) Cut edges between different clusters.

where $w(v_i, v_j)$ is the weight of the edge between points v_i and v_j , and $w(v_i, v_j) = (1/1 + \text{dist}(v_i, v_j))$ [27], [38]. The weight denotes the similarity between points v_i and v_j .

Definition 9 (Closeness): The closeness between clusters C_i and C_j is the average weight of cut edges straddling them. It is computed as follows:

$$\text{Close}(C_i, C_j) = \frac{\text{Conn}(C_i, C_j)}{|CE(C_i, C_j)|}. \quad (8)$$

Definition 10 (Similarity): The similarity between clusters C_i and C_j is a function that combines the connectivity and closeness. It is computed as

$$\text{Sim}(C_i, C_j) = \text{Conn}(C_i, C_j) \times \text{Close}(C_i, C_j)^\alpha \quad (9)$$

where α is a user-specified parameter. If $\alpha > 1$, then we give a higher importance to the closeness, and when $\alpha < 1$, we give a higher importance to the connectivity. Based on experience, we set $\alpha = 2$ in this paper.

B. Hierarchical Clustering Algorithm

The main idea of the proposed HC-LCCV is as follows: first, find the local cores and divide the data set into several clusters according to them; then, merge the small clusters with their most similar clusters; after that, a cluster hierarchy tree is constructed by repeatedly merging the most similar clusters¹ and the LCCV index in each layer of the hierarchy

¹When the number of connected subgraphs of the constructed USNG is larger than 2, HC-LCCV will terminate before the number of clusters reduces to 2, and the final number of clusters will equal the number of connected subgraphs. When the number of connected subgraphs is smaller than or equal to 2, and the final number of clusters will be 2.

Algorithm 3 HC-LCCV

Input: X (the data set)
Output: ocl (the optimal clustering results)
 $(\lambda, LN) = \text{NaN-Searching}(X)$;
for each point x in X **do**
 | Compute the density $\rho(x)$ according to Eq. (2);
end
 $(localCores, Rep) = \text{LORE}(LN, \rho)$;
 $n_l = \text{length}(localCores)$;
for $i = 1:n_l$ **do**
 $C_i = \phi$;
 for each point x in the data set **do**
 | **if** $Rep(x) == localCores(i)$ **then**
 | $C_i = C_i \cup x$;
 | **end**
 end
end
Compute the similarity matrix sim according to Definition 10;
Compute the graph-based distance between $localCores$;
 $n = \text{length}(X)$;
 $avgn = n/n_l$;
for each cluster C_i **do**
 $n_p = \text{length}(C_i)$;
 if $n_p < avgn$ **then**
 | $C_j = \arg \max_j (sim(i, j))$;
 | Merge clusters C_i and C_j ;
 | Update the similarity matrix sim ;
 end
end
while the number of clusters $N_{cl} > 2$ **do**
 $(C_i, C_j) = \arg \max_{i,j} (sim)$;
 | Merge clusters C_i and C_j and obtain the clustering result cl ;
 | Update the similarity matrix;
 | Calculate the $lccv(cl)$ value according to Formula (6);
end
 $ocl = \arg \max_{cl} \{lccv(cl)\}$;

tree is computed; finally, the best clustering results with the maximum average LCCV value is obtained. The procedure is described in Algorithm 3, where function NaN-Searching() is the natural neighbor searching algorithm, described in Algorithm 1.

In essence, the proposed hierarchical clustering algorithm mainly contains three processes: obtain the LNs, partition the data set based on local cores, and merge the initial clusters based on cluster similarities. The time complexity of the first phase is $O(n^2)$. When introducing k-dimensional-tree [41], its time complexity will be reduced to $O(n \log(n))$. The time complexity of the second phase is $O(n)$. Before the merging process, we must compute the similarities between clusters and the graph-based distances between local cores. The time complexity of computing similarities between clusters by counting

cut edges is $O(n_l \times n)$ (n_l is the number of local cores). Then, the time complexity of computing the graph-based distance is $O(n_l \times n \log(n))$. When merging clusters, the time complexity of updating the similarity matrix is $O(n)$ and calculating the LCCV index is $O(n_l^2)$ in each iteration. Therefore, the overall time complexity of HC-LCCV is $O(n_l \times n \log(n))$.

VI. EXPERIMENTAL ANALYSIS

To demonstrate the effectiveness of the LCCV index, we use 12 synthetic data sets and six real data sets. Based on the HC-LCCV algorithm, we compare the LCCV index with other indexes, including the DB, Silhouette, MDL, CVNN, and CSP; the time complexity of calculating these indexes is $O(n + n_c^2)$, $O(n^2)$, $O(n^2)$, $O(n \log n)$, and $O(n^3)$, respectively, where n_c is the number of clusters. We also apply the LCCV index to other algorithms, such as K-means and DBSCAN, and perform experiments on synthetic data sets. Moreover, we employ accuracy [42] to evaluate the partition of the synthetic data sets and real data sets. The clustering accuracy is computed as follows:

$$\text{Accuracy} = \frac{1}{n} \sum_{i=1}^n \delta(y_i, \text{map}(c_i)) \quad (10)$$

where y_i is the real cluster label, c_i is the serial number obtained by clustering, and

$$\delta(x, y) = \begin{cases} 1 & x = y \\ 0 & x \neq y \end{cases}$$

is a discriminate function. The larger the value of accuracy (accuracy $\in [0, 1]$) is, the better the clustering performance of the algorithm.

A. Experiments on Synthetic Data Sets

The synthetic data sets are shown in Fig. 4. The first three data sets contain spherical clusters and some noise points. Data Set 1 consists of five clusters, and a total of 513 points. Data Set 2 is composed of three spherical clusters with skewed distribution, and a total of 1873 points. Data Set 3 contains five clusters and some noise points, and a total of 1053 points. The rest of the data sets contain clusters with arbitrary shapes. Data Set 4 consists of four line clusters, with a total of 1268 points. Data Set 5 includes three spiral clusters, with a total of 312 points. Data Set 6 contains two circle clusters and one spherical cluster, with a total of 1897 points. Data Set 7 has two manifold clusters with different densities, and a total of 746 points. Data Set 8 involves four manifold clusters and a total of 630 points. Data Set 9 includes four clusters and some noise points. Two of the clusters are spherical with different densities, and the other two are concave, with a total of 582 points. Data Set 10 has four spherical clusters in two right angle line clusters, with a total of 1741 points. Data Set 11 contains three spherical clusters in a circle cluster, with a total of 1016 points. Data Set 12 contains seven clusters and a number of noise points for a total of 8537 points.

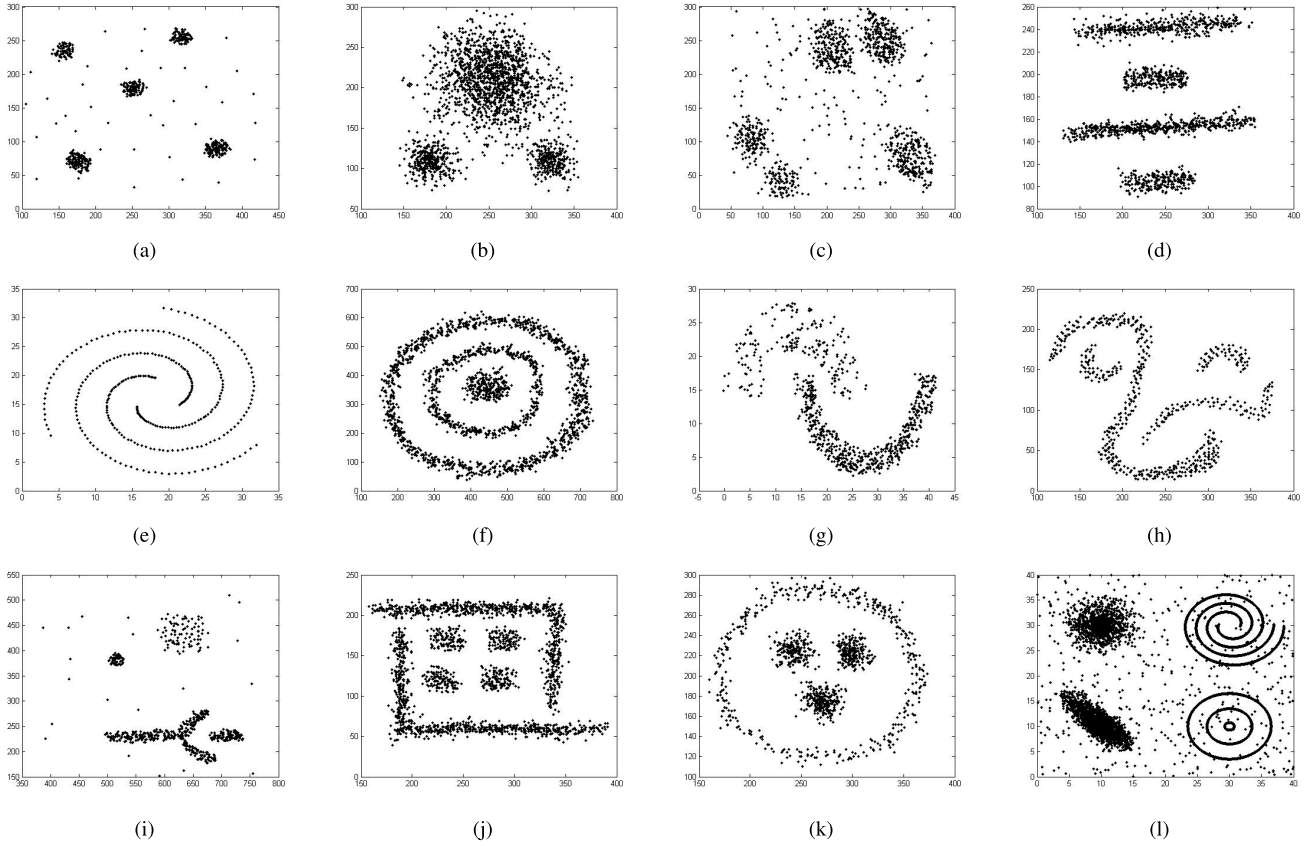


Fig. 4. Synthetic data sets. (a) Data Set 1. (b) Data Set 2. (c) Data Set 3. (d) Data Set 4. (e) Data Set 5. (f) Data Set 6. (g) Data Set 7. (h) Data Set 8. (i) Data Set 9. (j) Data Set 10. (k) Data Set 11. (l) Data Set 12.

Our experiment is conducted on a PC with an i5 Core, 2.8 GHz, 4 GB RAM, Windows 7, and MATLAB R2013a.²

The regarding right cluster number for Data Set 11 is four. The result of using the accuracy index to evaluate the partition of Data Set 11 is shown in Fig. 5(b) and it also demonstrates that when the optimal number of clusters is four, Data Set 11 is correctly partitioned. The experimental results from using the six validity indexes to evaluate clustering results are shown in Fig. 5(c)–(h), in which the red star points are the optimal validity index values. From the results, we learn that the optimal cluster numbers obtained by the CVNN and LCCV indexes are correct. The DB and Silhouette indexes obtain the optimal cluster number 9. The MDL and the CSP indexes obtain the optimal cluster number 3.

We first perform experiments on the original synthetic data sets, and the experimental results from using the six validity indexes to evaluate the optimal cluster numbers are shown in Table II, where NC denotes the correct cluster number. Then, we use the random method³ to reproduce each

²The MATLAB code is available from <https://github.com/DongdongCheng/lccv-source-code>.

³According to the original data set, we use a random method to recalculate the coordinates for each point. For a Data Set A that consists of n d -dimension objects, the j th dimension of i th object is $A(i, j)$. First, we compute the average length of each dimension $avel(j) = (\max(A(:, j)) - \min(A(:, j)))/n$, then, we generate a random number $r \in [-1, 1]$, and finally, $A'(i, j)$ is recalculated as $A'(i, j) = A(i, j) + 2 \times avel(j) \times r$.

TABLE II
OPTIMAL CLUSTERING NUMBERS OF THE
ORIGINAL SYNTHETIC DATA SETS

Datasets	NC	DB	Silhouette	MDL	CVNN	CSP	LCCV
Dataset 1	5	5	5	3	5	3	5
Dataset 2	3	3	3	21	2	2	3
Dataset 3	5	3	3	2	2	3	5
Dataset 4	4	13	14	14	4	4	4
Dataset 5	3	61	30	3	3	3	3
Dataset 6	3	30	31	3	3	3	3
Dataset 7	2	9	2	2	2	2	2
Dataset 8	4	20	20	5	4	4	4
Dataset 9	4	4	6	6	2	3	4
Dataset 10	6	18	17	5	5	5	6
Dataset 11	4	9	9	3	4	3	4
Dataset 12	7	4	6	2	2	2	7

data set 10. For example, taking Data Set 11, the correct numbers of clusters in the 10 experiments are all four. The experimental results are shown in Table III, where ON is the optimal number of clusters obtained by these indexes and accuracy is computed according to (10). The last row of Table III shows the NC correct rate and average accuracy of the results in the 10 experiments, and the NC correct rate is the proportion of the correct ON in the 10 experiments. The NC correct rate and average accuracy of every synthetic data set are listed in Tables IV and V, respectively.

The NC correct rate and clustering accuracy are consistent. The results illustrate that DB and Silhouette indexes are

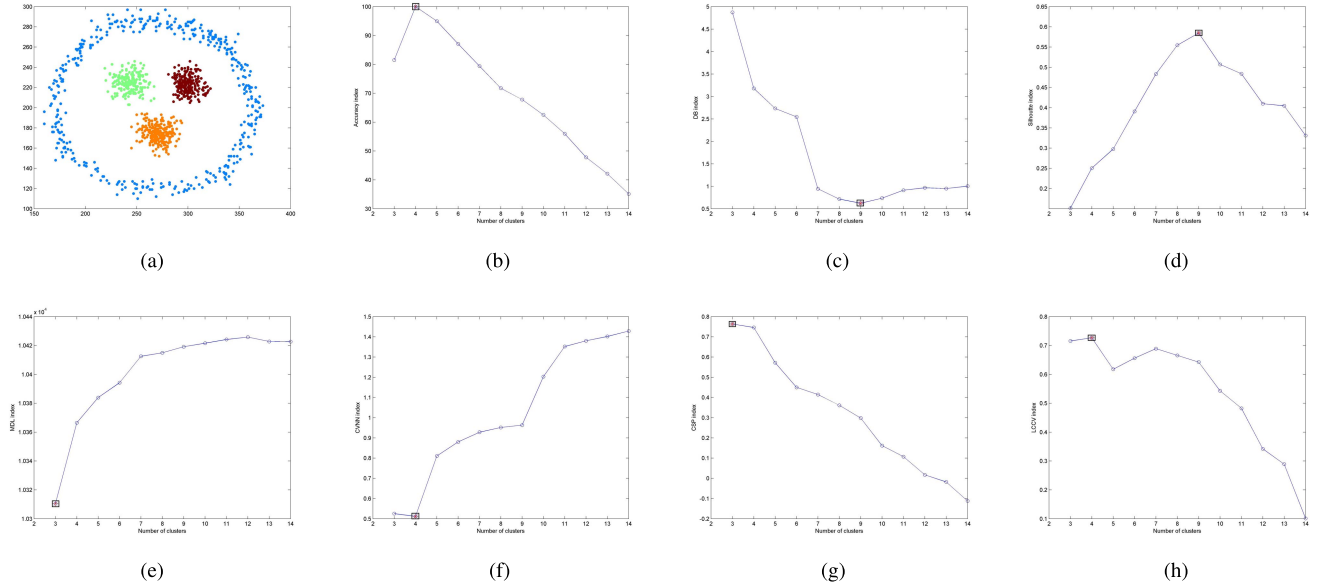


Fig. 5. Comparison of six indexes on Data Set 11. (a) Correct clustering result of Data Set 11. (b) Accuracy with the change of cluster number. (c)–(h) Cluster number–index (DB, Silhouette, MDL, CVNN, CSP, and LCCV) relationship graph of Data Set 11.

TABLE III
OPTIMAL CLUSTER NUMBER, CORRECT RATE OF CLUSTER NUMBER, AND ACCURACY OF DATA SET 11

Dataset	DB		Silhouette		MDL		CVNN		CSP		LCCV	
	ON	Accuracy	ON	Accuracy	ON	Accuracy	ON	Accuracy	ON	Accuracy	ON	Accuracy
Dataset 11_1	9	0.68	9	0.68	3	0.81	4	1.00	3	0.81	4	1.00
Dataset 11_2	21	0.36	7	0.73	3	0.81	4	1.00	3	0.81	4	1.00
Dataset 11_3	11	0.67	11	0.67	4	1.00	4	1.00	4	1.00	4	1.00
Dataset 11_4	9	0.69	9	0.69	3	0.81	4	1.00	3	0.81	4	1.00
Dataset 11_5	9	0.71	9	0.71	3	0.81	4	1.00	3	0.81	4	1.00
Dataset 11_6	10	0.68	10	0.68	3	0.81	4	1.00	3	0.81	4	1.00
Dataset 11_7	10	0.68	10	0.68	4	1.00	4	1.00	4	1.00	4	1.00
Dataset 11_8	9	0.68	9	0.68	4	1.00	4	1.00	4	1.00	4	1.00
Dataset 11_9	14	0.68	14	0.68	4	1.00	4	1.00	4	1.00	4	1.00
Dataset 11_10	10	0.68	10	0.68	3	0.81	4	1.00	3	0.81	4	1.00
	0%	0.65	0%	0.69	40%	0.89	100%	1.00	40%	0.89	100%	1.00

TABLE IV
NC CORRECT RATE OF SYNTHETIC DATA SETS

Datasets	DB	Silhouette	MDL	CVNN	CSP	LCCV
Dataset 1	100%	100%	0%	100%	10%	100%
Dataset 2	100%	100%	0%	0%	40%	100%
Dataset 3	40%	40%	0%	0%	0%	100%
Dataset 4	0%	0%	0%	100%	100%	100%
Dataset 5	0%	0%	40%	70%	100%	100%
Dataset 6	0%	0%	100%	80%	80%	100%
Dataset 7	0%	90%	100%	100%	100%	100%
Dataset 8	0%	0%	60%	90%	100%	100%
Dataset 9	100%	10%	50%	0%	0%	100%
Dataset 10	0%	0%	0%	0%	0%	100%
Dataset 11	0%	0%	40%	100%	40%	100%
Dataset 12	0%	0%	0%	0%	0%	100%

TABLE V
CLUSTERING ACCURACY OF SYNTHETIC DATA SETS

Datasets	DB	Silhouette	MDL	CVNN	CSP	LCCV
Dataset 1	1.00	1.00	0.62	1.00	0.61	1.00
Dataset 2	0.99	0.99	0.46	0.84	0.96	0.99
Dataset 3	0.82	0.82	0.47	0.44	0.57	0.99
Dataset 4	0.41	0.45	0.54	1.00	1.00	1.00
Dataset 5	0.13	0.13	0.86	0.90	1.00	1.00
Dataset 6	0.24	0.25	1.00	0.97	0.97	1.00
Dataset 7	0.49	0.94	1.00	1.00	1.00	1.00
Dataset 8	0.39	0.39	0.97	0.98	1.00	1.00
Dataset 9	1.00	0.66	0.89	0.74	0.85	1.00
Dataset 10	0.42	0.43	0.64	0.61	0.61	1.00
Dataset 11	0.65	0.69	0.89	1.00	0.89	1.00
Dataset 12	0.63	0.83	0.47	0.27	0.27	1.00

effective for evaluating spherical clusters. However, when dealing with data sets containing nonspherical clusters, the DB and Silhouette indexes fail to find the correct number. The MDL index is effective for Data Sets 6 and 7, and the results for Data Sets 5 and 8 are unstable. The CVNN index does good work on Data Sets 1, 4, 7, and 11, and the

performance on Data Sets 5, 6, and 8 is unstable. The CSP index performs well on Data Sets 4, 5, 7, and 8, and the results on Data Set 6 are not always good. The results show that the MDL, CVNN, and CSP indexes are effective for data sets containing nonspherical and well-separated clusters, but they are invalid when processing data sets containing

TABLE VI
TIME OF ALGORITHMS WITH SYNTHETIC DATA SETS

Datasets	DB	Silhouette	MDL	CVNN	CSP	LCCV
Dataset 1	1.23	2.01	1.68	1.05	2.11	1.22
Dataset 2	23.06	39.44	37.26	21.81	41.31	22.37
Dataset 3	5.80	11.04	8.22	5.07	10.93	5.20
Dataset 4	4.09	10.41	6.57	3.47	11.34	4.00
Dataset 5	0.54	1.62	0.76	0.43	1.02	0.45
Dataset 6	10.87	29.23	20.69	9.58	29.53	10.71
Dataset 7	1.84	3.71	3.44	1.54	3.74	1.57
Dataset 8	1.26	2.97	1.86	0.94	2.72	1.09
Dataset 9	1.76	3.61	2.52	1.50	3.10	1.60
Dataset 10	10.17	31.52	15.86	8.05	30.09	8.52
Dataset 11	2.56	5.77	3.99	2.17	6.56	2.58
Dataset 12	499.96	2047.76	1004.28	441.53	1708.97	452.02

TABLE VII
CHARACTERISTICS OF REAL DATA SETS

Datasets	Instances	Attributes	clusters
Wine	178	13	3
Cancer	569	30	2
Control	600	60	6
Breast	699	10	2
Banknote	1372	4	2
Pendigits	7494	16	10

TABLE VIII
OPTIMAL CLUSTERING NUMBERS OF REAL DATA SETS

Datasets	DB	Silhouette	MDL	CVNN	CSP	LCCV
Wine	12	3	2	2	2	3
Cancer	2	2	3	2	2	2
Control	3	3	3	3	3	6
Breast	2	2	3	2	2	2
Banknote	42	42	5	2	2	2
Pendigits	2	14	19	2	2	10

overlapping clusters or a large number of noise points. The LCCV index can obtain the correct optimal cluster number for the 12 synthetic data sets. From Table V, when given the optimal clustering numbers, the accuracy scores of these data sets using the LCCV index are 1 or close to 1, which means that they are correctly partitioned. We can conclude that the LCCV index is more effective for evaluating arbitrary-shaped clusters and more robust than the other indexes are.

The running times, obtained after executing every algorithm 10, are displayed in Table VI. From the table, DB, LCCV, and CVNN require a similar running time. LCCV takes a little longer time than that of CVNN, but it is much faster than Silhouette, MDL, and CSP.

B. Experiments on Real Data Sets

To further demonstrate the effectiveness of the LCCV index, we perform experiments on several benchmarking real data sets from UCI Machine Learning Repository. The characteristics of these data sets are shown in Table VII. The running times are listed in Table X. The experimental results from using validity indexes to evaluate the optimal number of clusters for these real data sets are shown in Tables VIII and IX. From Table VIII, we find that the LCCV index can obtain the correct optimal cluster number for the six real data sets. Other validity indexes are not as effective. From Table IX, the accuracy scores are less than 1, which means that some

TABLE IX
ACCURACY OF OPTIMAL CLUSTERING NUMBERS ON REAL DATA SETS

Datasets	DB	Silhouette	MDL	CVNN	CSP	LCCV
Wine	0.76	0.96	0.64	0.64	0.64	0.96
Cancer	0.78	0.78	0.68	0.78	0.78	0.78
Control	0.50	0.50	0.50	0.50	0.50	0.82
Breast	0.95	0.95	0.88	0.95	0.95	0.95
Banknote	0.13	0.13	0.74	0.90	0.90	0.90
Pendigits	0.21	0.72	0.69	0.21	0.21	0.82

TABLE X
TIME OF THE ALGORITHMS WITH REAL DATA SETS

Datasets	DB	Silhouette	MDL	CVNN	CSP	LCCV
wine	0.20	0.24	0.38	0.17	0.41	0.19
Cancer	1.43	1.90	8.73	1.31	2.36	1.37
Control	1.61	2.03	10.41	1.50	2.49	1.52
Breast	1.32	1.43	2.88	1.27	2.46	1.28
Banknote	11.48	19.91	21.50	10.50	24.65	10.96
Pendigits	403.37	717.06	931.13	408.49	1581.09	414.78

samples are not correctly partitioned. However, the accuracy scores of the LCCV index are higher than or equal to those of other indexes, that is, the clustering qualities of these real data sets obtained by the LCCV index are better than those of other indexes. The results show that the LCCV index is good at evaluating arbitrary-shaped clusters.

C. Applying the LCCV Index to Other Clustering Algorithms

The LCCV index is a commonly designed validity index based on local cores. To extend our method, we combine it with other clustering algorithms to determine the optimal cluster number. The extended algorithm based on LCCV (ELC), can be described in Algorithm 4.

Algorithm 4 ELC

Input: X (the data set)
Output: ocl (The optimal clustering results)
 $(\lambda, LN) = \text{NaN-Searching}(X)$;
for each point x **in** X **do**
 | Compute the density $\rho(x)$ according to Eq. (4);
end
 $(localCores) = \text{LORE}(LN, \rho)$;
Compute the graph-based distance between $localCores$;
for $k = k_{min}$ **to** k_{max} **do**
 | Use a certain clustering algorithm to cluster the
 | Data Set X and obtain the clustering result cl ;
 | Calculate the $lccv(cl)$ value according to Formula (6);
end
 $ocl = \arg \max_{cl} \{lccv(cl)\}$;

According to [20], we set $k_{min} = 2$ and $k_{max} = \sqrt{n}$. We use the ELC algorithm based on K-means and DBSCAN to perform experiments on the above synthetic data sets. For K-means, the value of k is the number of clusters we set. For DBSCAN, it has to set the searching radius Eps and the minimum number of points $Minpts$. In each iteration, we take k as the $Minpts$ and estimate Eps with k according to [43].

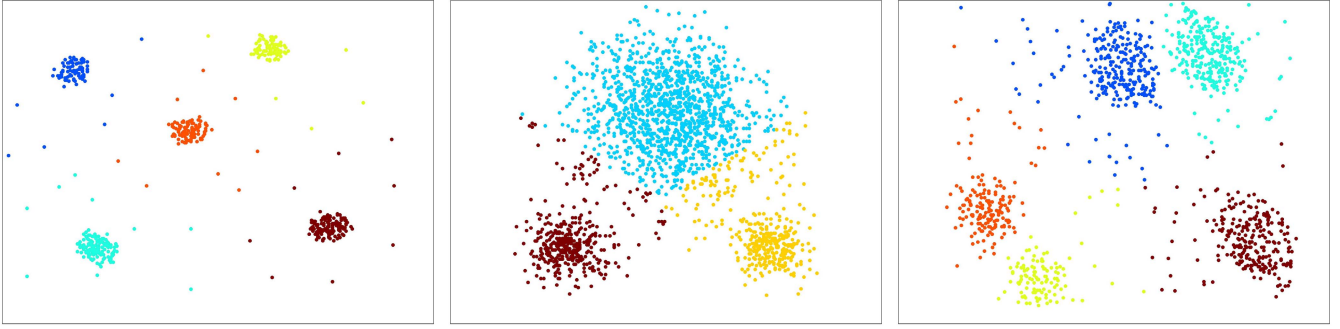


Fig. 6. Clustering results of ELC based on K-means algorithm on the first three data sets.

TABLE XI

RESULTS OF ELC ALGORITHM BASED ON K-MEANS AND DBSCAN

Datasets	NC	K-means		DBSCAN	
		ON	Accuracy	ON	Accuracy
Dataset 1	5	5	1.00	5	0.99
Dataset 2	3	3	0.93	3	0.91
Dataset 3	5	5	0.95	5	1.00
Dataset 4	4	6	0.61	4	0.99
Dataset 5	3	2	0.36	3	1.00
Dataset 6	3	6	0.32	3	1.00
Dataset 7	2	26	0.14	2	0.99
Dataset 8	4	21	0.32	4	1.00
Dataset 9	4	6	0.65	4	1.00
Dataset 10	6	37	0.24	6	1.00
Dataset 11	4	9	0.69	4	1.00
Dataset 12	7	4	0.72	7	0.96

TABLE XII

COMPARISON OF USING ALL POINTS AND LOCAL CORES TO COMPUTE LCCV INDEX

Datasets	NC	ON		Running time (s)	
		All points	Local cores	All Points	Local cores
Dataset 1	5	5	5	8.32	1.22
Dataset 2	3	3	3	154.69	22.37
Dataset 3	5	5	5	43.56	5.20
Dataset 4	4	4	4	53.21	4.00
Dataset 5	3	3	3	4.37	0.45
Dataset 6	3	3	3	141.78	10.71
Dataset 7	2	2	2	19.63	1.57
Dataset 8	4	4	4	13.56	1.09
Dataset 9	4	4	4	11.86	1.60
Dataset 10	6	6	6	116.41	8.52
Dataset 11	4	3	4	32.08	2.06
Dataset 12	7	7	7	4464.48	452.02

The experimental results are shown in Table XI. From Table XI, we can see that the ELC algorithm based on K-means obtains the correct optimal number of clusters for the first three data sets, and the clustering accuracy scores for the first three data sets are all more than 0.9, showing that most points are correctly partitioned. The clustering results of K-means for the first three data sets are displayed in Fig. 6. Since each point is always assigned to its nearest center, K-means cannot address nonspherical clusters. Therefore, it cannot obtain the correct cluster number for the other data sets. The ELC algorithm based on DBSCAN obtains the correct optimal number of clusters for all the data sets. The DBSCAN clustering results on the 12 synthetic data sets are displayed in Fig. 7, and the black points are noises detected by DBSCAN. From the clustering results and accuracy, most of the points are correctly partitioned.

VII. DISCUSSION

To efficiently evaluate the clustering results, we select local cores rather than all points to compute the LCCV index. We perform experiments on synthetic data sets to demonstrate that employing local cores produces the same results as employing all points or even better.

We use all points and local cores to compute the graph-based distance, and then calculate the LCCV index. The results are listed in Table XII, which show that employing local cores can obtain the same number of correct clusters as employing all points for all data sets except for Data Set 11.

For Data Set 11, the local cores retain the structural characteristics of the data set and exclude interference of some boundary points, so using local cores obtains the correct result but using all points does not. The running time of using all points to compute the LCCV index is far longer than that of just using local cores. The experiment shows that it is reasonable to use local cores to represent all points in a data set to compute the LCCV index.

For spherical clusters, the dissimilarity computed with graph-based distance is similar to that computed with the Euclidean distance. Take the data set in Fig. 2, for example, the Euclidean distances between R2 and R3, R4, and R5 are smaller than those between R2 and R1, R6, R7, R8, and R9. A certain number of noise points will not change the fact that the distances between points within a cluster are smaller than those in different clusters. Therefore, the LCCV index is robust for the data set containing spherical clusters. The noise points will rarely connect well-separated clusters to shorten the graph-based distances between points in different clusters. For manifold clusters, the graph-based distance is more applicable than the Euclidean distance for evaluating the dissimilarity between objects. Take a simple data set shown in Fig. 8(a); for example, points A and B are in different clusters, and points A and E are in the same cluster. The Euclidean distance between points A and B is 62, and between A and E is 112, whereas the graph-based distances between points A and B are 307, and between A and E are 117.

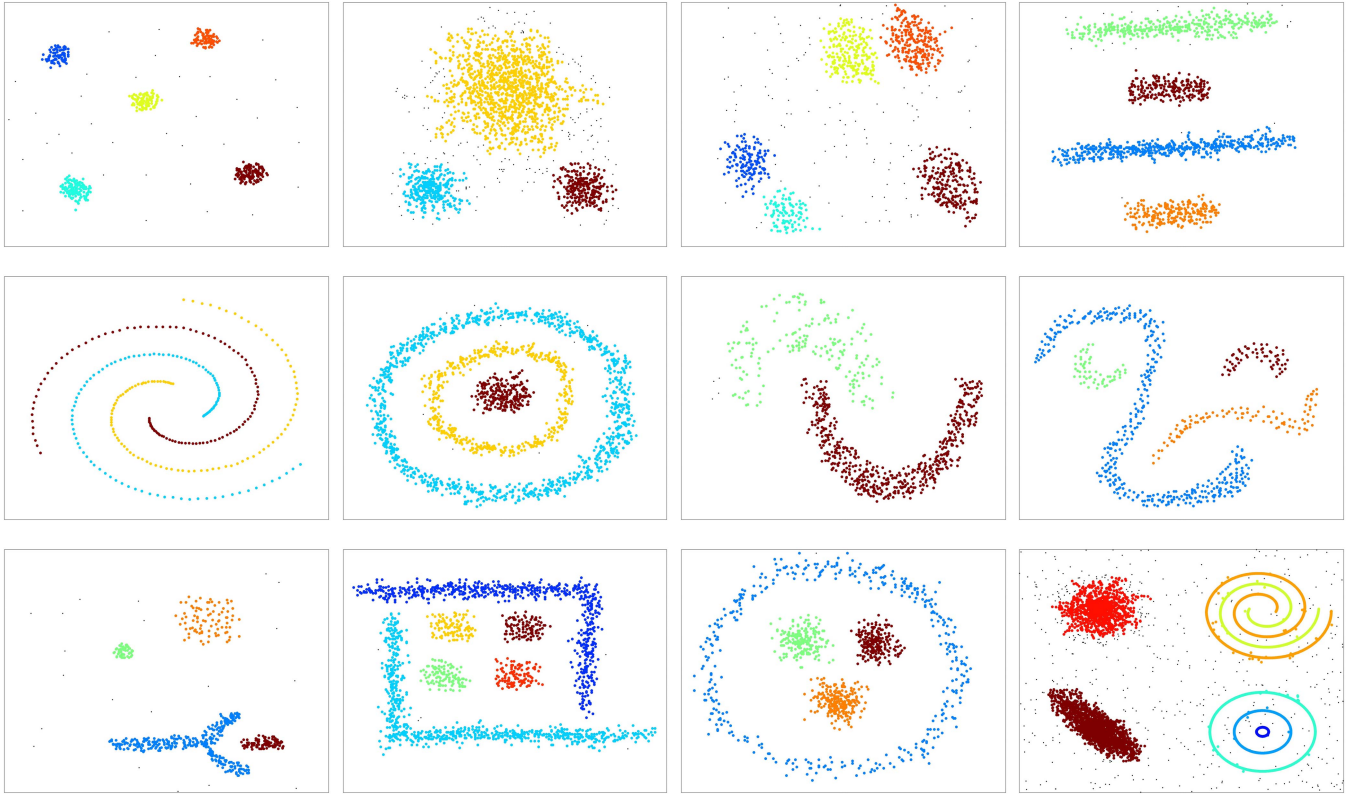


Fig. 7. Clustering results of ELC based on DBSCAN algorithm on synthetic data sets.

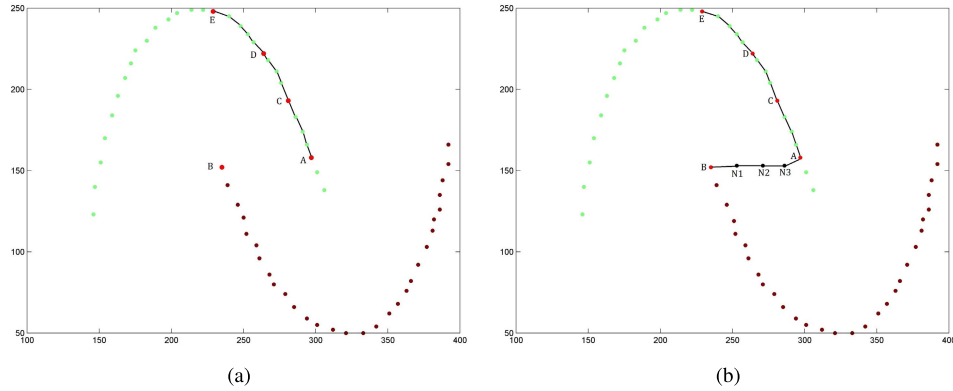


Fig. 8. Impact of noise points on manifold clusters. Graph-based distance (a) between A and B without noises and (b) between A and B with noises.

It is obvious that the graph-based distance is more suitable for measuring the dissimilarity between points A and B . However, when there are noises between points A and B , the situation is different. Just as shown in Fig. 8(b), when there are three noise points ($N1$, $N2$, and $N3$) between A and B , there is a path from B to A , and the graph-based distance between A and B changes to 63, smaller than that between points A and E . The existence of noise points change the relationship between points, that is, the graph-based distance between points in different clusters should be far from each other, but because of the noise points, they may become close to each other. In this case, the LCCV index may fail to obtain the correct cluster number.

To explore the impact of noise points, we perform experiments on six synthetic data sets by adding noise points ranging from 0% to 20%. The six data sets consist of two with spherical clusters (i.e., Data Sets 1 and 2), two with well-separated clusters (i.e., Data Sets 4 and 8), and two with complex structures (i.e., Data Sets 6 and 11). The optimal number of clusters and the accuracy obtained by HC-LCCV are shown in Fig. 9. For data sets with spherical clusters, the number of clusters stays the same with the change of noise points, and the accuracy is larger than 0.97 for Data Set 1 and 0.99 for Data Set 2 indicating that most points are correctly partitioned. For Data Sets 4 and 8, the number of clusters and the accuracy both remain unchanged, and the accuracy

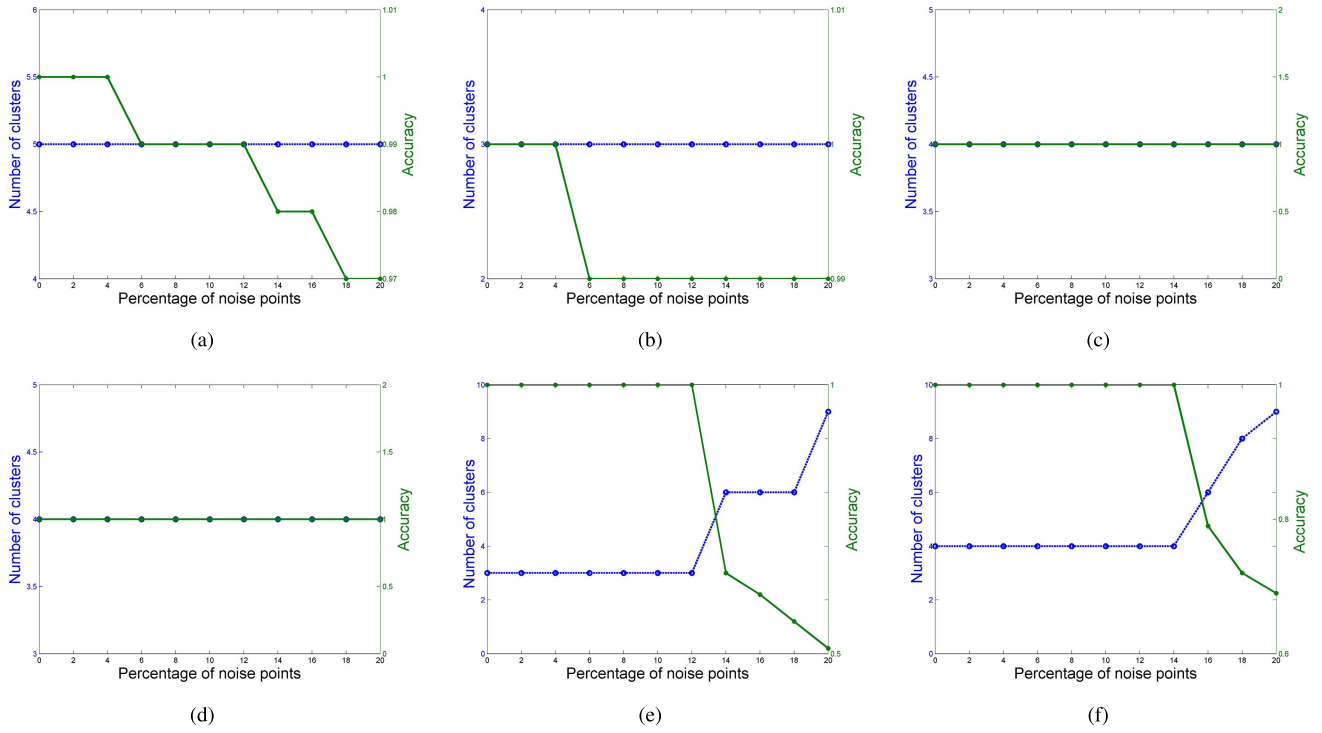


Fig. 9. Cluster number and accuracy with the increase in noise points. (a) Data Set 1. (b) Data Set 2. (c) Data Set 4. (d) Data Set 8. (e) Data Set 6. (f) Data Set 11.

scores are 1, indicating that the clustering results are exactly correct. For Data Sets 6 and 11, when the proportion of noise points is within a certain range (12% for Data Set 6 and 14% for Data Set 11), the LCCV index can obtain the correct optimal cluster number. However, when there are more noise points, the index will fail. From the results, we learn that the LCCV index is robust against data sets containing spherical clusters or well-separated clusters. For data sets with complex structures that have so many noise points that the graph-based distances between points in different clusters are smaller than those in the same cluster, the LCCV index will fail to obtain the correct number of clusters. Other indexes, such as the CVNN and CSP indexes, are also susceptible to noise points. How to solve the problem of noise points in clustering analysis is what we will answer in the future research.

VIII. CONCLUSION

In this paper, we propose a new cluster validity index based on local cores, LCCV. Since we use graph-based distance to evaluate the dissimilarity between local cores, the LCCV index is effective for obtaining the optimal cluster number for data sets containing clusters with arbitrary shapes. To obtain clusters with arbitrary shapes, we use a hierarchical clustering algorithm called HC-LCCV. The LCCV index is proposed to analyze the clustering results and determine the optimal number of clusters. The experimental results on synthetic and real data sets show that the LCCV index can evaluate clustering results effectively and is suitable for determining the number of clusters for data sets containing arbitrary-shaped clusters.

When there are many noise points in a data set with complex structures, the points may change the connections between local cores, and the LCCV index will fail to detect the correct number of clusters. A potential solution is to eliminate noise points before the clustering process. We would like to investigate ways to eliminate the noise points and improve the robustness of our method in our future work.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve this paper.

REFERENCES

- [1] R. Xu and D. Wunsch, II, "Survey of clustering algorithms," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, May 2005.
- [2] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Stat. Probab.*, Oakland, CA, USA, vol. 1, 1967, pp. 281–297.
- [3] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. Hoboken, NJ, USA: Wiley, 2009.
- [4] Y. Wang and L. Chen, "K-MEAP: Multiple exemplars affinity propagation with specified k clusters," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 12, pp. 2670–2682, Dec. 2016.
- [5] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, Jun. 2014.
- [6] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, Feb. 2007.
- [7] D. Cheng, Q. Zhu, J. Huang, L. Yang, and Q. Wu, "Natural neighbor-based clustering algorithm with local representatives," *Knowl.-Based Syst.*, vol. 123, pp. 238–253, May 2017.
- [8] J. Huang, Q. Zhu, L. Yang, D. Cheng, and Q. Wu, "QCC: A novel clustering algorithm based on quasi-cluster centers," *Mach. Learn.*, vol. 106, no. 3, pp. 337–357, 2017.

- [9] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. Int. Conf. Knowl. Discovery Data Mining*, 1996, pp. 226–231.
- [10] I. Gurrutxaga *et al.*, "SEP/COP: An efficient method to find the best partition in hierarchical clustering based on a new cluster validity index," *Pattern Recognit.*, vol. 43, no. 10, pp. 3364–3373, 2010.
- [11] I. Cattinelli, G. Valentini, E. Paulesu, and N. A. Borghese, "A novel approach to the problem of non-uniqueness of the solution in hierarchical clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 7, pp. 1166–1173, Jul. 2013.
- [12] U. Maulik and S. Bandyopadhyay, "Performance evaluation of some clustering algorithms and validity indices," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 12, pp. 1650–1654, Dec. 2002.
- [13] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Commun. Stat., Theory Methods*, vol. 3, no. 1, pp. 1–27, 1974.
- [14] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, no. 2, pp. 224–227, Apr. 1979.
- [15] M. Halkidi, M. Vazirgiannis, and Y. Batistakis, "Quality scheme assessment in the clustering process," in *Principles of Data Mining and Knowledge Discovery* (Lecture Notes in Computer Science), vol. 1910. Berlin, Germany: Springer, 2000, pp. 265–276.
- [16] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, Nov. 1987.
- [17] C. Böhm, C. Plant, J. Shao, and Q. Yang, "Clustering by synchronization," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2010, pp. 583–592.
- [18] S. H. Yue, J. S. Wang, T. Wu, and H. X. Wang, "A new separation measure for improving the effectiveness of validity indices," *Inf. Sci.*, vol. 180, no. 5, pp. 748–764, 2010.
- [19] Y. Liu, Z. Li, H. Xiong, X. Gao, J. Wu, and S. Wu, "Understanding and enhancement of internal clustering validation measures," *IEEE Trans. Cybern.*, vol. 43, no. 3, pp. 982–994, Jun. 2013.
- [20] S. Zhou, Z. Xu, and F. Liu, "Method for determining the optimal number of clusters based on agglomerative hierarchical clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 12, pp. 3007–3017, Dec. 2017.
- [21] W. W. Moss and J. A. Hendrickson, "Numerical taxonomy," *Encyclopedia Astrobiol.*, vol. 18, no. 1, pp. 227–258, 1973.
- [22] B. King, "Step-wise clustering procedures," *J. Amer. Stat. Assoc.*, vol. 62, no. 317, pp. 86–101, 1967.
- [23] A. Guénoche, P. Hansen, and B. Jaumard, "Efficient algorithms for divisive hierarchical clustering with the diameter criterion," *J. Classification*, vol. 8, no. 1, pp. 5–30, 1991.
- [24] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," *ACM SIGMOD Rec.*, vol. 25, pp. 103–114, Jun. 1996.
- [25] C.-R. Lin and M.-S. Chen, "Combining partitionial and hierarchical algorithms for robust and efficient data clustering with cohesion self-merging," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 2, pp. 145–159, Feb. 2005.
- [26] A. Bouguettaya, Q. Yu, X. Liu, X. Zhou, and A. Song, "Efficient agglomerative hierarchical clustering," *Expert Syst. Appl.*, vol. 42, no. 5, pp. 2785–2797, 2015.
- [27] G. Karypis, E.-H. Han, and V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," *Computer*, vol. 32, no. 8, pp. 68–75, Aug. 1999.
- [28] J. C. Dunn, "Well-separated clusters and optimal fuzzy partitions," *J. Cybern.*, vol. 4, no. 1, pp. 95–104, 2008.
- [29] Y. Jung, H. Park, D.-Z. Du, and B. L. Drake, "A decision criterion for the optimal number of clusters in hierarchical clustering," *J. Global Optim.*, vol. 25, no. 1, pp. 91–111, 2003.
- [30] L. F. Chen, Q. S. Jiang, and S. R. Wang, "A hierarchical method for determining the number of clusters," *J. Softw.*, vol. 19, no. 1, pp. 62–72, 2008.
- [31] G. Guo, L. Chen, Y. Ye, and Q. Jiang, "Cluster validation method for determining the number of clusters in categorical sequences," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 12, pp. 2936–2948, Dec. 2017.
- [32] M. B. Gorzalczy and F. Rudziński, "Generalized self-organizing maps for automatic determination of the number of clusters and their multiprototypes in cluster analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 2833–2845, Jul. 2018.
- [33] N. A. Yousri, M. S. Kamel, and M. A. Ismail, "A novel validity measure for clusters of arbitrary shapes and densities," in *Proc. 19th Int. Conf. Pattern Recognit.*, vols. 1–6, 2008, pp. 3454–3457.
- [34] Q. Zhu, J. Feng, and J. Huang, "Natural neighbor: A self-adaptive neighborhood method without parameter K," *Pattern Recognit. Lett.*, vol. 80, pp. 30–36, Sep. 2016.
- [35] G. Wang and Q. Song, "Automatic clustering via outward statistical testing on density metrics," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 8, pp. 1971–1985, Aug. 2016.
- [36] E. Tu, L. Cao, J. Yang, and N. Kasabov, "A novel graph-based k -means for nonlinear manifold clustering and representative selection," *Neurocomputing*, vol. 143, pp. 109–122, Nov. 2014.
- [37] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000.
- [38] P. Yang, Q. Zhu, and B. Huang, "Spectral clustering with density sensitive similarity function," *Knowl.-Based Syst.*, vol. 24, no. 5, pp. 621–628, 2011.
- [39] H. Jia, S. Ding, L. Meng, and S. Fan, "A density-adaptive affinity propagation clustering algorithm based on spectral dimension reduction," *Neural Comput. Appl.*, vol. 25, nos. 7–8, pp. 1557–1567, 2014.
- [40] Q. Zhu, D. Cheng, and J. Huang, "A hierarchical clustering algorithm based on saturated neighbor graph," in *Proc. Int. Conf. Ind. Inform.-Comput. Technol., Intell. Technol., Ind. Inf. Integr.*, 2016, pp. 47–50.
- [41] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, 1975.
- [42] W.-Y. Chen, Y. Song, H. Bai, C.-J. Lin, and E. Y. Chang, "Parallel spectral clustering in distributed systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 3, pp. 568–586, Mar. 2011.
- [43] M. Daszykowski, B. Walczak, and D. L. Massart, "Looking for natural patterns in data: Part 1. Density-based approach," *Chemometrics Intell. Lab. Syst.*, vol. 56, no. 2, pp. 83–92, 2001.



Dongdong Cheng received the bachelor's degree in computer science from Chongqing Normal University, Chongqing, China, in 2013, and the master's degree in computer science from Chongqing University, Chongqing, in 2016, where she is currently pursuing the Ph.D. degree with the Computer Science College.

Her current research interests include clustering analysis and data mining.



Qingsheng Zhu (M'16) was a Visiting Scholar with London University, London, U.K., and a Visiting Professor with the University of Illinois at Chicago, Chicago, IL, USA. He is currently a Professor of computer science with Chongqing University, Chongqing, China, and the Director of the Chongqing Key Laboratory, Software Theory and Technology, Chongqing. His current research interests include service-oriented software engineering, data mining and outlier detection, modeling, and simulating for plant generation.



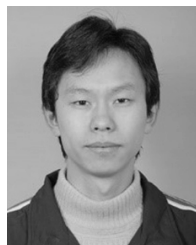
Jinlong Huang received the Ph.D. degree from Chongqing University, Chongqing, China, in 2017.

He is currently a Lecturer with the College of Big Data and Intelligent Engineering, Yangtze Normal University, Chongqing. His current research interests include outlier detection and clustering analysis.



Quanwang Wu received the B.S., M.S., and Ph.D. degrees in computer science from Chongqing University, Chongqing, China, in 2007, 2010, and 2013, respectively.

From 2014 to 2015, he was a Special Researcher with the Digital Content and Media Sciences Research Division, National Institute of Informatics, Tokyo, Japan. He is currently a Lecturer with the Computer College, Chongqing University. His current research interests include service-oriented computing (e.g., service composition and service selection) and cloud computing (e.g., virtual machine consolidation and workflow scheduling).



Lijun Yang received the Ph.D. degree from Chongqing University, Chongqing, China, in 2017.

He is currently a Lecturer with the School of Computer Science and Technology, Southwest Minzu University, Chengdu, China. His current research interests include classification, prototype reduction, and clustering analysis.