Soft Comput
DOI 10.1007/s00500-017-2590-y

METHODOLOGIES AND APPLICATION

Kernel-based multiobjective clustering algorithm with automatic attribute weighting

Zhiping Zhou[1] · **Shuwei Zhu**[2]

Abstract Clustering algorithms with attribute weighting have gained much attention during the last decade. However, they usually optimize a single-objective function that can be a limitation to cope with different kinds of data, especially those with non-hyper-spherical shapes and/or linearly non-separable patterns. In this paper, the multiobjective optimization approach is introduced into the kernel-based attribute-weighted clustering algorithm, in which two objective functions separately considering the intracluster compactness and intercluster separation are optimized simultaneously. Meanwhile, the sampling operation and efficient clustering ensemble method are incorporated with the projection similarity validity index approach to obtain the clustering solution, which can effectively reduce the computing time especially for large data. Experiments on many data sets demonstrate that, the proposed algorithm in general outperforms the existing attribute-weighted algorithms and the computing efficiency for selection of the final solution is improved by a large margin. Moreover, its merit in terms of the partition and cluster interpretation tools is shown.

Keywords Attribute weighting · Kernel clustering · Multiobjective optimization · Clustering ensemble · Projection similarity validity

1 Introduction

Clustering is a method of creating groups of objects based on similarity degrees of relevant features, which has been used in many areas such as data mining, pattern recognition and machine learning. Generally, the problem of clustering, especially that of the partition-based community, can be posed as an optimization problem, adopting a cluster validity criteria to be optimized that may represent various properties of clusters, such as compactness, separation, and connectivity. However, traditional clustering algorithms usually optimize one cluster validity criteria (cluster compactness), that may only suit a particular structure of datasets. Recently, multiobjective clustering (MOC for short) (Mukhopadhyay et al. 2014) approaches are becoming popular owing to their obvious superiority to capture diverse characteristics of the datasets, among which the multiobjective clustering with automatic k-determination (MOCK for short) (Handl and Knowles 2007) proposed in 2007 is widely regarded as the most classical one. After that, a lot of MOC algorithms were proposed in terms of different aspects, for example, considering the categorical characteristics of the datasets (Saha and Maulik 2014; Yang et al. 2015; Mukhopadhyay et al. 2009), incorporating soft subspace principle (Zhu et al. 2012; Xia et al. 2013), and being used in the field of image segmentation (Mukhopadhyay and Maulik 2011; Sag and Cunkas 2015; Zhao et al. 2015).

Generally, conventional clustering algorithms treat all attributes (or features) equally when computing the distance measurement. This may not be reasonable in some cases, for instance, the cluster structure in the dataset is

Communicated by V. Loia.

✉ Shuwei Zhu
zswjiang@163.com

Zhiping Zhou
zzp@jiangnan.edu.cn

[1] Engineering Research Center of Internet of Things Technology Applications Ministry of Education, Jiangnan University, Wuxi 214122, Jiangsu, China

[2] College of Electronics and Information Engineering, Tongji University, Shanghai 201804, China

Published online: 06 April 2017

subject to a subset of features, or different features (subsets of features) contribute differently on the clustering. Sine the middle of last decade, the attribute weighting based soft subspace clustering algorithms are prevalent in this regard, whose major superiority lies in automatically updating the attribute weights for different classes during the clustering process. However, most existing attribute-weighted clustering algorithms commonly utilize the Euclidean distance as the dissimilarity measure, which may perform poorly for the datasets with more complex construction (i.e., clusters are not hyper-spherical and/or linearly separable). Hence several kinds of clustering methods were proposed to address this problem, among which the kernel-based ones have attracted a lot of attention. Recently, some kernel clustering algorithms with attribute weighting (Ferreira and Carvalho 2014a, b; Ferreira et al. 2016; Shen et al. 2006; Wang et al. 2016) have been proposed, which show obvious superiority over the conventional kernel clustering methods.

In this paper, a novel multiobjective kernel clustering algorithm with automatic attribute weighting (MOKCW for short) is proposed, which simultaneously optimizes two clustering validity criterions considering the intercluster compactness and the intercluster separation, respectively. Experiment results show that the proposed method outperforms the state-of-the-art attribute-weighted clustering algorithms. Hereby, the main contributions of this work could be summarized as follows.

- A novel kernel-based multiobjective clustering approach is proposed. To our best knowledge, this is the first attempt where multiobjective optimization is introduced into attribute-weighted kernel clustering algorithm.
- In terms of the distance between each pair of centers and that between each center and the global center, an effective objective function is defined to measure the intercluster separation.
- To efficiently obtain the final clustering result from the non-dominated solutions, an improved projection similarity validity index (*PSVIndex* for short) method incorporated with clustering ensemble (CE for short) and random sampling is developed.
- Comprehensive results are showed on some benchmark datasets along with detailed analysis, which obviously demonstrate the superiority of the proposed method.

Rest of the paper is organized as follows. In Sect. 2, a brief background about multiobjective clustering, attribute-weighted subspace clustering, and kernel clustering is presented respectively, and also the motivation is given. In Sect. 3, all the details of the proposed algorithm MOKCW are analyzed. In Sect. 4, experimental studies for the performance of MOKCW along with the comparative experimental results are given, followed by the conclusions in Sect. 5.

2 Related work

2.1 Multiobjective clustering

In general, a multiobjective optimization problem optimizes several conflicting objective functions in order to obtain a number of Pareto-optimal solutions. For MOC approaches, they usually perform search over several cluster validity indices simultaneously with some major steps, including encoding the chromosome, developing the clustering objective functions as well as obtaining the final clustering result from the Pareto-optimal solution set. In terms of chromosome encoding policies, Garcia-Piquer et al. (2014) analyzed three most commonly used methods in detail, namely label-based, prototype-based, and graph-based ones, in which the second one was thought to be more applicable to large datasets, for the length of each individual did not rely on the number of instances. In the existing MOC algorithms, most adopt centroids, a type of prototype, to represent the individuals.

It should be noted that the choice of suitable objective functions in MOC methods are very important, since they should be conflicting and beneficial to find the partitional structure of datasets. Typically, the combination of two objective functions can be found in the literature; for example, the overall cluster deviation dev and cluster connectedness $conn$ were utilized in MOCK (Handl and Knowles 2007), CAOS (clustering algorithms based on multiobjective strategies platform) (Garcia-Piquer et al. 2012, 2014) and some other methods (Faceli et al. 2009; Li et al. 2014; Prakash and Singh 2015); two indices J_m and XB (Xie-Beni) in terms of within-cluster and between-cluster information respectively were utilized in Saha and Maulik (2014), Zhu et al. (2012), Saha et al. (2011), Ma et al. (2015), Yang et al. (2011), Zhong et al. (2013); and the fuzzy compactness π and fuzzy separation sep that were similar to J_m and XB were utilized in Yang et al. (2015), Mukhopadhyay et al. (2009), Mukhopadhyay and Maulik (2011), in which sep can measure the intercluster separation more directly than XB. Besides, some other couples of objective functions were also appropriate for MOC approaches, for example, the well-known fuzzy index J_m and an extension of the overlap and separation index (OSI for short) (Capitaine and Frlicot 2011) to be named as OS (overall overlap-separation) (Wikaisuksakul 2014); the S_Dbw validity index (Halkidi and Vazirgiannis 2001) was divided into two terms (Sag and Cunkas 2015), scattering $scat(NC)$ measuring the compactness of the clusters as well as density $Dens\_bw(NC)$ measuring the separation of the clusters; and the non-local spatial information derived from images was introduced into π and sep to formulate two novel fitness functions (Zhao et al. 2015), obtaining more robust image segmentation results. Note that, some researchers have established MOC models that optimize more than two objective functions, especially some methods using archived multi-

objective simulated annealing (AMOSA for short) as the underlying optimization tool in Saha and Bandyopadhyay (2013), Saha et al. (2015, 2016), Alok et al. (2016). However, due to the fact that generally two objective functions can ensure the performance of MOC algorithms, we do not detailedly analyze those with over two objective functions.

As per the nature of the multiobjective optimization algorithms, when the MOC algorithm stops, a set of Pareto-optimal solutions named as P_S are generated, through which the final clustering solution can be achieved. Some techniques have been proposed by different MOC algorithms, among which, three types are prevalent: (1) one internal cluster validity index (CVI for short) is calculated for each solution and then the one with the best value is selected, like that in Mukhopadhyay and Maulik (2011), Zhao et al. (2015), Li et al. (2014), Liu et al. (2015); (2) the semi-supervised methods are used to calculate one external CVI with a fraction of the true class labels as the prior knowledge (Saha et al. 2011, 2013, 2015, 2016; Ma et al. 2015; Wikaisuksakul 2014; Saha and Bandyopadhyay 2013; Alok et al. 2016); (3) the cluster ensemble strategy is employed to integrate all solutions of the P_S and obtain a particular final solution, and related works can be found in Saha and Maulik (2014), Zhu et al. (2012), Coelho et al. (2010), Mukhopadhyay et al. (2013), Benaichouche et al. (2016). Since no prior class information of the real-world data set is got before the clustering operation, the second type cannot be applied to most real problems despite their popularity in the literature. In addition, some other effective methods (Yang et al. 2015; Xia et al. 2013) have been proposed. In this study, an effective and efficient method is proposed by incorporating clustering ensemble into the *PSVIndex* method to obtain the final clustering solution through P_S, thereby reducing the computing time by a large margin, especially when coping with large data sets. It can work with any MOC algorithms if the similarities among objects on relevant dimensions are figured out.

2.2 Attribute-weighted subspace clustering

For most conventional clustering algorithms, attribute weights are regarded to contribute equally during the clustering procedure, but this is not appropriate in many cases especially when coping with high-dimensional datasets. In this regard, several methods have been proposed that automatically assign dissimilar weights to different attributes in terms of their contribution to clustering, one attribute weight can get a large value if the distribution on this dimension is compact. Among them, the weighting k-means (W-k-means for short) with a vector to represent the global weights for all clusters (Huang et al. 2005), and the attribute-weighting clustering algorithm (AWA for short) with a matrix to represent the local weights for different clusters (Chan et al. 2004) are the earlier methods. Afterward, various methods have been proposed, like various kinds of soft subspace clustering (SSC for short) algorithms (Xia et al. 2013; Wang et al. 2016; Jing et al. 2007; Gan and Wu 2008; Gan and Ng 2015), the mixed attribute-weighting algorithm (MWKM for short) (Bai et al. 2011) for high-dimensional categorical data, and the Minkowski metric weighted k-means (MWK-Means for short) (Amorim and Mirkin 2012) that utilizes Minkowski distance to replace the Euclidian distance. Nowadays, the SSC algorithms have been the most popular ones to cope with high-dimensional data, among which the fuzzy subspace clustering (FSC for short) (Gan and Wu 2008) and the entropy weighting k-means (EWKM for short) (Jing et al. 2007) have gained more attention than the others. To overcome the drawback of earlier SSC algorithms only considering within-cluster information, the enhanced soft subspace clustering algorithm (ESSC for short) (Deng et al. 2010) was proposed using between-cluster information to add a new term in the distance measurement. However, a new parameter η balancing the intracluster term and intercluster term was not easily determined in different cases. The objective function of the ESSC method is expressed as Eq. (1).

$$
\begin{aligned}
&J_{\mathrm{ESSC}} = \sum_{i=1}^{n}\sum_{k=1}^{K} u_{ki}^{m} \sum_{j=1}^{d} w_{kj} D_{kj} + \gamma \sum_{k=1}^{K}\sum_{j=1}^{d} w_{kj} \ln w_{kj},\\
&s.t.\ u_{ki} \in [0,1],\ \sum_{i=1}^{n} u_{ki} = 1,\ w_{kj} \in [0,1],\\
&\sum_{j=1}^{d} w_{kj} = 1.
\end{aligned}
\tag{1}
$$

where, $v_{0j} = \frac{\sum_{i=1}^{n} x_{ij}}{n}$ is the global center of the whole dataset and then $D_{kj} = (x_{ij} - v_{kj})^2 - \eta(v_{kj} - v_{0j})^2$ is the enhanced dissimilarity measure.

In Huang et al. (2014a), three k-means-type algorithms were extended by integrating both the intracluster compactness and the intercluster separation, while the latter was designed in the denominator of the objective function so that no new parameter was introduced. Among them, the extension of attribute-weighting clustering algorithm (E-AWA for short) could achieve the best overall performance, whose objective function was expressed as Eq. (2) with the global center v_0 as the same as that of Eq. (1).

$$
\begin{aligned}
&J_{E\text{-AWA}} = \sum_{i=1}^{n}\sum_{k=1}^{K} u_{ki} \sum_{j=1}^{d} w_{kj}^{\beta} \frac{(x_{ij} - v_{kj})^2}{(v_{kj} - v_{0j})^2},\\
&s.t.\ u_{ki} \in \{0,1\},\ \sum_{i=1}^{n} u_{ki} = 1,\ w_{kj} \in [0,1],\\
&\sum_{j=1}^{d} w_{kj} = 1.
\end{aligned}
\tag{2}
$$

Although the two methods ESSC and E-AWA are superior to conventional SSC algorithms, they also have some drawbacks such as trapping into local optimal. A fuzzy MOC algorithm named as MOSSC (multiobjective evolutionary algorithm-based soft subspace clustering) (Zhu et al. 2012), and a crisp MOC algorithm named as MOEASSC (multiobjective evolutionary approach-based soft subspace clustering) (Xia et al. 2013) both simultaneously optimize two clustering criterions, leading to high-quality clustering results. In MOEASSC, two conflicting objective functions J_{In} considering intracluster information and J_{Add} considering intercluster information are formulated as Eqs. (3) and (4).

$$J_{In} = \sum_{i=1}^{n}\sum_{k=1}^{K} u_{ki} \sum_{j=1}^{d} w_{kj}(x_{ij} - v_{kj})^2,$$
$$s.t.\ u_{ki} \in \{0,1\},\ \sum_{i=1}^{n} u_{ki} = 1,\ w_{kj} \in [0,1],$$
$$\sum_{j=1}^{d} w_{kj} = 1. \tag{3}$$

$$J_{Add} = \sum_{i=1}^{K} \left(Aw_i \Big/ (Sep_i + \sigma) + \sum_{j=1}^{d} w_{kj} \log w_{kj}, \right) \tag{4}$$

where

$$Aw_i = \sum_{k=1}^{d} \delta_j w_{kj} \Big/ \sum_{k=1}^{d} \delta_j, \quad \delta_j = \begin{cases} 1, & \text{if } w_{kj} > 1/d \\ 0, & \text{else} \end{cases},$$
$$Sep_i = \sum_{p=1}^{K}\sum_{j=1}^{d} (v_{kj} - v_{pj})^2.$$

Note that, for MOEASSC, the computation equation of attribute weights is not derived according to the well-known expectation maximization (EM) method, like that in Zhu et al. (2012), Jing et al. (2007), Gan and Wu (2008), Deng et al. (2010), Huang et al. (2014a). They are based on the distance of the cluster centers and formulated by Eq. (5). A larger D_{kj} denotes that the kth cluster is easier to be separated in the jth dimension, and w_{kj} gets a larger value accordingly.

$$w_{kj} = \exp(D_{kj}) \Big/ \sum_{l=1}^{d} \exp(D_{kl}), \tag{5}$$

where

$$D_{kj} = \sum_{p=1}^{K} |v_{pj} - v_{kj}|.$$

2.3 Kernel clustering and our motivation

Generally, partition clustering algorithms employ the Euclidean distance as the dissimilarity measure, which performs well for datasets with hyper-spherical and/or linearly separable distribution. However, they may perform poorly if the data structure is more complex (i.e., clusters with non-hyper-spherical shapes and/or linearly non-separable). To capture the nonlinear structure in data, some effective methods are proposed, among which the kernel-based ones have gained a great deal of attention. They may make the data become linearly separable by mapping into a space of high dimension. In the literature, kernel clustering algorithms are under two main approaches: clustering in feature space such as KCM-F and KFCM-F (with F standing for the feature space), in which centers are obtained, and clustering with kernelization of the metric such as KCM-K and KFCM-K (with K standing for the kernelization), in which the distance is computed by means of kernels (Graves and Pedrycz 2010). Note that, for clustering in feature space, there is no need to directly compute the cluster centers as the kernel matrix is computed at the beginning of the program. However, the complexity of computing the matrix is $O(Kn^2d)$, where K, n and d are respectively the number of clusters, objects and attributes, so that the complexity will become too high if the value of n is very large. In order to weigh the importance of different features during the clustering procedure, the weighted fuzzy kernel clustering algorithm (WFKCA for short) was proposed (Shen et al. 2006), where attribute weighting was firstly introduced into kernel clustering method. Recently, some comprehensive works about kernel clustering with automatic attribute weighting can be found in Ferreira and Carvalho (2014a, b), Ferreira et al. (2016). Among them, the hard-type method KCM (Ferreira and Carvalho 2014a; Ferreira et al. 2016) and fuzzy-type method KFCM (Ferreira and Carvalho 2014b) under both situations of the feature space and the kernelization were utilized as the basis methods and various kinds of their enhanced versions with attribute weighting were proposed. According to the experiment analysis of these algorithms, the ones using local adaptive distance are superior to the ones using global adaptive distance in most cases. Regarding that, generally, clustering in feature space is not applicable to large datasets, thereby abandoned by us. Then the methods with kernelization of the metric are merely considered to conduct our research, whose general function with local automatic feature weighting is expressed as follows (Ferreira et al. 2016; Ferreira and Carvalho 2014b).

$$J = \sum_{k=1}^{K}\sum_{i=1}^{n} u_{ki}^{m} \sum_{j=1}^{d} w_{kj}^{\beta} \left\| \varphi(x_{ij}) - \varphi(v_{kj}) \right\|^2, \tag{6}$$

where, m=1 and $u_{ki} \in \{0, 1\}$ for hard clustering; $m > 1$ and $u_{ki} \in [0, 1]$ for fuzzy clustering; attribute weights $w_{kj} (1 \leq k \leq K, \ 1 \leq j \leq d)$ subject to two constraints: (1) $w_{kj} \in [0, 1], \sum_{j=1}^{d} w_{kj} = 1$ and $\beta > 1$, or (2) $w_{kj} > 0, \prod_{j=1}^{d} w_{kj} = 1$ and $\beta = 1$.

Note that, the computations of attribute weights are very different from each other when using the above two constraints, which are not given out here as can be found in Ferreira et al. (2016) concerning hard clustering and Ferreira and Carvalho (2014b) concerning fuzzy clustering. In this paper, kernel-based hard clustering methods are utilized to develop the MOC algorithm. According to Ferreira et al. (2016), the performances of two algorithms named as VKCM-K-GP and VKCM-K-LP showed superiority over the others, and the latter one performed even better. For the method VKCM-K-GP, it subjected to the constraint that the product of the attribute weights on all clusters was equal to one, while the later one VKCM-K-LP having the constraint that the product of the attribute weights on each cluster was equal to one. Additionally, the fuzzy kernel-based clustering algorithm VKFCM-K-LP, which can be seen as the fuzzy version of VKCM-K-LP, outperformed the others in Ferreira and Carvalho (2014b).

It is known that the between-cluster information is important for finding the true partition as well as the within-cluster information. Recently, several enhanced clustering algorithms were proposed by adopting the intercluster separation in the design of the objective functions to further improve their performances, including the ESSC method (Deng et al. 2010), the E-AWA method (Huang et al. 2014a) as well as some other methods (Wu et al. 2005; Bai et al. 2013; Bai and Liang 2014; Ji and Wang 2014; Huang et al. 2014b). It deserves noticing that new parameters, whose values are hard to be set in different cases, should be introduced in these methods. However, to our best knowledge, no method has considered the between-cluster information among the existing attribute-weighted kernel clustering algorithms, which may be a limitation to their performance. In view of the obvious superiority of MOC methods over single-objective ones that can achieve more accurate and stable results with no new parameters, a novel kernel-based clustering algorithm with attribute weighting under the multiobjective optimization approach, named as MOKCW, is proposed in this paper. This method simultaneously optimizes two separate objective functions considering within-cluster and between-cluster information, respectively.

3 Kernel-based multiobjective clustering with attribute weighting

3.1 Multiobjective optimization

In the view of the fact that most partition clustering algorithms can achieve the results by minimizing the objective functions, here we merely consider minimization problems with respect to multiobjective optimization. Assuming that a vector in the decision variable space of d dimensions is $\mathbf{x} = [x_1, x_2, \ldots, x_d]^{\mathrm{T}}$, then the form of multiobjective optimization problems that optimizes m fitness functions can be defined as following (Handl and Knowles 2007; Mukhopadhyay et al. 2009).

$$\begin{aligned} &\min \ y = F(\mathbf{x}) = [f_1(\mathbf{x}), \ f_2(\mathbf{x}), \ \ldots, \ f_m(\mathbf{x})]^{\mathrm{T}}, \\ &s.t. \ \begin{cases} g_i(\mathbf{x}) \leq 0, \ i = 1, 2, \ldots, p, \\ h_j(\mathbf{x}) = 0, \ j = 1, 2, \ldots, q, \end{cases} \end{aligned} \tag{7}$$

where, $g_i(\mathbf{x}) \leq 0, \ i = 1, 2, \ldots, p$ are p inequality constraints, and $h_j(\mathbf{x}) = 0, \ j = 1, 2, \ldots, q$ are q equality constraints.

Assuming two variables $\mathbf{u} = [u_1, u_2, \ldots, u_d]^{\mathrm{T}}$ and $\mathbf{v} = [v_1, v_2, \ldots, v_d]^{\mathrm{T}}$ in the d dimension space, if and only if: $\forall i \in \{1, \ldots, d\}, \ F(u_i) \leq F(v_i) \ \wedge \ \exists j \in \{1, \ldots, d\}, \ F(u_j) < F(v_j)$, then it can be said that $\mathbf{u}$ Pareto dominates $\mathbf{v}$, namely $\mathbf{u} \prec \mathbf{v}$. If there exists no vector $\mathbf{x}$ such that $\mathbf{x} \prec \mathbf{x}^*$, $\mathbf{x}^*$ is called non-dominated solution, and a set named as P_S is comprised of all $\mathbf{x}^*$, whose fitness values constitute the set named as P_F. In this study, the well-known non-dominated sorting genetic algorithm-Π (NSGA-Π for short) (Deb et al. 2002) is used as the underlying optimization tool, which can effectively cope with various kinds of multiobjective optimization problems. It consists of some steps such as: chromosome representation; selection, crossover, mutation operations; the non-dominated sorting with crowding distance estimation; and the elitism strategy which is typically distinct from other multiobjective optimization algorithms. However, it deserves noting that, MOKCW uses the different solution selection of Pareto-optimal front and mutation step as described in Sect. 3.4.

3.2 Objective functions

According to Sect. 2.1, the objective functions are important to MOC, here J_c and F_s are respectively formulated measuring the compactness and dispersion of the data partition. In terms of the first index, namely compactness, we employ the objective function expressed as Eq. (6), and VKCM-K-

LP is the best algorithm using this function, and hence the constraint that the product of the attribute weights on each cluster is equal to one with β=1. Note that m=1 is set for hard clustering, the first function J_c is shown as Eq. (8), the same to that of VKCM-K-LP, where n is the number of objects, d is the number of attributes, and K is the number of clusters known before clustering.

$$J_{\mathrm{c}} = \sum_{k=1}^{K}\sum_{i=1}^{n} u_{ki} \sum_{j=1}^{d} w_{kj} \left\|\varphi(x_{ij}) - \varphi(v_{kj})\right\|^2,$$
$$s.t.\ u_{ki} \in \{0,1\},\ \sum_{i=1}^{n} u_{ki} = 1,$$
$$w_{kj} > 0 \prod_{j=1}^{d} w_{kj} = 1. \tag{8}$$

In Eq. (8), $\left\|\varphi(x_{ij}) - \varphi(v_{kj})\right\|$ is the kernel distance between object $\mathbf{x}_i$ and cluster center $\mathbf{v}_k$ on the jth dimension, which is generally expressed as Eq. (9), and $\kappa(x_{ij}, v_{kj})$ is the kernel metric.

$$\left\|\varphi(x_{ij}) - \varphi(v_{kj})\right\|^2 = \kappa(x_{ij}, x_{ij}) - 2\kappa(x_{ij}, v_{kj}) + \kappa(v_{kj}, v_{kj}) \tag{9}$$

Examples of commonly used kernel functions are Gaussian, given by $\kappa(x_{ij}, v_{kj}) = e^{-(x_{ij}-v_{kj})^2/2\sigma_j^2}$, $\sigma_j > 0$, polynomial of degree d, given by $\kappa(x_i, x_k) = (\gamma x_i^{\mathrm{T}} x_k + \theta)^d$, $\gamma > 0$, $\theta > 0$. In this paper, Gaussian kernel, the most commonly used in the literature as well as used in Ferreira and Carvalho (2014a, b), Ferreira et al. (2016) is employed to develop the functions. Besides, the benchmark method: VKCM-K-LP and VKFCM-K-LP use Gaussian kernel (Ferreira and Carvalho 2014a; Ferreira et al. 2016), which should be implemented here to have fair comparison. Thus, $\kappa(x_{ij}, x_{ij}) = 1$ and $\left\|\varphi(x_{ij}) - \varphi(v_{kj})\right\|^2 = 2(1 - \kappa(x_{ij}, v_{kj}))$. The computations of cluster center v_{kj}, feature weight w_{kj} and partition allocation u_{ki} are respectively shown as Eqs. (10), (11), and (12) (Ferreira et al. 2016).

$$v_{kj} = \frac{\sum_{i=1}^{n} u_{ki}\kappa(x_{ij}, v_{kj}) x_{ij}}{\sum_{i=1}^{n} u_{ki}\kappa(x_{ij}, v_{kj})} \tag{10}$$

$$w_{kj} = \frac{\left\{\prod_{l}^{d}\left(\sum_{i=1}^{n} u_{ki}\|\varphi(x_{il}) - \varphi(v_{kl})\|^2\right)\right\}^{1/d}}{\sum_{i=1}^{n} u_{ki}\left\|\varphi(x_{ij}) - \varphi(v_{kj})\right\|^2} \tag{11}$$

$$u_{ki} = \begin{cases} 1, & \varphi^2(\mathbf{x}_i, \mathbf{v}_k) \le \varphi^2(\mathbf{x}_i, \mathbf{v}_h),\ \ 1 \le h \le K \\ 0, & \text{else} \end{cases} \tag{12}$$

where,

$$\varphi^2(\mathbf{x}_i, \mathbf{v}_k) = \sum_{j=1}^{d} w_{kj} \left\|\varphi(x_{ij}) - \varphi(v_{kj})\right\|.$$

In terms of between-cluster information, the second objective function is defined based on XB, a well-known internal CVI, which has been widely used for some MOC methods in Saha and Maulik (2014), Zhu et al. (2012), Saha et al. (2011), Ma et al. (2015), Yang et al. (2011), Zhong et al. (2013) and its extension with prototypes in kernel space is expressed as follows.

$$XB = \frac{\sum_{i=1}^{n}\sum_{k=1}^{K} u_{ki}^2 \|\varphi(\mathbf{v}_k) - \varphi(\mathbf{x}_i)\|^2}{n \times D_{\mathrm{s}}}, \tag{13}$$

where,

$$D_{\mathrm{s}} = \min_{t \ne k} \|\varphi(\mathbf{v}_k) - \varphi(\mathbf{v}_t)\|^2.$$

As it can be seen from Eq. (13), the denominator of XB mainly employs a term D_s measuring the separation of clusters with the minimum distance between each pair of different centers. However, in some cases, this index is not applicable as it may present unstable results (Wu et al. 2014), especially when two centers are allocated closely in the real partition, and a solution that can detect this case will be dropped by computing XB. To alleviate this problem, a novel CVI named as Wu-and-Li index (WLI for short) was proposed in Wu et al. (2014), whose equation was shown as following.

$$WLI = \frac{WL_n}{2 \times WL_d}, \tag{14}$$

where,

$$WL_n = \sum_{k=1}^{K} \frac{\sum_{i=1}^{n} u_{ki}^2 \|\mathbf{v}_k - \mathbf{x}_i\|^2}{\sum_{i=1}^{n} u_{ki}},$$
$$WL_d = \frac{1}{2}\left(\min_{t \ne k} \|\mathbf{v}_k - \mathbf{v}_t\|^2 + \mathrm{median}_{t \ne k} \|\mathbf{v}_k - \mathbf{v}_t\|^2\right).$$

As we can see from Eq. (14), the denominator of WLI mainly employs two terms considering both the minimum and the median distances between each pair of centers, which can partially allow the existence of closely allocated centers to some extent. Except for the distance between each pair of cluster centers, the distance between each cluster center and the global center (the mean value of the entire data set) is also important to measure the separation of clusters, such as the mechanism of ESSC and E-AWA. Hence, we define a new term D_{s2} that can effectively measure the separation index by the kernelization of weighted distance between K cluster centers and the global center, which is expressed as following.

$$D_{s2} = \sum_{k=1}^{K}\sum_{j=1}^{d} w_{kj} \left\|\varphi(v_{kj}) - \varphi(v_{0j})\right\|^2 \tag{15}$$

In this paper two terms D_s in Eq. (13) and D_{s2} are simultaneously used to develop the denominator of the second objective function F_s. Commonly, D_{s2} is smaller than D_s owing to the attribute weighting, the mean value of them like the term WL_d of WLI expressed as Eq. (14) is unreasonable for most cases because the result is very close to $D_s/2$. Hence, we employ the product of D_s and D_{s2} as the denominator of F_s, and also employ J_c as the numerator, which is similar to the forms of XB and WLI. Note that, the term D_s included in Eq. (13) is expressed as D_{s1} here to be easier distinguished from D_{s2}, and hence F_s is represented as Eq. (16). As described before, in Saha and Maulik (2014), Zhu et al. (2012), Saha et al. (2011), Ma et al. (2015), Yang et al. (2011), Zhong et al. (2013), some MOC methods utilized J_m and XB as the objective functions that are conflicting owing to the separation measurement in the denominator of XB, despite the numerator of the later is similar to the former. Here, J_c and F_s have the same property, and the effectiveness of F_s will be discussed and confirmed in the later Sect. 5.2.2.

$$F_s = \frac{J_c}{D_s \times D_{s2}} = \frac{J_c}{D_{s1} \times D_{s2}} \tag{16}$$

3.3 Chromosome encoding and initialization

A mixed chromosome encoding strategy is adopted to represent each individual, where the first half is the centers and the second half is the weights so that the objective functions can be computed easily by decoding. Here, an individual vector can be represented as $\mathbf{x} = (v_{11}, \ldots, v_{1d}, \ldots, v_{K1}, \ldots, v_{Kd}, w_{11}, \ldots, w_{1d}, \ldots, w_{K1}, \ldots, w_{Kd})$, whose length is $2 \times K \times d$. In the initialization step, the dataset is randomly partitioned into K clusters and the partition matrix U can be got afterward, then the cluster centers are computed by $v_{kj} = (\sum_{i=1}^{n} u_{ki} x_{ij}) / \sum_{i=1}^{n} u_{ki}, 1 \le k \le K, \ 1 \le j \le d$. In order to generate a more widely distributed region for the attribute weights of the initial population, half of its members are $\mathbf{w}_k = (1, \ldots, 1), \ 1 \le k \le K$, while another half are generated by using Eq. (17).

$$w_{kj} = \frac{\left\{ \prod_l^d \left(\sum_{i=1}^{n} u_{ki} (x_{il} - v_{kl})^2 \right) \right\}^{1/d}}{\sum_{i=1}^{n} u_{ki} (x_{ij} - v_{kj})^2} \tag{17}$$

3.4 Genetic operations: selection, crossover, and mutation

Selection: The chromosome selection is a process of choosing the individuals for reproduction. There are several selection methods such as tournament selection, roulette wheel selection, steady state selection, rank selection and elitism, among which the first two are most popular (Deb et al. 2002). In the tournament selection, the chromosomes are randomly selected from the large population at first, then they compete against each other and the one with higher fitness based on nondominance rank and crowding distance is selected for next generation. While for the roulette wheel selection, the chromosomes are selected based on the probability distribution of their fitness values, and those with higher probability values will be more likely to be selected for reproduction.

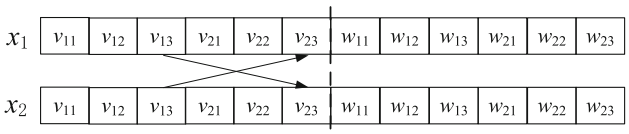

Fig. 1 The one-point crossover of the chromosomes

In this study, the roulette wheel method is adopted. The main reason is that different probability can be obtained, which gives all of chromosomes an opportunity to be chosen. For each individual, it will be selected by using a probability computed with the following rank-based evaluation function.

$$F(\mathbf{x}_i) = \alpha(1-\alpha)^{i_{rank}-1} \tag{18}$$

where $\mathbf{x}_i$ represents the ith chromosome, i_{rank} is the rank of $\mathbf{x}_i$ that to be lower for better chromosome, and α is a parameter indicating the selective pressure of the algorithm. Accordingly, the individual with a lower rank value will be selected with a higher probability.

Crossover: It is assumed that the product of the attribute weights on each cluster should be equal to one, hence the range of these weights are not in the interval (0,1) and the differences among them are relatively notable. Thus, there is no variance of attribute weights during the crossover and mutation operation. Here, the one-point crossover method is used and only the first half part of each couple of chromosomes are performed in this process, as shown in Fig. 1.

Mutation: During the mutation process, a change is made to each gene of the first half part of chromosomes selected with the probability P_m, then a random number ξ in the interval [0,1] with uniform distribution can be used to make the change with the procedure representing as below.

$$x_{ij}^{\text{new}} = \begin{cases} l_j + r_j \times \xi, & \text{if } rand_j < P_m, \\ x_{ij}^{\text{old}}, & \text{otherwise,} \end{cases} \tag{19}$$

where, $r_j (1 \le j \le d)$ is the range of the jth dimension of the dataset, if the max value of the jth dimension is u_j, the min value of that is l_j, then $r_j = u_j - l_j$, and also $r_j \in [0, 1]$ if the dataset is normalized before the clustering process.

3.5 The procedure of the proposed algorithm

Our proposed algorithm MOKCW is summarized in Fig. 2. The parameters included are total generation number $T_{\max}$,

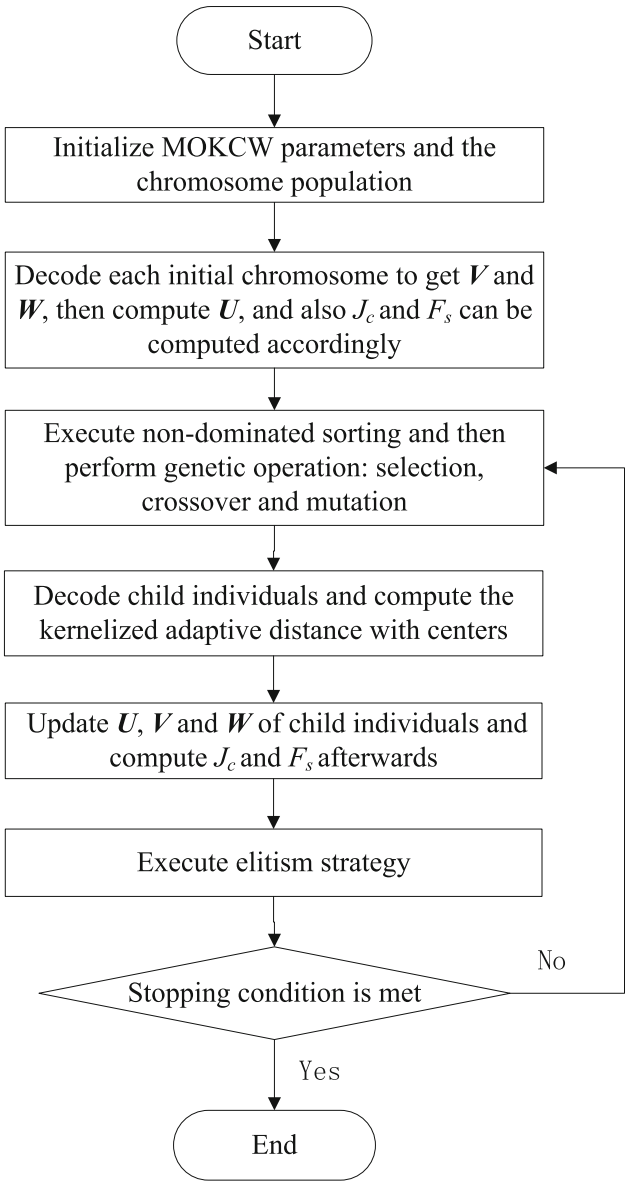

Fig. 2 Flowchart of the proposed algorithm MOKCW

population size N, number of clusters K, mutation probability P_m. In the flowchart, V, W, and U respectively represents cluster center matrix, attribute weight matrix, and partition matrix. In the end of MOKCW, a non-dominated solution set P_S is obtained, and through each one of the set a clustering result can be achieved. It deserves to note that, for each generation, the kernelized adaptive distances are computed twice, where the first time is needed due to the change of each individual by genetic operations.

3.6 Obtain the final clustering solution

As discussed above, the final clustering solution should be obtained through P_S when the proposed algorithm MOKCW stops, and some popular methods have been introduced in Sect. 2.1. As a matter of fact, the internal CVI is usually similar to the objective functions adopted in this study to some extent, we could not use the first method owing to the bias to the construction of data. As the distribution of non-dominated solutions produced in this study is diverse, there is no need to retain the information of all solutions, and thus cluster ensemble may not generate a desirable final solution. Besides, the semi-supervised based method cannot be utilized for most real problems as there is no any priori information of the data partition labels. Therefore, an effective and stable method based on similarity among objects on relevant dimensions is used in this paper, which has also been used in MOEASSC. However, it should be pointed that the equation of *PSVIndex* in Xia et al. (2013) was not strictly formulated, whose appropriate formula could be shown as following.

$$PSVIndex = \sum_{l=1}^{K}\left(\sum_{i=1}^{n_l}\sum_{j=1,j\neq i}^{n_l} SPDis(i,j)\right), \tag{20}$$

where,

$$SPDis(i,j) = \sum_{s=1}^{d}\log(\left|Project_{is} - Project_{js}\right| + 1.0). \tag{21}$$

In Eq. (21), $Project_{is}$ denotes the projected interval of the ith point on the sth dimension, namely projection coordinate. For example, the interval [0,1] is evenly divided into 10 segments, namely $totalSeg = 10$, and then projected coordinate in the interval [0,0.1) is 1, while that in the interval [0.9,1] is 10. Commonly, the objects in the same class have similar projection coordinates leading to a small $SPDis$ value, and hence the best solution can be chosen by the smallest *PSVIndex*. According to Eq. (20), the time complexity of selection step is $O(NKP^2d)$, P=max(n_l), and $n_l(1 \le l \le K)$ denotes the number of objects of the lth cluster. If the data size is large, then P should be large as well, and this method will be time-consuming. In order to improve its efficiency when coping with relatively large data, an efficient cluster ensemble (CE) method with random sampling strategy is employed here and incorporated into *PSVIndex* that is named as *PSVIndex* + CE. Due to the fact that a subset of the dataset can also get a small *PSVIndex* value if the clustering result of the whole dataset is similar to the true partition, the sampling rate $n_s\%$ can be given a small value (i.e., no more than 20%), which will lead to a reduction in computation complexity by a large margin. However, the result may be unreliable if the *PSVIndex* value is calculated by merely one subset. Generally, cluster ensemble methods can improve the quality and robustness of results,

it consists of generating a set of clusterings from the same dataset and combining them into a final clustering (Strehl and Ghosh 2003). Hereby, we adopt five different subsets to obtain five solutions, and then merge them into one solution as the final result by using the CE method. In this paper, two efficient hypergraph-based CE approaches, namely MCLA (Strehl and Ghosh 2003) and HGBF (Fern and Brodley 2004), are utilized to conduct this procedure, whose time complexity are respectively $O(nK^2r^2)$ and $O(nKr)$, where r is the number of solutions, such that $r = 5$ here. Besides, two other CE approaches CSPA and HGPA proposed in Strehl and Ghosh (2003) are not adopted because the efficiency and/or the performance of them are dissatisfactory in the literature. In a word, the procedure of *PSVIndex* + CE is described as Algorithm 1.

Algorithm 1 *PSVIndex* + CE

(1) The first part:
for t=1 to 5 **do**
  Get subset S_t by sampling the n_s% of the dataset S;
  for j=1 to N **do**
    Get the cluster labels of S_t according to the jth partition result, and calculate the $PSVIndex_j$ value according to Eq. (20);
  end for
  Obtain the best solution with a index value $l^{(t)}$ by $\min_j PSVIndex_j$, and the clustering partition of S is $P^{(t)}$;
end for
(2) The second part:
Step1 Randomly keep one solution $P^{(\tau)}(1 \le \tau \le T)$ if there is a group consisted of several solutions with the same index value $l^{(t)}(1 \le t \le 5)$, and calculate the size $\xi^{(\tau)}$ of each corresponding group, then T partition results can be achieved.
Step2 If T=1, namely only one solution exists, thus the single partition P is the clustering result.
Step3 If T=2, let the $l^{(\varpi)}$th one by $\varpi = \max_\tau \xi^{(\tau)}(\tau = 1, 2)$ to be the final solution and the partition $P^{(\varpi)}$ is the clustering result.
Step4 If $T \ge 3$, combine the T partition vectors with class labels of the dataset to constitute a new matrix, through which the final clustering result is achieved by using the CE method MCLA or HGBF.

Among the traditional clustering methods, most have specified the number of clusters to be equal to the known number of classes so that certain privileged information about the dataset should be available, which is cheating for unsupervised learning.

Recently, two papers (Jos-Garcła and Gmez-Flores 2016; Hancer and Karaboga 2017) about the survey of focusing on determination of cluster number present relatively comprehensive approaches, where the evolutionary computation (EC) based clustering, either single-objective or multiobjective approaches, are popular due to the global search potential, and some typical MOC methods can be found in Handl and Knowles (2007), Zhao et al. (2015), Wikaisuksakul (2014), Saha and Bandyopadhyay (2013), Mukhopadhyay et al. (2013). For the multiobjective clustering algorithms, the encoding scheme plays a crucial role in both computational and clustering performance, and the centroid-based scheme is utilized in most cases, such as that in Zhao et al. (2015), Wikaisuksakul (2014), Saha and Bandyopadhyay (2013), Mukhopadhyay et al. (2013), for the length of individuals generated by this scheme is not usually very long that may efficiently improve the computational performance of applied EC operators on the individuals. In this case, the variable-length individuals with the specified range of K from $K_{\min}$ to $K_{\max}$ are generated. For the MOC methods with variable-length individuals, if two objective functions are utilized with an increasing number of K, a trade-off between them is required to partition data for the appropriate number of clusters. However, in our experimental study of the two objective functions J_c and F_s, they are not conflicting when the values of K increase for most datasets. Also, it deserves noticing that the values of attribute weights decrease as the values of K increase, hence the objective functions will achieve a smaller value with a large K, and the corresponding individual will be survived with a large probability. Thus, it can be concluded that the variable-length encoding strategy may not be effective for MOKCW.

In this study, a two-step method named as the *PSVIndex*-gap statistic that can identify the appropriate number of clusters is utilized in this paper, which is also utilized for MOEASSC in Xia et al. (2013) and the effectiveness has been confirmed. We have developed an improved version of *PSVIndex* combined with the clustering ensemble strategy, and the *PSVIndex*-gap statistic is still suitable for our method. In the first stage, the MOKCW algorithm was carried out with each value of K, $K = 1, 2, \ldots, K_{\max}$, and then a solution with the minimum value of *PSVIndex* (denoted as $PSVIndex_C$) was recorded. In the second stage, the gap statistic method developed in Tibshirani et al. (2001) was modified to identify the number of clusters as the same as that in Xia et al. (2013), by which the within-cluster dispersion was replaced by the *PSVIndex* that was unbiased to the two objective functions proposed in MOKCW. The detailed process is described as below:

Step1: Generate B reference datasets for each value of K as described in Tibshirani et al. (2001), and the MOKCW is utilized to cluster each one, and then the minimum values of *PSVIndex* for different K can be obtained and denoted as $PSVIndex^*_{Cb}$, $C = 1, 2, \ldots, Cmax$, $b = 1, 2, \ldots, B$. Afterward, the gap statistical values are calculated by

$$Gap(C) = \sum_{b=1}^{B} \log(PSVIndex^*_{Cb}) \Big/ B - \log(PSVIndex_C) \quad (22)$$

Step2: The mean value of $PSVIndex^*_{Cb}$ is calculated as $\bar{l} = \sum_{b=1}^{B} \log(PSVIndex^*_{Cb}) \big/ B$, and the standard deviation is computed by

$$sd_C = \left[\sum_{b=1}^{B} \{\log(PSVIndex^*_{Cb}) - \bar{l}\}^2 \Big/ B \right]^{1/2} \tag{23}$$

then $s_C = sd_C \sqrt{1 + 1/B}$. After that, the suitable number of clusters is the smallest one that satisfies $Gap(C) \geq Gap(C+1) - s_{C+1}$.

3.7 Time complexity

The worse-case time complexity of the proposed method MOKCW is $O(T_{\max} NnKd + NKP^2d)$, where the parameters have all been described in the above context with no repeated definitions, and then the detailed analysis is addressed below.

(1) Computation time of J_c and F_s are both $O(NnKd)$.
(2) For each generation, crossover and mutation operations require $O(2NKd)$ and $O(P_m NKd)$ time, respectively.
(3) Non-dominated sorting requires $O(MN^2)$ time, where M=2 is the number of objective functions.
(4) The selection step to obtain the clustering solution requires $O(NKP^2d)$, P=max(n_l) time.

Generally K is much smaller than n, and thus the complexity of MOKCW is dominated by the computation of objective functions and selection step. The number of generations is $T_{\max}$ so that the total complexity of MOKCW becomes $O(T_{\max} NnKD + NKP^2d)$ if *PSVIndex* is utilized and that can also be $O(T_{\max} NnKD + NK(n_s P)^2 d)$ if *PSVIndex* + CE is utilized.

4 Partition and cluster interpretation

To evaluate the overall heterogeneity of the data, the intracluster and intercluster data heterogeneity, and the contribution of each attribute to the cluster formation, etc., the indexes for partition and cluster interpretation are needed, and an approach introduced in Chavent et al. (2006) is generally valid even when the overall dispersion dose not decompose into the overall dispersion within clusters plus the overall dispersion between clusters. Also, it has been adapted suitably in Ferreira and Carvalho (2014a, b), Ferreira et al. (2016) to different types of kernel-based attribute clustering algorithms, respectively, where the detailed analysis can be found and the definitions of overall and within clusters dispersion measures, as well as their corresponding decompositions according to clusters, attributes, and both clusters and attributes are utilized for our method.

The overall heterogeneity of all n data points is measured by the overall dispersion, with Eq. (6) replacing the cluster centroids $\mathbf{v}_i$ by the overall centroid $\mathbf{v}$, and the general equation for both hard and fuzzy clustering algorithms (i.e., VKFCM-K-LP and VKCM-K-LP) with the constraint that the product of the attribute weights on each cluster is equal to one is formulated as (citer12,r13)

$$T = \sum_{k=1}^{K} \sum_{i=1}^{n} u_{ki}^m \sum_{j=1}^{d} w_{kj} \left\| \varphi(x_{ij}) - \varphi(v_j) \right\|^2 \tag{24}$$

where the constraints of membership are respectively: (1) $m = 1$ and $u_{ki} \in \{0, 1\}$ for hard clustering; (2) $m > 1$ and $u_{ki} \in [0, 1]$ for fuzzy clustering.

As can be observed from Eq. (24), T measures how dispersed the patterns are with respect to the overall centroid. In this paper, as described above, the Gaussian kernel is employed so that the updated equation of the overall cluster centroid $\mathbf{v} = (v_1, v_2, \ldots, v_d)$, which minimizes the overall dispersion T, is expressed as Eq. (25). Note that, the detailed proof can be found in Ferreira et al. (2016), Ferreira and Carvalho (2014b), and hence we do not give out the analysis.

$$v_j = \frac{\sum_{k=1}^{K} w_{kj} \sum_{i=1}^{n} u_{ki}^m \kappa(x_{ij}, v_j) x_{ij}}{\sum_{k=1}^{K} w_{kj} \sum_{i=1}^{n} u_{ki}^m \kappa(x_{ij}, v_j)} \tag{25}$$

In the above equation, the constraints of membership u_{ki} are similar to that of Eq. (24), where the first one is used for VKCM-K-LP and the second one for VKFCM-K-LP. As can be seen from Eq. (24), the overall dispersion T decomposes according to attributes ($T = \sum_{j=1}^{d} T_j$) and according to clusters ($T = \sum_{k=1}^{K} T_k$), as well as according to clusters and variables ($T = \sum_{k=1}^{K} \sum_{j=1}^{d} T_{kj}$). And also, the overall heterogeneity within cluster given in Eq. (6) is measured by the within-cluster dispersion similarly, which can decompose according to variables ($J = \sum_{j=1}^{d} J_j$), and according to clusters ($J = \sum_{k=1}^{K} J_k$), as well as according to clusters and variables ($J = \sum_{k=1}^{K} \sum_{j=1}^{d} J_{kj}$).

Note that, T and J means the overall dispersion without clustering and after clustering, respectively; T_j and J_j means the attribute-specific overall dispersion without clustering and after clustering concerning the j-th attribute, respectively; T_k and J_k means the cluster-specific overall dispersion without clustering and after clustering, respectively; T_{kj} and J_{kj} respectively means the cluster–attribute-specific overall dispersion without clustering and after clustering concerning the j-th attribute. It can be easily seen that: $T \geq J$, $T_k \geq J_k (k = 1, 2, \ldots, K)$, $T_j \geq J_j (j = 1, 2, \ldots, d)$ and $T_{kj} \geq J_{kj} (k = 1, 2, \ldots, K,\ j = 1, 2, \ldots, d)$.

The overall heterogeneity index is measured by the difference between T and J normalized by the T:

$$Q(P) = \frac{T - J}{T} = 1 - \frac{J}{T} \tag{26}$$

The overall heterogeneity index concerning the j-th attribute is measured by the difference between T_j and J_j normalized by T_j:

$$Q_j(P) = \frac{T_j - J_j}{T_j} = 1 - \frac{J_j}{T_j} \tag{27}$$

For the cluster-specific overall dispersion after clustering, the relative contribution of the cluster P_k to the overall within-cluster dispersion J is given by $J(k) = J_k \big/ J$, $k = 1, 2, \ldots K$. For the cluster-specific overall dispersion without clustering, the quality of a cluster P_k is measured by the difference between T_k and J_k normalized by T_k:

$$Q(P_k) = \frac{T_k - J_k}{T_k} = 1 - \frac{J_k}{T_k} \tag{28}$$

The quality of a cluster P_k concerning the j-th attribute is measured by the difference between T_{kj} and J_{kj} normalized by T_{kj}:

$$Q_j(P_k) = \frac{T_{kj} - J_{kj}}{T_{kj}} = 1 - \frac{J_{kj}}{T_{kj}} \tag{29}$$

The above indexes all take their values from the interval [0,1], and the value closer to 1 denotes a better result with respect to each aspect of the partition/cluster quality while that close to 0 denotes a poor quality. Additionally, it deserves noticing that the usefulness of the aforementioned partition and cluster interpretation has been shown in Ferreira et al. (2016), Ferreira and Carvalho (2014b) with the application on the benchmark datasets.

5 Experiment study

5.1 Datasets and parameter setting

In this section, the performance of MOKCW is evaluated by conducting experiments on several benchmark datasets, which is also compared with that of three SSC algorithms ESSC (Deng et al. 2010), E-AWA (Huang et al. 2014a), MOEASSC (Xia et al. 2013) described in Sect. 2.2, and the kernel-based attribute-weighted algorithms VKCM-K-LP (Ferreira et al. 2016), a crisp version of clustering method, and VKFCM-K-LP (Ferreira and Carvalho 2014b), the fuzzy version. Among them only MOEASSC is a multiobjective method and the others are single-objective ones. For ESSC and E-AWA, we adopt k-means++ (Arthur and Vassilvitskii 2007) to find some points as the initial centers that are apart from each other for the sake of reducing unfavorable impact of initialization. We do not include in our experiments other SSC algorithms like EWKM (Jing et al. 2007), FSC (Gan and Wu 2008) because MOEASSC has already been shown superiority over them, and other kernel-based feature weighted clustering algorithms like VKCM-K-LS, VKCM-K-GP (Ferreira et al. 2016) as well as VKFCM-K-LS, VKFCM-K-GP (Ferreira and Carvalho 2014b) because VKCM-K-LP or VKFCM-K-LP has already been shown to outperform all of them. The experiments are conducted on a computer with Intel Core i7-4770, CPU 3.40 GHz and 16 GB RAM by using MATLAB2010.

Table 1 The characters of the datasets

Datasets	K	d	n	Datasets	K	d	n
Iris	3	4	150	Abalone	3	8	4177
Wine	3	13	178	WFRN	4	4	5456
Newthyroid	3	5	215	SVMguide1	2	4	7089
Breast	6	9	106	Thyroid	3	6	7200
Vertebral	3	6	310	Waveform	3	40	5000
Bupa	3	6	345	Magic	2	10	19,020
WDBC	2	30	569	Occupancy	2	5	20,560
Image	7	16	2310	Shuttle	7	9	43,500
Seismic	2	4	2584	5Gaussians	5	2	100,000

All the algorithms considered in this paper are applied to 18 data sets: 16 are real-life data sets obtained from the UCI Machine Learning Repository, namely Iris, Wine, Newthyroid, Breast, Vertebral, Bupa, WDBC, Image, Seismic, Abalone, WFRN, Thyroid, Waveform, Magic, Occupancy, Shuttle; one data set SVMguide1 obtained from the LIBSVM (Chang and Lin 2011) library; and one data set 5Gaussian utilized in Ferreira et al. (2016) that is larger than the others. These data sets are mostly adopted in Ferreira and Carvalho (2014a, b), Ferreira et al. (2016) and also widely used in the literature to evaluate the clustering performance. As shown in Table 1 the data sets considered are briefly described, where K is the true number of classes, d and n are, respectively, the number of features and objects. For most SSC algorithms, the experiments are conducted on the data sets standardized into the interval [0,1] (denoted as "Standardization"), which can alleviate the uneven impact of different attributes' ranges on updating the weights. However, the experiment is conducted on the data sets without standardization (denoted as "None") in Ferreira et al. (2016), and conducted on both the non-standardized and standardized versions of the data sets in Ferreira and Carvalho (2014b). In this paper, the cases of none and standardization are both considered to make a more comprehensive analysis, and the standardization is based on the minimum and maximum values of each attribute.

Table 2 Parameter settings of all algorithms

Algorithms	Parameter setting
ESSC	$m = \frac{\min(n,D-1)}{\min(n,D-1)-2}$, $\gamma = 100$, $\eta = 0.1$
E-AWA	$\beta = 8$
MOEASSC	$P_c = 0.5$, $P_m = K/D$, $\sigma = 10^{-7}$
VKFCM-K-LP	$m = 2$
MOKCW	$P_m = 0.1$, $\alpha = 0.1$

As the ESSC method is sensitive to the parameters shown in Eq. (1), in which γ and η are respectively selected from a set containing several numbers (Deng et al. 2010), but the best parameters for different datasets can be hardly set without any prior knowledge. To make a fair comparison, each parameter is given one value according to the suggestion in related works. The total iteration number of all single-objective algorithms is 100, while they can stop in advance if $J^{t-1} - J^t \leq 10^{-9}$, where t denotes the iteration times. For the two MOC algorithms, the population size is $N = 20$, the total generation is $T_{\max} = 40$. Other parameters of all algorithms are set as shown in Table 2, and it deserves noting that there is no parameter in VKCM-K-LP. The term $2\sigma^2$ is important to the Gaussian kernel employed in this paper, and we utilize the same value of VKCM-K-LP set in Ferreira et al. (2016). The average of the 0.1 and 0.9 quantiles of $\|x_{ij} - x_{kj}\|^2, i \neq k, j = 1, \ldots, d$ are assigned for the terms $2\sigma_j^2 (j = 1 \ldots, d)$. For relatively large datasets Abalone, WFRN, SVMguide1, Thyroid, Waveform, *PSVIndex* + CE described in the Sect. 3.6 with the sampling rate 10%, namely $n_s = 10$, is used to obtain the final clustering result of both the two MOC approaches. Moreover, for datasets Magic, Occupancy, Shuttle, and 5Gaussians that are even more larger, *PSVIndex* + CE with the sampling rate 5%, namely $n_s = 5$, is adopted. But for other relatively small datasets, we utilize *PSVIndex* to obtain the final solution. In both *PSVIndex* and *PSVIndex* + CE, *totalSeg* = 20 is adopted to calculate the projection coordinates.

5.2 Experiment result and analysis

To evaluate the performance of the clustering results of all algorithms and make a overall comparison, three well-known external CVIs accuracy (*Acc*) and rand index (*RI*) (Huang et al. 2005), as well as normalized mutual information (*NMI*) (Deng et al. 2010) are adopted here. They all take their values from the interval [0,1], in which 1 indicating the perfect match between the clustering result and the true partition, whereas the value close to 0 indicates a result found by chance, and thus the larger values they are the better clustering result achieved. All algorithms considered in this paper are executed 20 times independently, and their performances are compared in terms of the means and the standard deviations of *Acc*, *RI*, *NMI* shown in Tables 3 and 4. For each data set, since in some cases the best performance obtained is similar to the second one, thereby the best and the second best values obtained by the proposed method MOCKW on each performance metric is marked respectively, if existed. Besides, if MOEASSC is applied to the data sets without standardization, some features with very large ranges are abound to get too large weight values according to the computation of weights expressed in Eq. (5), resulting in a very poor clustering performance. Thus, results of MOEASSC on non-standardized data sets are not shown in Tables 3 and 4. Also, note that, in Ferreira et al. (2016) the experimental result is the best case selected among several executions on each data set, and hence the true comprehensive performance of different kernel clustering methods cannot be concretely evaluated in comparison.

5.2.1 *Clustering performance of the algorithms*

It can be firstly observed from Tables 3 and 4 that, in some cases, VKCM-K-LP obviously outperforms ESSC, E-AWA, and VKFCM-K-LP among the four single-objective attribute-weighted clustering methods, which is an important motivation for us. By comparing the two single-objective kernel method, namely VKCM-K-LP and VKFCM-K-LP, the former shows superiority over the later, especially for datasets Newthyroid, Seismic, WFRN, Occupancy and Shuttle. Moreover, in most cases, the proposed approach MOKCW can obtain the best result or the second best result on the three indices under both the "none" situation and the "standardization" situation. In some cases MOKCW is beaten, for instance, by VKCM-K-LP on dataset Iris (none); by E-AWA on dataset Vertebral (none), Thyroid (both none and standardization) and Occupancy (none); by MOEASSC on dataset Wine, Seismic(standardization), and 5Gaussians; by ESSC on dataset Magic. But note that for some cases of them, the difference is just on a small margin. As we can observe that the performance ranking results of each approach on the three indices are very similar, which can be also found in Xia et al. (2013), Ferreira et al. (2016), Deng et al. (2010), Huang et al. (2014a). However, in some cases this phenomenon does not occur, such as that for dataset Breast, the proposed method get the best *Acc* and *RI* values, but VKCM-K-LP get the best *NMI* value; for dataset Vertebral, E-AWA is shown obvious superiority over MOKCW on *Acc*, but vice verse on *NMI*. Also, for datasets Bupa, Thyroid, Waveform, Shuttle, the inconsistence of the three indices can be found, as *NMI* is usually disaccord with *Acc* and *RI*.

We can also observe from Tables 3 and 4 that the overall performance of ESSC under the "none" situation is worse than that under the "standardization" situation, especially for data sets Iris, Wine, Thyroid, WDBC, Waveform. And

Table 3 The results of all algorithms on *Acc*, *RI*, and *NMI* (Mean ± SD)

Datasets	Algorithms	None			Standardization		
		Acc	*RI*	*NMI*	*Acc*	*RI*	*NMI*
Iris	ESSC	0.764 ± 0.125	0.806 ± 0.066	0.662 ± 0.081	0.839 ± 0.077	0.844 ± 0.048	0.716 ± 0.066
	E-AWA	0.815 ± 0.065	0.819 ± 0.035	0.647 ± 0.069	0.787 ± 0.070	0.805 ± 0.039	0.636 ± 0.066
	MOEASSC	–	–	–	0.900 ± 0.000	0.886 ± 0.000	0.778 ± 0.000
	VKCM-K-LP	0.952 ± 0.019	0.941 ± 0.021	0.855 ± 0.021	0.955 ± 0.016	0.944 ± 0.018	0.858 ± 0.018
	VKFCM-K-LP	0.947 ± 0.000	0.934 ± 0.000	0.832 ± 0.000	0.947 ± 0.000	0.934 ± 0.000	0.832 ± 0.000
	MOKCW	0.943 ± 0.020	0.931 ± 0.021	0.835 ± 0.034[b]	0.960 ± 0.000[a]	0.950 ± 0.000[a]	0.864 ± 0.000[a]
Wine	ESSC	0.640 ± 0.051	0.690 ± 0.032	0.414 ± 0.028	0.955 ± 0.000	0.940 ± 0.000	0.847 ± 0.000
	E-AWA	0.876 ± 0.113	0.868 ± 0.080	0.732 ± 0.109	0.897 ± 0.104	0.887 ± 0.074	0.763 ± 0.108
	MOEASSC	–	–	–	0.956 ± 0.003	0.942 ± 0.003	0.854 ± 0.006
	VKCM-K-LP	0.917 ± 0.016	0.894 ± 0.018	0.759 ± 0.041	0.907 ± 0.024	0.883 ± 0.027	0.733 ± 0.061
	VKFCM-K-LP	0.938 ± 0.000	0.919 ± 0.000	0.796 ± 0.000	0.938 ± 0.000	0.919 ± 0.000	0.796 ± 0.000
	MOKCW	0.926 ± 0.008[b]	0.903 ± 0.009[b]	0.778 ± 0.021[b]	0.921 ± 0.006	0.897 ± 0.007	0.768 ± 0.016
Newthyroid	ESSC	0.676 ± 0.094	0.610 ± 0.067	0.229 ± 0.106	0.893 ± 0.001	0.826 ± 0.002	0.620 ± 0.004
	E-AWA	0.682 ± 0.050	0.625 ± 0.042	0.385 ± 0.089	0.684 ± 0.041	0.632 ± 0.017	0.390 ± 0.051
	MOEASSC	–	–	–	0.888 ± 0.000	0.818 ± 0.000	0.603 ± 0.000
	VKCM-K-LP	0.885 ± 0.135	0.856 ± 0.116	0.687 ± 0.164	0.864 ± 0.146	0.834 ± 0.128	0.655 ± 0.177
	VKFCM-K-LP	0.809 ± 0.000	0.737 ± 0.000	0.541 ± 0.000	0.809 ± 0.000	0.737 ± 0.000	0.541 ± 0.000
	MOKCW	0.946 ± 0.002[a]	0.912 ± 0.004[a]	0.770 ± 0.007[a]	0.946 ± 0.003[a]	0.913 ± 0.005[a]	0.772 ± 0.009[a]
Breast	ESSC	0.347 ± 0.029	0.639 ± 0.061	0.317 ± 0.038	0.501 ± 0.026	0.782 ± 0.015	0.522 ± 0.014
	E-AWA	0.485 ± 0.040	0.729 ± 0.029	0.530 ± 0.037	0.510 ± 0.040	0.759 ± 0.035	0.542 ± 0.012
	MOEASSC	–	–	–	0.527 ± 0.029	0.781 ± 0.022	0.532 ± 0.041
	VKCM-K-LP	0.545 ± 0.037	0.789 ± 0.020	0.555 ± 0.027	0.544 ± 0.041	0.790 ± 0.031	0.557 ± 0.028
	VKFCM-K-LP	0.524 ± 0.009	0.805 ± 0.005	0.553 ± 0.013	0.523 ± 0.009	0.802 ± 0.007	0.551 ± 0.012
	MOKCW	0.586 ± 0.043[a]	0.811 ± 0.015[a]	0.527 ± 0.030	0.579 ± 0.045[a]	0.810 ± 0.015[a]	0.535 ± 0.032
Vertebral	ESSC	0.507 ± 0.090	0.568 ± 0.075	0.136 ± 0.127	0.474 ± 0.000	0.633 ± 0.000	0.269 ± 0.000
	E-AWA	0.616 ± 0.073	0.688 ± 0.041	0.351 ± 0.065	0.631 ± 0.064	0.694 ± 0.034	0.360 ± 0.061
	MOEASSC	–	–	–	0.479 ± 0.052	0.623 ± 0.004	0.249 ± 0.007
	VKCM-K-LP	0.499 ± 0.020	0.651 ± 0.017	0.318 ± 0.041	0.510 ± 0.039	0.667 ± 0.020	0.350 ± 0.048
	VKFCM-K-LP	0.516 ± 0.000	0.661 ± 0.000	0.330 ± 0.000	0.517 ± 0.003	0.661 ± 0.001	0.330 ± 0.001
	MOKCW	0.527 ± 0.025[b]	0.687 ± 0.008[b]	0.401 ± 0.018[a]	0.506 ± 0.012	0.697 ± 0.007[a]	0.431 ± 0.019[a]

Table 3 continued

Datasets	Algorithms	None			Standardization		
		Acc	*RI*	*NMI*	*Acc*	*RI*	*NMI*
Bupa	ESSC	0.472 ± 0.040	0.514 ± 0.026	0.017 ± 0.017	0.473 ± 0.000	0.504 ± 0.004	0.004 ± 0.000
	E-AWA	0.438 ± 0.068	0.505 ± 0.010	0.013 ± 0.008	0.439 ± 0.058	0.503 ± 0.004	0.016 ± 0.010
	MOEASSC	–	–	–	0.536 ± 0.026	0.502 ± 0.006	0.006 ± 0.004
	VKCM-K-LP	0.428 ± 0.024	0.501 ± 0.004	0.007 ± 0.006	0.433 ± 0.022	0.502 ± 0.005	0.008 ± 0.006
	VKFCM-K-LP	0.431 ± 0.003	0.498 ± 0.001	0.002 ± 0.001	0.428 ± 0.004	0.498 ± 0.001	0.002 ± 0.001
	MOKCW	0.554 ± 0.013[a]	0.505 ± 0.003[b]	0.006 ± 0.003	0.552 ± 0.006[a]	0.504 ± 0.001[a]	0.012 ± 0.003[b]
WDBC	ESSC	0.810 ± 0.081	0.704 ± 0.094	0.390 ± 0.128	0.928 ± 0.000	0.866 ± 0.000	0.623 ± 0.000
	E-AWA	0.887 ± 0.088	0.815 ± 0.096	0.540 ± 0.177	0.916 ± 0.002	0.846 ± 0.003	0.597 ± 0.005
	MOEASSC	–	–	–	0.930 ± 0.003	0.869 ± 0.006	0.634 ± 0.017
	VKCM-K-LP	0.939 ± 0.002	0.885 ± 0.004	0.664 ± 0.005	0.940 ± 0.002	0.886 ± 0.004	0.664 ± 0.004
	VKFCM-K-LP	0.942 ± 0.000	0.891 ± 0.000	0.672 ± 0.000	0.942 ± 0.000	0.891 ± 0.000	0.672 ± 0.000
	MOKCW	0.940 ± 0.007[b]	0.886 ± 0.012[b]	0.675 ± 0.019[a]	0.942 ± 0.003[a]	0.890 ± 0.005[b]	0.668 ± 0.013[b]
Image	ESSC	0.463 ± 0.064	0.797 ± 0.030	0.425 ± 0.045	0.606 ± 0.067	0.862 ± 0.020	0.624 ± 0.018
	E-AWA	0.585 ± 0.036	0.843 ± 0.026	0.582 ± 0.038	0.585 ± 0.067	0.846 ± 0.029	0.584 ± 0.057
	MOEASSC	–	–	–	0.620 ± 0.075	0.863 ± 0.022	0.628 ± 0.064
	VKCM-K-LP	0.616 ± 0.026	0.869 ± 0.010	0.626 ± 0.022	0.624 ± 0.029	0.869 ± 0.010	0.625 ± 0.027
	VKFCM-K-LP	0.624 ± 0.042	0.870 ± 0.014	0.649 ± 0.025	0.623 ± 0.035	0.871 ± 0.011	0.649 ± 0.023
	MOKCW	0.626 ± 0.033[a]	0.871 ± 0.012[a]	0.616 ± 0.034	0.635 ± 0.028[a]	0.874 ± 0.011[a]	0.631 ± 0.016[b]
Seismic	ESSC	–	–	–	0.788 ± 0.000	0.666 ± 0.000	0.030 ± 0.000
	E-AWA	0.691 ± 0.001	0.573 ± 0.001	0.030 ± 0.001	0.691 ± 0.001	0.573 ± 0.001	0.030 ± 0.001
	MOEASSC	–	–	–	0.885 ± 0.000	0.797 ± 0.000	0.041 ± 0.000
	VKCM-K-LP	0.828 ± 0.001	0.715 ± 0.001	0.034 ± 0.001	0.828 ± 0.001	0.715 ± 0.001	0.034 ± 0.001
	VKFCM-K-LP	0.730 ± 0.042	0609 ± 0.036	0.019 ± 0.004	0.748 ± 0.033	0.624 ± 0.029	0.021 ± 0.004
	MOKCW	0.828 ± 0.000[a]	0.716 ± 0.000[a]	0.034 ± 0.001[a]	0.829 ± 0.001[b]	0.717 ± 0.001[b]	0.034 ± 0.000[b]
Abalone	ESSC	0.509 ± 0.049	0.571 ± 0.072	0.147 ± 0.037	0.507 ± 0.000	0.612 ± 0.000	0.163 ± 0.000
	E-AWA	0.509 ± 0.016	0.610 ± 0.008	0.157 ± 0.009	0.511 ± 0.014	0.610 ± 0.008	0.159 ± 0.011
	MOEASSC	–	–	–	0.503 ± 0.005	0.610 ± 0.001	0.161 ± 0.001
	VKCM-K-LP	0.510 ± 0.001	0.618 ± 0.001	0.163 ± 0.001	0.512 ± 0.001	0.619 ± 0.001	0.162 ± 0.001
	VKFCM-K-LP	0.513 ± 0.002	0.620 ± 0.002	0.165 ± 0.002	0.513 ± 0.000	0.619 ± 0.000	0.165 ± 0.000
	MOKCW	0.510 ± 0.001[b]	0.618 ± 0.001[b]	0.163 ± 0.001[b]	0.512 ± 0.002[b]	0.619 ± 0.001[b]	0.163 ± 0.001[b]

[a] The best performance obtained by MOCKW, [b] the second best performance obtained by MOCKW

Table 4 The results of all algorithms on *Acc*, *RI*, and *NMI* (Mean ± SD)

Datasets	Algorithms	None			Standardization		
		Acc	*RI*	*NMI*	*Acc*	*RI*	*NMI*
WFRN	ESSC	0.435 ± 0.039	0.580 ± 0.052	0.160 ± 0.062	0.425 ± 0.026	0.594 ± 0.030	0.166 ± 0.046
	E-AWA	0.559 ± 0.095	0.680 ± 0.057	0.322 ± 0.107	0.651 ± 0.107	0.720 ± 0.048	0.392 ± 0.090
	MOEASSC	–	–	–	0.432 ± 0.045	0.554 ± 0.025	0.141 ± 0.045
	VKCM-K-LP	0.553 ± 0.048	0.696 ± 0.035	0.334 ± 0.055	0.552 ± 0.047	0.701 ± 0.018	0.346 ± 0.034
	VKFCM-K-LP	0.506 ± 0.005	0.659 ± 0.002	0.269 ± 0.012	0.501 ± 0.023	0.657 ± 0.007	0.265 ± 0.018
	MOKCW	0.576 ± 0.038[a]	0.721 ± 0.010[a]	0.398 ± 0.027[a]	0.584 ± 0.040[b]	0.723 ± 0.011[a]	0.398 ± 0.025[a]
SVMguide1	ESSC	0.746 ± 0.137	0.657 ± 0.120	0.339 ± 0.203	0.804 ± 0.000	0.685 ± 0.000	0.282 ± 0.000
	E-AWA	0.855 ± 0.063	0.767 ± 0.058	0.506 ± 0.112	0.875 ± 0.001	0.782 ± 0.001	0.536 ± 0.001
	MOEASSC	–	–	–	0.780 ± 0.001	0.656 ± 0.002	0.241 ± 0.003
	VKCM-K-LP	0.829 ± 0.087	0.731 ± 0.075	0.451 ± 0.146	0.857 ± 0.002	0.755 ± 0.003	0.497 ± 0.004
	VKFCM-K-LP	0.858 ± 0.000	0.757 ± 0.000	0.509 ± 0.000	0.858 ± 0.000	0.757 ± 0.000	0.509 ± 0.000
	MOKCW	0.900 ± 0.005[a]	0.819 ± 0.008[a]	0.586 ± 0.011[a]	0.906 ± 0.007[a]	0.830 ± 0.011[a]	0.601 ± 0.015[a]
Thyroid	ESSC	0.369 ± 0.000	0.381 ± 0.000	0.002 ± 0.000	0.360 ± 0.000	0.381 ± 0.000	0.002 ± 0.000
	E-AWA	0.578 ± 0.091	0.502 ± 0.052	0.207 ± 0.037	0.592 ± 0.081	0.509 ± 0.046	0.210 ± 0.030
	MOEASSC	–	–	–	0.368 ± 0.000	0.381 ± 0.000	0.002 ± 0.000
	VKCM-K-LP	0.504 ± 0.049	0.428 ± 0.018	0.141 ± 0.033	0.506 ± 0.050	0.426 ± 0.017	0.140 ± 0.033
	VKFCM-K-LP	0.535 ± 0.030	0.461 ± 0.010	0.046 ± 0.016	0.527 ± 0.025	0.552 ± 0.016	0.263 ± 0.049
	MOKCW	0.524 ± 0.026	0.445 ± 0.019	0.191 ± 0.020[b]	0.517 ± 0.016	0.437 ± 0.012	0.184 ± 0.023
Waveform	ESSC	0.385 ± 0.031	0.545 ± 0.010	0.037 ± 0.029	0.512 ± 0.000	0.667 ± 0.000	0.363 ± 0.000
	E-AWA	0.548 ± 0.065	0.675 ± 0.018	0.381 ± 0.024	0.525 ± 0.024	0.670 ± 0.001	0.375 ± 0.001
	MOEASSC	–	–	–	0.513 ± 0.001	0.665 ± 0.003	0.356 ± 0.019
	VKCM-K-LP	0.514 ± 0.004	0.667 ± 0.001	0.366 ± 0.001	0.513 ± 0.004	0.667 ± 0.001	0.365 ± 0.001
	VKFCM-K-LP	0.624 ± 0.001	0.672 ± 0.001	0.361 ± 0.001	0.624 ± 0.001	0.672 ± 0.001	0.361 ± 0.001
	MOKCW	0.623 ± 0.002[b]	0.672 ± 0.002[b]	0.366 ± 0.003[b]	0.623 ± 0.001[b]	0.671 ± 0.002[b]	0.359 ± 0.004

Table 4 continued

Datasets	Algorithms	None			Standardization		
		Acc	*RI*	*NMI*	*Acc*	*RI*	*NMI*
Magic	ESSC	0.598 ± 0.064	0.527 ± 0.027	0.020 ± 0.033	0.619 ± 0.000	0.528 ± 0.000	0.012 ± 0.000
	E-AWA	0.537 ± 0.000	0.503 ± 0.000	0.003 ± 0.000	0.537 ± 0.001	0.503 ± 0.001	0.003 ± 0.001
	MOEASSC	–	–	–	0.526 ± 0.015	0.502 ± 0.002	0.002 ± 0.002
	VKCM-K-LP	0.559 ± 0.000	0.507 ± 0.000	0.002 ± 0.000	0.559 ± 0.000	0.507 ± 0.000	0.002 ± 0.000
	VKFCM-K-LP	0.539 ± 0.000	0.503 ± 0.000	0.002 ± 0.000	0.539 ± 0.000	0.503 ± 0.000	0.002 ± 0.000
	MOKCW	0.559 ± 0.000[b]	0.507 ± 0.000[b]	0.002 ± 0.000[b]	0.567 ± 0.004[b]	0.509 ± 0.001[b]	0.003 ± 0.001[b]
Occupancy	ESSC	–	–	–	0.741 ± 0.172	0.672 ± 0.165	0.279 ± 0.243
	E-AWA	0.848 ± 0.090	0.758 ± 0.085	0.480 ± 0.146	0.866 ± 0.061	0.775 ± 0.059	0.508 ± 0.106
	MOEASSC	–	–	–	0.564 ± 0.012	0.508 ± 0.004	0.044 ± 0.011
	VKCM-K-LP	0.748 ± 0.187	0.690 ± 0.178	0.348 ± 0.305	0.670 ± 0.163	0.630 ± 0.149	0.259 ± 0.261
	VKFCM-K-LP	0.638 ± 0.101	0.558 ± 0.108	0.139 ± 0.178	0.625 ± 0.076	0.542 ± 0.083	0.114 ± 0.137
	MOKCW	0.789 ± 0.059[b]	0.694 ± 0.098[b]	0.364 ± 0.241[b]	0.910 ± 0.036[a]	0.837 ± 0.049[a]	0.606 ± 0.065[a]
Shuttle	ESSC	0.465 ± 0.102	0.526 ± 0.056	0.245 ± 0.044	0.428 ± 0.011	0.537 ± 0.025	0.498 ± 0.012
	E-AWA	0.519 ± 0.054	0.491 ± 0.016	0.063 ± 0.034	0.452 ± 0.088	0.472 ± 0.027	0.123 ± 0.072
	MOEASSC	–	–	–	0.416 ± 0.034	0.518 ± 0.023	0.407 ± 0.040
	VKCM-K-LP	0.658 ± 0.054	0.670 ± 0.039	0.387 ± 0.080	0.642 ± 0.075	0.656 ± 0.052	0.368 ± 0.086
	VKFCM-K-LP	0.629 ± 0.197	0.576 ± 0.083	0.072 ± 0.095	0.525 ± 0.196	0.535 ± 0.082	0.121 ± 0.094
	MOKCW	0.674 ± 0.029[a]	0.679 ± 0.020[a]	0.384 ± 0.089[b]	0.662 ± 0.032[a]	0.671 ± 0.027[a]	0.385 ± 0.087[b]
5Gaussians	ESSC	–	–	–	0.731 ± 0.092	0.859 ± 0.061	0.743 ± 0.082
	E-AWA	0.838 ± 0.062	0.934 ± 0.023	0.879 ± 0.020	0.853 ± 0.073	0.941 ± 0.025	0.887 ± 0.022
	MOEASSC	–	–	–	0.992 ± 0.007	0.994 ± 0.006	0.975 ± 0.017
	VKCM-K-LP	0.891 ± 0.083	0.960 ± 0.028	0.928 ± 0.045	0.906 ± 0.099	0.966 ± 0.034	0.940 ± 0.056
	VKFCM-K-LP	0.924 ± 0.106	0.967 ± 0.044	0.930 ± 0.093	0.918 ± 0.103	0.964 ± 0.045	0.919 ± 0.099
	MOKCW	0.973 ± 0.012[a]	0.980 ± 0.008[a]	0.969 ± 0.023[a]	0.976 ± 0.011[b]	0.983 ± 0.009[b]	0.964 ± 0.026[b]

[a] The best performance obtained by MOCKW, [b] the second best performance obtained by MOCKW

Table 5 Average performance rankings of different algorithms on all datasets regarding *Acc*, *RI*, and *NMI*

Algorithms	None			Standardization		
	Acc	*RI*	*NMI*	*Acc*	*RI*	*NMI*
ESSC	4.6 (5)	4.5 (5)	4.4 (5)	4.6 (6)	3.9 (4)	3.9 (5)
E-AWA	3.2 (4)	3.2 (4)	3.1 (4)	3.8 (5)	4.2 (5)	3.6 (4)
MOEASSC	–	–	–	3.7 (4)	4.3 (6)	4.1 (6)
VKCM-K-LP	2.7 (2)	2.6 (2)	2.4 (2)	3.5 (3)	3.0 (2)	3.2 (2)
VKFCM-K-LP	2.8 (3)	2.8 (3)	3.0 (3)	3.2 (2)	3.2 (3)	3.5 (3)
MOKCW	1.5 (1)	1.7 (1)	2.0 (1)	1.7 (1)	1.5 (1)	2.1 (1)

even for data sets Seismic, Occupancy, 5Gaussians, it cannot work normally. Meanwhile, this phenomenon occurs in E-AWA for datasets Breast, WFRN and SVMguide1 as well. But for the algorithms VKCM-K-LP and VKFCM-K-LP, the performances under both the "none" situation and the "standardization" situation are close to each other for most datasets, obvious differences can be found by VKCM-K-LP on data sets SVMguide1 and Occupancy, as well as by VKFCM-K-LP on dataset Shuttle. For instance, when regarding dataset SVMguide1, VKCM-K-LP performs unsteadily with a relatively large standard deviation value 0.087 under the "none" situation, while a small standard deviation value 0.002 is get under the "standardization" situation. To analyze our 20 groups of experiment results on dataset SVMguide1 (none) in detail, the values of *Acc*, *RI*, *NMI* by VKCM-K-LP are most near 0.857, 0.756, 0.499 respectively, but two groups are near 0.574, 0.511, 0.024 respectively. The proposed method MOKCW performs very steadily on this dataset, and *Acc*, *RI*, *NMI* can get small standard deviations that are 0.005, 0.008, 0.011 respectively. Meanwhile, the *Acc*, *RI*, *NMI* values of MOKCW have increased by 8.56%, 12.04%, 29.93% compared to that of VKCM-K-LP under the "none" situation.

Table 5 shows the average performance rankings of all methods on the 18 datasets regarding *Acc*, *RI*, *NMI* computed from Tables 3 and 4, making a more evident comparison. Also, Fig. 3 shows the histogram of mean values of the three indices in comparison for different algorithms. As can be observed from both Table 5 and Fig. 3, the performance of our proposed method is obvious superior to that of the others, whichever index considered. Additionally, the performance of MOEASSC is not as better as that in Xia et al. (2013), where some complex datasets considered in this paper are not included, and also, it may be owing to the computation of attribute weights expressed by Eq. (5) that does not allow good results for these datasets. For instance, the clustering results of MOEASSC on WFRN, Thyroid and Occupancy are abnormal, but it needs noticing that MOEASSC shows obvious superiority for data sets Seismic and 5Gaussians.

Tables 6 and 7, also computed from Tables 3 and 4, show, respectively, the percentage of times that each algorithm obtained the best performance rankings (first or second) under the "none" situation and the "standardization" situation. It can be noticed from these tables that the MOKCW algorithm appeared among the best performing methods with an obvious superiority than the others.

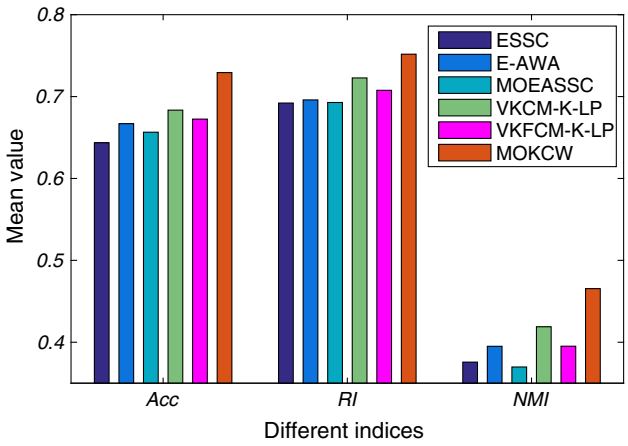

Fig. 3 Mean values of *Acc*, *RI*, and *NMI* using different algorithms in the 18 datasets under the "standardization" situation

As mentioned before, the proposed method MOKCW performs very steadily on dataset SVMguide1 (none), while the VKCM-K-LP approach don't. This is not a specific case, also existing in other approaches. Generally, the clustering performance not only lies in the precision, but also depends on the stability. Figure 4 shows the box-plots of *RI* along with the mean values for different algorithms on all datasets (standardization) considered in this paper, which can analyze the statistical distribution in detail. We can analyze the stability according to the ranges of upper and lower bounds for different algorithms, also the two bounds are capable to indicate the clustering precision performance. The proposed method MOKCW generally has smaller ranges indicating better stability than that of VKCM-K-LP, especially for datasets Vertebral, Bupa, WFRN, Thyroid, and Occupancy. Moreover, higher bounds got by MOKCW indicating better clustering precision. Thus, we can conclude that the incorporation of multiobjective optimization into VKCM-K-LP is

Table 6 Percentual of best performance rankings (first or second) under the "None" situation

Algorithms	*Acc*	*RI*	*NMI*
ESSC	11.11 (5)	11.11 (5)	11.11 (5)
E-AWA	22.22 (4)	27.78 (4)	33.33 (4)
MOEASSC	–	–	–
VKCM-K-LP	38.89 (3)	33.33 (3)	44.44 (2)
VKFCM-K-LP	50.00 (2)	55.55 (2)	44.44 (2)
MOKCW	88.89 (1)	88.89 (1)	77.78 (1)

Table 7 Percentual of best performance rankings (first or second) under the "Standardization" situation

Algorithms	*Acc*	*RI*	*NMI*
ESSC	16.67 (6)	16.67 (5)	16.67 (6)
E-AWA	27.78 (2)	27.78 (3)	44.44 (2)
MOEASSC	22.22 (5)	16.67 (5)	22.22 (4)
VKCM-K-LP	27.78 (2)	27.78 (3)	22.22 (4)
VKFCM-K-LP	27.78 (2)	33.33 (2)	27.78 (3)
MOKCW	83.33 (1)	88.89 (1)	77.78 (1)

very effective, leading to an improvement of the clustering performance by a large margin.

5.2.2 *Study of the objective functions, attribute weighting and the execution time*

As described in Sect. 3.2, the second objective function F_s of MOKCW is the extension of *XB*, it utilizes both the kernelization of the distances between each pair of different centers and the weighted distances between cluster centers and the global center to measure the dispersion of partition result. To analyze this adjustment, here the kernel MOC approach simultaneously optimizing J_c expressed as Eq. (8) and *XB* expressed as Eq. (13) is named as MOKCW2. We have carried out MOKCW2 on all the data sets in this paper considered under both the "none" and the "standardization" situation and compared the results with that of MOKCW in terms of *Acc*, *RI* and *NMI*. Figure 5 shows the line chart for *Acc* comparison of MOKCW and MOKCW2 on all data sets considered in this study. We can observe that MOKCW performs better than MOKCW2 as a whole, in some cases such as Breast, Vertebral, WFRN, SVMguide1, Thyroid and Waveform, the superiority of MOKCW is obvious especially under the "none" situation. Therefore, it can be concluded that our improved objective function F_s is more beneficial to the evolution procedure of MOC, which can produce better clustering solutions.

In Shen et al. (2006), Zhou et al. (2016), the investigation about the distribution of four attributes of the Iris dataset was conducted as well as showing four features of its 150 members, which aimed at giving an intuitive understanding of the physical properties of the attribute weight assignment. The results showed that attribute 3 and attribute 4 of dataset Iris are more compact in each cluster, thus they should be more important and contribute much more than other two attributes in clustering. This can also be verified by Table 8 here, in which attribute weights of Iris are obtained by four algorithms MOEASSC, VKCM-K-LP, VKFCM-K-LP, and MOKCW. Each algorithm is independently executed 10 times and the mean value for each case is recorded. Note that, the constraints of weights for MOEASSC are $w_{kj} \in [0, 1], \sum_{j=1}^{d} w_{kj} = 1$, while that for the other three methods are $w_{kj} > 0, \prod_{j=1}^{d} w_{kj} = 1$. Besides, since the clustering performance of ESSC and E-AWA on Iris are undesirable and unstable according to Table 3 and Fig. 4, the attribute weights obtained are not presented in Table 8. It can be observed that attribute 3 and attribute 4 have higher weights than attribute 1 and attribute 2 for each attribute-weighted clustering algorithm, especially for VKCM-K-LP and MOKCW that can achieve higher clustering accuracy.

It should be pointed that MOEASSC and MOKCW need more execution time compared to single-objective clustering algorithms owing to the multiobjective optimization procedure, which has been a problem for all MOC methods. Also, during each generation of MOKCW, the kernelized adaptive distances need to be computed twice to update the child population. Table 9 shows the runtime of different algorithms considered in this paper, it is clearly observed that MOEASSC and MOKCW indicate larger results especially for datasets with large size, which represents a worse time efficiency. Hence, the MOC approaches cannot be used for time strict cases. It should be noted that, for single-objective clustering algorithms, usually they will stop in advance so that the total iterations are less than 100; for example, sometimes only near 10 iterations are needed. However, the two MOC methods will stop until $T_{\max}$ generations are executed that is worthless in some cases, and thus we should do some research about the stop criterion to reduce unnecessary time cost.

5.2.3 *Experiments on obtaining the best solution and cluster number*

For some larger datasets, the method *PSVIndex* + CE described in Sect. 3.6 is used to obtain the final clustering result. Table 10 shows the performance of three methods *PSVIndex*, *PSVIndex*+MCLA and *PSVIndex*+HGBF for datasets WFRN, SVMguide1, Thyroid, Occupancy, Shut-

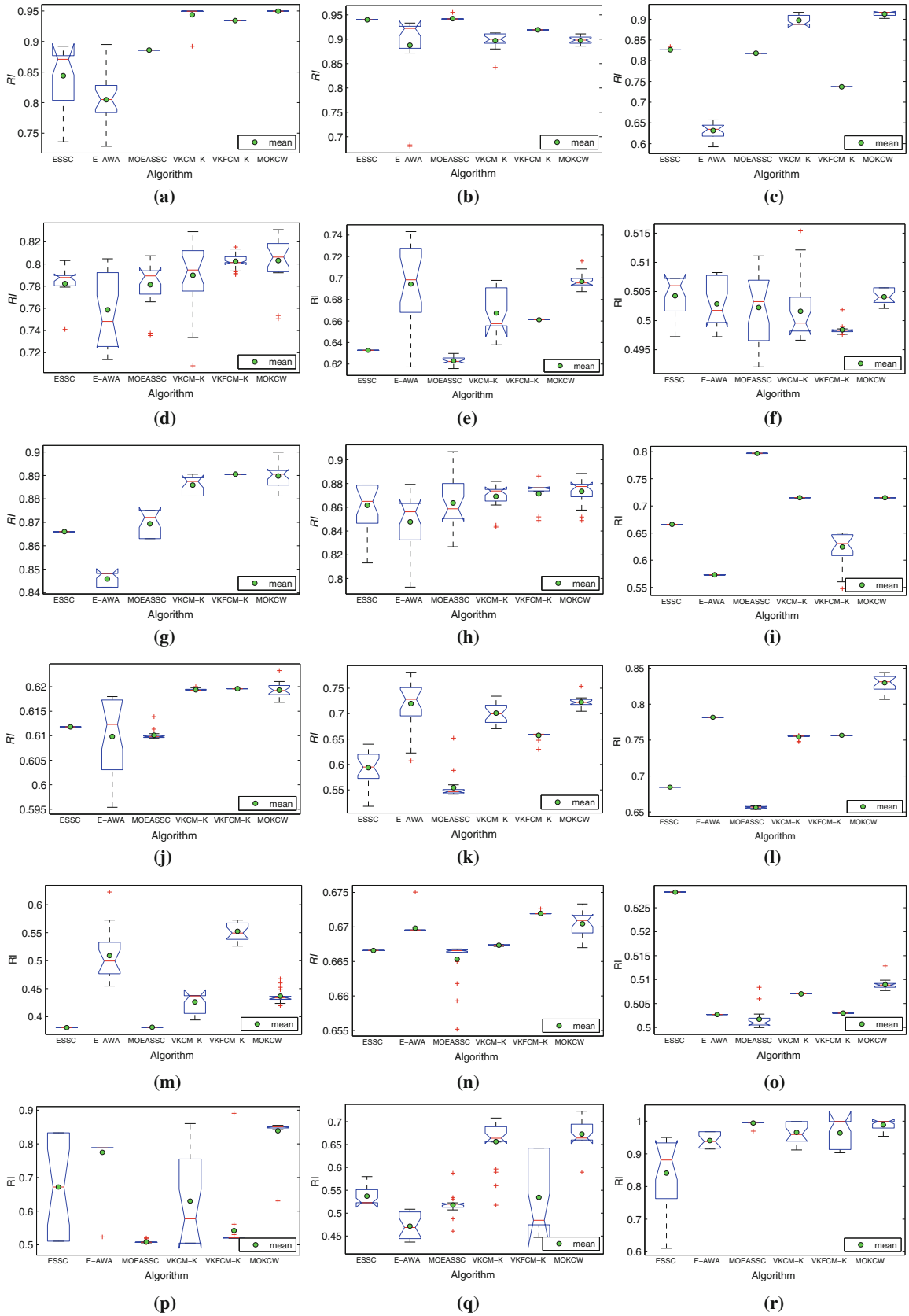

Fig. 4 Box-plots for *RI* of all algorithms. **a** Iris, **b** Wine, **c** Newthyroid, **d** Breast, **e** Vertebral, **f** Bupa **g** WDBC, **h** Image, **i** Seismic, **j** Abalone, **k** WFRN, **l** SVMguide1, **m** Thyroid, **n** Waveform, **o** Magic, **p** Occupancy, **q** Shuttle, **r** 5Gaussians

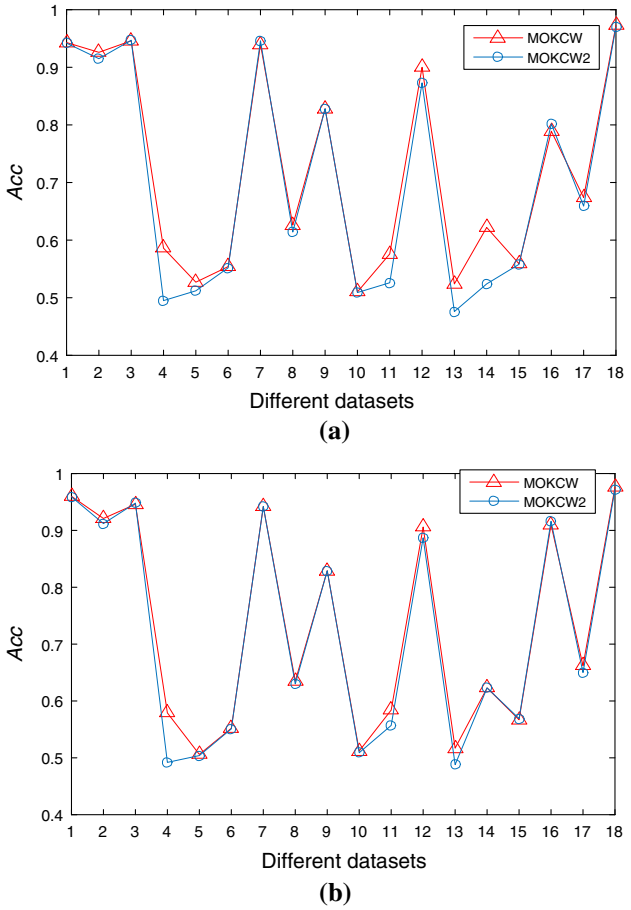

Fig. 5 The comparison of MOKCW and MOKCW2 on *Acc*. **a** None, **b** standardization. *Note* The horizontal ordinates of 1–18 respectively denotes datasets from Iris to 5Gaussians with the same order as shown in Fig. 4

tle, and 5Gaussians. Each method has been executed 50 times, and then the average values of *Acc*, *RI*, *NMI*, and runtime are computed. It should be noted that different results of MOKCW on dataset Abalone, Waveform, Magic, and Occupancy are always the same, and hence the corresponding values are not shown in Table 10. Among the 50 executions, the cases of $T \geq 3$ are respectively 37, 29, 33, 21 and 24 times for WFRN, SVMguide1, Thyroid, Shuttle, and 5Gaussians. The two *PSVIndex* + CE methods perform similarly, whose index values are a little better than that of *PSVIndex*. As described in Sect. 3.7, the computing complexity of *PSVIndex*, HGBF, MCLA is respectively $(NK(n_sP)^2d)$, $O(Knr)$, $O(nK^2r^2)$. Commonly K is small and r is 5 in this paper, hence the computing time of the two CE methods are short compared with *PSVIndex* such that can be ignored. Table 11 shows the runtime for selection methods (*PSVIndex* and *PSVIndex* + CE) in comparison, with which the two MOC methods are conducted on datasets Abalone, WFRN, SVMguide1, Thyroid, Waveform, Magic, Occupancy, Shuttle, and 5Gaussians. We can observe that the runtime obtaining the final clustering solution using *PSVIndex* + CE has been decreased by a large margin especially for some larger datasets Magic, Occupancy, Shuttle, and 5Gaussians, which indicates a very obvious efficiency superiority. For instance, MOEASSC and MOKCW respectively needs 12012.8 and 9464.27 s conducted on 5Gaussians using *PSVIndex*, whereas 149.822 s and 118.057 s are needed using *PSVIndex* + CE.

Table 8 Attribute weight assignment of different weighted clustering algorithms on Iris dataset

Algorithm	Cluster	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$
MOEASSC	1	0.224	0.204	0.279	0.293
	2	0.202	0.143	0.321	0.334
	3	0.239	0.147	0.289	0.324
VKCM-K-LP	1	0.163	0.018	33.72	9.850
	2	0.220	0.149	3.004	10.15
	3	0.203	0.231	3.681	5.795
VKFCM-K-LP	1	0.458	0.147	4.438	3.346
	2	0.601	0.471	1.857	1.905
	3	0.552	0.603	1.984	1.513
MOKCW	1	0.164	0.018	33.73	9.847
	2	0.220	0.149	3.011	10.16
	3	0.202	0.226	3.661	5.987

Table 12 shows the result of the number of clusters by using the *PSVIndex*-gap method for 5 times, where the result of Score K is the most frequently achieved one out of the 5 values and the value in the brackets denotes the time of finding the true number. As dataset 5Gaussians with merely two attributes can be drawn directly and the number of clusters can be determined, it does not need to be conducted on with the *PSVIndex*-gap method. For datasets Iris, Wine, Newthyroid, Vertebral, SVMguide1, Magic, and Occupancy, we can always obtain the score K values that are equal to the true K values of Table 1. Meanwhile, the true number of clusters can be obtained in most cases for datasets Bupa, Waveform, Thyroid so that the score K values are also equal to the true K values. However, wrong number of clusters are obtained for datasets Breast, WDBC, Image, Seismic, Abalone, WFRN, and Shuttle, despite the fact in some cases the correct number of clusters is obtained. This phenomenon also occurred in Xia et al. (2013) as the theoretical explanation cannot be provided. It maybe owing to the fact that the performances of the clustering method on these datasets are not very desirable to achieve the true result, except for dataset WDBC. Generally, the *PSVIndex*-gap method is suitable for both MOKCW and MOEASSC to check the correct number of clusters, but also it needs to be further improved to achieve more accurate results.

Table 9 Runtime of different algorithms (s)

Datasets	ESSC	E-AWA	MOEASSC	VKCM-K-LP	VKFCM-K-LP	MOKCW
Iris	0.090	0.013	0.475	0.056	0.067	1.145
Wine	0.058	0.020	0.579	0.072	0.121	1.786
Thyroid	0.042	0.019	0.611	0.077	0.127	1.299
Breast	0.169	0.020	0.561	0.059	0.268	1.890
Vertebral	0.082	0.022	0.839	0.128	0.149	1.557
Bupa	0.158	0.028	0.957	0.085	0.226	1.661
WDBC	0.068	0.018	3.199	0.558	0.788	4.533
Image	2.973	0.174	14.421	1.217	7.221	22.067
Seismic	0.419	0.033	2.146	1.608	2.126	2.903
Abalone	1.309	0.072	5.332	0.896	2.381	9.041
WFRN	1.176	0.089	6.464	1.119	1.603	7.948
SVMguide1	0.914	0.035	12.419	0.667	0.719	8.534
Thyroid	1.058	0.116	8.535	0.922	2.038	15.108
Waveform	2.006	0.594	20.427	2.007	13.313	59.836
Magic	3.342	0.302	6.421	1.517	3.029	54.693
Occupancy	0.459	0.163	23.795	0.448	2.841	26.081
Shuttle	58.082	6.934	213.891	25.423	29.872	221.364
5Gaussians	25.101	2.076	185.805	1.032	4.831	183.232

Table 10 Validity index results by different selection methods

Datasets	Methods	*Acc*	*RI*	*NMI*
WFRN	*PSVIndex*	0.592	0.719	0.385
	PSVIndex + MCLA	0.592	0.721	0.389
	PSVIndex + HGBF	0.593	0.722	0.39
SVMguide1	*PSVIndex*	0.899	0.819	0.585
	PSVIndex + MCLA	0.908	0.833	0.606
	PSVIndex + HGBF	0.905	0.828	0.598
Thyroid	*PSVIndex*	0.468	0.413	0.137
	PSVIndex + MCLA	0.482	0.418	0.146
	PSVIndex + HGBF	0.478	0.419	0.148
Shuttle	*PSVIndex*	0.658	0.664	0.377
	PSVIndex + MCLA	0.663	0.670	0.386
	PSVIndex + HGBF	0.662	0.671	0.385
5Gaussians	*PSVIndex*	0.978	0.984	0.966
	PSVIndex + MCLA	0.978	0.984	0.967
	PSVIndex + HGBF	0.976	0.983	0.964

Table 11 Runtime of obtaining the clustering solution by *PSVIndex* and *PSVIndex* + CE (s)

Datasets	Algorithms	*PSVIndex*	*PSVIndex* + CE
Abalone	MOEASSC	43.662	2.414
	MOKCW	28.879	1.294
WFRN	MOEASSC	73.279	3.925
	MOKCW	49.422	2.621
SVMguide1	MOEASSC	191.037	9.593
	MOKCW	175.293	8.908
Thyroid	MOEASSC	117.014	5.874
	MOKCW	85.060	4.421
Waveform	MOEASSC	114.813	5.907
	MOKCW	59.499	3.086
Magic	MOEASSC	1565.56	19.381
	MOKCW	1137.72	14.116
Occupancy	MOEASSC	1723.75	21.526
	MOKCW	1634.43	20.342
Shuttle	MOEASSC	2469.25	30.674
	MOKCW	1943.79	24.157
5Gaussians	MOEASSC	12012.8	149.822
	MOKCW	9464.27	118.057

5.2.4 Partition interpretation and cluster interpretation: the SVMguide1 data set

In order to show that how much the developed clusters are different from those produced by other methods, the partition and cluster interpretation indices introduced in Sect. 4 are utilized, and we consider the previous results obtained with the application of the VKCM-K-LP, VKFCM-K-LP, and MOKCW on the SVMguide1 dataset.

Firstly, the partition interpretation is taken into consideration, hence the overall heterogeneity index $Q(P)$ given by Eq. (26) and the overall heterogeneity index concerning the j-th attribute $Q_j(P)$ given by Eq. (27) are adopted. The partitions provided by the VKCM-K-LP, VKFCM-K-LP and

Table 12 Results of the number of clusters by the *PSVIndex*-gap method

Datasets	Score K	Datasets	Score K
Iris	3 (5)	Abalone	2 (1)
Wine	3 (5)	WFRN	2 (1)
Newthyroid	3 (5)	SVMguide1	2 (5)
Breast	2 (0)	Thyroid	3 (3)
Vetebral	3 (5)	Waveform	3 (4)
Bupa	3 (4)	Magic	2 (5)
WDBC	4 (0)	Occupancy	2 (5)
Image	4 (0)	Shuttle	3 (0)
Seismic	3 (2)	5Gaussians	–

Table 13 Quality of the partition concerning single attributes ($Q_j(P)$) for SVMguide1 dataset (%)

Algorithm	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$
VKCM-K-LP	88.16	93.97	79.23	86.29
VKFCM-K-LP	92.30	92.46	79.20	85.64
MOKCW	88.09	94.65	77.37	87.14

Table 14 Cluster heterogeneity indexes for the SVMguide1 data set (%)

Algorithm	Cluster	Cardinal	$J(k)$	$Q(P_k)$
VKCM-K-LP	1	3848	37.30	94.19
	2	3241	62.70	78.15
VKFCM-K-LP	1	4042	43.84	92.92
	2	3047	56.16	83.06
MOKCW	1	3603	33.33	95.40
	2	3486	66.67	77.11

Table 15 Cluster heterogeneity indexes of the attributes for the SVMguide1 data set (%)

Algorithm	Cluster	$\mathbf{x}_1$	$\mathbf{x}_2$	$\mathbf{x}_3$	$\mathbf{x}_4$
VKCM-K-LP	1	95.03	97.59	70.19	73.24
	2	33.18	41.74	82.40	89.38
VKFCM-K-LP	1	96.79	97.03	70.00	69.88
	2	74.86	72.73	84.86	91.35
MOKCW	1	95.30	98.09	71.20	65.88
	2	35.02	42.33	80.45	89.14

MOKCW take the values of $Q(P)$ as respectively 89.23%, 89.48% and 90.42%, which means that MOKCW outperforms the other two methods with respect to the quality of partition. Also, we can conclude that the set of attributes has the average discrimination power of all attributes that is relatively high, which can separate the dataset into homogeneous clusters well. For the index $Q_j(P)$, as observed from Table 13, the discriminate power of attribute $\mathbf{x}_2$ is higher than the average discriminate power for both algorithms VKCM-K-LP and MOKCW, and the discriminate powers of attributes $\mathbf{x}_1$ and $\mathbf{x}_2$ are higher than the average discriminate power for the algorithm VKFCM-K-LP.

Then, we take the cluster interpretation into consideration, hence the index $J(k)$ measuring the relative contribution of the cluster P_k to the overall within-cluster dispersion J, the index $Q(P_k)$ measuring the quality of a cluster P_k, as well as the index $Q_j(P_k)$ measuring the quality of a cluster P_k concerning the j-th attribute are adopted. Table 14 indicates the $J(k)$ and $Q(P_k)$ values for the VKCM-K-LP, VKFCM-K-LP and MOKCW. It can be concluded that, for the three algorithms, cluster 2 is more heterogeneous according to $J(k)$, while cluster 1 has a better quality index according to $Q(P_k)$.

Table 15 indicates the $Q_j(P_k)$ values for the three algorithms VKCM-K-LP, VKFCM-K-LP, and MOKCW, that is the cluster heterogeneity index of each attribute. We can observe that, the qualities of cluster 1 concerning attributes $\mathbf{x}_1$ and $\mathbf{x}_2$ are much better than the other cases for all the three algorithms. Moreover, it should be pointed that a larger $Q_j(P_k)$ value compared to $Q(P_k)$ value means that the j-th attribute characterizes the cluster Pk. For all the three algorithms, comparing the value of $Q_j(P_k)$ in Table 15 with the value of $Q(P_k)$ in Table 14, it can be concluded that attribute $\mathbf{x}_1$ and $\mathbf{x}_2$ characterize cluster 1, whereas cluster 2 is characterized by attributes $\mathbf{x}_3$ and $\mathbf{x}_4$.

5.2.5 Statistical significance test

To verify that the result is statistically significant when comparing different clustering approaches, the Wilcoxon rank-sum test has been conducted at the 5% significance level, through which the p values can be calculated. In this study, the p values for *RI* comparison of MOKCW and other algorithms at a time are represented in Table 16. As a null hypothesis, it is assumed that there are no significant differences between *RI* of the two groups (a group corresponding to MOKCW and another corresponding to some other algorithms), whereas the alternative hypothesis is that there is a significant difference in the mean values of the two groups.

Note that as this is a multiple comparison test, we have set the p values threshold to 0.01 (0.05/5) according to Bonferroni inequality to achieve an overall 5% significance level. Table 16 shows that most p values reported are less than 0.01, which is a strong evidence against the null hypothesis. Hence we can conclude that the better *RI* values produced are statistically significant and have not occurred by chance. Similar results are obtained for *Acc* and *NMI* comparisons of MOKCW and other algorithms, establishing the significant superiority of the proposed method.

Table 16 The p values for RI comparison between MOKCW and other methods produced by the Wilcoxon rank-sum test

Datasets	ESSC	E-AWA	MOEASSC	VKCM-K-LP	VKFCM-K-LP
Iris	7.6468E−009	7.9334E−009	4.6827E−010	Same	4.6827E−010
Wine	7.6187E−009	Same	1.0742E−008	Same	7.6187E−009
Newthyroid	6.8159E−009	4.5378E−008	4.7442E−009	2.2741E−005	4.7442E−009
Breast	3.4037E−005	3.7020E−005	5.6122E−004	Same	Same
Vertebral	7.9919E−009	Same	6.7193E−008	1.0265E−004	1.5124E−008
Bupa	0.0066	Same	Same	0.0039	6.2860E−008
WDBC	6.1036E−009	2.8920E−008	3.0773E−008	0.0068	Same
Image	Same	9.2709E−005	Same	Same	Same
Seismic	3.1034E−009	1.7750E−008	1.4868E−008	Same	3.2853E−008
Abalone	8.0065E−009	2.9334E−007	6.7478E−008	Same	Same
WFRN	6.7956E−008	Same	6.7860E−008	4.5979E−004	6.1004E−008
SVMguide1	8.0065E−009	2.8636E−008	6.4034E−008	2.9458E−008	8.0065E−009
Thyroid	8.0065E−009	1.6439E−007	6.6438E−008	Same	6.7478E−008
Waveform	7.7176E−009	Same	6.5970E−008	1.0959E−006	0.0036
Magic	7.9919E−009	3.9533E−008	1.3239E−007	7.9919E−009	1.9415E−008
Occupancy	1.9169E−007	2.3837E−007	6.7956E−008	0.0013	1.1981E−006
Shuttle	2.8322E−008	6.7288E−008	6.7098E−008	Same	1.7643E−007
5Gaussians	2.0387E−008	1.2490E−005	0.0031	0.0083	6.9462E−005

6 Conclusion

In this paper, a new clustering method named as MOKCW is developed, in which the main innovation lies in three effective aspects. The first one is the introduction of multiobjective optimization into feature weighted kernel clustering algorithms, which is the first attempt to our best knowledge. In terms of the optimization viewpoint, the second contribution is owing to a novel objective function F_s, where the distance between each pair of centers and that between each center and the global center are computed as the denominator term to measure the intercluster separation. In order to cope with large datasets, we develop a novel method named as *PSVIndex* + CE to efficiently obtain the final clustering solution by incorporating the clustering ensemble strategy into the original *PSVIndex* approach, which is the third contribution.

The performances of all methods considered in this study are evaluated through comprehensive experiments carried out with eighteen benchmark datasets, and the results indicate that our proposed method performed far better than the state-of-the-art methods. Meanwhile, the usefulness of F_s is demonstrated in terms of the overall clustering precision, while that of *PSVIndex* + CE is demonstrated in terms of time efficiency especially regarding large datasets. Moreover, an application with the SVMguide1 dataset shows the merit of the partition and cluster interpretation tools.

There are some scopes of future research to extend the proposed method. To effectively reduce the runtime, other faster algorithm can be utilized as the underlying optimization tool. It is possible that some more suitable criterions will be further investigated, and we may develop novel objective functions that are capable of getting clustering result close to the true partition. Furthermore, some other possible applications of the proposed method to more complicated tasks will be investigated.

Acknowledgements This study was funded by the Natural Science Foundation of China (Grant No. 61373126) and the Fundamental Research Funds for the Central Universities of China (Grant No. JUSRP51510).

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Alok AK, Saha S, Ekbal A (2016) Multi-objective semi-supervised clustering for automatic pixel classification from remote sensing imagery. Soft Comput 20(12):4733–4751

Arthur D, Vassilvitskii S (2007) k-means++: the advantages of careful seeding. In: Proceedings of the 18 annual ACM-SIAM symposium on discrete algorithms, pp 1027–1035

Bai L, Liang J (2014) The k-modes type clustering plus between-cluster information for categorical data. Neurocomputing 133:111–121

Bai L, Liang J, Dang C, Cao F (2011) A novel attribute weighting algorithm for clustering high-dimensional categorical data. Pattern Recognit 44(12):2843–2861

Bai L, Liang J, Dang C, Cao F (2013) A novel fuzzy clustering algorithm with between-cluster information for categorical data. Fuzzy Sets Syst 215:55–73

Benaichouche AN, Oulhadj H, Siarry P (2016) Multiobjective improved spatial fuzzy c-means clustering for image segmentation combining Pareto-optimal clusters. J Heuristics 22(4):383–404

Capitaine HL, Frlicot C (2011) A cluster-validity index combining an overlap measure and a separation measure based on fuzzy-aggregation operators. IEEE Trans Fuzzy Syst 19(3):580–588

Chan EY, Ching WK, Ng MK, Huang JZ (2004) An optimization algorithm for clustering using weighted dissimilarity measures. Pattern Recognit 37(5):943–952

Chang C-C, Lin C-J (2011) LIBSVM: a library for support vector machines. ACM Trans Intell Syst Technol: TIST 2(3):1–27

Chavent M, de Carvalho FA, Lechevallier Y, Verde R (2006) New clustering methods for interval data. Comput Stat 21(2):211–229

Coelho AL, Fernandes E, Faceli K (2010) Inducing multi-objective clustering ensembles with genetic programming. Neurocomputing 74(1):494–498

de Amorim RC, Mirkin B (2012) Minkowski metric, feature weighting and anomalous cluster initializing in K-means clustering. Pattern Recognit 45(3):1061–1075

Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):97–182

Deng Z, Choi K-S, Chung F-L, Wang S (2010) Enhanced soft subspace clustering integrating within-cluster and between-cluster information. Pattern Recognit 43(3):767–781

Faceli K, de Souto MC, de Arajo DS, de Carvalho AC (2009) Multi-objective clustering ensemble for gene expression data analysis. Neurocomputing 72(13):2763–2774

Fern XZ, Brodley CE (2004) Solving cluster ensemble problems by bipartite graph partitioning. In: Proceedings of the 21 international conference on Machine learning, pp 1–8

Ferreira MR, de Carvalho FA (2014a) Kernel-based hard clustering methods in the feature space with automatic variable weighting. Pattern Recognit 47(9):3082–3095

Ferreira MR, De Carvalho FDA (2014b) Kernel fuzzy c-means with automatic variable weighting. Fuzzy Sets Syst 237:1–46

Ferreira MR, de Carvalho FDA, Simoes EC (2016) Kernel-based hard clustering methods with kernelization of the metric and automatic weighting of the variables. Pattern Recognit 51:310–321

Gan G, Wu J (2008) A convergence theorem for the fuzzy subspace clustering (FSC) algorithm. Pattern Recognit 41(6):1939–1947

Gan G, Ng MK-P (2015) Subspace clustering with automatic feature grouping. Pattern Recognit 48(11):3703–3713

Garcia-Piquer A, Fornells A, Orriols-Puig A, Corral G, Golobardes E (2012) Data classification through an evolutionary approach based on multiple criteria. Knowl Inf Syst 33(1):35–56

Garcia-Piquer A, Fornells A, Bacardit J, Orriols-Puig A, Golobardes E (2014) Large-scale experimental evaluation of cluster representations for multiobjective evolutionary clustering. IEEE Trans Evol Comput 18(1):36–53

Graves D, Pedrycz W (2010) Kernel-based fuzzy clustering and fuzzy clustering: a comparative experimental study. Fuzzy Sets Syst 161(4):522–543

Halkidi M, Vazirgiannis M (2001) Clustering validity assessment: finding the optimal partitioning of a data set. In: Proceedings of the 2001 IEEE international conference on data mining, pp 187–194

Hancer E, Karaboga D (2017) A comprehensive survey of traditional, merge-split and evolutionary approaches proposed for determination of cluster number. Swarm Evol Comput 32:49–67

Handl J, Knowles J (2007) An evolutionary approach to multiobjective clustering. IEEE Trans Evol Comput 11(1):56–76

Huang JZ, Ng MK, Rong H, Li Z (2005) Automated variable weighting in k-means type clustering. IEEE Trans Pattern Anal Mach Intell 27(5):657–668

Huang X, Ye Y, Zhang H (2014a) Extensions of kmeans-type algorithms: a new clustering framework by integrating intracluster compactness and intercluster separation. IEEE Trans Neural Netw Learn Syst 25(8):1433–1446

Huang X, Ye Y, Guo H, Cai Y, Zhang H, Li Y (2014b) DSKmeans: a new kmeans-type approach to discriminative subspace clustering. Knowl Based Syst 70:293–300

Jing L, Ng MK, Huang JZ (2007) An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. IEEE Trans Knowl Data Eng 19(8):1026–1041

Ji J, Wang K-L (2014) A robust nonlocal fuzzy clustering algorithm with between-cluster separation measure for SAR image segmentation. IEEE J Sel Top Appl Earth Obs Remote Sens 7(12):4929–4936

Jos-Garcła A, Gmez-Flores W (2016) Automatic clustering using nature-inspired metaheuristics: a survey. Appl Soft Comput 41:192–213

Li Y, Wei Y, Wang Y, Jiao L (2014) Multi-objective evolutionary for synthetic aperture radar image segmentation with non-local means denoising. Nat Comput 13(1):39–53

Liu R, Zhang L, Li B (2015) Synergy of two mutations based immune multi-objective automatic fuzzy clustering algorithm. Knowl Inf Syst 45(1):133–157

Ma A, Zhong Y, Zhang L (2015) Adaptive multiobjective memetic fuzzy clustering algorithm for remote sensing imagery. IEEE Trans Geosci Remote Sens 53(8):4202–4217

Mukhopadhyay A, Maulik U (2011) A multiobjective approach to MR brain image segmentation. Appl Soft Comput 11(1):872–880

Mukhopadhyay A, Maulik U, Bandyopadhyay S (2009) Multiobjective genetic algorithm-based fuzzy clustering of categorical attributes. IEEE Trans Evol Comput 13(5):991–1005

Mukhopadhyay A, Maulik U, Bandyopadhyay S (2013) An interactive approach to multiobjective clustering of gene expression patterns. IEEE Trans Biomed Eng 60(1):35–41

Mukhopadhyay A, Maulik U, Bandyopadhyay S, Coello CAC (2014) Survey of multiobjective evolutionary algorithms for data mining: part II. IEEE Trans Evol Comput 18(1):20–35

Prakash J, Singh P (2015) An effective multiobjective approach for hard partitional clustering. Memet Comput 7(2):93–104

Sag T, Cunkas M (2015) Color image segmentation based on multiobjective artificial bee colony optimization. Appl Soft Comput 34:389–401

Saha S, Bandyopadhyay S (2013) A generalized automatic clustering algorithm in a multiobjective framework. Appl Soft Comput 13(1):89–108

Saha I, Maulik U (2014) Incremental learning based multiobjective fuzzy clustering for categorical data. Inf Sci 267:35–57

Saha I, Maulik U, Plewczynski D (2011) A new multi-objective technique for differential fuzzy clustering. Appl Soft Comput 11(2):2765–2776

Saha S, Ekbal A, Gupta K, Bandyopadhyay S (2013) Gene expression data clustering using a multiobjective symmetry based clustering technique. Comput Biol Med 43(11):1965–1977

Saha S, Spandana R, Ekbal A, Bandyopadhyay S (2015) Simultaneous feature selection and symmetry based clustering using multiobjective framework. Appl Soft Comput 29:479–486

Saha S, Alok AK, Ekbal A (2016) Brain image segmentation using semi-supervised clustering. Expert Syst Appl 52(15):50–63

Shen H, Yang J, Wang S, Liu X (2006) Attribute weighted mercer kernel based fuzzy clustering algorithm for general non-spherical datasets. Soft Comput 10(11):1061–1073

Strehl A, Ghosh J (2003) Cluster ensembles—a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 3:583–617

Tibshirani R, Walther G, Hastie T (2001) Estimating the number of clusters in a data set via the gap statistic. J R Stat Soc Ser B (Stat Methodol) 63(2):411–423

Wang J, Deng Z, Choi K-S, Jiang Y, Luo X, Chung F-L, Wang S (2016) Distance metric learning for soft subspace clustering in composite kernel space. Pattern Recognit 52:113–134

Wikaisuksakul S (2014) A multi-objective genetic algorithm with fuzzy c-means for automatic data clustering. Appl Soft Comput 24:679–691

Wu K-L, Yu J, Yang M-S (2005) A novel fuzzy clustering algorithm based on a fuzzy scatter matrix with optimality tests. Pattern Recogn Lett 26(5):639–652

Wu C, Ouyang C, Chen L, Lu L (2014) A new fuzzy clustering validity index with a median factor for centroid-based clustering. IEEE Trans Fuzzy Syst 23(3):701–718

Xia H, Zhuang J, Yu D (2013) Novel soft subspace clustering with multi-objective evolutionary approach for high-dimensional data. Pattern Recognit 46(9):2562–2575

Yang D, Jiao L, Gong M, Liu F (2011) Artificial immune multi-objective SAR image segmentation with fused complementary features. Inf Sci 181(13):2797–2812

Yang C-L, Kuo R, Chien C-H, Quyen NTP (2015) Non-dominated sorting genetic algorithm using fuzzy membership chromosome for categorical data clustering. Appl Soft Comput 18(1):20–35

Zhao F, Liu H, Fan J (2015) A multiobjective spatial fuzzy clustering algorithm for image segmentation. Appl Soft Comput 30:48–57

Zhong Y, Zhang S, Zhang L (2013) Automatic fuzzy clustering based on adaptive multi-objective differential evolution for remote sensing imagery. IEEE J Sel Top Appl Earth Obs Remote Sens 6(99):1–12

Zhou J, Chen L, Chen CLP, Zhang Y, Li H (2016) Fuzzy clustering with the entropy of attribute weights. Neurocomputing 198:34–125

Zhu L, Cao L, Yang J (2012) Multiobjective evolutionary algorithm-based soft subspace clustering. In: Proceedings of the 2012 IEEE international conference on Evolutionary Computation, pp 1–8