A Distance-Type-Insensitive Clustering Approach

Xiaowei Gu [1,2], Plamen Angelov [1,2 *], Zhijin Zhao [3]

[1] School of Computing and Communications, Lancaster University, UK;

[2] Lancaster Intelligent, Robotic and Autonomous Systems Centre (LIRA), Lancaster University, UK;

[3] School of Communication Engineering, Hangzhou Dianzi University, China.

Email: {x.gu3, p.angelov}@lancaster.ac.uk, zhaozj03@hdu.edu.cn

Abstract

In this paper, we offer a method aiming to minimize the role of distance metric used in clustering. It is well known that distance metrics used in clustering algorithms heavily influence the end results and also make the algorithms sensitive to imbalanced attribute/feature scales. To solve these problems, a new clustering algorithm using a per-attribute/feature ranking operating mechanism is proposed in this paper. Ranking is a rarely used discrete, nonlinear operator by other clustering algorithms. However, it also has unique advantages over the dominantly used continuous operators. The proposed algorithm is based on the ranks of the data samples in terms of their spatial separation and is able to provide a more objective clustering result compared with the alternative approaches. Numerical examples on benchmark datasets prove the validity and effectiveness of the proposed concept and principles.

Key words: clustering, distance metric, ranking, spatial separation

1. Introduction

Clustering is an important tool for statistical data analysis [1]. The main goal of clustering is to group homogeneous data samples into clusters [2]. Clustering technique is used for identifying the underlying patterns and multimodal distribution behind the empirically observed data samples and, thus, it is the key to data mining and complex system identification [3].

Clustering algorithms use distances, based on which they estimate the underlying data distribution. However, it is well known that different distance types have different abilities in disclosing the ensemble properties and mutual distribution of the data, and the differences are even more significant in higher dimensional data spaces [4], [5]. Choosing the most suitable distance type for a specific problem is of great importance for a meaningful clustering result. However, this requires a certain degree of *prior* knowledge of the problem itself [6]. Moreover, practically all the clustering algorithms use only one type of distance at one time [7].

In this paper, in order to minimize the influence of the distance in clustering and to obtain a more objective partition, a fundamentally new approach using a novel ranking operation is proposed. It is called Ranking Operation-based Clustering (ROC) algorithm. Instead of using the continuous algorithmic operators that the vast majority of clustering algorithms rely on, the proposed ROC algorithm uses the per-attribute/feature ranking (in terms of the spatial divergence of the data samples) to disclose their ensemble properties. Ranking operation is widely used in our daily life, but clustering approaches avoid using it because ranks are nonlinear, discrete operators [2]. Ranking operation is able to provide sufficient spatial divergence information of the data for clustering; meanwhile, it ignores the unnecessary details that may lead to discrepancy. In comparison with the existing clustering approaches, the ROC algorithm has the following unique advantages:

1) the clustering result is invariant to the type of distance metric used;

2) the clustering result is invariant to the scales of attributes/features.

In addition, it is free from user- and problem- specific parameters and it does not impose any *prior* assumptions of generation model on the empirically observed data. Therefore, the ROC algorithm is able to produce clustering results with higher objectiveness.

* Corresponding author

The remainder of this paper is organized as follows. Section 2 briefly reviews the related works, which provides the background and motivation of this paper. Section 3 presents the main procedure of the proposed ROC algorithm. Numerical examples are given in section 4 for demonstrating the concept. The final section concludes this paper and points out the directions for future work.

2. Critical Analysis of the Related Works

Based on the clustering mechanisms used by the existing approaches, one can generally group them into five general categories [8], [9]:

1) hierarchical clustering algorithms, e.g., agglomerative [10]–[13], divisive [14], [15];

2) centroid-based clustering algorithms, e.g., k-means [16], [17], k-medoids [18];

3) density-based clustering algorithms, e.g., eClustering [19], DBSCAN [20], subtractive [21];

4) distribution-based clustering algorithms, e.g., Gaussian mixture models [22] , mean-shift [23];

5) fuzzy clustering algorithms, e.g., fuzzy c-means [24], [25], fuzzy-possibilistic c-means [26]–[28].

Due to the limited space of this paper, it is impossible to cover all of the existing clustering algorithms. The interested readers are referred to, for example, [8], [29] for more details.

Although, different clustering algorithms operate and behave differently, practically all of them directly or indirectly rely on the use of distances. Hierarchical clustering algorithms use the mutual distances between data samples to build up a dendrogram, and achieve the result by cutting the dendrogram at the desired similarity level. Centroid-based clustering algorithms iteratively minimize an error criterion function, which is formulated based on the distances between the data samples to the respective cluster centres. Density-based, distribution-based and fuzzy clustering algorithms formulate the algorithmic operators, statistical models or criteria based on the mutual distribution information of the data samples estimated using distances. Therefore, the clustering results obtained by these algorithms are highly relevant (dependent) to the types of distance used.

Generally, Euclidean and Mahalanobis types of distance are the most commonly used types by clustering algorithms. However, studies have shown that for many problems, the widely used distance metrics, e.g., Euclidean distance, city block distance and their generalization, Minkowski distance as well as the Mahalanobis distance are less effective [4]. Cosine similarity, despite of being a pseudo-metric, is more frequently used in very high dimensional problems, e.g., natural language processing [30], speech processing [31] because it does not suffer from the so-called “curse of dimensionality” [4], [5]. Other popular similarity measures include Jaccard coefficient, Pearson correlation coefficient [32]. However, similarity measures are not full metrics, and they can hide information and be misleading [33]. There are also hybrid distances proposed for clustering the mixed-type data, namely, the data with interval and categorical variables [34]. However, defining such a distance function is usually very challenging. In short, choosing the best distance is always problem-dependent, and it requires expert knowledge.

Another frequently occurring problem directly related to the effectiveness of distances is the highly imbalanced scales of different attributes/features of the dataset. If some of the attributes/features have much larger scales, they will overwhelmingly squeeze the role of other attributes/features during the calculation of the selected distances. As a result, the imbalances may cause significant difficulties in clustering. To solve this problem, the common practice is to employ attribute/feature re-scaling techniques, e.g., normalization, standardization [3], as pre-processing.

There are also some methods proposed to learn a distance metric from data to boost the performance of clustering or classification [35]–[37]. However, the learning process is computationally expensive due to the iterative search for the optimal solution, and the learnt distance metric is not meaningful for a different problem. *Prior* knowledge about the problem is also required in order to perform the distance metric learning.

Furthermore, it is very difficult to judge the quality of the clustering. There have been many measures proposed for evaluating the quality of clustering results, and the most well-known ones include: Silhouette Coefficient [38], Calinski-Harabasz index [39], Davies-Bouldin index [40], etc. However, they use some types

of distance to measure the spatial separation between data samples and cluster centres, which raises the question of the objectiveness and correctness of these quality measures because they can give very different judgements on the quality of a clustering result with different distance types. It is also possible to objectively measure the quality by using the Rand index [41] or Purity [2], [42] and the ground truth, namely, the actual class labels of the data samples. However, in many real-world applications, ground truth is usually an extremely valuable but scarce resource [43]. Without the ground truth, there is no proper way to check whether clusters are correctly created.

There are two data partitioning algorithms introduced recently that deserve special attentions. One of them is the self-organized direction-aware data partitioning (SODA) algorithm [7]. Unlike other clustering algorithms, the SODA algorithm involves a (linear) combination of a distance metric and a cosine dissimilarity-based component to estimate both, the spatial and angular divergences of the data. Therefore, it utilizes two types of divergence information, namely, spatial and angular, for data partitioning at the same time. The SODA algorithm, firstly, projects data samples onto a number of direction-aware planes based on their spatial and angular divergences. Then, prototypes are identified from the direction-aware planes as the local maxima of the data densities (both, spatial-based and angular-based), and *data clouds* are formed around them partitioning the data space. The other algorithm is the autonomous data partitioning (ADP) [2]. The ADP algorithm introduces for the first time the ranking operation to data partitioning. ADP identifies the prototypes from data by rank-ordering the data samples in terms of their data density value and mutual distances. Then, it iteratively filters out the more meaningful prototypes as the local maxima of global data densities and forms *data clouds* afterwards. The ideas of both, SODA and ADP algorithms are innovative, but they still fail to avoid and minimize the influence of the distances on the partitioning result.

Clustering is an unsupervised machine learning technique for recognising the unknown patterns behind the data in an exploratory manner. As it was stated in Section 1, selecting a proper distance measure for a clustering algorithm is one of the preconditions for obtaining a meaningful result, and this requires *prior* knowledge. However, in many real-world problems, *prior* knowledge is very limited. Learning a distance metric from data is practically impossible with a poor understanding of the problem, and, thus, one has to choose an existing, well-known and easy-to-use distance measure instead. With different types of distance, the clustering results obtained by a clustering algorithm can vary a lot. In addition, involving attribute/feature re-scaling techniques for pre-processing usually results in an entirely new partitioning. It is practically impossible to tell which distance measure or pre-processing technique enables the clustering algorithm to perform the best based on the very limited *prior* knowledge,

To address these issues, a feasible solution is to minimize the role of distance measures in the clustering. Following this concept, we propose the ROC algorithm, which is insensitive to the type of distance metric used as well as to the imbalances in the scales of the data attributes/features thanks to its per-attribute/feature ranking operating mechanism. Moreover, the ROC algorithm is free from user- and problem- specific parameters and requires no *prior* assumptions to be made. The proposed algorithm can be very useful under the circumstances where *prior* knowledge is very limited. These merits show the very strong potential of the ROC algorithm in solving real-world problems.

In the next sections of this paper, the proposed ROC algorithm will be descripted in detail.

3. The ROC Algorithm

3.1. Algorithm Overview

The ROC algorithm starts by applying the ranking operation to each attribute/feature of data for extracting the ensemble properties. Then, it aggregates the information obtained separately from each attribute/feature for estimating the multimodal distribution. In the end, *data clouds* are formed using the local peaks of the multimodal distribution as prototypes to partition the data space. The main procedure of the proposed ROC algorithm is given by Fig.1 in the form of a flowchart.

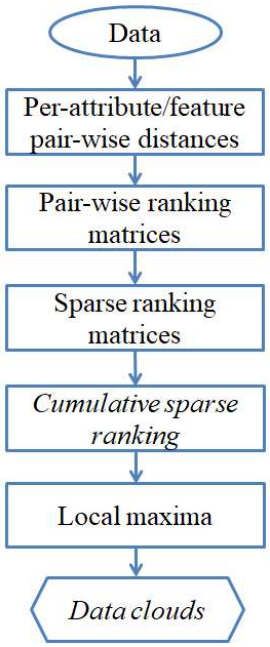


Fig. 1. The main procedure of the Rank Operation-Based Clustering (ROC) algorithm

As it is depicted in Fig. 1, the ROC algorithm performs clustering through the following main steps:

Step 1. Calculate per-attribute/feature pair-wise distances from the data;

Step 2. Rank-order the pair-wise distances and transform the ranking indices into ranking matrices;

Step 3. Filter the ranking matrices and obtain the sparse ranking matrices;

Step 4. Calculate the *cumulative sparse ranking* at each data sample;

Step 5. Identify the local maxima of *cumulative sparse ranking*;

Step 6. Create a Voronoi tessellation with the local maxima and form *data clouds*.

The detailed algorithmic procedure of each step is given in the following sub-section.

3.2. Detailed Algorithmic Procedure

First of all, let us define the static dataset in the N-dimensional real data space $\mathbf{R}^N$ as $\{\boldsymbol{x}\}_K = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_K\}$, $\boldsymbol{x}_k = \left[x_k^1, x_k^2, ..., x_k^N\right]^T \in \mathbf{R}^N$, where the subscript k denotes the time instance at which $\boldsymbol{x}_k$ is observed. The static dataset $\{\boldsymbol{x}\}_K$ can be further expressed by a $N \times K$ dimensional matrix:

$$\mathbf{X}_K = \left[\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_{K-1}, \boldsymbol{x}_K\right] = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_{K-1}^1 & x_K^1 \\ x_1^2 & x_2^2 & \cdots & x_{K-1}^2 & x_K^2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_1^{N-1} & x_2^{N-1} & \cdots & x_{K-1}^{N-1} & x_K^{N-1} \\ x_1^N & x_2^N & \cdots & x_{K-1}^N & x_K^N \end{bmatrix}_{N \times K} \tag{1}$$

Step 1

Considering the i^{th} ($i=1,2,...,N$) attribute/feature of the dataset $\{\boldsymbol{x}\}_K$, namely, the i^{th} row of $\mathbf{X}_K$ (equation (1)) denoted by $\boldsymbol{X}_K^i=\left[x_1^i,x_2^i,\cdots,x_K^i\right]_{1\times K}$, the pair-wise distances between any two different elements of $\boldsymbol{X}_K^i$ can be formulated in the following matrix form:

$$\mathbf{d}^i=\begin{bmatrix} 0 & d_{1,2}^i & \cdots & d_{1,K-1}^i & d_{1,K}^i \\ d_{1,2}^i & 0 & \cdots & d_{2,K-1}^i & d_{2,K}^i \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ d_{1,K-1}^i & d_{2,K-1}^i & \cdots & 0 & d_{K-1,K}^i \\ d_{1,K}^i & d_{2,K}^i & \cdots & d_{K-1,K}^i & 0 \end{bmatrix}_{K\times K} \tag{2}$$

where $d_{k,j}^i=\left\|x_k^i-x_j^i\right\|$ denotes the most widely used Euclidean distance between x_k^i and x_j^i ($k<j$ and $k,j=1,2,...,K$). However, it has to be stressed that $d_{k,j}^i$ can be the distance metric of any type, and the result will not change due to the ranking operation (the theoretical proof will be presented in subsection 3.3). The pair-wise distance matrix, $\mathbf{d}^i$ can be further expressed in a compact form as the following vector:

$$\boldsymbol{d}^i=\left[d_{1,2}^i,d_{1,3}^i,...,d_{1,K}^i,d_{2,3}^i,...,d_{K-2,K}^i,d_{K-1,K}^i\right]_{1\times\frac{(K-1)K}{2}} \tag{3}$$

By applying the same principle to all *N* attributes/features of $\{\boldsymbol{x}\}_K$, one is able to obtain a set of pair-wise distance matrices and the corresponding vectors: $\mathbf{d}^1,\mathbf{d}^2,...,\mathbf{d}^N$ and $\boldsymbol{d}^1,\boldsymbol{d}^2,...,\boldsymbol{d}^N$, respectively.

For example, considering a dataset consisting of four data samples with two attributes/features, $\{\boldsymbol{x}\}_4=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3,\boldsymbol{x}_4\}$ ($\boldsymbol{x}_1=[0,0.8]^T$, $\boldsymbol{x}_2=[0.6,0.2]^T$, $\boldsymbol{x}_3=[0.4,0.7]^T$ and $\boldsymbol{x}_4=[0.9,1.2]^T$), one can obtain two pair-wise distance matrices (one per attribute/feature), $\mathbf{d}^1$ and $\mathbf{d}^2$:

$$\mathbf{d}^1=\begin{bmatrix} 0 & d_{1,2}^1 & d_{1,3}^1 & d_{1,4}^1 \\ d_{1,2}^1 & 0 & d_{2,3}^1 & d_{2,4}^1 \\ d_{1,3}^1 & d_{2,3}^1 & 0 & d_{3,4}^1 \\ d_{1,4}^1 & d_{2,4}^1 & d_{3,4}^1 & 0 \end{bmatrix}=\begin{bmatrix} 0 & 0.6 & 0.4 & 0.9 \\ 0.6 & 0 & 0.2 & 0.3 \\ 0.4 & 0.2 & 0 & 0.5 \\ 0.9 & 0.3 & 0.5 & 0 \end{bmatrix}_{4\times4} \quad \text{and}$$

$$\mathbf{d}^2=\begin{bmatrix} 0 & d_{1,2}^2 & d_{1,3}^2 & d_{1,4}^2 \\ d_{1,2}^2 & 0 & d_{2,3}^2 & d_{2,4}^2 \\ d_{1,3}^2 & d_{2,3}^2 & 0 & d_{3,4}^2 \\ d_{1,4}^2 & d_{2,4}^2 & d_{3,4}^2 & 0 \end{bmatrix}=\begin{bmatrix} 0 & 0.6 & 0.1 & 0.4 \\ 0.6 & 0 & 0.5 & 1.0 \\ 0.1 & 0.5 & 0 & 0.5 \\ 0.4 & 1.0 & 0.5 & 0 \end{bmatrix}_{4\times4},$$

and two pair-wise distance vectors:

$$\boldsymbol{d}^1=\left[d_{1,2}^1,d_{1,3}^1,d_{1,4}^1,d_{2,3}^1,d_{2,4}^1,d_{3,4}^1\right]_{1\times6}=[0.6,\ 0.4,\ 0.9,\ 0.2,\ 0.3,\ 0.5]_{1\times6} \quad \text{and}$$

$$\boldsymbol{d}^2=\left[d_{1,2}^2,d_{1,3}^2,d_{1,4}^2,d_{2,3}^2,d_{2,4}^2,d_{3,4}^2\right]_{1\times6}=[0.6,\ 0.1,\ 0.4,0.5,\ 1.0,\ 0.5]_{1\times6}.$$

Step 2

The elements of the pair-wise distance vector, $\boldsymbol{d}^i$ ($i=1,2,...,N$) are **ranked** in a descending order in terms of their values, and a new vector with the rank-ordered elements is obtained:

$$\hat{\boldsymbol{d}}^i=\left[\hat{d}_1^i,\hat{d}_2^i,...,\hat{d}_{(K-1)K/2}^i\right]_{1\times\frac{(K-1)K}{2}} \tag{4}$$

If there is $d_{k,j}^i=d_{m,n}^i$ ($k\le m$ and $j<n$), $d_{k,j}^i$ is ranked before $d_{m,n}^i$ in $\hat{\boldsymbol{d}}^i$.

Based on the pair-wise distance vector and the corresponding rank-ordered vector, $\boldsymbol{d}^i$ and $\hat{\boldsymbol{d}}^i$, the ranking indices of the elements of $\boldsymbol{d}^i$ can be obtained, which is given by the following ranking index vector, $\boldsymbol{r}^i$:

$$\boldsymbol{r}^i = \left[r_{1,2}^i, r_{1,3}^i, ..., r_{1,K}^i, r_{2,3}^i, ..., r_{K-2,K}^i, r_{K-1,K}^i \right]_{1\times\frac{(K-1)K}{2}} \tag{5}$$

where $r_{k,j}^i$ is the corresponding ranking index of $d_{k,j}^i$ in $\hat{\boldsymbol{d}}^i$; $r_{k,j}^i \in \left\{1,2,3,...,\frac{(K-1)K}{2}\right\}$ and $r_{k,j}^i \neq d_{m,n}^i$ if $k \neq m$ or $j \neq n$.

Further, $\boldsymbol{r}^i$ can be transformed back to a pair-wise ranking matrix in a similar form as equation (2):

$$\mathbf{r}^i = \begin{bmatrix} 0 & r_{1,2}^i & \cdots & r_{1,K-1}^i & r_{1,K}^i \\ r_{1,2}^i & 0 & \cdots & r_{2,K-1}^i & r_{2,K}^i \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ r_{1,K-1}^i & r_{2,K-1}^i & \cdots & 0 & r_{K-1,K}^i \\ r_{1,K}^i & r_{2,K}^i & \cdots & r_{K-1,K}^i & 0 \end{bmatrix}_{K\times K} \tag{6}$$

By applying the same procedure to all the attributes/features, we can finally obtain N different ranking matrices like the above one: $\mathbf{r}^1, \mathbf{r}^2, ..., \mathbf{r}^N$, which are composed of ranking indices (only integers) derived from the pair-wise distance matrices calculated per attribute/feature.

Let us continue the previous example. With $\boldsymbol{d}^1$ and $\boldsymbol{d}^2$, the corresponding rank-ordered vectors $\hat{\boldsymbol{d}}^1$ and $\hat{\boldsymbol{d}}^2$ are expressed as:

$$\hat{\boldsymbol{d}}^1 = \left[\hat{d}_1^1, \hat{d}_2^1, \hat{d}_3^1, \hat{d}_4^1, \hat{d}_5^1, \hat{d}_6^1 \right]_{1\times6} = \left[d_{1,4}^1, d_{1,2}^1, d_{3,4}^1, d_{1,3}^1, d_{2,4}^1, d_{2,3}^1 \right]_{1\times6} = \left[0.9,\ 0.6,\ 0.5,\ 0.4,\ 0.3,\ 0.2\right]_{1\times6} \text{ and}$$

$$\hat{\boldsymbol{d}}^2 = \left[\hat{d}_1^2, \hat{d}_2^2, \hat{d}_3^2, \hat{d}_4^2, \hat{d}_5^2, \hat{d}_6^2 \right]_{1\times6} = \left[d_{2,4}^2, d_{1,2}^2, d_{2,3}^2, d_{3,4}^2, d_{1,4}^2, d_{1,3}^2 \right]_{1\times6} = \left[1.0,\ 0.6,\ 0.5,\ 0.5,\ 0.4,\ 0.1\right]_{1\times6}.$$

Accordingly, the ranking index vectors are obtained as:

$$\boldsymbol{r}^1 = \left[2,\ 4,\ 1,\ 6,\ 5,\ 3\right]_{1\times6} \text{ and } \boldsymbol{r}^2 = \left[2,\ 6,\ 5,\ 3,\ 1,\ 4\right]_{1\times6}.$$

and the pair-wise ranking matrices are given as follows:

$$\mathbf{r}^1 = \begin{bmatrix} 0 & 2 & 4 & 1 \\ 2 & 0 & 6 & 5 \\ 4 & 6 & 0 & 3 \\ 1 & 5 & 3 & 0 \end{bmatrix}_{4\times4} \text{ and } \mathbf{r}^2 = \begin{bmatrix} 0 & 2 & 6 & 5 \\ 2 & 0 & 3 & 1 \\ 6 & 3 & 0 & 4 \\ 5 & 1 & 4 & 0 \end{bmatrix}_{4\times4}.$$

Step 3

Based on the obtained ranking matrices: $\mathbf{r}^1, \mathbf{r}^2, ..., \mathbf{r}^N$, one can calculate the *cumulative ranking* at each data sample denoted by $\gamma(\boldsymbol{x}_k)$ ($k = 1,2,...,K$) using the following equation:

$$\gamma(\boldsymbol{x}_k) = \sum_{j=1}^{K}\sum_{i=1}^{N} \mathbf{r}^i(k,j) \tag{7a}$$

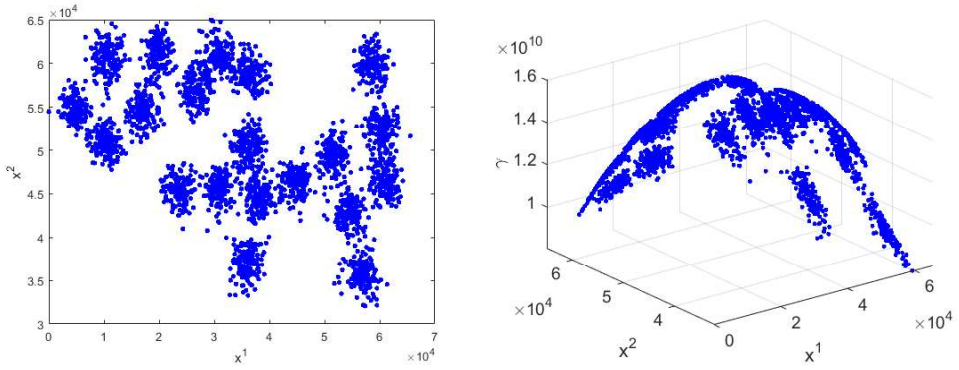
where $\mathbf{r}^i(k,j)$ stands for the element at the k^{th} row, j^{th} column of the ranking matrix, $\mathbf{r}^i$, and there is:

$$\mathbf{r}^i(k,j) = \begin{cases} r_{k,j}^i & k \leq j \\ r_{j,k}^i & k > j \end{cases} \tag{7b}$$

Cumulative ranking aggregates the information of spatial proximity between the data samples estimated from each attribute/feature of the data. However, without using the actual distance values for calculation, it uses the integer ranking indices instead, and, thus, it is able to ignore the unnecessary details. *Cumulative ranking* is closely related to the concept of *cumulative proximity*, $q(\boldsymbol{x}_k)$ [44], [45], which is defined as a measure of mutual positions of the data samples. If city block distance is used, $q(\boldsymbol{x}_k)$ has a very similar form as the *cumulative ranking*:

$$q(\boldsymbol{x}_k) = \sum_{j=1}^{K}\sum_{i=1}^{N}\left\|x_k^i - x_j^i\right\|^2 \tag{8}$$

One may also consider the *cumulative ranking* as a form of *cumulative proximity* by comparing equations (7a) and (8). Similarly, the *cumulative ranking* also results in a unimodal distribution. The data sample having the highest value of $\gamma(\boldsymbol{x}_k)$ is the one that determines the peak of the data distribution. This is because that the sum of the distances between this data sample and all other data samples within the data space are the smallest, and, thus, the sum of the ranking indices between the centre and all other data samples are the largest. Comparatively, the values of the *cumulative ranking* of the data samples that are located at the edge of the data space will be much lower. An example of the *cumulative ranking* of all the data samples is depicted in Fig. 2, where the A1 dataset is used [46].



(a) $\{\boldsymbol{x}\}_K$ (b) $\gamma(\boldsymbol{x})$

Fig. 2. The *cumulative ranking* $\gamma(\boldsymbol{x})$ of A1 dataset

In order to extract the multimodal distribution of the ranking from data, one can transform the ranking matrices $\mathbf{r}^1, \mathbf{r}^2, ..., \mathbf{r}^N$ into sparse matrices $\mathbf{s}^1, \mathbf{s}^2, ..., \mathbf{s}^N$, which has the similar form to equation (6) ($i = 1, 2, ..., N$):

$$\mathbf{s}^i = \begin{bmatrix} \mathbf{s}^i(1,1) & \mathbf{s}^i(1,2) & \cdots & \mathbf{s}^i(1,K) \\ \mathbf{s}^i(2,1) & \mathbf{s}^i(2,2) & \cdots & \mathbf{s}^i(2,K) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{s}^i(K,1) & \mathbf{s}^i(K,2) & \cdots & \mathbf{s}^i(K,K) \end{bmatrix}_{K\times K} \tag{9}$$

and, the transformation is done by a filtering operation using the following rules ($k, j = 1, 2, ..., K$):

$$\mathbf{s}^i(k,j) \leftarrow \mathbf{r}^i(k,j) \quad \textit{if } \mathbf{r}^i(k,j) \geq T_o \tag{10a}$$

$$\mathbf{s}^i(k,j) \leftarrow 0 \quad \textit{if } \mathbf{r}^i(k,j) < T_o \tag{10b}$$

where $\mathbf{s}^i(k,j)$ stands for the element at the k^{th} row, j^{th} column of the $K \times K$ dimensional sparse matrix, $\mathbf{s}^i$; T_o is the threshold controlled by the free parameter, a, which is calculated by the following equation:

$$T_o = \frac{(K-1)K}{2} \cdot a \tag{11}$$

and there is $a \in (0,1)$. Apparently, the higher a is, the sparser $\mathbf{s}^1, \mathbf{s}^2, ..., \mathbf{s}^N$ will be.

Following the previous example, we filter the ranking matrices $\mathbf{r}^1$ and $\mathbf{r}^2$ with the threshold $T_o = 3$, namely, $a = 0.5$, and obtain the sparse ranking matrices as follows:

$$\mathbf{s}^1 = \begin{bmatrix} 0 & 0 & 4 & 0 \\ 0 & 0 & 6 & 5 \\ 4 & 6 & 0 & 3 \\ 0 & 5 & 3 & 0 \end{bmatrix}_{4\times 4} \text{ and } \mathbf{s}^2 = \begin{bmatrix} 0 & 0 & 6 & 5 \\ 0 & 0 & 3 & 0 \\ 6 & 3 & 0 & 4 \\ 5 & 0 & 4 & 0 \end{bmatrix}_{4\times 4}.$$

Step 4

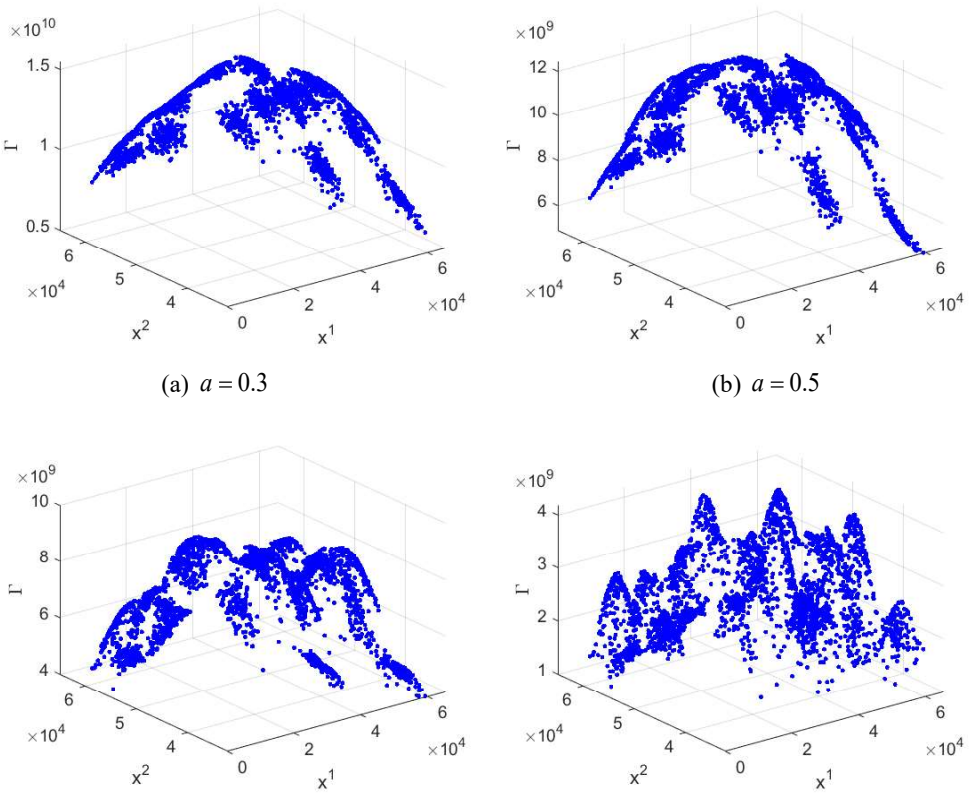
Similarly to equation (7a), one can obtain the *cumulative sparse ranking* at each data sample based on the sparse ranking matrices, $\mathbf{s}^1, \mathbf{s}^2, ..., \mathbf{s}^N$, denoted by $\Gamma(\boldsymbol{x}_k)$ ($k = 1, 2, ..., K$):

$$\Gamma(\boldsymbol{x}_k) = \sum_{j=1}^{K} \sum_{i=1}^{N} \mathbf{s}^i(k,j) \tag{12}$$

Combining equations (10a), (10b) and (12) one can see that the lower a is, the closer $\Gamma(\boldsymbol{x})$ is to $\gamma(\boldsymbol{x})$. While, the higher a is, the closer $\Gamma(\boldsymbol{x})$ is to a zero matrix. In comparison with $\gamma(\boldsymbol{x})$, the *cumulative sparse ranking* is able to disclose the multimodal distribution of the data. An example of the *cumulative sparse ranking* of all the data samples from the A1 dataset [46] is depicted in Fig. 3, where a is set to be $a = 0.3$, 0.5, 0.7 and 0.9, respectively. Comparing Figs. 3(a)-(d), one can conclude that $\Gamma(\boldsymbol{x})$ is closer to a unimodal distribution with a smaller value of a, meanwhile, $\Gamma(\boldsymbol{x})$ is able to give more details of the multimodal distribution with a larger a.

However, it has to be stressed that a is not a user- or problem-specific parameter, and its value can be determined without any *prior* knowledge of the problem. The value of a influences the granularity of the clustering result obtained by the proposed algorithm. The higher a is, the more detailed partitioning the ROC algorithm obtains. This will be demonstrated through numerical examples in section 4.

In the rest of this paper, we use $a = 0.9$ by default except when it is specifically declared otherwise.



(a) $a = 0.3$ (b) $a = 0.5$

(c) $a = 0.7$ (d) $a = 0.9$

Fig. 3. The *cumulative sparse ranking* $\Gamma(\boldsymbol{x})$ of A1 dataset

Step 5

In this step, the local maxima are identified from $\Gamma(\boldsymbol{x})$ in an efficient way, and they will be used as prototypes to partition the data into *data clouds* forming a Voronoi tessellation [47] in the next step. In contrast to the conventionally defined clusters, *data clouds* do not have regular shapes, e.g., hyper-sphere, or hyper-ellipsoid, and pre-defined parameters, but are formed from data samples around the nearest prototypes directly representing the local ensemble properties [48].

Firstly, for each data sample $\boldsymbol{x}_k$ ($k = 1,2,...,K$), we identify the M nearest data samples to it by the following equation:

$$\{\boldsymbol{x}\}_k^* \leftarrow \{\boldsymbol{x}_{n_1}, \boldsymbol{x}_{n_2}, ..., \boldsymbol{x}_{n_M}\}; \quad \{n_1, n_2, ..., n_M\} = \underset{j=1,...,k-1,k+1,...K}{\arg\max} \left(R(\boldsymbol{x}_k, \boldsymbol{x}_j)\right) \tag{13}$$

where $n_1, n_2, ..., n_M$ are the indices of the M nearest data samples; $R(\boldsymbol{x}_k, \boldsymbol{x}_j)$ is the sum of ranking indices of the per-attribute/feature distances between $\boldsymbol{x}_j$ and $\boldsymbol{x}_k$:

$$R(\boldsymbol{x}_k, \boldsymbol{x}_j) = \sum_{i=1}^{N} \mathbf{r}^i(k, j) \tag{14}$$

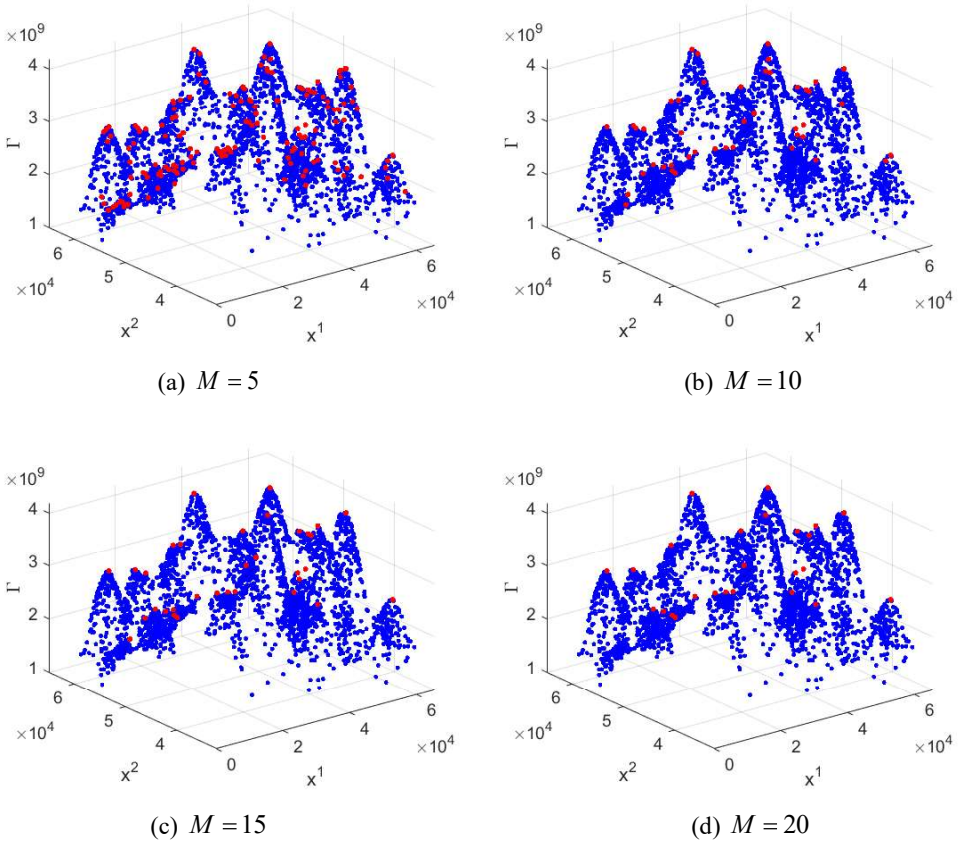
$\mathbf{r}^i(k, j)$ is the element at the k^{th} row, j^{th} column of the ranking matrix $\mathbf{r}^i$, and its value is decided by equation (7b); M is an integer value determining the amount of data samples that are potentially close to $\boldsymbol{x}_k$.

Then, all the local maxima, denoted by $\{\boldsymbol{p}\}$, are identified by the following rule ($k=1,2,...,K$):

$$IF\left(\Gamma(\boldsymbol{x}_k) > \max_{\boldsymbol{x}^* \in \{\boldsymbol{x}\}_k^*}\left(\Gamma(\boldsymbol{x}^*)\right)\right) \quad THEN\left(\{\boldsymbol{p}\} \leftarrow \boldsymbol{x}_k\right) \tag{15}$$

The local maxima identified from the *cumulative sparse ranking* $\Gamma(\boldsymbol{x})$ in the example given by Fig. 3 with $a=0.9$ are depicted in Fig. 4, where the red dots represent the identified local maxima. The value of M is set to be 5, 10, 15 and 20, respectively.

In general, the smaller M is, the more detailed partitioning the proposed ROC algorithm is able to achieve. However, it has to be stressed that M is not a user- and problem- specific parameter. In this paper, we use $M=10$ by default.



(a) $M=5$ (b) $M=10$

(c) $M=15$ (d) $M=20$

Fig. 4. The identified local maxima (red dots) from the *cumulative sparse ranking* $\Gamma(\boldsymbol{x})$

Step 6

Finally, a Voronoi tessellation is created and *data clouds*, denoted by $\{\boldsymbol{C}\}$, are formed by using the identified local maxima $\{\boldsymbol{p}\}$ as their prototypes with the following rule ($k=1,2,...,K$):

$$\boldsymbol{C}_n \leftarrow \boldsymbol{C}_n + \boldsymbol{x}_k; \quad n \leftarrow \arg\max_{\boldsymbol{x}_m \in \{\boldsymbol{p}\}}\left(R(\boldsymbol{x}_k, \boldsymbol{x}_m)\right) \tag{16}$$

where $R(\boldsymbol{x}_k, \boldsymbol{x}_m) = \sum_{i=1}^{N} \mathbf{r}^i(m,k)$; m is the original index of $\boldsymbol{x}_m$ in $\{\boldsymbol{x}\}_K$.

The final clustering result of the example given by Fig. 4(b) is depicted in Fig. 5, where the dots in different colours represent data samples of different *data clouds*; the dots in red are the local maxima.

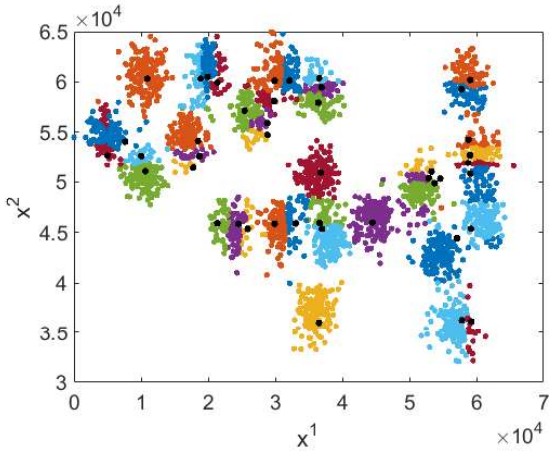


Fig. 5. The clustering results of A1 dataset

3.3. Proof for the Distance-Type-Insensitivity

In this subsection, the proof for the distance-type-insensitivity of per-attribute/feature ranking operation is given.

For a full distance metric, it has to satisfy the non-negativity and subadditivity conditions [49], namely, inequalities (17a) and (17b):

Non-negativity: $d(x,y) \geq 0$ (17a)

Subadditivity: $d(x,y) + d(x,z) \geq d(y,z)$ (17b)

In the proposed ROC algorithm, the pair-wise distance matrix (equation (2)) is calculated on each attribute/feature of the data separately for the ranking operation. Assuming that for the i^{th} attribute/feature of the data, the following inequalities (18a) and (18b) are satisfied for Euclidean distance ($k, j, m, n = 1, 2, ..., K$ and $k \neq m$ or $j \neq n$):

$$d^i_{k,j} = \left\| x^i_k - x^i_j \right\| \geq d^i_{m,n} = \left\| x^i_m - x^i_n \right\| \quad (18a)$$

$$r^i_{k,j} < r^i_{m,n} \quad (18b)$$

One can always find a point denoted by y on the i^{th} dimension of the data space, $\mathbf{R}^N$ that meets the following equation:

$$\left\| x^i_k - x^i_j \right\| = \left\| x^i_m - y \right\| = \left\| x^i_m - x^i_n \right\| + \left\| x^i_n - y \right\| \quad (19a)$$

With any other types of distance metric, the following inequality is also satisfied thanks to the non-negativity and subadditivity conditions (equations (17a) and (17b)):

$$d(x^i_k, x^i_j) = d(x^i_m, y) \geq d(x^i_m, x^i_n) + d(x^i_n, y) \quad (19b)$$

and, thus, the rank indices of $d^i_{k,j}$ and $d^i_{m,n}$ still satisfy the inequality (18b).

This indicates that, the values of elements of $\boldsymbol{d}^i$ may be different if different types of distance metrics are used, but their ranking orders in $\hat{\boldsymbol{d}}^i$ remain the same. Therefore, for any type of distance metric used, the proposed ROC algorithm will always obtain the same ranking sequence, $\boldsymbol{r}^i$.

As the "core" of the ROC algorithm, the per-attribute/feature ranking operation gives the following very important advantages to the algorithm:

1) it is insensitive to the type of distance metrics used;

2) it is insensitive to the imbalanced attribute/feature scales;

and, as a consequence, normalization and standardization are unnecessary for the ROC algorithm.

3.4. Computational Complexity Analysis

The majority of the computations take place during the first three steps of the ROC algorithm. The computational complexity of calculating the pair-wise distance sequences, ranking and filtering for one attribute/feature is $O\left(K^2\right)$. Thus, for the data with N attributes/features, the overall computational complexity of this step is $O\left(N \cdot K^2\right)$. The computational complexity of the remaining steps of the ROC algorithm is linear in regards to the number of data samples in the static data space, namely, $O\left(K\right)$.

Therefore, the overall complexity of the proposed ROC algorithm is $O\left(N \cdot K^2\right)$.

4. Numerical Examples

In this section, numerical examples based on well-known benchmark datasets are presented to demonstrate the general concept and principles of the proposed ROC algorithm.

4.1. Experimental Setting

The following benchmark datasets are involved in the numerical experiments:

1) A1 dataset [46], mentioned earlier;

2) A2 dataset [46];

3) A3 dataset [46];

4) Steel plates faults dataset [50];

5) Cardiotocography dataset [51];

6) Wine quality dataset [52];

7) Multiple feature dataset [53];

8) Optical recognition of handwritten digits dataset [54];

9) Occupancy detection dataset [55].

The details of these datasets are given in Table 1.

Since the proposed ROC algorithm is for offline application, its performance is compared dominantly with the popular offline clustering algorithms. Nonetheless, some well-known evolving algorithms are also involved for a comprehensive comparison. In this study, the following clustering algorithms are used for comparison:

1) DBSCAN algorithm [20];

2) Mean-shift algorithm [23];

3) Subtractive algorithm [21];

4) Affinity propagation algorithm [13];

5) Nonparametric mode identification algorithm [56];

6) Nonparametric mixture model algorithm [57];

7) SODA algorithm (offline version) [7];

8) ADP algorithm (both, offline version and evolving version) [2];

9) eClustering algorithm [58];

10) Evolving local means algorithm [42].

The settings of these algorithms used in the numerical experiments are given in Table 2.

Table 1. Details of the benchmark datasets

Dataset	Number of classes	Number of samples	Number of attributes/features
A1	20	3000	2 +1 class label
A2	35	5250	2+1 class label
A3	50	7500	2+1 class label
Cardiotocography	10	2126	21+1 class label
Steel plates faults	7	1941	27+1 class label
Wine quality [a]	7	6497	11+1 class label
Multiple feature	10	2000	649+1 class label
Optical recognition of handwritten digits	10	5620	62+1 class label
Occupancy detection [b]	2	20560	5+1 class label

[a] Two sub-datasets related to red and white wines are combined;
[b] The time stamps in the original dataset have been removed.

In order to objectively evaluate the quality of the clustering results, the following measures are used:

1) number of *data clouds*/clusters (C), which should be equal to or larger than the number of classes (N_C) in the dataset.

2) Rand index (R) [41], which is used for measuring the accuracy of the clustering results. The Rand index is formulated as [59]:

$$R = 1 + \frac{2\sum_{i=1}^{N_C}\sum_{j=1}^{C} s_{i,j}^2 - \sum_{i=1}^{N_C} s_i^2 - \sum_{j=1}^{C} s_j^2}{(K-1)K} \tag{20}$$

where s_i is the number of data samples of the i^{th} class ($i = 1,2,...,N_C$); s_j is the number of data samples that are grouped in the j^{th} *data cloud*/cluster ($j = 1,2,...,C$); $s_{i,j}$ denotes the number of data samples of the i^{th} class that are grouped in the j^{th} *data cloud*/cluster. The value range of Rand index is $[0,1]$, and the value should be as close to 1 as possible. The higher R is, the better the clustering result is.

3) Purity (P) [2], which is also an index for measuring the accuracy of the clustering results. The Purity of a particular clustering result is calculated by the following equation:

$$P = \frac{\sum_{j=1}^{C} s_j^D}{K} \tag{21}$$

where s_j^D is the number of data samples with the dominant class label in the j^{th} *data cloud*/cluster ($j=1,2,...,C$). The value range of Purity is $[0,1]$, and its value should be as close to 1 as possible. A higher value of Purity indicates that the clustering algorithm shows stronger separation ability.

4) execution time (t_{exe}), which directly indicates the computational efficiency of the algorithms, and its value should be as small as possible.

Table 2. Settings of the comparative algorithms

Algorithm	Free Parameter(s)	Experimental Setting
DBSCAN	1) cluster radius, *r*; 2) minimum number of data samples within the radius, *k*;	1) the value of the knee point of the sorted *k*-dist graph; 2) *k*=4 [20];
MS	1) bandwidth, *p*; 2) kernel function type;	1) *p*=0.15; 2) Gaussian kernel;
Sub	initial cluster radius, *r*;	*r*=0.3 [21];
AP	1) maximum number of iterative refinements; 2) cumulative number of iterations for monitoring the exemplar decisions; 3) dampening factor, λ ;	1) 200 ; 2) 20 ; 3) $\lambda=0.5$ [13];
NMI	1) bandwidth sequence; 2) *prior* data distribution model;	1) $[0.1\hat{\sigma},0.2\hat{\sigma},...,2.0\hat{\sigma}]$; $\hat{\sigma}$ is the maximum value of the standard deviations calculated from each attribute/feature of the data [56]; 2) Gaussian distribution;
NMM	1) *prior* scaling parameter, α ; 2) maximum number of iterative refinements; 3) *prior* data distribution model;	1) $\alpha=1$; 2) 200; 2) Gaussian distribution;
SODA	granularity, *g*;	*g*=6 [7]
ADP	none;	
eClustering	1) learning parameter, ρ ; 2) spread, *r*; 3) membership threshold, ε ;	1) $\rho=0.5$; 2) $r=0.5$; 3) $\varepsilon=e^{-1}$ [58];
ELM	initial cluster radius, *r;*	$r=0.15$ [42] ;

(DBSCAN: DBSCAN algorithm; MS: mean-shift clustering algorithm; Sub: subtractive algorithm; AP: affinity propagation algorithm; NMI: nonparametric mode identification algorithm; NMM: nonparametric mixture model algorithm; SODA: self-organized direction-aware data partitioning algorithm; ADP: autonomous data partitioning algorithm; eClustering: eClustering algorithm; ELM: evolving local means algorithm)

The reason for using the Rand index and Purity as the quality measures in the numerical examples of this paper is that both measures are calculated based on the obtained cluster indices and the ground truth. Most of other well-known quality indicators, e.g., Silhouette Coefficient [38], Calinski-Harabasz index [39], Davies-Bouldin index [40], use some types of distance measure for evaluation.

However, both quality measures share the same deficiency that when the number of *data clouds*/clusters (C) tends to be closer to the number of data samples (N_S) in the dataset, they would give higher values, but, in practice, having almost as many clusters as data samples is highly questionable. To avoid this problem, we further involve another criterion that C should be equal or larger than the number of classes (N_C), but smaller than 5% of N_S, namely,

$$N_c \le C \le 5\% \cdot N_S \tag{22}$$

Otherwise, the clustering results are viewed as invalid ones.

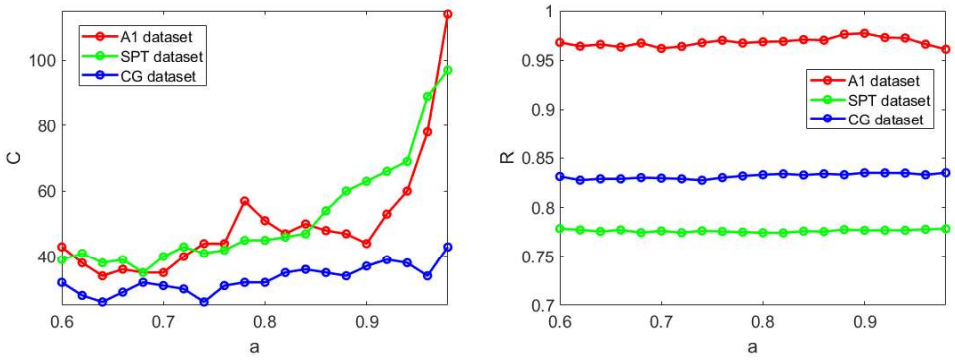
In the numerical examples presented in this section if it is not specifically declared, the two parameters of the ROC algorithm are set as: $a = 0.9$ and $M = 10$. However, we have to stress that a and M are not problem- and user- specific parameters and require no *prior* knowledge to be determined.

4.2. Influence of Parameters *a* and *M*

In this subsection, the influence of the parameters a and M on the performance of the proposed ROC algorithm is investigated. In the following numerical experiments, A1, steel plates faults and cardiotocography datasets are used.

The first numerical experiment studies the influence of a on the clustering results. In this experiment, the value of a changes from 0.4 to 0.98, and the relationship between different values of a and the performance of the ROC algorithm in terms of number of *data clouds* and Rand index are depicted in Fig. 6, where $M = 10$.

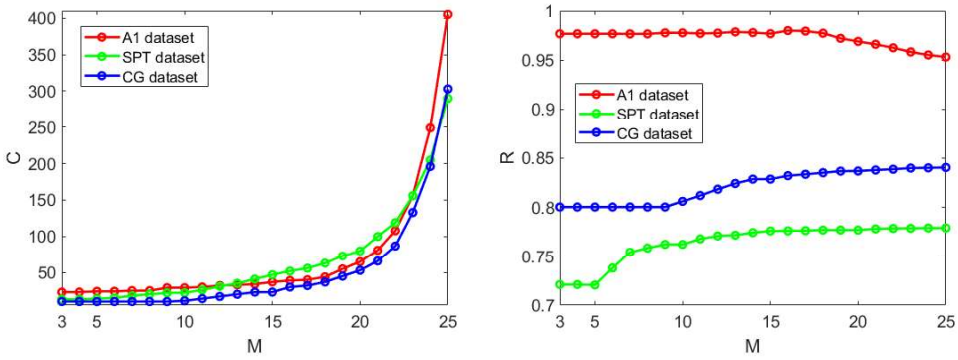
The second numerical experiment investigates the influence of different values of M on the clustering results, where the value of M changes from 3 to 25. The relationship between different values of M and the performance of the ROC algorithm in terms of number of *data clouds* and Rand index are depicted in Fig. 7. In this example, $a = 0.9$.



(a) Number of *data clouds* (b) Rand index

Fig. 6. The influence of different values of a on the clustering results

(C: number of *data clouds*/clusters; R: Rand index; CG: cardiotocography dataset; SPF: steel plates faults dataset)



(a) Number of *data clouds* (b) Rand index

Fig. 7. The influence of different values of M on the clustering results

(C: number of *data clouds*/clusters; R: Rand index; CG: cardiotocography dataset; SPF: steel plates faults dataset)

From Figs. 6 and 7 one can see that different values of a and M (especially, values of $a \leq 0.9$ and $M \leq 20$) have only a limited influence on the quality of the clustering results. However, they influence the level of granularity of the partitions. As it has been stated before, both a and M are not user- and problem-specific parameters and can be defined without *prior* knowledge of the problem.

4.3. Influence of Using Different Distance Types

In this subsection, the influence of the distance types on the clustering is investigated. The A2, A3 and wine quality datasets are used for the numerical experiments. The obtained results by the ROC algorithm are tabulated in Table 3, where the *i)* Euclidean distance (which is the same as the city block distance in 1D space), *ii)* standardized Euclidean distance, *iii)* Minkowski distance (here, we use the L^3 norm) and *iv)* Chebyshev distance are used respectively. We also show the clustering results obtained on the A3 dataset in Fig. 8 as a 2D visualization.

As one can see from the Table 3 and Fig. 8, there is no difference in the clustering results when different types of distance are used, which demonstrates the main advantages of the proposed approach:

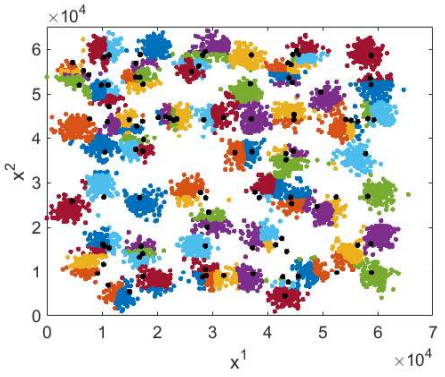
1) the distance-type-invariance and;

2) removing the need for normalizing or standardizing the data.

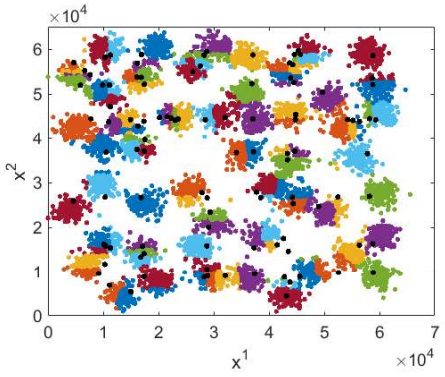
Table 3 Performance comparison using different types of distance

Dataset	Distance	C	R
A2	Euclidean	96	0.9861
	Standardized Euclidean	96	0.9861
	Minkowski	96	0.9861
	Chebyshev	96	0.9861
A3	Euclidean	104	0.9913
	Standardized Euclidean	104	0.9913
	Minkowski	104	0.9913
	Chebyshev	104	0.9913
WQ	Euclidean	161	0.6699
	Standardized Euclidean	161	0.6699
	Minkowski	161	0.6699
	Chebyshev	161	0.6699

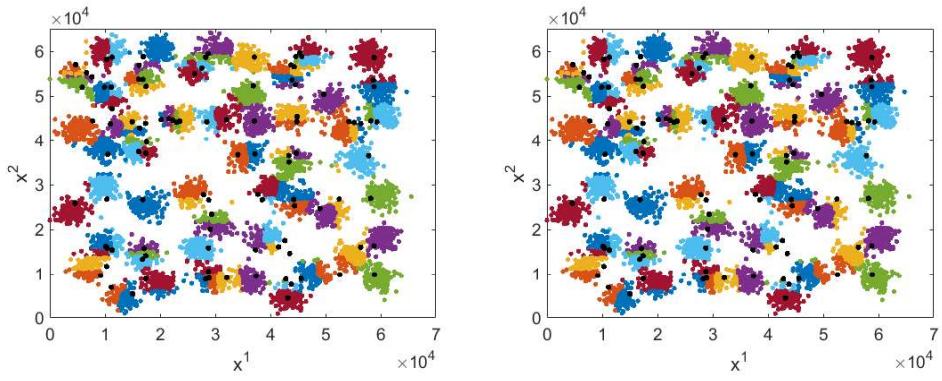
(C: number of *data clouds*/clusters; R: Rand index; WQ: wine quality dataset)



(a) Euclidean distance



(b) Standardized Euclidean distance



(c) Minkowski (L^3 norm) (d) Chebyshev

Fig. 8. Clustering results with different types of distance

4.4. Performance Comparison

In this subsection, we test the performance of the proposed ROC algorithm on benchmark datasets and further compare it with alterative clustering algorithms. For a better evaluation, all the numerical experiments are performed 20 times by randomly re-ordering the data samples, and the clustering quality measures are reported in the form of:

$$mean \pm standard\ \ deviation \tag{23}$$

Firstly, the results obtained by the comparative algorithms on A1, A2, A3, steel plates faults, cardiotocography and wine quality datasets are given in Table 4, where the best clustering results on each benchmark dataset are highlighted, and the invalid results are presented in brackets.

Table 4. Statistical performance comparison on benchmark datasets

Dataset	Algorithm	C	R	P	t_{exe}
A1	ROC	43.35±0.67	0.9775±0.0002	0.9509±0.0022	2.08±0.12
	DBSCAN	26.00±0.00	0.9744±0.0000	0.8288±0.0002	0.84±0.04
	MS	*(2.00±0.00)*	*(0.5230±0.0253)*	*(0.1000±0.0000)*	*(0.10±0.01)*
	Sub	*(9.00±0.00)*	*(0.9031±0.0000)*	*(0.4480±0.0000)*	*(0.35±0.09)*
	AP	*(1373.40±279.19)*	*(0.9565±0.0050)*	*(0.8827±0.0474)*	*(65.22±2.62)*
	NMI	*(7.00±0.00)*	*(0.8650±0.0000)*	*(0.3500±0.0000)*	*(7.41±0.33)*
	NMM	*(4.45±0.89)*	*(0.6861±0.1297)*	*(0.1981±0.0434)*	*(132.14±12.84)*
	SODA	*(9.00±0.00)*	*(0.9156±0.0000)*	*(0.4483±0.0000)*	*(0.46±0.06)*
	OADP	20.00±0.00	**0.9968±0.0000**	**0.9837±0.0000**	**0.37±0.03**
	EADP	*28.65*±3.18	*0.9670*±0.0080	0.7795±0.0601	0.31±0.04
	eClustering	*(*2.70±1.75*)*	*(*0.4669±0.2869*)*	*(*0.1342±0.0856*)*	*(*0.10±0.02*)*
	ELM	*(*1.35±0.59*)*	*(*0.1574±0.1774*)*	*(*0.0675±0.0294*)*	*(*0.23±0.04*)*
A2	ROC	97.70±1.42	**0.9855±0.0006**	0.9537±0.0099	6.29±0.11
	DBSCAN	48.00±0.00	0.9798±0.0000	0.8243±0.0002	**2.47±0.07**
	MS	*(2.50±0.51)*	*(0.5508±0.0293)*	*(0.0682±0.0126)*	*(0.03±0.01)*
	Sub	*(11.00±0.00)*	*(0.9225±0.0000)*	*(0.3143±0.0000)*	*(0.69±0.09)*
	AP	*(2763.10±412.36)*	*(0.9732±0.0017)*	*(0.8659±0.0341)*	*(190.12±2.39)*
	NMI	*(13.00±0.00)*	*(0.9314±0.0000)*	*(0.3714±0.0000)*	*(18.11±0.22)*
	NMM	*(5.80±1.70)*	*(0.7703±0.0738)*	*(0.1483±0.0352)*	*(266.51±34.72)*
	SODA	*(13.00±0.00)*	*(0.9247±0.0000)*	*(0.3707±0.0000)*	*(0.80±0.08)*
	OADP	*(28.00±0.00)*	*(0.9840±0.0000)*	*(0.7895±0.0000)*	*(1.10±0.10)*
	EADP	*(30.35±3.30)*	*(0.9552±0.0086)*	*(0.5225±0.0538)*	*(0.55±0.05)*
	eClustering	*(2.15±1.53)*	*(0.3414±0.3106)*	*(0.0606±0.0412)*	*(0.16±0.03)*

	ELM	*(1.60±0.60)*	*(0.1887±0.1555)*	*(0.0457±0.0171)*	*(0.41±0.07)*
A3	ROC	102.85±0.99	**0.9915±0.0001**	**0.9476±0.0006**	13.05±0.32
	DBSCAN	68.00±*0.00*	0.9841±0.0000	0.8331±0.0001	**4.89±0.16**
	MS	*(4.00±0.00)*	*(0.7487±0.015)*	*(0.0800±0.0000)*	*(0.05±0.01)*
	Sub	*(13.00±0.00)*	*(0.9314±0.0000)*	*(0.2600±0.0000)*	*(1.06±0.08)*
	AP	*(4829.40±556.92)*	*(0.9787±0.0019)*	*(0.8665±0.0429)*	*(381.35±6.17)*
	NMI	*(21.00±0.00)*	*(0.9591±0.0000)*	*(0.4199±0.0000)*	*(30.06±0.50)*
	NMM	*(6.05±1.27)*	*(0.7873±0.0538)*	*(0.1069±0.0237)*	*(384.72±45.23)*
	SODA	*(11.00±0.00)*	*(0.9113±0.0000)*	*(0.2187±0.0000)*	*(1.10±0.07)*
	OADP	*(28.00±0.00)*	*(0.9768±0.0000)*	*(0.5565±0.0000)*	*(2.18±0.05)*
	EADP	*(31.80±3.52)*	*(0.9633±0.0069)*	*(0.4689±0.0506)*	*(0.77±0.05)*
	eClustering	*(2.20±1.01)*	*(0.3813±0.2704)*	*(0.0440±0.0201)*	*(0.23±0.02)*
	ELM	*(1.25±0.44)*	*(0.0947±0.1331)*	*(0.0250±0.0089)*	*(0.52±0.08)*
SPF	ROC	46.30±5.21	**0.7776±0.0013**	**0.6248±0.0115**	9.76±0.05
	DBSCAN	19.00±0.00	0.4894±0.0002	0.4858±0.0000	0.33±0.01
	MS	*(892.00±2.65)*	*(0.7884±0.0001)*	*(0.9075±0.0016)*	*(0.29±0.02)*
	Sub	*(4.00±0.00)*	*(0.5656±0.0000)*	*(0.4147±0.0000)*	*(0.38±0.04)*
	AP	*(1417.95±315.45)*	*(0.7608±0.0292)*	*(0.8563±0.1051)*	*(26.53±0.14)*
	NMI	9.00±0.00	0.3236±0.0000	0.3653±0.0000	68.00±0.46
	NMM	*(2.00±0.00)*	*(0.2223±0.0000)*	*(0.3472±0.0000)*	*(71.83±0.42)*
	SODA	23.00±0.00	0.6849±0.0000	0.5095±0.0000	0.32±0.04
	OADP	14.00±0.00	0.6883±0.0000	0.4498±0.0000	**0.15±0.01**
	EADP	24.75±2.94	0.7336±0.0102	0.4750±0.0137	0.23±0.03
	eClustering	12.05±2.39	0.6579±0.0434	0.4256±0.0190	0.16±0.03
	ELM	*(5.70±1.78)*	*(0.4023 ±0.1148)*	*(0.3758±0.0539)*	*(2.49±1.24)*
CG	ROC	22.95±3.39	0.8252±0.0051	0.5266±0.0244	9.20±0.03
	DBSCAN	15.00±0.00	0.3658±0.0000	0.3166±0.0000	0.40±0.01
	MS	*(508.60±3.55)*	*(0.8381±0.0002)*	*(0.7676±0.0043)*	*(0.22±0.02)*
	Sub	*(165±0.00)*	*(0.8403±0.0000)*	*(0.6961±0.0000)*	*(0.48± 0.04)*
	AP	44.00±0.00	0.8330±0.0000	0.5292±0.0000	11.24±0.06
	NMI	*(321.75±0.79)*	*(0.8265±0.0001)*	*(0.6303±0.0010)*	*(36.00±0.26)*
	NMM	*(3.80±0.41)*	*(0.3773±0.0033)*	*(0.3006±0.0009)*	*(98.80±0.53)*
	SODA	102.00±0.00	0.8163±0.0000	0.5113±0.0000	0.56±0.03
	OADP	90.00±0.00	**0.8342±0.0000**	**0.5691±0.0000**	**0.23±0.01**
	EADP	54.45±3.24	0.8293±0.0014	0.5149±0.0083	0.25±0.04
	eClustering	*(7.45±2.16)*	*(0.7143±0.0640)*	*(0.3671±0.0198)*	*(0.11±0.01)*
	ELM	19.9±6.69	0.4455±0.1039	0.3654± 0.0326	1.45±0.20
WQ	ROC	161.80±9.76	**0.6701±0.0002**	**0.5488±0.0041**	50.40±0.35
	DBSCAN	18.00±0.00	0.3737±0.0000	0.4411±0.0000	3.69±0.03
	MS	12.00±0.00	0.3318±0.0000	0.4388±0.0000	0.76±0.04
	Sub	7.00±0.00	0.6312±0.0000	0.4771±0.0000	1.50±0.08
	AP	*(3031.00±1062.69)*	*(0.6619±0.0080)*	*(0.7083±0.0914)*	*(286.67±1.57)*
	NMI	8.00±0.00	0.4327±0.0000	0.4373±0.0000	513.67±12.09
	NMM	*(2.85±0.37)*	*(0.3287±0.0000)*	*(0.4366±0.0001*	*(245.15±1.45)*
	SODA	61.00±0.00	0.6056±0.0000	0.4562±0.0000	1.21±0.04
	OADP	21.00±0.00	0.6366±0.0000	0.4456±0.0000	1.26±0.02
	EADP	37.35±3.05	0.6542±0.0027	0.4488±0.0023	0.76±0.06
	eClustering	8.95±3.47	0.5922 ±0.0213	0.4387±0.0027	**0.36±0.03**
	ELM	9.10±2.55	0.3790±0.0421	0.4428±0.0055	1.75±0.16

(C: number of *data clouds*/clusters; R: Rand index; P: Purity; t_{exe}: execution time; CG: cardiotocography dataset; SPF: steel plates faults dataset; WQ: wine quality dataset; ROC: ranking operation-based Clustering algorithm; DBSCAN: DBSCAN algorithm; MS: mean-shift clustering algorithm; Sub: subtractive algorithm; AP: affinity propagation algorithm; NMI: nonparametric mode identification algorithm; NMM: nonparametric mixture model algorithm; SODA: self-organized direction-aware data partitioning algorithm; OADP: autonomous data partitioning algorithm (offline version); EADP: autonomous data partitioning algorithm (evolving version); eClustering: eClustering algorithm; ELM: evolving local means algorithm)

As one can see from Table 4, the ROC algorithm is able to outperform the state-of-the-art approaches on the majority of the involved benchmark datasets. The advantages of the proposed algorithm are more obvious on datasets with complex structure.

Meanwhile, one may notice the limitations of the ROC algorithm from the same table that the computational efficiency of the proposed algorithm deteriorates quickly with the increase of the dimensionality and cardinality of the dataset. This is because of the per-attribute/feature ranking operation used in the ROC algorithm. The ROC algorithm needs to calculate the ranking matrix on each attribute/feature of the dataset.

Nonetheless, we have to stress that large-scale datasets are always challenging for the clustering problems. One can involve feature selection techniques to reduce the dimensionality of the data, e.g., principle component analysis (PCA) [60], which can speeds up the computation process.

In the following examples, we use the multiple feature, optical recognition of handwritten digits and occupancy detection datasets to demonstrate the concept. Since the dimensionality and/or cardinality of the benchmark datasets involved are very high, PCA is used for dimensionality reduction. The nine clustering algorithms involved use the ($p+1$) principle components of the data samples, where p corresponds to the first p scores with the sum above 90%, which means that the principle components have contained the most of the spatial information of the data. The results are tabulated in Table 5, where one can see that the proposed ROC algorithm outperforms most of the comparative algorithms.

Table 5. Statistical performance comparison on the principle component analysis (PCA) scores of the large-scale benchmark datasets

Dataset	Algorithm	C	R	P	t_{exe}
MF	ROC	41.00±0.00	**0.9035±0.0000**	**0.6270±0.0000**	0.84±0.02
	DBSCAN	*(4.00±0.00)*	*(0.1702±0.0004)*	*(0.1230±0.0002)*	*(0.36±0.01)*
	MS	*(3.00±0.00)*	*(0.6838±0.0304)*	*(0.2719±0.0205)*	*(0.01±0.00)*
	Sub	*(5.00±0.00)*	*(0.7973±0.0000)*	*(0.3745±0.0000)*	*(0.18±0.06)*
	AP	*(1280.00±361.12)*	*(0.8859±0.0144)*	*(0.8144±0.1025)*	*(29.27±1.22)*
	NMI	*(2.00±0.00)*	*(0.4940±0.0000)*	*(0.1965±0.0000)*	*(16.47±0.42)*
	NMM	*(3.15±0.75)*	*(0.6531±0.1405)*	*(0.2809±0.0553)*	*(74.20±7.88)*
	SODA	*(8.00±0.00)*	*(0.8154±0.0000)*	*(0.4270±0.0000)*	*(0.30±0.03)*
	OADP	20.00±0.00	0.8943±0.0000	0.5650±0.0000	**0.16±0.01**
	EADP	22.80±2.59	0.8815±0.0091	0.5211±0.0353	0.23±0.05
	eClustering	*(3.65±1.46)*	*(0.6407±0.1004)*	*(0.2659±0.0650)*	*(0.08±0.02)*
	ELM	*(1.30±0.47)*	*(0.1690±0.1120)*	*(0.1223±0.0357)*	*(0.18±0.03)*
ORD	ROC	115.00±0.00	0.9072±0.0000	0.8751±0.0000	84.84±19.10
	DBSCAN	*(5.00±0.00)*	*(0.3596±0.0001)*	*(0.2196±0.0000)*	*(2.87±0.14)*
	MS	*(3676.30±5.85)*	*(0.9049±0.0001)*	*(1.0000±0.0000)*	*(11.09±0.51)*
	Sub	*(4447.00±0.00)*	*(0.9003±0.0000)*	*(1.0000±0.0000)*	*(13.82±0.80)*
	AP	195.00±0.00	0.9054±0.0000	0.9721±0.0000	**52.49±1.44**
	NMI	*(5611.00±0.00)*	*(0.9001±0.0000)*	*(1.0000±0.0000)*	*(433.85±18.37)*
	NMM	58.20±3.01	**0.9332±0.0018**	**0.9666±0.0075**	1954.40±89.52
	SODA	*(1782.00±0.00)*	*(0.9111±0.0000)*	*(0.9948±0.0000)*	*(6.95±0.24)*
	OADP	*(286.00±0.00)*	*(0.9111±0.0000)*	*(0.9778±0.0000)*	*(1.76±0.19)*
	EADP	*(248.75±5.60)*	*(0.9078±0.0004)*	*(0.9781±0.0018)*	*(0.80±0.05)*
	eClustering	*(8.80±2.91)*	*(0.8151±0.0572)*	*(0.4252±0.0771*	*(0.44±0.04)*
	ELM	*(8.95±3.28)*	*(0.1944±0.0428)*	*(0.1489±0.0247)*	*(10.92±6.09)*
OD	ROC	665.25±0.79	0.3585±0.0000	**0.9902±0.0000**	325.12±101.05
	DBSCAN	203.00±0.00	0.5635±0.0000	0.8471±0.0000	27.08±1.17
	MS	1.95±0.22	0.6449±0.0001	0.7692±0.0001	**0.16±0.05**
	Sub	4.00±0.00	0.5421±0.0000	0.9074±0.0000	7.40±0.26
	AP	System Crashed			
	NMI	15.00±0.00	**0.8449±0.0000**	0.9685±0.0000	376.66±8.11
	NMM	4.20±1.47	0.7026±0.0120	0.8633±0.0510	903.12±147.35

	SODA	22.00±0.00	0.6723±0.0000	0.9591±0.0000	2.81±0.21
	ADP	18.00±0.00	0.6571±0.0000	0.9873±0.0000	14.45±0.35
	EADP	38.10±3.19	0.4806±0.0179	0.9872±0.0026	2.20±0.09
	eClustering	4.65±2.32	0.6530±0.1357	0.8957±0.0698	0.93±0.04
	ELM	2.30±1.03	0.6658±0.0315	0.7943±0.0355	1.88±0.16

(C: number of *data clouds*/clusters; R: Rand index; P: Purity; t_{exe}: execution time; MF: multiple feature dataset; ORD: optical recognition of handwritten digits dataset; OD: occupancy detection dataset; ROC: ranking operation-based Clustering algorithm; DBSCAN: DBSCAN algorithm; MS: mean-shift clustering algorithm; Sub: subtractive algorithm; AP: affinity propagation algorithm; NMI: nonparametric mode identification algorithm; NMM: nonparametric mixture model algorithm; SODA: self-organized direction-aware data partitioning algorithm; OADP: autonomous data partitioning algorithm (offline version); EADP: autonomous data partitioning algorithm (evolving version); eClustering: eClustering algorithm; ELM: evolving local means algorithm)

5. Conclusion and Future Direction

In order to minimize the role of distance measures in clustering algorithms, this paper proposed a novel clustering algorithm named ROC. Instead of using the smooth, continuous operators that the current clustering algorithms rely on, the proposed ROC algorithm utilizes per-attribute/feature ranking operation in terms of the spatial divergence of the empirically observed data to disclose the ensemble properties, and, further, approximates the real distribution in an objective manner. Ranking operator is rarely used by other approaches, but it has very unique properties compared with the conventional operators. These unique properties give the special advantages to the ROC algorithm surpassing alternative clustering approaches:

1) being insensitive to the type of distance metric that is used;

2) being insensitive to the imbalanced attribute/feature scales;

3) being free from user- and problem- specific parameters;

4) being free from *prior* assumptions on data generation model.

The second point further makes the attribute/feature rescaling techniques such as normalization and standardization unnecessary for the ROC algorithm. These advantages allow the ROC algorithm to produce objective clustering results without the requirement of *prior* knowledge about the problems. Therefore, the ROC algorithm serves as a strong clustering analysis tool for real-world problems where usually only very limited *prior* knowledge is available.

Numerical examples on benchmark datasets have demonstrate that the ROC algorithm consistently outperforms other approaches in terms of the clustering quality measured by the Rand index and Purity. Nonetheless, numerical results also show that the computational efficiency of the ROC algorithm is relatively lower than the state-of-the-art approaches.

As the “core” of the proposed ROC algorithm, the per-attribute/feature ranking operating mechanism is able to minimize the influence of the distance metrics on the clustering results. However, we have to admit that it also brings the deficiency of higher computational complexity. Furthermore, the current ROC algorithm is limited to offline applications because of this operating mechanism. As future work, we will improve the proposed ROC algorithm in three directions:

1) using alternative ranking operations to speed up the computation;

2) developing an approach to recursively update the clustering result with new data samples;

3) developing an online version for streaming data processing.

The first direction allows the ROC algorithm to perform clustering on high-dimensional data efficiently without using dimensionality reduction techniques, which may lead to a subjective result. The second and third directions enable the ROC algorithm to perform clustering on data streams either on the basis of the result primed offline or from “the scratch”. This will make the proposed algorithm a strong data analysis tool in the era of “big data”.

References

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

[2] X. Gu, P. P. Angelov, and J. C. Principe, "A method for autonomous data partitioning," *Inf. Sci. (Ny).*, vol. 460–461, pp. 65–82, 2018.

[3] P. Angelov, *Autonomous learning systems: from data streams to knowledge in real time*. John Wiley & Sons, Ltd., 2012.

[4] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional space," in *International Conference on Database Theory*, 2001, pp. 420–434.

[5] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is 'nearest neighbors' meaningful?," in *International Conference on Database Theoryheory*, 1999, pp. 217–235.

[6] X. Gu and P. P. Angelov, "Self-organising fuzzy logic classifier," *Inf. Sci. (Ny).*, vol. 447, pp. 36–51, 2018.

[7] X. Gu, P. Angelov, D. Kangin, and J. Principe, "Self-organised direction aware data partitioning algorithm," *Inf. Sci. (Ny).*, vol. 423, pp. 80–95, 2018.

[8] O. Maimon and L. Rokach, *Data mining and knowledge discovery handbook*. Springer, Boston, MA, 2005.

[9] V. Estivill-Castro, "Why so many clustering algorithms-a position paper," *ACM SIGKDD Explor. Newsl.*, vol. 4, pp. 65–75, 2002.

[10] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: a new data clustering algorithm and its applications," *Data Min. Knowl. Discov.*, vol. 1, no. 2, pp. 141–182, 1997.

[11] G. Karypis, E.-H. Han, and V. Kumar, "Chameleon: hierarchical clustering using dynamic modeling," *Computer (Long. Beach. Calif).*, vol. 32, no. 8, pp. 68–75, 1999.

[12] W. H. E. Day and H. Edelsbrunner, "Efficient algorithms for agglomerative hierarchical clustering methods," *J. Classif.*, vol. 1, pp. 7–24, 1984.

[13] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science (80-.).*, vol. 315, no. 5814, pp. 972–976, 2007.

[14] A. Gucnoche, P. Hansen, and B. Jaumard, "Efficient algorithms for divisive hierarchical clustering with the diameter criterion," *Joumal Classif.*, vol. 8, pp. 5–30, 1991.

[15] T. Xiong, S. Wang, A. Mayers, and E. Monga, "DHCC: Divisive hierarchical clustering of categorical data," *Data Min. Knowl. Discov.*, vol. 24, pp. 103–135, 2012.

[16] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," *5th Berkeley Symp. Math. Stat. Probab.*, vol. 1, no. 233, pp. 281–297, 1967.

[17] S. Zhong, "Efficient online spherical k-means clustering," in *Proceedings of the International Joint Conference on Neural Networks*, 2005, pp. 3180–3185.

[18] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis (vol.344)*. John Wiley & Sons, 2009.

[19] P. Angelov, "An approach for fuzzy rule-base adaptation using on-line clustering," *Int. J. Approx. Reason.*, vol. 35, no. 3, pp. 275–289, 2004.

[20] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *International Conference on Knowledge Discovery and Data Mining*, 1996, vol. 96, pp. 226–231.

[21] S. L. Chiu, "Fuzzy model identification based on cluster estimation," *J. Intell. Fuzzy Syst.*, vol. 2, no. 3, pp. 267–278, 1994.

[22] A. Corduneanu and C. M. Bishop, "Variational Bayesian model selection for mixture distributions," *Proc. Eighth Int. Conf. Artif. Intell. Stat.*, pp. 27–34, 2001.

[23] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, 2002.

[24] J. C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters," *J. Cybern.*, vol. 3, no. 3, 1973.

[25] J. C. Dunn, "Well-separated clusters and optimal fuzzy partitions," *J. Cybern.*, vol. 4, no. 1, pp. 95–104, 1974.

[26] N. R. Pal, K. Pal, J. M. Keller, and J. C. Bezdek, "A possibilistic fuzzy c-means clustering algorithm," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 4, pp. 517–530, 2005.

[27] E. Rubio, O. Castillo, F. Valdez, P. Melin, C. I. Gonzalez, and G. Martinez, "An extension of the fuzzy possibilistic clustering algorithm using type-2 fuzzy logic techniques," *Adv. Fuzzy Syst.*, vol. 2017, 2017.

[28] E. Rubio, O. Castillo, and P. Melin, "Interval type-2 fuzzy system design based on the interval type-2 fuzzy c-means algorithm," in *Fuzzy Technology*, Springer,Cham, 2016, pp. 133–146.

[29] C. C. Aggarwal and C. K. Reddy, Eds., *Data clustering: algorithms and applications*. CRC press, 2013.

[30] F. A. Allah, W. I. Grosky, and D. Aboutajdine, "Document clustering based on diffusion maps and a comparison of the k-means performances in various spaces," in *IEEE Symposium on Computers and Communications,* 2008, pp. 579–584.

[31] M. Senoussaoui, P. Kenny, P. Dumouchel, and T. Stafylakis, "Efficient iterative mean shift based cosine dissimilarity for multi-recording speaker clustering," *IEEE Int. Conf. Acoust. Speech Signal Process.*, pp. 7712–7715, 2013.

[32] A. Huang, "Similarity measures for text document clustering," in *The Sixth New Zealand Computer Science Research Student Conference*, 2008, pp. 49–56.

[33] X. Gu, P. P. Angelov, D. Kangin, and J. C. Principe, "A new type of distance metric and its use for clustering," *Evol. Syst.*, vol. 8, no. 3, pp. 167–178, 2017.

[34] A. H. Foss, M. Markatou, and B. Ray, "Distance metrics and clustering methods for mixed-type data," *Int. Stat. Rev.*, pp. 1–30, 2018.

[35] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning with application to clustering with side-information," *Adv. Neural Inf. Process. Syst.*, vol. 15, pp. 505–512, 2003.

[36] K. Weinberger, J. Blitzer, and L. Saul, "Distance metric learning for large margin nearest neighbor classification," *Adv. Neural Inf. Process. Syst.*, vol. 18, p. 1473, 2006.

[37] M. G. C. A. Cimino, B. Lazzerini, and F. Marcelloni, "A novel approach to fuzzy clustering based on a dissimilarity relation extracted from data using a TS system," *Pattern Recognit.*, vol. 39, no. 11, pp. 2077–2091, 2006.

[38] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, 1987.

[39] T. Caliński and J. Harabasz, "A dendrite method for cluster analysis," *Commun. Stat. Methods*, vol. 3, no. 1, pp. 1–27, 1974.

[40] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 2, pp. 224–227, 1979.

[41] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *J. Am. Stat. Assoc.*, vol. 66, no. 336, pp. 846–850, 1971.

[42] R. Dutta Baruah and P. Angelov, "Evolving local means method for clustering of streaming data," *IEEE Int. Conf. Fuzzy Syst.*, pp. 10–15, 2012.

[43] M. Canini, W. Li, A. W. Moore, and R. Bolla, "GTVS: boosting the collection of application traffic ground truth," in *International Workshop on Traffic Monitoring and Analysis*, 2009, pp. 54–63.

[44] P. P. Angelov, X. Gu, and J. Principe, "A generalized methodology for data analysis," *IEEE Trans. Cybern.*, vol. 48, no. 10, pp. 2981–2993, 2018.

[45] P. Angelov, "Outside the box: an alternative data analytics framework," *J. Autom. Mob. Robot. Intell. Syst.*, vol. 8, no. 2, pp. 53–59, 2014.

[46] I. Kärkkäinen and P. Fränti, "Dynamic local search algorithm for the clustering problem," 2002.

[47] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial tessellations: concepts and applications of Voronoi diagrams*, 2nd ed. Chichester, England: John Wiley & Sons., 1999.

[48] P. Angelov and R. Yager, "A new type of simplified fuzzy rule-based system," *Int. J. Gen. Syst.*, vol. 41, no. 2, pp. 163–185, 2011.

[49] B. McCune, J. B. Grace, and D. L. Urban, *Analysis of ecological communities*. 2002.

[50] M. Buscema and W. Tastle, "A new meta-classifier," in *Annual Meeting of the North American Fuzzy Information Processing Society*, 2010, pp. 1–7.

[51] D. Ayres-de-Campos, J. Bernardes, A. Garrido, J. Marques-de-Sa, and L. Pereira-Leite, "SisPorto 2.0: A program for automated analysis of cardiotocograms," *J. Matern. Fetal. Med.*, vol. 9, no. 5, pp. 311–318, 2000.

[52] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decis. Support Syst.*, vol. 47, pp. 547–553, 2009.

[53] M. van Breukelen, R. P. W. Duin, D. M. J. Tax, and J. E. den Hartog, "Handwritten digit recognition by combined classifiers," *Kybernetika*, vol. 34, no. 4, pp. 381–386, 1998.

[54] E. Alpaydin and C. Kaynak, "Cascading classifiers," *Kybernetika*, vol. 34, no. 4, pp. 369–374, 1998.

[55] L. M. Candanedo and V. Feldheim, "Accurate occupancy detection of an office room from light, temperature, humidity and CO2 measurements using statistical learning models," *Energy Build.*, vol. 112, pp. 28–39, 2016.

[56] J. Li, S. Ray, and B. G. Lindsay, "A nonparametric statistical approach to clustering via mode identification," *J. Mach. Learn. Res.*, vol. 8, no. 8, pp. 1687–1723, 2007.

[57] D. M. Blei and M. I. Jordan, "Variational inference for Dirichlet process mixtures," *Bayesian Anal.*, vol. 1, no. 1 A, pp. 121–144, 2006.

[58] P. Angelov, P. Sadeghi-Tehran, and R. Ramezani, "An approach to automatic real-time novelty detection, object identification, and tracking in video streams based on recursive density estimation and evolving Takagi–Sugeno fuzzy systems," *Int. J. Intell. Syst.*, vol. 29, no. 2, pp. 1–23, 2014.

[59] J. M. Santos and M. Embrechts, "On the use of the adjusted rand index as a metric for evaluating supervised classification," in *International Conference on Artificial Neural Networks*, 2009, pp. 175–184.

[60] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: Data mining, inference, and prediction*. Burlin: Springer, 2009.