Applied Soft Computing Journal 78 (2019) 324–345

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

New fuzzy C-means clustering method based on feature-weight and cluster-weight learning

Mahdi Hashemzadeh *, Amin Golzari Oskouei, Nacer Farajzadeh

Faculty of Information Technology and Computer Engineering, Azarbaijan Shahid Madani University, Tabriz, Iran

HIGHLIGHTS

- Feature weighting and cluster weighting are used to improve the FCM clustering method.
- Feature weighting is carried out based on importance of each feature in the clusters.
- Weights of the clusters are calculated considering the importance of their features.
- Feature weighting and cluster weighting are performed simultaneously.

ARTICLE INFO

Article history:
Received 22 August 2018
Received in revised form 18 February 2019
Accepted 22 February 2019
Available online 27 February 2019

Keywords:
Fuzzy C-means
Clustering
Feature weighting
Cluster weighting
Local feature weighting

ABSTRACT

Among fuzzy clustering methods, fuzzy c-means (FCM) is the most recognized algorithm. In this algorithm, it is assumed that all the features are of equal importance. In real applications, however, the importance of the features are different and there exist some features that are more important than the others. These important features should basically have more effects than the other features in the forming of optimal clusters. The basic FCM algorithm does not support this idea. Also, the FCM algorithm suffers from another problem; the algorithm is very sensitive to initialization, whereas a bad initialization leads to a poor local optima. Some improved versions of FCM have been proposed in the literature, each of which has somehow mitigated the first problem or the second one. In this paper, motivated by these weaknesses of the FCM, the goal is to solve the two problems at the same time. In doing so, an automatic local feature weighting scheme is proposed to properly weight the features of each clusters. And, a cluster weighting process is performed to mitigate the initialization sensitivity of the FCM. Feature weighting and cluster weighting are performed simultaneously and automatically during the clustering process resulting in high quality clusters, regardless of the initial centers. Extensive experiments conducted on a synthetic dataset and 16 real world datasets indicate that the proposed algorithm outperforms the state-of-the-arts algorithms. The convergence proof of the proposed algorithm is also provided.

1. Introduction

Automatic data partitioning-clustering is one of the most important tools in data mining, and the related areas such as machine learning, machine vision and pattern recognition [1,2]. The goal is to divide a set of samples into homogeneous partitions (clusters) based on an objective function, so as to increase intra-cluster similarity and decrease inter-cluster similarity. In recent decades, various algorithms have been introduced for clustering of which k-means [3] and fuzzy c-means (FCM) [4] are the most well-known algorithms among the rest. Some other versions of these algorithms have also been proposed (more details and evaluations can be found in [5–9]). However, these two algorithms have always drawn the interest of researchers due to some key ideas that they possess; simple structure, easy implementation, fast convergence, and need for low storage capacity to name.

When the boundaries among the clusters are vague, FCM demonstrates a better performance than k-means [10,11] and has shown successful results in various applications such as [12–20]. This is due to the fuzzy nature of this method and thus, it applies soft-clustering in contrast to *k*-means that does hard clustering. Furthermore, the fuzzy membership matrix in this method provides us a better understanding of the complicated relationships between the clusters and the samples [21].

In most real world applications, obviously, some of the extracted features are irrelevant to the target problem or are less

* Correspondence to: Azarbaijan Shahid Madani University, Tabriz-Azarshahr Road, Tabriz, 5375171379, Iran.
E-mail addresses: hashemzadeh@azaruniv.ac.ir (M. Hashemzadeh), a.golzari@azaruniv.ac.ir (A.G. Oskouei), n.farajzadeh@azaruniv.ac.ir (N. Farajzadeh).

https://doi.org/10.1016/j.asoc.2019.02.038
1568-4946/

important than the others. FCM and k-means algorithms however, do not take the importance of the features into account, and therefore, their performances drop markedly in some applications [22]. In addition, these algorithms are sensitive to the initial cluster points, whereas a bad initialization may lead to a poor local optima [23,24].

In order to take the advantages of the important features over the others in the clusters formation, one may apply a weighting scheme on the features [9,25,26]. There are two groups of methods for doing so. The first group includes the algorithms that assign weights to the features globally. That is, a given feature is assigned with only one weight along all the clusters. The methods in the second group assign local weights to the features, and thus, a given feature has different weights in different clusters. This scheme has shown a better performance than the global weighting [9]. For example, the performance of a global weighting method presented in [27] was improved by Chan et al. [28] employing a local weight scheme.

For handling the sensitivity of FCM to initialization, there are also two groups of approaches. In the first group, the policy is to systematically prevent the formation of poor-quality clusters during the algorithm restarts (repeats). While in the second group, the policy is to make the algorithms independent of a random initialization, hence they do not need any restarting procedure. For example, in global k-means algorithm [29] and some of its modifications such as methods in [30,31] the process starts from a single cluster, and incrementally other clusters are added according to a criterion. In fact, at each step of algorithm run, a new cluster is added to the solution. Despite the fact that these algorithms are somehow able to mitigate the sensitivity of the algorithms to the initialization, their computational cost in general is very high.

In this paper, in order to overcome the above mentioned problems simultaneously, we propose a new FCM clustering algorithm based on a combination of feature weighting and cluster weighting. Feature weighting is carried out locally and automatically during the clustering procedure. At the same time, in order to deal with the problem of initialization, we apply cluster weighting as well. The weights of the clusters are calculated at each restart based on the sum of their members' intra-cluster distances and member's current features' weights. Calculating the weights of the clusters in this way prevents the creation of clusters that have a large sum of intra-cluster weighted-feature distances (SIWD[1]). And consequently, allows high quality clusters to be formed regardless of the initial centers.

In addition to the combination of feature weighting and cluster weighting, an objective function based on a non-Euclidean distance metric is used in the proposed method. The advantage of using this distance metric is that the noise and outliers cannot affect the feature weighting process very much [32,33].

The contributions of this work are threefold:

(1) Feature weighting is carried out locally in a way that the features have different weights depending on their importance in the clusters, thus increasing the quality of clustering;
(2) The weight of the clusters is calculated dynamically while taking the importance of the features in each cluster into consideration;
(3) Feature weighting and cluster weighting are performed simultaneously and automatically during the clustering process resulting in high quality clusters regardless of the initial centers.

The performance of the proposed algorithm is evaluated over a large number of standard datasets and extensive experiments are performed to examine the effectiveness of each solution proposed for feature weighting and initialization problem. The results show that our algorithm achieves a higher performance compared to the existing competitors.

The rest of the paper is organized as follows: Section 2, gives a brief review to the related work; in Section 3, the proposed algorithm, its convergence proof and its complexity analysis are provided; Section 4 presents the results of experiments; and finally, Section 5 deals with conclusion and ideas for future work.

2. Related work

In this section, we review the existing clustering methods that have been proposed to improve the clustering performance of the basic FCM and k-means algorithms; Section 2.1 reviews the related work based on feature weighting scheme, and Section 2.2 reviews the methods against initialization sensitivity.

2.1. Global and local feature weighting

There are two groups of methods for weighting the features in the clustering approaches. As a pioneering work (in 1984), SYNCLUS algorithm used global feature weighting technique in order to determine the importance of different features in the clustering process [34]. This algorithm starts by an initial set of feature weights and uses the k-means algorithm to cluster the data into k clusters. Then, it tries to find optimal weights by optimizing a weighted mean-square cost function. These steps are iterated until the procedure converges to an optimal set of weights. Although SYNCLUS Algorithm computes the feature weights automatically, the feature group weights should be given manually. Moreover, this algorithm is very time-consuming [35], so it may not be efficiently used for large datasets.

In 2004, Wang et al. [36] also used global feature weighting scheme for FCM. To weight the features, their method utilized a learning based approach and an evolutionary fitness function. The gradient descent algorithm was employed to find proper weights. The reported results showed improvements on the original FCM. However, the evolutionary function and the similarity measure used in this method were very complicated. The complexity of this method is high. This is due to a traditional way of feature-weight learning process that depends on a high number of iterations. Also, this method is based on an assumption that the distribution of the data is uniform which is not the case in most of the real world datasets.

Huang et al. [27] in 2005 proposed W-k-means clustering algorithm. They added an additional step to the basic k-means algorithm in order to determine the weights of features at each iteration. The weight of each feature was estimated based on the sum of the within-cluster variances of the feature. As such, noise features can be identified and their effects on the clustering result can be reduced. However, their method requires users to subjectively specify an additional parameter, so-called the exponent of feature weights. Hence, it is laborious for users to determine a proper value for this parameter to obtain a high clustering quality result [37]. In addition, feature weights generated using this method might not highlight the representative features (or dim irrelevant features) well. It is also observed that the weight differences among features are somehow unobvious when the number of features is high [37].

[1] In this paper, $Dz_k = \sum_{n=1}^{N}\sum_{m=1}^{M} u_{nk}{}^{\alpha} w_{km}^{q} d^2(x_{nm} - c_{km})$ is defined as the sum of the intra-cluster weighted-feature distance (SIWD). Here, N refers to the number of data samples, M is the number of features, u_{nk} indicates the degree of membership of the nth sample to the kth cluster, c_{km} indicates the center of kth cluster which is defined by u_{nk}, w_{km} refers to the weight of mth feature in the kth cluster, α is the fuzzification coefficient ($\alpha > 1$), and $d^2(x_{nm} - c_{km})$ is a distance criteria of non-Euclidean type.

In 2008, Hung et al. [38] presented a FCM-based algorithm for image segmentation that was carried out through global feature weighting. They estimated the feature weights using the bootstrap technique. Although their method has high computational complexity, their proposed approach has shown a good effectiveness and performance and it performed better than the aforementioned FWL [36] approach. However, the feature weights calculated by this algorithm may be unsuitable for some extreme cases [39]. In other words, in some datasets, the weight of the features does not properly represent the importance of the features [39].

In 2014, Xing et al. [40] presented two methods to improve the performance of FCM algorithm. In their basic method, they introduced a global feature weighting algorithm, namely IFWFCM. IFWFCM automatically computed feature weights in the clustering process. They then generalized the proposed method to a kernelized version by utilizing a kernelized distance measure (IFWFCM_KD). Experimental results revealed that the performance of IFWFCM is better than IFWFCM_KD on certain datasets.

Recently, in contrast to the global feature weighting methods, local feature weighting methods have gained more attentions. For example, the method proposed by Frigui et al. [9] determines the weights of features for each cluster independently through an optimization procedure during the clustering process. Compared to a global weighting method proposed in [38], their method showed a higher accuracy. They applied their algorithm for image segmentation and achieved interesting results.

Frigui et al. [9] introduced two local feature weighting methods based on simultaneous clustering and attribute discrimination techniques (SCAD1 and SCAD2). In the second version of their algorithm (SCAD2), a central weighting scheme was used that can make the algorithm very sensitive to the initialization of centers. Also, in these two algorithms, in some situations, the obtained weights of the features do not properly represent the importance of features in the clusters [41].

Jing et al. [42] extended k-means algorithm to determine feature weights in each clusters, and used the weights' values to find the important features. To this end, they defined an objective function using the weight entropy for their clustering algorithm. Although their algorithm was sensitive to initial centers, in the case of proper selection of initial centers, it was able to improve the clustering performance. In 2014, Zhi et al. [33] presented a hard c-means based clustering method which automatically assigned local weights for features during the clustering process. They used new distance metric that reduced the effect of noise and outliers in the clustering process. In fact, it was an improved version of the method presented in [28]. Although the proposed method was not sensitive to noise and outliers, it was sensitive to the selection of the initial points.

In 2014, Ferreira et al. [43] carried out extensive researches on feature weighting both locally and globally. Using basic FCM and kernel distance, they introduced a clustering method which was able to improve the clustering qualities by utilizing adaptive distances. According to the experimental analysis of these algorithms, the method using local adaptive distance are superior to the method using global adaptive distance in most cases [44]. Although these methods can produce better results than traditional FCM, they are unsuitable for clustering large datasets [44].

In 2016, a maximum-entropy-regularized weighted fuzzy c-means (EWFCM) method was introduced in [22]. In this method, in order to find the optimal feature weights, the feature-weight entropy regularization technique was introduced to be included in the objective function of the clustering process. Also, the kernel based EWFCM (KEWFCM) method was proposed for clustering the data that includes non-spherical shaped clusters.

In 2016, in [45], two techniques were proposed for weighting a multivariate FCM method. In the first technique, each sample, feature and cluster has a proper weight. The aim is to determine the relevance of each feature when calculating the degree of membership for a sample regarding a cluster. The second one introduced a weighted distance which was used to consider the variability of each feature and cluster. The weights were computed at each iteration of algorithm. These adaptive distances allow the clustering algorithm to find clusters with different sizes and shapes.

In a recently proposed algorithm [46], a new objective function based on a kernel metric is proposed. They used local feature weighting scheme to find those clusters that have linearly non-separable patterns or non-hyper-spherical shapes. In their work, a multi-objective optimization method was proposed to be used in a feature weighted clustering process. Two separate objective functions, taking into account the inter cluster separation and intra cluster compactness, were optimized simultaneously.

In conclusion, algorithms that utilize the local feature weighting are highly accurate and perform better than global feature weighting algorithms. On the other hand, both types of feature weighting algorithms are highly sensitive to the selection of initial centers. If the initial centers are not appropriately selected, the algorithm's efficiency drops sharply. Moreover, some of the existing local feature weighting methods are not able to accurately weigh the features based on their real importance in some situations, so the accuracy of the clustering is reduced.

2.2. Methods against initialization sensitivity

In this section, a review of the methods that have been proposed to solve bad initialization problem are provided. These methods can be categorized into two groups.

In the first group, the policy is to automatically prevent the formation of poor-quality clusters during the algorithm restarts (repeats). For example, in [47], the initial centers were determined using a stochastic function to cover the entire data space. k-means++ method [47] initializes the centers in k-means algorithm by choosing the samples that are further far from each other. The main drawback of this method from the scalability point of view is its inherent sequential nature. That is, the choice of the next center depends on the current set of centers [48]. Accordingly, k-means++ can be only applied on datasets of moderate size and only for modest values of k [48]. Methods in [49,50] randomly select initial centers, and during the clustering process, penalize the clusters with respect to the winning frequency of their representatives. Method in [49] was combined with a frequency sensitive competitive learning as a counter mechanism that penalizes the assignment of a data point to a crowded cluster. The principle is intuitively appealing, but it was approached without the emphasis on efficiency or quality. Also, complexity analysis and the corresponding empirical comparisons were not provided [51]. Method in [50] is suitable for clustering linearly non-separable patterns or non-hyper-spherical datasets, as it uses kernel distances. However, this method has difficulties in finding a suitable kernel function to map the data points from the input space to the linear kernel space.

Method in [52] introduced a technique to refine the initial point to a point probably to be close to the modes of the joint probability density of the samples. The main goal of this method was to cope with large datasets. The authors of this method argued that their algorithm leads to a refined starting seed that is not corrupted by outliers or other influential data points. However, the main problem with random methods is that they do not guarantee obtaining an optimal solution [53]. A study in [54] introduced MinMax k-means method to deal with the sensitivity of k-means algorithm to the initial points by modifying k-means objective. This method starts using arbitrary centers and then, attempts to refine them by minimizing the maximum intra-cluster

sum of squared error. This method has a good performance for balanced datasets, however, it does not work well for imbalanced databases.

In the second group, the policy is to make the algorithms independent of a random initialization; hence, they do not need any restarting procedure. A global kernel k-means method was presented in [55], which utilized a kernel based clustering technique to incrementally identify nonlinearly separable clusters. This method is an incremental approach, where at each stage, one cluster is added through a global search process consisting of several repetitions of kernel k-means from proper initializations. Another incremental and deterministic clustering algorithm was presented in [56] to deal with the bad initialization problem. In this algorithm, a kernel function is used to map data samples from the input space to a higher dimensional feature space. Then, the clustering error is optimized using the data points in new space. Both methods presented in [55] and [56] have a high computational complexity.

We can conclude that although the aforementioned algorithms are not sensitive to the selection of the initial points, they suffer from three shortcomings. First, they assume that all the features have the equal importance. Second, their computational complexities are relatively high [54]. Three, as most of these algorithms use Euclidean distance, they are sensitive to noise and outliers [33].

Considering the all above described limitations of existing methods, we propose a FCM clustering method that is able to perform feature weighting and cluster weighting, simultaneously. Feature weighting is done locally and at the same time, cluster weighting is applied to cope with the problem of initialization. Carrying out these two operation in a concurrent manner during the clustering process, form high quality clusters regardless of the initial centers. Also, a non-Euclidean distance metric is used in our objective function which makes the algorithm more robust against the effects of outliers in the feature weighting process.

3. Proposed algorithm

In this section, we introduce our clustering algorithm. First, an overview of the proposed algorithm is presented. Then, the steps of the algorithm are described in detail, and finally, convergence proof of the algorithm and the analysis of its complexity are provided.

3.1. Algorithm overview

Motivated by the problems of standard FCM clustering algorithm, we aim to design an approach that employs a local feature weighting scheme to increase the accuracy of the clustering, and a cluster weighting technique to alleviate the sensitivity of the clustering to bad initialization as well. In addition, since the Euclidean distance metric is sensitive to noise and outliers [32], we define a new objective function based on a non-Euclidean distance metric. This distance metric is introduced in [33] and is not sensitive to noise and outliers.

The objective function in the proposed method is as follows:

$$F(\mathbf{U},\mathbf{C},\mathbf{W},\boldsymbol{z}) = \sum_{n=1}^{N}\sum_{k=1}^{K}\sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} z_k^{p} d^2(x_{nm} - c_{km}). \quad (1)$$

In Eq. (1), N refers to the number of data samples, M is the number of features, K is the number of clusters, $\mathbf{U} = [u_{nk}]$ is a K by N matrix in which u_{nk} indicates the degree of nth sample's membership to the center of kth cluster, $\mathbf{C} = [c_{km}]$ is a K by M matrix, where c_{km} indicates the center of kth cluster and is defined by u_{nk}. Also, in this equation, $\mathbf{W} = [w_{km}]$ is a K by M matrix, where w_{km} refers to the weight of mth feature in kth cluster, $\boldsymbol{z} = [z_k]$ is a vector with the length of K, where z_k refers to kth cluster weight, and the term $d^2(x_{nm} - c_{km})$ indicates a non-Euclidean distance metric and is defined as follows:

$$1 - exp(-\gamma_m (x_{nm} - c_{km})^2), \quad (2)$$

where γ_m indicates inverse of the variance of mth feature of x dataset:

$$\gamma_m = \frac{I}{var_m}, var_m = \sum_{n=1}^{N} \frac{(x_{nm} - \bar{x}_m)^2}{N}, \bar{x}_m = \sum_{n=1}^{N} \frac{x_{nm}}{N}, I = (0, 1]. \quad (3)$$

And α is the fuzzification coefficient ($\alpha > 1$) (fuzzifier). It is commonly used by fuzzy clustering methods to determine the level of fuzziness in the formed clusters. A large α results in smaller membership values, and thus, fuzzier clusters are formed. In the limit $\alpha = 1$, the memberships converge to 0 or 1 implying a hard partitioning. In the absence of domain knowledge, α is commonly set to 2 [22,45].

Similar to the method presented in [54], the appropriate value of parameter $p(0 \leq p < 1)$ is found during the operation of the proposed clustering algorithm. This parameter controls the sensitivity of the cluster weight updating to the relative difference of the SIWDs (sum of the intra-cluster weighted-feature distance). This parameter prevents the formation of clusters with big SIWDs and thus, makes them balanced in terms of the total SIWDs. This metric indicates the quality of clustering [54].

Following to the method presented in [27], the principle that should be considered for weighting features in each cluster is as follows: assign a larger weight to a feature that has a smaller variance in the related cluster, and a smaller weight to a feature that has a larger variance. In this paper, we refer to this principle as the "*feature weighting principle*". Inspired by the research carried out in [33], in order to be able to implement this principle, we employ parameter q in Eq. (1) (in range $q > 1$ and $q < 0$). Further details on how to calculate proper values for p and q, and study their influences on the performance of the method are discussed in Section 3.2.

As stated in introduction, k-partitioning algorithms are sensitive to initialization. This sensitivity exists in both algorithms with and without feature weighting. In fact, feature weighting is necessary but not enough. After a bad initialization, it is possible that some clusters with large SIWDs are merged together, and those with low SIWD are broken into smaller clusters. Therefore, even if there are some clusters with balanced SIWDs in the dataset, after running the algorithm, it is possible that some clusters with unbalanced SIWDs are formed. This is the case that happens in k-means based algorithms.

Cluster weighting solves this problem to a large extent. The principle that is used for weighting clusters is as follows: assign a larger weight to a cluster that has a larger SIWD, and a smaller weight to a cluster that has a smaller SIWD. We call this principle the "*cluster weighting principle*". By assigning a higher weight to a cluster with a big SIWD, we can prevent formation of clusters with an unbalanced SIWD.

Considering the criterion drawn from SIWD, cluster weighting has the best efficiency on the datasets where there are groups (clusters) having a nearly balanced SIWD. This is due to emergence of the clusters with a balanced SIWD during the algorithm restart. However, in practice, there are not always such balanced groups (clusters) in the datasets. To solve this problem, feature weighting is implemented along with the cluster weighting. In this way, one can achieve the best efficiency of a clustering algorithm irrespective of the structure of the data and the initialization.

In order to complete the formulation of the objective function presented in Eq. (1), the following limitations should be considered:

$$u_{nk} \in [0,1], \sum_{k=1}^{K} u_{nk} = 1, \quad \text{where } 1 \le n \le N \text{ and } 1 \le k \le K;$$
$$w_{km} \in [0,1], \sum_{m=1}^{M} w_{km} = 1 \quad \text{where } 1 \le k \le K \text{ and } 1 \le m \le M;$$
$$z_k \in [0,1], \sum_{k=1}^{K} z_k = 1 \quad \text{where } 1 \le k \le K \text{ and } 0 \le p < 1. \tag{4}$$

Considering the limitations presented in Eq. (4), the Lagrange multiplier of the objective function in Eq. (1) can be written as follows:

$$\tilde{F} = \sum_{n=1}^{N}\sum_{k=1}^{K}\sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} z_k^{p} d^2(x_{nm} - c_{km}) - \delta\left(\sum_{k=1}^{K} u_{nk} - 1\right) - \psi\left(\sum_{m=1}^{M} w_{km} - 1\right) - \omega\left(\sum_{k=1}^{K} z_k - 1\right), \tag{5}$$

where, δ, ψ, ω are the parameters of Lagrange multiplier.

Once the Lagrange equation in Eq. (5) is solved given the conditions in KKT[2], we can draw the updating polices for u_{nk}, w_{km}, c_{km} and z_k based on Eqs. (6)–(9), respectively (see Appendix for details):

$$u_{nk} = \frac{1}{\sum_{l=1}^{K}\left[\frac{z_k^{p}\sum_{m=1}^{M} w_{km}^{q} d^2(x_{nm}-c_{km})}{z_l^{p}\sum_{m=1}^{M} w_{lm}^{q} d^2(x_{nm}-c_{lm})}\right]^{\frac{1}{(\alpha-1)}}} \tag{6}$$

The membership degree of a given sample n to cluster k is calculated by Eq. (6), where u_{nk} indicates the degree of nth sample's membership to the center of kth cluster, $d^2(x_{nm} - c_{km})$ indicates the distance metric (Eq. (2)), α indicates the fuzzification coefficient ($\alpha > 1$), z_k refers to kth cluster weight, M is the number of features, p is determined automatically during iteration and q is set manually (in range $q > 1$ and $q < 0$).

The centers are chosen randomly from the samples before the algorithm start. Then, the centers are updated by Eq. (7) during the repetition:

$$c_{km} = \frac{\sum_{n=1}^{N} {u_{nk}}^{\alpha} \exp\left(-\gamma_m (x_{nm} - c_{km})^2\right) x_{nm}}{\sum_{n=1}^{N} {u_{nk}}^{\alpha} \exp\left(-\gamma_m (x_{nm} - c_{km})^2\right)}, \tag{7}$$

where, c_{km} indicates the center of kth cluster, N refers to the number of data samples. u_{nk} is calculated by Eq. (6), and x_{nm} indicates the mth feature in nth sample.

The weights of the features in each cluster are updated by Eq. (8):

$$w_{km} = \begin{cases} \frac{1}{h_m} & \text{if } Dw_{km} = 0 \text{ and } h_m = |\{s: Dw_{ks} = 0\}| \\ 0 & \text{if } Dw_{km} \neq 0 \text{ and } \exists s \text{ where } Dw_{ks} = 0 \\ \frac{1}{\sum_{s=1}^{M}\left[\frac{Dw_{km}}{Dw_{ks}}\right]^{\frac{1}{q-1}}} & \text{if } Dw_{ks} \neq 0 \text{ where } \forall 1 \le s \le M, \end{cases} \tag{8}$$

where, $Dw_{km} = \sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{nm} - c_{km})$.

[2] Karush–Kuhn–Tucker.

The weights of the clusters are updated by Eq. (9):

$$z_k = \begin{cases} \frac{1}{g_k} & \text{if } Dz_k = 0 \text{ and } g_k = |\{l: Dz_l = 0\}| \\ 0 & \text{if } Dz_k \neq 0 \text{ and } \exists l \text{ where } Dz_l = 0 \\ \frac{1}{\sum_{l=1}^{K}\left[\frac{Dz_k}{Dz_l}\right]^{\frac{1}{p-1}}} & \text{if } Dz_l \neq 0 \text{ where } \forall 1 \le l \le K, \end{cases} \tag{9}$$

where, $Dz_k = \sum_{n=1}^{N}\sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} d^2(x_{nm} - c_{km})$. The details of the calculations above are provided in Appendix.

3.2. Discussions on parameters p and q

As mentioned earlier, similar to the method presented in [54], the value of p is chosen in range $0 \le p < 1$. By setting the value of p equal to 1, the weights of the clusters are defined as follows:

$$z_k = \begin{cases} \approx 1. & k = \text{argmax}_{1 \le k \le K} Dz_k \\ \approx 0. & \text{otherwise.} \end{cases} \tag{10}$$

In each restart of the algorithm, only the cluster that has a higher Dz_k gains a weight close to 1, and as a result, all the samples of that cluster are randomly transferred to one of the clusters having a weight close to zero. This clearly leads to the formation of an empty cluster and the algorithm will not continue properly. If $p > 1$, the objective function mainly focuses on the weighting of the clusters, and as a consequence, weights become more influential than the other parameters. Hence, it does not converge to a minimum point. If $p = 0$, the weights of the clusters become useless in the algorithm. We therefore conclude that the most appropriate value for p is something between zero and one.

Furthermore, p has an inverse relationship with the similarity measure between the weights of the clusters (see Eq. (9)). That is, the value of the weights becomes less (more) similar to one another as the value of p increases (decreases). In such a case, the relative difference between the value of Dz_k increases between clusters. To clarify this, we define $\frac{z_k}{z_{k'}}$ as the similarity ratio between the weights of the clusters:

$$\frac{z_k}{z_{k'}} = \left(\frac{Dz_k}{Dz_{k'}}\right)^{\frac{1}{1-p}}. \tag{11}$$

If the value of this ratio is close to 1, it would indicate a greater similarity between the weights of the clusters. This ratio can be redefined as follows considering p:

$$\frac{{z_k}^{p}}{{z_{k'}}^{p}} = \left(\frac{Dz_k}{Dz_{k'}}\right)^{\frac{p}{1-p}}. \tag{12}$$

If $0 \le p < 1$, then the values of $1/(1-p)$ and $p/(1-p)$ increase as the value of p increases. As a result, the ratios of the two above-mentioned equations are less inclined towards 1, thus, the weights of the clusters and ${z_k}^{p}$ will be less similar to each other. By decreasing the value of p, the values of $1/(1-p)$ and $p/(1-p)$ decrease. As a result, the ratios of Eqs. (11) and (12) tend more towards 1, and thus, the weights of the clusters and ${z_k}^{p}$ become more similar to each other. Accordingly, for a larger values of p, more weights are assigned to the clusters with a larger SIWD. In fact, the objective function in Eq. (1) punishes clusters that have more SIWD and prevents the creation of larger and unbalanced SIWDs. This technique results in very good clusters even with a very bad initialization.

Considering that there might be structures in the dataset with various SIWDs, some algorithms are unable to detect the proper clusters among these structures. Algorithms that do not use cluster weighting, such as FCM and k-means, may form clusters that are unbalanced in terms of SIWD. This means that during the running of the algorithm, clusters with a lower SIWD may be

divided into smaller clusters, or some of the clusters with larger SIWDs may be combined. If the clusters are unbalanced in terms of SIWD, the clustering algorithm is of a low quality.

The parameter q is adjusted based on the above defined *feature weighting principle* [27]. According to this principle, it can be said that if $Dw_{km} > Dw_{k\acute{m}}$, then $w_{km} < w_{k\acute{m}}$. Therefore, the optimal value of q can be analyzed as follows:

- If $q = 0$, the presence or absence of w_{km} has no effect in the objective function in Eq. (1). If $q = 1$, w_{km} will be 1 for small Dw_{km}, and 0 for the rest of the weights. In this case, the objective function in Eq. (1) is minimized, and hence, the resulting cluster is the outcome of only one involving feature. This indeed is not a proper setting.
- When $0 < q < 1$, a larger w_{km} will result in a larger Dw_{km} and a larger w_{km}^q accordingly. This clearly disagrees with the feature weighting principle. If one set $q > 1$, then the larger Dw_{km} will result in a smaller w_{km}, and w_{km}^q will become smaller as well. Hence, the influence of mth feature on the kth cluster decreases. Considering $q < 0$ while Dw_{km} is large, will result in a larger w_{km} as well. It should be noted that since q is negative, w_{km}^q becomes smaller.

With regard to these explanations, setting $q < 0$ or $q > 1$ is of recommended. Based on the goals that we are following for clustering, these intervals are used for parameters p and q in the proposed algorithm.

3.3. Algorithm description

Details of the proposed algorithm are presented in Fig. 1. According to the explanations given in Section 3.2, the performance of the proposed algorithm with the simultaneous weighting of the clusters and the features is based on the optimal adjustment of parameters p and q, that is discussed in next subsection.

3.3.1. Parameter tuning

Similar to the method presented in [54], a repeat-based algorithm is used to find the optimal value for p. Three values p_{init}, p_{step} and p_{max} are defined and the algorithm starts with a small value for $p(p_{init})$. In each iteration, the value of p is increased by p_{step} until the maximum value p_{max} is reached. If an empty cluster or a cluster with only one sample appears, the value of p is reduced by p_{step}, regardless of whether p is equal to p_{max} or not. In this step, we select the values of u_{nk}, w_{km} and z_{km} based on the previous p. The algorithm continues until the difference between the two successive values of the objective function is smaller than the threshold ε, or the number of iterations reaches a maximum. Considering the explanation presented in the previous section, p_{max}should not be chosen very large or close to 1.

To improve the stability of the algorithm, similar to the method proposed in [54], we add a memory effect parameter to the weights. By using this parameter, smoother transitions of the values are performed between consecutive iterations. In other words, the influence of the weights from the previous iteration to the current update is controlled. Accordingly, the final weights at each iteration are obtained by using Eqs. (13) and (14) as follows:

$$\boldsymbol{W}^{(t)} = \beta \boldsymbol{W}^{(t-1)} + (1-\beta)\,\boldsymbol{W}^{(t)} \tag{13}$$

$$\boldsymbol{z}^{(t)} = \beta \boldsymbol{z}^{(t-1)} + (1-\beta)\,\boldsymbol{z}^{(t)} \tag{14}$$

where, parameter $\beta \in [0,1)$ is to control such transition. The application of this step in the proposed algorithm is shown in Fig. 2.

Weighting the clusters and the features simultaneously may slow down the convergence speed of the algorithm. However, we will show (Section 4.3) that choosing the value of β carefully will eliminate this drawback.

3.3.2. Proposed method vs. bad initialization

In this section we aim to illustrate the performance of the proposed method against the bad initialization of cluster centers. To this end, we design a test that shows how our proposed algorithm alternates between the $\boldsymbol{U}$, $\boldsymbol{C}$, $\boldsymbol{W}$ and $\boldsymbol{z}$ optimization steps to get a local optimum of $\boldsymbol{F}$. As illustrated in Fig. 3(a), we run the algorithm on a synthetic dataset, including some samples in four clusters in a two-dimensional feature space. Then, we track the path of the initial centers to the final centers of the clusters obtained by the algorithm.

The obtained centers for four clusters along the first seven restarts of algorithm are depicted in Fig. 3(a) using different colors. The arrows show the path between obtained consecutive cluster centers. As it is evident in this figure, although inappropriate initial centers are deliberately selected, the algorithm can easily achieve a good local optima. As shown in Fig. 3(b), the value obtained for the objective function is reduced in the second iteration markedly. The obtained centers in the second iteration are the reason of such behavior. The value of the objective function has been significantly reduced from second to seventh iterations, and there are, however, slight decreases from iterations 7 to 52. This shows that the proposed algorithm achieves nearly optimal solution in the seventh iteration. By performing the feature weighting along with the cluster weighting in a concurrent manner, the algorithm achieves an optimal solution very fast irrespective of the initialization.

3.4. Convergence proof

In this section, convergence proof of the proposed algorithm is presented. The goal is to prove that the objective function in Eq. (5) reaches its minimum value or in other words, it converges.

Theorem 1. *Let matrix of centers* **C**, *matrix of weights* **W**, *and vector* **z** *be fixed, matrix* **U** *is a strict local minimum of Eq. (5) if and only if it is obtained via Eq. (6).*

Proof. u_{nk} in Eq. (6) is calculated after taking the derivative of Eq. (5) and setting it equal to zero (see Appendix for details); as a result, u_{nk} is either minimal or maximal. Now, if we show that the second partial derivative of Eq. (5) is positive, it can be proved that u_{nk} defined by Eq. (6) is a local minimum of Eq. (5); the derivative of Eq. (5) with respect to u_{nk} is as follows:

$$\sum_{m=1}^{M} \alpha\,(\alpha-1)\,{u_{nk}}^{\alpha-2} w_{km}^q z_k^p d^2(x_{nm} - c_{km}). \tag{15}$$

Since $d^2(x_{nm} - c_{km}) \geq 0$, $w_{km} \geq 0$, $z_k \geq 0$ and $\alpha > 1$, it can be shown that Eq. (15) is positive. As a result, Eq. (6) is a local minimum of Eq. (5).

Theorem 2. *Let membership matrix* **U**, *matrix of weights* **W** *and vector* **z** *be fixed, matrix of centers* **C** *is a strict local minimum of Eq. (5) if and only if* c_{km} *is obtained via Eq. (7).*

Proof. Proving Theorem 2 is just like proving Theorem 1. That is, we only need to show that the derivative of Eq. (5) with respect to c_{km} is positive. Considering Eq. (16), the second partial derivative of Eq. (5) with respect to c_{km} is a positive phrase. As a result, Eq. (7) is also a local minimal of Eq. (5).

$$\sum_{n=1}^{N} 2\left[\gamma_m {u_{nk}}^{\alpha} w_{km}^q z_k^p exp\left(-\gamma_m (x_{nm} - c_{km})^2\right)\right]$$
$$-\left[{\gamma_m}^2 {u_{nk}}^{\alpha} w_{km}^q z_k^p exp\left(-\gamma_m (x_{nm} - c_{km})^2\right)(2x_{nm} - 2c_{km})^2\right] \tag{16}$$

Algorithm 1. Proposed clustering algorithm.

Input: Dataset $\boldsymbol{\chi} = \{x_n\}_{n=1}^{N}$, Initial centers $C^{(0)}$, Number of clusters K, Number of attributes M, Secondary parameters t_{max}, p_{max}, p_{init}, p_{step}, ε, Exponent of attribute weight q, Fuzzy degree α;
Output: Membership matrix **U**, Cluster centers matrix **C;**

1: set $t = 0$
2: set $p_{init} = 0$
3: set $z_k^{(0)} = \frac{1}{K}$, $\forall k = 1 \dots K$
4: set $w_{km}^{(0)} = \frac{1}{M}$, $\forall k = 1 \dots K$, $\forall m = 1 \dots M$
5: set empty= **FALSE** //No empty or singleton clusters yet detected
6: $p = p_{init}$
7: **repeat**
8: $t = t + 1$
9: Update the cluster assignments matrix **U** by Eq. (6)
10: **if** empty or singleton clusters have occurred at time t **then** //reduce p.
11: empty=**TRUE**
12: $p = p - p_{step}$
13: **if** $p < p_{init}$ **then**
14: **return** NULL
15: **end if**
//Revert to the assignments and weights corresponding to the reduce p.
16: $u_{nk}^{(t)} = \left[\mathbf{U\_history}^{(p)}\right]_{nk}, \forall k = 1 \dots K$, $\forall n = 1 \dots N$
17: $z_k^{(t-1)} = \left[\boldsymbol{Z}\mathbf{\_history}^{(p)}\right]_{k}, \forall k = 1 \dots K$
18: $w_{km}^{(t-1)} = \left[\boldsymbol{w}\mathbf{\_history}^{(p)}\right]_{km}, \forall m = 1 \dots M$, $\forall k = 1 \dots K$
19: **end if**
20: Update the cluster center matrix **C** by Eq. (7)
21: **if** $p < p_{max}$ **and** empty=**FALSE** then //increase p.
22: $\mathbf{U\_history}^{(p)} = [u_{nk}^{(t)}]$ //store the current assignment in matrix $\mathbf{U\_history}^{(p)}$.
23: $\boldsymbol{W}\mathbf{\_history}^{(p)} = [w_{km}^{(t-1)}]$ //store the previous feature weights in matrix $\boldsymbol{W}\mathbf{\_history}^{(p)}$.
24: $\boldsymbol{Z}\mathbf{\_history}^{(p)} = [z_k^{(t-1)}]$ //store the previous cluster weights in matrix $\boldsymbol{Z}\mathbf{\_history}^{(p)}$.
25: $p = p + p_{step}$
26: **end if**
27: Update the feature weight matrix **W** by Eq. (8)
28: Update the cluster weight matrix **z** by Eq. (9)
29: **until** $\left|F^{(t)} - F^{(t-1)}\right| < \varepsilon$ ***or*** $t \geq t_{max}$
30: **return** **U**, **C**

Fig. 1. Proposed clustering algorithm.

Algorithm 2. Proposed clustering algorithm (improved).

Input: Dataset $\boldsymbol{\chi} = \{x_n\}_{n=1}^{N}$, Initial centers $C^{(0)}$, Number of clusters K, Number of attributes M, Secondary parameters t_{max}, p_{max}, p_{init}, p_{step}, ε, Exponent of attribute weight q, Memory effect β, Fuzzy degree α;
Output: Membership matrix **U**, Cluster centers matrix **C;**

1:
⋮ {Lines 1 to 28 are drawn from Algorithm 1 (Figure 1)}
28:
29: $\mathbf{W}^{(t)} = \beta\mathbf{W}^{(t-1)} + (1-\beta)\,\mathbf{W}^{(t)}$
30: $\mathbf{z}^{(t)} = \beta\mathbf{z}^{(t-1)} + (1-\beta)\,\mathbf{z}^{(t)}$
31: **until** $\left|F^{(t)} - F^{(t-1)}\right| < \varepsilon$ ***or*** $t \geq t_{max}$
32: **return** **U**, **C**

Fig. 2. Proposed clustering algorithm (improved).

Theorem 3. *Let membership matrix* **U**, *vector of weights* **z** *and matrix of centers* **C** *be fixed, matrix of weights* **W** *is a local minimal of Eq. (5) if and only if* w_{km} *is calculated via Eq. (8).*

Proof. Here again, we only need to show that the derivative of Eq. (5) with respect to w_{km} is positive. The second partial derivative of Eq. (5) with respect to w_{km} is equal to:

$$\sum_{n=1}^{N} {u_{nk}}^{\alpha} q\,(q-1)\, w_{km}^{q-2} z_k^{p} d^2(x_{nm} - c_{km}). \tag{17}$$

Since $d^2\,(x_{nm} - c_{km}) \geq 0$, $w_{km} \geq 0$, $z_k \geq 0$, $q > 1$ or $q < 0$, it can be shown that Eq. (17) is positive. As a result, Eq. (8) is a local minimal of Eq. (5).

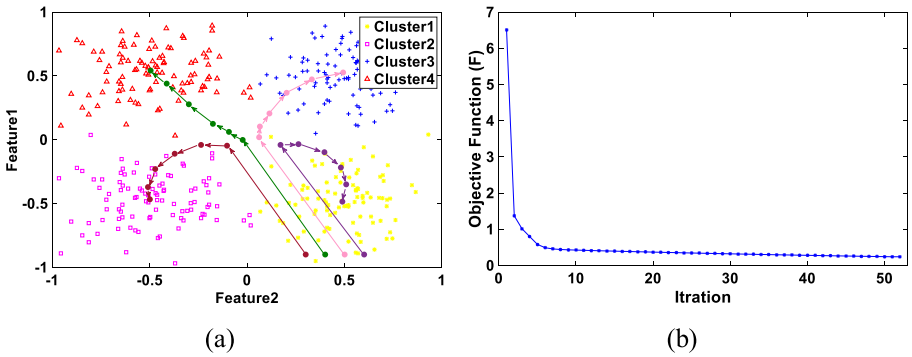

Fig. 3. Performance of the proposed algorithm on a synthetic dataset, including some samples in four clusters in a two-dimensional feature space. (a) The path of the initial centers to the obtained final centers of the clusters. (b) The value of the objective function along the algorithm iterations.

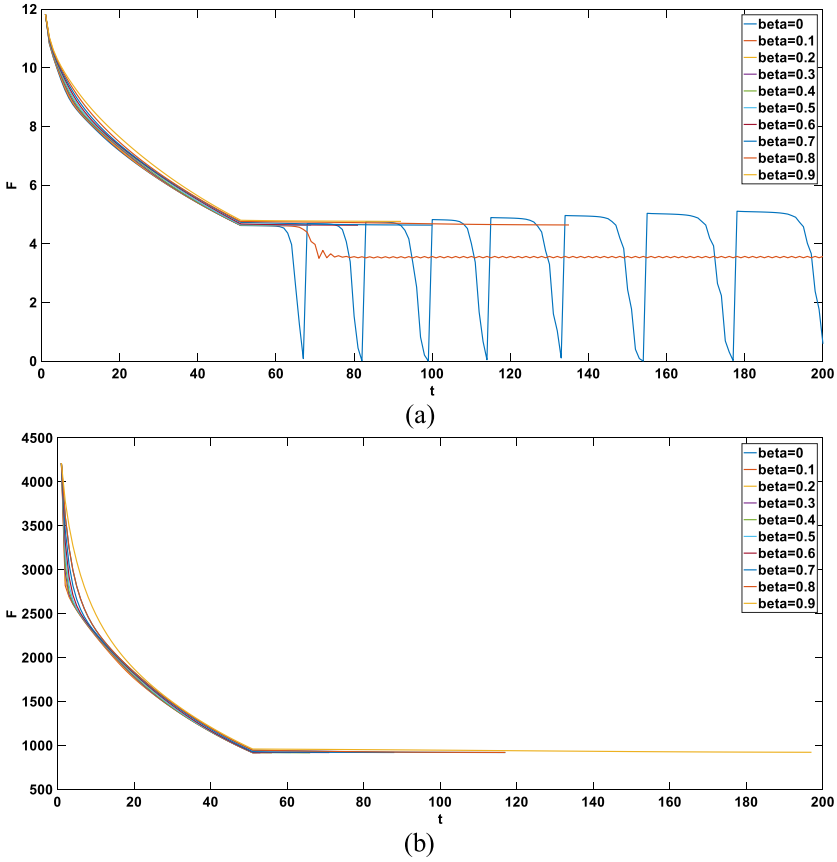

Fig. 4. The convergence rate (algorithm's iterations) of the proposed algorithm for different βs on (a) synthetic dataset and (b) *Ecoli* dataset.

Theorem 4. *Let membership matrix* **U**, *matrix of weights* **W** *and matrix of centers* **C** *be fixed, matrix of weight* ***z*** *is a local minimal of Eq.* (5) *if and only if* z_k *is calculated via Eq.* (9).

Proof. Similar to Theorems 1–3, we need to show that the derivative of Eq. (5) with respect to z_k is positive. Considering Eq. (18), the second partial derivative of Eq. (5) with respect to z_k is a positive phrase. As a result, Eq. (9) is also a local minimal of Eq. (5).

$$\sum_{n=1}^{N}\sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} p\,(p-1)\, z_k^{p-2} d^2(x_{nm}-c_{km}). \tag{18}$$

Since our aim is to make Eq. (5) minimal, and considering Theorems 1–4, and also taking the updating functions of membership matrix **U** Eq. (6), matrix of centers **C** Eq. (7), matrix of weights **W** Eq. (8), and matrix of weights **z** Eq. (9) into account, we can ensure that the objective function in Eq. (5) will finally reach its minimal value and converge.

3.5. Remarks

The computational complexity of the proposed method depends on four updating stages: updating membership matrix **U**, centers matrix **C**, weights matrix **W**, and clusters **z**. The computational complexity of each stage is equal to *NKM*, where *N* refers to the number of data samples, *K* is the number of clusters, and *M* is the number of features. Since each of the stages is run separately, the overall computational complexity is equal to $t\,(NMK + NMK + NMK + NMK) = 4tNMK$ where t is number of iterations.

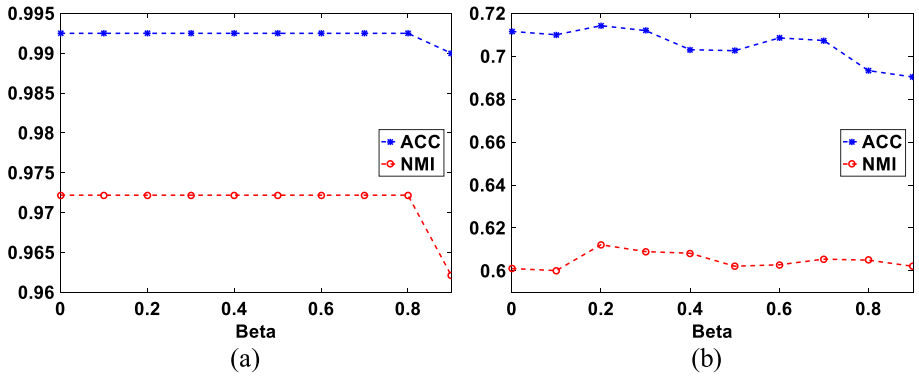

Fig. 5. The efficiency of the proposed method for different βs on (a) synthetic dataset, and (b) Ecoli dataset.

4. Experiments

In this section, the performance of the proposed algorithm is evaluated, and the results are compared with state-of-the-art algorithms, namely the standard k-means (k-means) [3], the standard FCM [4], simultaneous clustering and attribute discrimination (SCAD2) [9], k-means++ [47], the hard c-means clustering with local weighting (RLFWHCM) [33], fuzzy c-means clustering with the entropy of feature weight (EWFCM) [22], and the improved version of k-means with Min–Max method (MMKMC) [54].

Threshold value ε and the maximum number of restarts are the common parameters in all the algorithms. In the experiments, we set them to 10^{-5} and $t = 200$, respectively. Parameter α is common between other fuzzy algorithms and our method and is set to 2 in all algorithms. Additionally, we adjust $p_{step} = 0.01$, $p_{init} = 0$ and $p_{max} = 0.5$, and for each test dataset (see Section 4.1), we consider the number of clusters equal to the number of labeled classes.

4.1. Dataset

In the experiments carried out to evaluate the performance of the proposed method, two sets of data, say synthetic datasets and real world datasets, are used.

(1) Synthetic dataset

In order to easily examine the efficiency of the solutions proposed in our method to solve each of the existing challenges, we use a synthetic dataset introduced in [57]. In this dataset, there are 400 four-dimensional samples in 4 classes, 100 samples per class. The distribution of the samples is Gaussian. However, for different clusters, the means (μ) and the standard deviations (σ) are different.

(2) Real world dataset

To evaluate the performance of the proposed algorithm and compare its results with other state-of-the-art methods, we use 16 standard and real world datasets from UCI data repository [58]. The details of this dataset are summarized in Table 1.

Table 1
Characteristics of the real world dataset.

Dataset	Number of data points	Number of dimensions	Number of classes
Ecoli	336	8	8
Iris	150	4	3
Wine	178	13	3
Glass	214	10	6
Ionosphere	351	34	2
Heberman	306	3	2
Spectf heart	267	44	2
Vowel	871	3	6
Pima Indians Diabetes	768	8	2
Breast Cancer Wisconsin (Original)	699	10	2
Dermatology	366	33	6
Zoo	101	17	7
Letter Recognition (A,B,C)	2291	16	3
Statlog (heart)	270	13	2
Balance	625	4	3
Breast Cancer Wisconsin (Diagnostic)	569	32	2

4.2. Performance metrics

In the experiments, two metrics namely clustering *accuracy* and *normalized mutual information* (NMI) are used.

Accuracy: This measure is used when the cluster's structure is specified in the data [59]. This measure is calculated as Eq. (19):

$$ACC = \frac{\sum_{k=1}^{K} d_k}{N}, \tag{19}$$

where d_k indicates the number of samples that have been correctly assigned to K clusters, and N refers to the total number of all samples in the dataset.

NMI: The normalized mutual information metric is a symmetric measure that is able to quantify the statistical information shared between two clusters' distributions [60]. This measure is defined as Eq. (20):

$$NMI\ (R, Q) = \frac{\sum_{i=1}^{I}\sum_{j=1}^{J} P\ (i, j)\ log\ \frac{P(i,j)}{P(i)P(j)}}{\sqrt{H\ (R)\ H\ (Q)}}, \tag{20}$$

where R and Q are two partitions of the input dataset including I and J clusters, respectively. $P\ (i)$ is the probability that a randomly selected sample from the input data is assigned to cluster R_i, $P\ (i, j)$ is the probability that a sample belongs to both clusters R_i and Q_i. $H\ (R)$ is the entropy associated with all the probabilities $P\ (i)\ (1 \leq i \leq I)$ in partition R [60].

In this paper, following [22], the average values of *ACC*s and NMIs are obtained from averaging the results of 100 runs of the algorithms.

4.3. Adjusting parameter β

Parameter $\beta (0 \leq \beta < 1)$ as stated in Section 3.3, controls the influence of the weights that are computed in the previous iteration. Applying the cluster weighting and the feature

Table 2
Comparison of the performance of the proposed method and RLFWHCM on real world datasets.

Dataset	Method	q									
		−10	−8	−6	−4	−2	2	4	6	8	10
Ecoli	Ours	**0.7142**	0.6250	0.6250	0.5752	0.5854	0.6666	0.4464	0.4434	0.4434	0.4434
		0.6121	0.6017	0.6032	0.5992	0.5911	0.4658	0.3284	0.2999	0.2804	0.2755
	RLFWHCM	0.6741	0.6473	0.5735	0.5276	0.5101	0.5014	0.4841	0.4471	0.4523	0.5254
		0.5809	0.5684	0.5639	0.5518	0.5581	0.2379	0.2154	0.2261	0.2382	0.2117
Iris	Ours	0.9067	0.9000	0.9000	0.8934	0.8800	**0.9666**	0.9600	0.9533	0.9533	0.9466
		0.7892	0.7639	0.7578	0.7433	0.7226	**0.8801**	0.8641	0.8572	0.8572	0.8449
	RLFWHCM	0.8987	0.8987	0.8980	0.8980	0.8547	0.9200	0.9174	0.9140	0.9127	0.9127
		0.8005	0.8005	0.8003	0.7949	0.7706	0.8430	0.8325	0.8328	0.8180	0.8180
Wine	Ours	0.9382	0.9382	0.9382	0.9382	**0.9388**	0.9157	0.9326	0.9382	0.9382	0.9382
		0.7955	0.7955	0.7955	0.7955	**0.7967**	0.7391	0.7844	0.7955	0.7955	0.7955
	RLFWHCM	0.9267	0.9181	0.9121	0.9310	0.9188	0.7966	0.8895	0.8931	0.8963	0.9135
		0.7696	0.7696	0.7696	0.7696	0.7939	0.4363	0.7137	0.7349	0.7509	0.7509
Glass	Ours	0.4701	**0.4766**	0.4677	0.4691	0.4537	0.3666	0.3939	0.4074	0.3866	0.3774
		0.3019	0.3127	**0.3169**	0.3057	0.3033	0.1391	0.1834	0.2145	0.1796	0.1534
	RLFWHCM	0.4556	0.4593	0.4565	0.4495	0.4415	0.3560	0.3551	0.3551	0.3551	0.4065
		0.2939	0.2953	0.2968	0.2861	0.2995	0.1191	0.1168	0.1168	0.1168	0.1808
Dermatology	Ours	0.7237	0.7049	**0.7300**	0.7065	0.7112	0.7003	0.6721	0.6721	0.5300	0.5271
		0.7710	0.7694	**0.7781**	0.7767	0.7684	0.6612	0.5610	0.5500	0.5677	0.5637
	RLFWHCM	0.7051	0.7103	0.7106	0.7013	0.6893	0.5785	0.5288	0.5355	0.5256	0.5245
		0.7578	0.7609	0.7705	0.7722	0.7634	0.3501	0.4772	0.4283	0.5223	0.5032
Zoo	Ours	0.8168	0.8339	0.8356	0.8455	0.8287	0.7079	**0.8614**	0.8515	0.8317	0.7180
		0.8164	0.8152	0.8334	0.8375	**0.8600**	0.7195	0.7324	0.7385	0.7531	0.7267
	RLFWHCM	0.7169	0.7169	0.7169	0.7169	0.7169	0.7169	0.7169	0.7169	0.7169	0.7169
		0.8107	0.8128	0.8148	0.8148	0.8408	0.6291	0.6983	0.6710	0.6379	0.6710
Ionosphere	Ours	**0.7094**	**0.7094**	**0.7094**	**0.7094**	**0.7094**	**0.7094**	**0.7094**	**0.7094**	**0.7094**	**0.7094**
		0.1299	**0.1299**	**0.1299**	**0.1299**	**0.1299**	**0.1299**	**0.1299**	**0.1299**	**0.1299**	**0.1299**
	RLFWHCM	0.6675	0.6675	0.6675	0.6675	0.6675	0.6558	0.6661	0.6661	0.6661	0.6670
		0.1208	0.1210	0.1213	0.1211	0.1122	0.1122	0.1122	0.1122	0.1122	0.1122
Breast Cancer Wisconsin (Original)	Ours	0.9636	0.9636	0.6349	**0.9663**	**0.9663**	0.9601	0.9618	0.9621	0.9565	0.9567
		0.7607	0.7607	0.7678	**0.7751**	**0.7751**	0.6531	0.7637	0.7662	0.7309	0.7363
	RLFWHCM	0.9624	0.9630	0.9630	0.9633	0.9640	0.8668	0.9243	0.9552	0.9565	0.9567
		0.7239	0.7238	0.7266	0.7253	0.7344	0.5357	0.7034	0.7097	0.7207	0.7185
Letter Recognition	Ours	0.8496	0.8495	0.8491	0.8427	0.8347	**0.8912**	0.8608	0.8565	0.8548	0.8536
		0.6343	0.6340	0.6336	0.6164	0.6124	**0.6803**	0.6457	0.6409	0.6390	0.6374
	RLFWHCM	0.7803	0.7803	0.7796	0.7767	0.7421	0.7331	0.7574	0.7963	0.8077	0.8070
		0.6192	0.6182	0.6184	0.5980	0.5962	0.6744	0.6164	0.6241	0.6250	0.6240
Breast Cancer Wisconsin (Diagnostic)	Ours	**0.9420**	**0.9420**	**0.9420**	**0.9420**	**0.9420**	0.9308	0.9332	0.9332	0.9402	0.9384
		0.6736	0.6736	0.6736	0.6736	0.6703	0.6294	0.6647	0.6503	**0.6885**	0.6825
	RLFWHCM	0.7696	0.7696	0.7696	0.7696	0.7939	0.4363	0.7137	0.7349	0.7509	0.7509
		0.6801	0.6807	0.6825	0.6862	0.6794	0.5532	0.6367	0.6634	0.6419	0.6539
Pima Indians Diabetes	Ours	0.6627	0.6641	0.6742	0.6817	**0.6855**	0.5768	0.5932	0.6213	0.6335	0.6354
		0.0779	0.0805	0.0911	0.1012	**0.1039**	0.0303	0.0351	0.0469	0.0566	0.0586
	RLFWHCM	0.6036	0.6054	0.6088	0.6197	0.6272	0.5680	0.5859	0.5867	0.5878	0.5877
		0.0329	0.0362	0.0366	0.0356	0.0379	0.0252	0.0333	0.0310	0.0315	0.0320
Heberman	Ours	0.6072	0.6085	0.6114	0.5780	0.5397	**0.7549**	0.7418	0.7418	0.7386	0.7386
		0.0362	0.0363	0.0379	0.0197	0.0121	**0.0948**	0.0854	0.0854	0.0822	0.0822
	RLFWHCM	0.5990	0.6003	0.5911	0.5715	0.5382	0.6503	0.6130	0.6062	0.6042	0.6045
		0.0261	0.0271	0.0231	0.0187	0.0111	0.0440	0.0365	0.0343	0.0339	0.0351
Statlog (heart)	Ours	**0.8444**	**0.8444**	**0.8444**	0.8370	0.8407	0.8370	0.8370	0.8370	0.8407	0.8370
		0.3732	**0.3732**	**0.3732**	0.3564	0.3638	0.3564	0.3564	0.3564	0.3647	0.3564
	RLFWHCM	0.8077	0.8129	0.8148	0.8188	0.8329	0.7185	0.7811	0.7892	0.7966	0.7981
		0.3021	0.3112	0.3164	0.3212	0.3514	0.1667	0.2534	0.2633	0.2780	0.2802
Balance	Ours	0.6304	0.6300	0.6308	0.6300	0.6248	0.5421	0.5594	**0.6512**	0.6240	0.6336
		0.1184	0.1183	0.1172	0.1171	0.1185	0.1166	0.1161	0.1213	**0.1229**	0.1222
	RLFWHCM	0.5262	0.5262	0.5288	0.5285	0.5235	0.5400	0.5308	0.5342	0.5272	0.5283
		0.1150	0.1150	0.1150	0.1155	0.1167	0.1161	0.1131	0.1138	0.1202	0.1196
Vowel	Ours	0.6146	0.6265	0.6134	0.6148	0.6163	0.5607	0.6321	0.6385	**0.6451**	0.6299
		0.5650	0.5618	0.5564	0.5550	0.5464	0.4705	0.5314	0.5604	**0.5760**	0.5751
	RLFWHCM	0.6267	0.6191	0.6211	0.6196	0.6276	0.5695	0.6019	0.6127	0.6148	0.6156
		0.5633	0.5608	0.5703	0.5646	0.5603	0.4831	0.5235	0.5398	0.5453	0.5464

(*continued on next page*)

Table 2 (*continued*).

Dataset	Method	q									
		−10	−8	−6	−4	−2	2	4	6	8	10
Spectf heart	Ours	0.6104 0.2028	0.6104 0.2028	0.6067 0.2028	0.6048 0.2052	0.6044 **0.2077**	0.6253 0.1327	0.6078 0.1935	0.6119 0.1981	0.6127 0.1981	0.6134 0.1981
	RLFWHCM	0.6655 0.1441	**0.6659** 0.1260	0.6655 0.1259	0.6655 0.1425	0.6655 0.1272	0.6303 0.1138	0.6617 0.1334	0.6651 0.1423	0.6651 0.1435	0.6647 0.1453

Table 3
The average results of the proposed method and RLFWHCM on all real world datasets.

Method	Performance metric	q									
		−10	−8	−6	−4	−2	2	4	6	8	10
Ours	ACC	**0.7498**	**0.7450**	**0.7254**	**0.7386**	**0.7345**	**0.7313**	**0.7311**	**0.7389**	**0.7270**	**0.7182**
	NMI	**0.4778**	**0.4760**	**0.4785**	**0.4743**	**0.4730**	**0.4312**	**0.4352**	**0.4376**	**0.4382**	**0.4330**
RLFWHCM	ACC	0.7116	0.7100	0.7048	0.7015	0.6946	0.6398	0.6704	0.6755	0.6772	0.6862
	NMI	0.4588	0.4579	0.4595	0.4573	0.4595	0.3399	0.3884	0.3902	0.3960	0.4376

weighting in each cluster simultaneously may slow down the convergence speed of the algorithm. Therefore, we believe that selecting a proper value for β increases the convergence speed, and hopefully, increases the accuracy of the algorithm as well. For this purpose, we evaluate the proposed algorithm for different βs. We perform the experiments on both the synthetic dataset and the real world dataset, *Ecoli*. The clusters in the synthetic dataset are balanced and they are similar to one another in terms of sum of the intra-cluster distance, where in contrast, the clusters are unbalanced in *Ecoli* dataset.

The convergence speed (number of iterations) of the proposed algorithm for different βs on synthetic dataset as well as on *Ecoli* dataset are shown in Fig. 4. According to this figure, when we run the proposed algorithm for larger βs, its convergence time (iterations) also increases due to the frequent number of restarts of the algorithm. For the synthetic dataset, it converges after 92, 135, 100, 81, 69 and 60 restarts for β equals to 0.9, 0.8, 0.7, 0.6, 0.5 and 0.4, respectively. For *Ecoli* dataset, the algorithm using the values of β equal to 0.9, 0.8, 0.7, 0.6, 0.5 and 0.4 converges after 197, 117, 88, 81, 61, and 66 restarts, respectively. The reason for the increase in the number of restarts for large βs can be sought in the greater effects of the previous weights.

When we implement the proposed algorithm without involving the weights from the previous restart in the current restart, say $\beta = 0$, the algorithm stops at 200th restart after a few ups and downs on the synthetic dataset. The reason for this behavior is that the empty clusters are created while the algorithm is running; when the empty clusters are created, the algorithm returns the weights and the assignments of the previous restart, and starts over with a new value for p.

For $\beta = 0.1$, it is considered that the number of ups and downs decreases. Nevertheless, the number of restarts is still 200. For $\beta = 0.2$ and $\beta = 0.3$, the algorithm converges after 52 and 54 restarts, respectively.

In *Ecoli* dataset, the algorithm converges for $\beta = 0$ at 117th restart, and for 0.1, 0.2 and 0.3, it converges at 56th, 52th, and 53th restarts, respectively. Hence, we can conclude that a proper range for β in our algorithm is [0.2, 0.3]. In addition, the efficiency of the proposed algorithm for different βs on two datasets is shown in Fig. 5. As shown in this figure, for almost all values of β, the algorithm has almost the same performance on synthetic dataset. On the *Ecoli* dataset, the algorithm performs better when β is equal to 0, 0.1, 0.2, and 0.3. Also, from Fig. 5, we can conclude that parameter β has less effect in the balanced datasets than the unbalanced ones.

4.4. Experiment 1: effect of feature weighting

In this section, we investigate the performance of the proposed algorithm regarding the local weighting of the features. That is, we aim to examine whether the proposed algorithm and its competitors, EWFCM [22], RLFWHCM [33] and SCAD2 [9] properly assign weights to features or not. To this end, we use the synthetic dataset. To demonstrate the importance of each feature in this dataset, we visualize the data in different 2D subspaces. Fig. 6 presents the result of this visualization. From this figure, it is understood that Cluster1 and Cluster2 are mostly formed based on Feature1 and Feature2, and Cluster3 and Cluster4 are mostly formed based on Feature2 and Feature3. This means that the first and the second features in Cluster1 and Cluster2 are more important than the third and the fourth features. Similarly, the second and the third features in Cluster3 and Cluster4 are more important than the first and the second features.

We run the proposed algorithm and the other three algorithms. After the completion of the clustering process, we evaluate the weights of the features found by each algorithm. Fig. 7 compares the values of the weights obtained for each of the features in different clusters using different algorithms. As shown in this figure, the proposed algorithm compared to the other algorithms, assigns more weight to the first and the second features in Cluster1 and Cluster2, and to the second and the third features in Cluster3 and Cluster4.

The differences in the weights found for the various features are fully consistent with the prediction made based on the data visualization in Fig. 6. This clearly shows that our algorithm does local feature weighting properly. The SCAD2 algorithm correctly performed weighting of the features in Cluster1 and Cluster3. However, in Cluster2 and Cluster4, the weighting of the features was not done correctly. This is due to the fact that the importance of the features in those clusters was not properly recognized. So in SCAD2, the weight of the features does not properly represent the importance of each feature in each cluster.

4.5. Experiment 2: effect of cluster weighting

In this section, we evaluated the effect of cluster weighting against bad initializations. In order to select some different initial points, we select a two-dimensional sub-space of the synthetic dataset so that we can illustrate its data points and select the required initial points manually. We select the sub-space containing the first and the second features. The distribution of data in this space is shown in Fig. 6(a).

We define 8 sets of points P1, P2, P3, P4, P5, P6, P7, and P8 manually as the initial points of the cluster centers located on

Table 4
Comparison of the proposed algorithm and MMKMC on the real world datasets.

Dataset	Method	β		
		0.3	0.2	0
Ecoli	Ours	0.7120 0.6089	**0.7142** **0.6121**	0.7105 0.6010
	MMKMC	0.5440 0.5817	0.5773 0.5842	0.5523 0.5523
Iris	Ours	**0.9666** **0.8801**	**0.9666** **0.8801**	**0.9666** **0.8801**
	MMKMC	0.8866 0.7207	0.8866 0.7207	0.8866 0.7207
Wine	Ours	0.9382 0.7955	0.9382 0.7955	**0.9387** **0.7967**
	MMKMC	0.6685 0.4236	0.6674 0.4224	0.6707 0.4254
Glass	Ours	0.4598 0.3013	0.4766 0.3169	0.4652 0.2989
	MMKMC	0.4924 0.3582	**0.5210** **0.3684**	0.4929 0.3625
Dermatology	Ours	0.7237 0.7767	**0.7300** 0.7781	0.6509 0.6102
	MMKMC	0.6737 **0.7917**	0.6767 0.7893	0.6795 0.7981
Zoo	Ours	0.8455 **0.8599**	0.8393 0.8593	**0.8613** 0.7065
	MMKMC	0.7293 0.7467	0.7293 0.7494	0.7293 0.7538
Ionosphere	Ours	**0.7094** **0.1299**	**0.7094** **0.1299**	0.3589 0.0229
	MMKMC	0.6695 0.0654	0.6638 0.0612	0.6638 0.0612
Breast Cancer Wisconsin (Original)	Ours	0.9414 0.6742	**0.9663** **0.7750**	0.9474 0.7091
	MMKMC	0.9326 0.6461	0.9458 0.6938	0.8989 0.5562
Letter Recognition	Ours	0.8908 0.6797	0.8911 0.6802	**0.8912** **0.6803**
	MMKMC	0.7298 0.4534	0.7302 0.4555	0.7302 0.4555
Breast Cancer Wisconsin (Diagnostic)	Ours	**0.9420** **0.6885**	**0.9420** **0.6885**	**0.9420** **0.6885**
	MMKMC	0.8312 0.4170	0.8312 0.4170	0.8312 0.4170
Pima Indians Diabetes	Ours	0.6809 0.1001	0.6834 0.1024	**0.6855** **0.1039**
	MMKMC	0.6627 0.0307	0.6627 0.0307	0.6627 0.0307
Heberman	Ours	**0.7549** **0.0947**	**0.7549** **0.0947**	**0.7549** **0.0947**
	MMKMC	0.5186 0.0006	0.5218 0.0001	0.5117 0.0018
Statlog (heart)	Ours	**0.8444** **0.3731**	**0.8444** **0.3731**	**0.8444** **0.3731**
	MMKMC	0.5925 0.0199	0.5925 0.0199	0.5925 0.0199
Balance	Ours	0.5488 **0.1229**	**0.6512** 0.0926	0.5604 0.1161
	MMKMC	0.5016 0.1017	0.4915 0.0906	0.4985 0.1016
Vowel	Ours	0.6336 0.5686	**0.6451** 0.5733	0.6384 **0.5760**
	MMKMC	0.5157 0.4792	0.4997 0.4721	0.5166 0.4814

(*continued on next page*)

Table 4 (*continued*).

Dataset	Method	β		
		0.3	0.2	0
Spectf heart	Ours	0.6142 **0.2077**	0.6125 **0.2077**	**0.6254** **0.2077**
	MMKMC	0.6104 0.0963	0.6104 0.0963	0.6104 0.0963

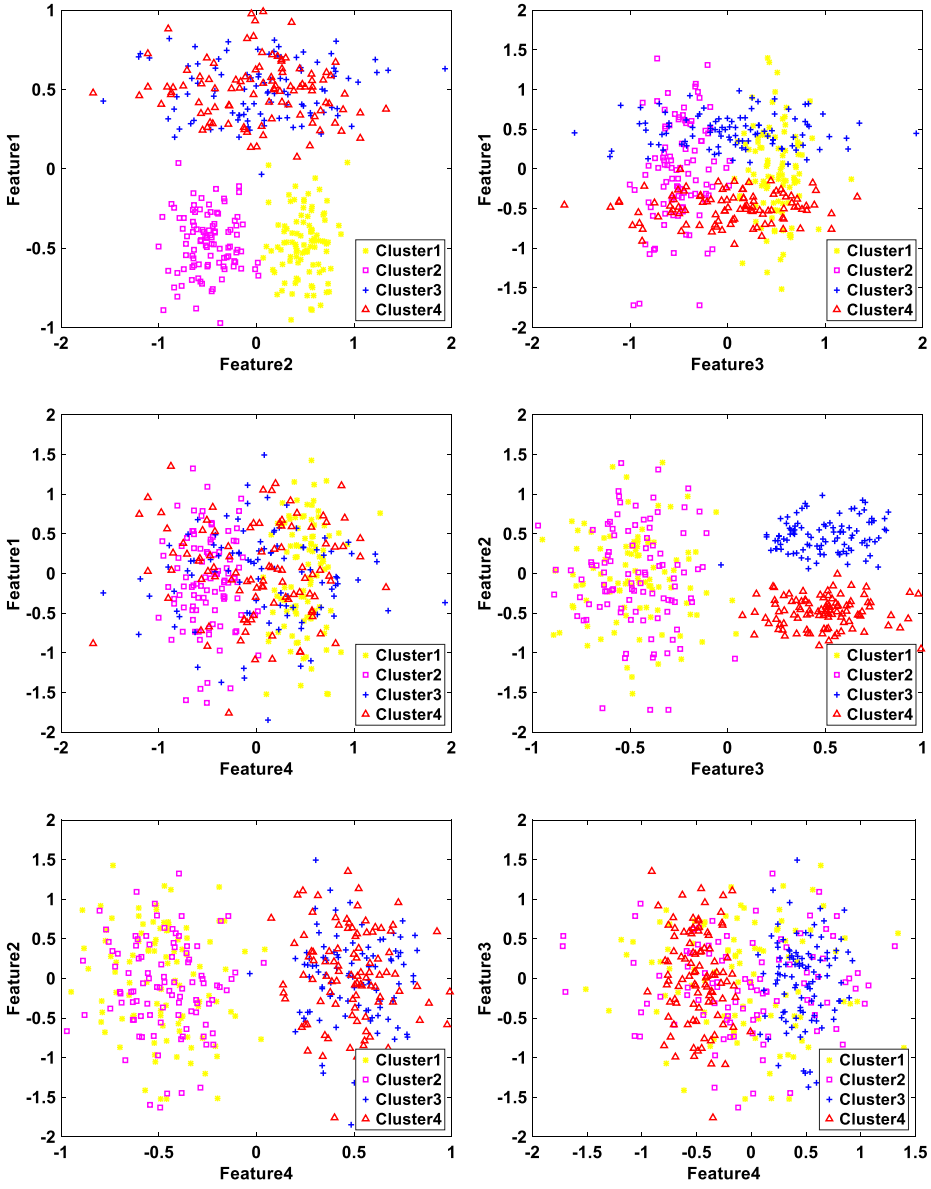

Fig. 6. Visualization of synthetic dataset in various subspaces of 2D features.

the sub-space. Each set contains four different 2D points. In the selection of these points, we consider to select different types of points that have both appropriate and inappropriate conditions for clustering. Selected points for this experiment are shown in Fig. 8. As shown in this figure, points in P4, P5, P6, P7, and P8 sets are inappropriate initial points (bad initializations).

We run the proposed algorithm once by assigning weights to the clusters employing our algorithm, and once without assigning weights. In the case of latter experiment, we assign the same weight for all the clusters so that $z_k = \frac{1}{K}$, where z_k represents kth cluster weight and K represents the total number of clusters. The clustering results from both experiments are presented in Fig. 8. As shown in this figure, the proposed algorithm has a better performance through cluster weighting than the case without cluster weighting. Therefore, due to cluster weighting, the proposed algorithm has a similar performance for any eight sets of the selected points. This clearly indicates the robustness of the algorithm against bad initializations. When the weight is not assigned to the clusters, the efficiency of the proposed algorithm is decreased. It is however evident in Fig. 8 that the performance drops markedly in the case of bad initializations compared to the appropriate initial points.

We also run k-means, FCM, SCAD2, RLFWHCM, MMKMC and EWFCM algorithms on this dataset using the same initial points.

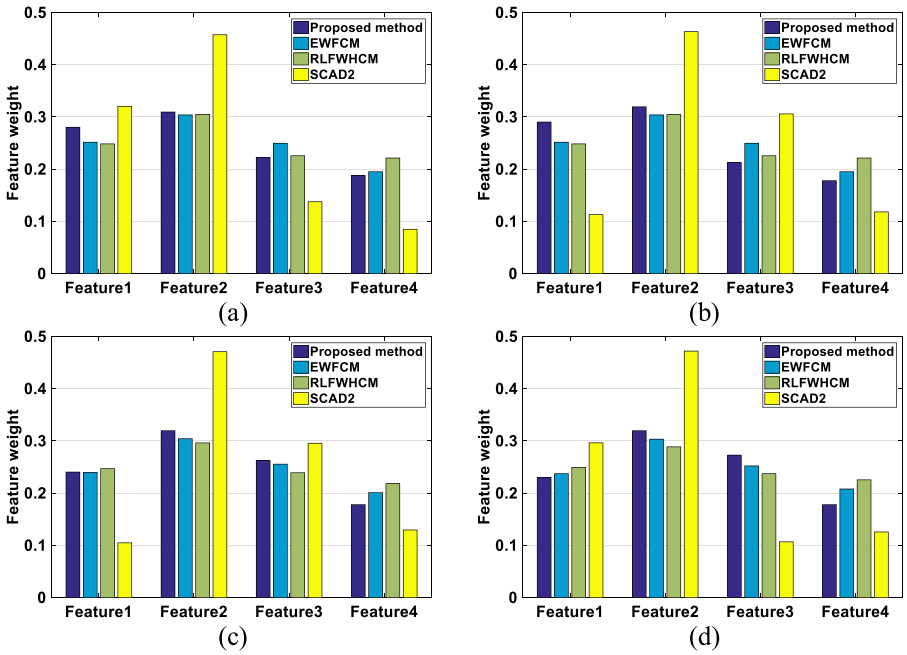

Fig. 7. Weights of the features assigned in synthetic dataset for (a) Cluster1, (b) Cluster2, (c) Cluster3, and (d) Cluster4.

In the proposed algorithm and RLFWHCM, we set the values of q and β equal to -10 and 0.3, respectively. The results are given in Fig. 9.

As shown in Fig. 9, the performance of the proposed algorithm is much better than the other methods, especially in the case of bad initializations. The second best result belongs to EWFCM algorithm. The reason why EWFCM is superior to the other two algorithms, RLFWHCM and MMKMC, is due to the shape and size of the clusters formed in this method. As such, it is possible to achieve a good performance even in case of a bad initialization. For this algorithm, the difference between the highest and the lowest accuracies is 0.125%. The other five algorithms, k-means, FCM, SCAD2, RLFWHCM and MMKMC, are severely sensitive to initialization, where the differences between the highest and the lowest accuracies are 0.22, 0.14, 0.135, 0.130 and 0.130 for k-means, FCM, SCAD2, RLFWHCM and MMKMC, respectively.

Based on these results, it is obvious that the cluster weighting scheme adopted in the proposed method robustly mitigates the sensitivity of the algorithm to initialization.

4.6. Experiment 3: clustering results on real world datasets

In this section, the clustering results of the proposed algorithm on the real world datasets are presented and compared with the results of the other methods. There are some common parameters between the proposed algorithm and some of the competitors. Using different values of these parameters may yield totally different clustering results. Therefore, comparisons are done separately to make a thorough study on the effect of different values of the adopted parameters in the efficiency of the algorithms.

4.6.1. Proposed algorithm vs. RLFWHCM

Here, the clustering results of the proposed algorithm on the real world datasets are presented and compared with the results of RLFWHCM method [33]. The common parameter between the proposed algorithm and RLFWHCM is q. For β, we choose a value between 0, 0.2 and 0.3 (see Section 3.3.1). Table 2 shows the obtained results for both methods. In each row of this table, the first line indicates ACC, and the second line represents NMI measure. The best results are in boldfaced.

As shown in Table 2, the proposed algorithm achieves better performances for all the datasets compared to RLFWHCM, on average 5.2012% for ACC and 2.8240% for NMI. It is worth mentioning that the proposed algorithm has the best performances for all values of q on the datasets *Wine*, *Glass*, *Dermatology*, *Zoo*, *Ionosphere*, *Breast Cancer Wisconsin* (Diagnostic), *Letter Recognition*, *Heberman*, *Breast Cancer Wisconsin* (Original), *Pima Indians Diabetes*, *Statlog* (heart), and *Balance*. However, for qs smaller than -2, it has a rather poor performance in terms of NMI.

It should be noted that the accuracy measure could not be used always as the only performance metric. Particularly, it should never be used as a measure when there is one dominant class (here the clusters) in the data [61]. In a dataset with a highly imbalance data, a model can assign the label of the dominant class for all the predictions and achieve a high classification accuracy which is not desired in the problem domain. This behavior (say problem) has been shown by RLFWHCM in a way that ACC measure is high on the *Ecoli* dataset, whereas NMI measure is very low, which indicates that the clustering has not been done appropriately.

Table 3 shows the average performances of the two methods for all the datasets considering different qs. The results in this table also confirm that the proposed algorithm performs better than RLFWHCM, on average 6.8998% for ACC and 8.8785% for NMI.

4.6.2. Proposed algorithm vs. MMKMC

In this section, we compare the performance of the proposed algorithm with MMKMC [54]. The common parameter between these two methods is β. Parameter q in the proposed algorithm is set to the value that yields the best result (among the values -10 and 10) (see Section 4.6.1). Table 4 compares the results of the two methods on all the real world datasets. In each row, the first line represents ACC and the second line indicates NMI.

As shown in Table 4, the proposed algorithm compared to MMKMC, performs better for all values of β on *Wine*, *Iris*, *Ecoli*, *Zoo*, *Ionosphere*, *Spectf heart*, *Breast Cancer Wisconsin* (Diagnostic), *Letter Recognition*, *Heberman*, *Breast Cancer Wisconsin* (Original),

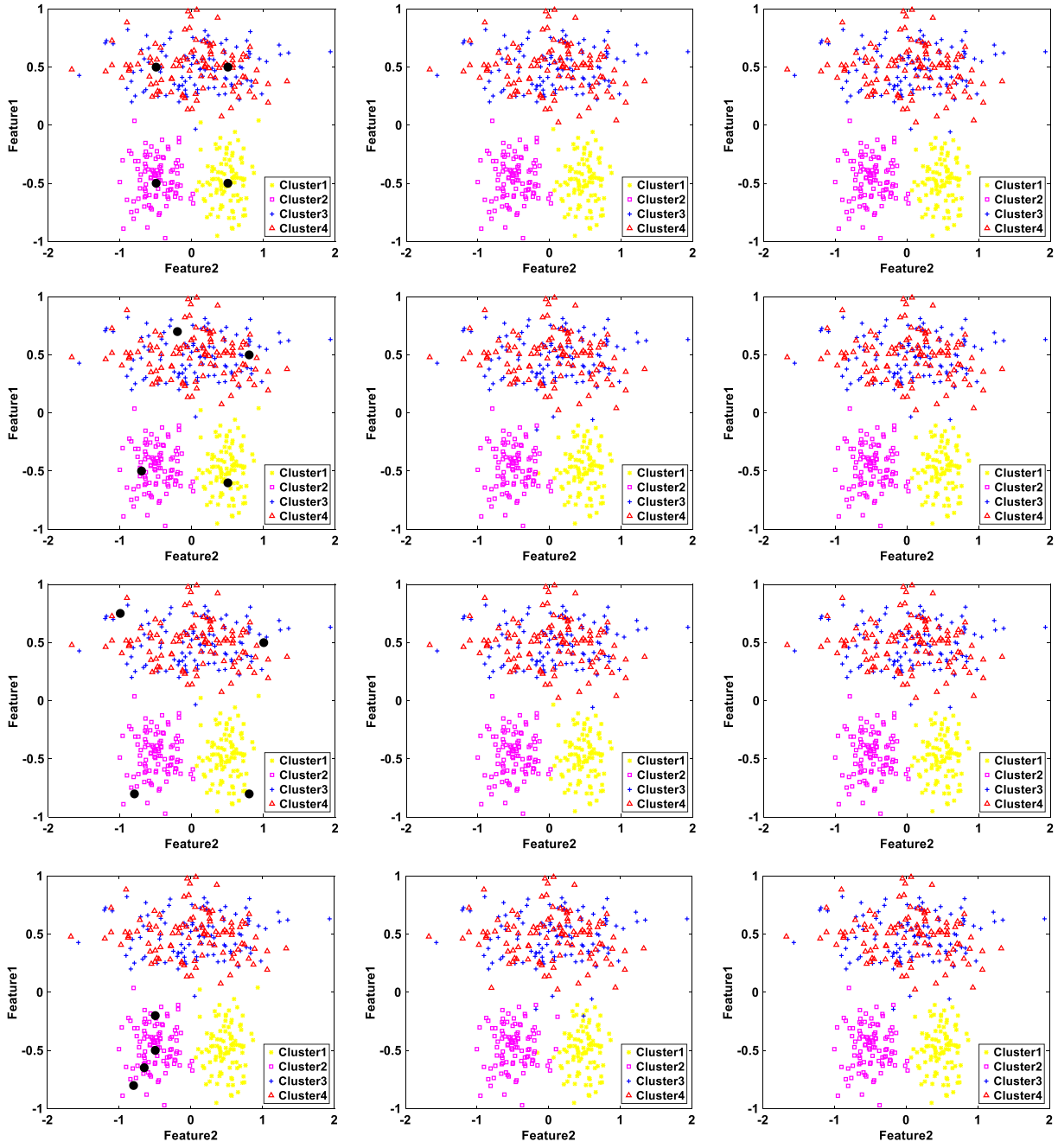

Fig. 8. The effect of cluster weighting in the proposed method. Column (a) indicates each set of the initial points. Point sets P1, P2, and P3 in rows 1, 2, and 3 are the appropriate initial points, and point sets P4, P5, P6, P7, and P8 in rows 4, 5, 6, 7, and 8 are inappropriate points (bad initializations). Column (b) shows clustering results obtained by the proposed method without using cluster weighting. Column (c) shows clustering results obtained by the proposed method using cluster weighting.

Statlog (heart), *Vowel* and *Balance* in terms of both ACC and NMI. For $\beta = 0.3$ and $\beta = 0.2$, our method shows higher performance on *Dermatology* in terms of ACC. However, it has a rather poor performance for all values of β in terms of NMI metrics. On *Glass*, the performance of MMKMC is better than ours in terms of both ACC and NMI metrics.

Table 5 shows the average performances. According to this table, the performance of the proposed method for all values of β is better than MMKMC and the best result is achieved for $\beta = 0.2$.

4.6.3. Proposed algorithm vs. k-means, FCM, SCAD2, k-means++ and EWFCM

In this section, we compare the performance of our method with k-means [3], FCM [4], SCAD2 [9], k-means++ [47] and EWFCM [22]. The results are shown in Table 6. In each row, the first line represents ACC, and the second line indicates NMI. The results of the proposed method are reported based on the best values of β and q (see Sections 4.6.1 and 4.6.2).

Table 6 shows that the proposed algorithm performs better on *Iris*, *Ecoli*, *Dermatology*, *Zoo*, *Breast Cancer Wisconsin* (Diagnostic), *Letter Recognition*, *Breast Cancer Wisconsin* (Original), *Statlog*

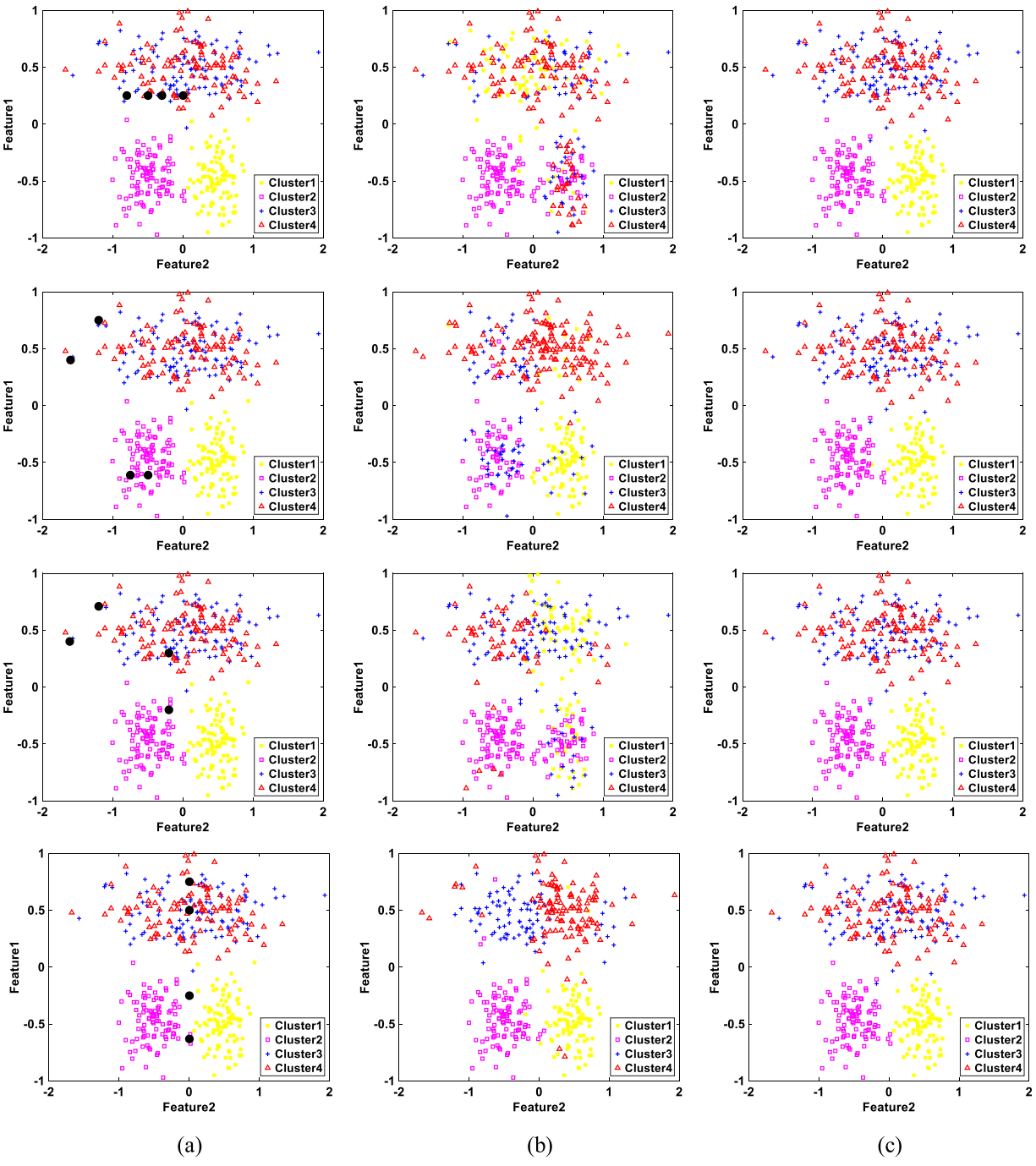

Fig. 8. (*continued*).

Table 5
The average performance of the proposed method and MMKMC on all the real world datasets.

Method	Metric	$\beta = 0.3$	$\beta = 0.2$	$\beta = 0$
Ours	ACC	**0.7584**	**0.7714**	**0.7321**
	NMI	**0.4901**	**0.4959**	**0.4470**
MMKMC	ACC	0.6599	0.6629	0.6579
	NMI	0.3708	0.3732	0.3646

(heart), *Pima Indians Diabetes*, *Vowel* and *Balance* datasets in terms of ACC and NMI. However, EWFCM has a fairly better performance on three datasets *Heberman*, *Ionosphere* and *Glass*. Additionally, although the proposed algorithm has a poor performance in terms of ACC on *Spectf heart*, it achieves a better performance in terms of NMI.

For EWFCM algorithm, the NMI rate is very low on *Spectf heart* dataset, indicating a low clustering quality. Table 7 shows the average performance on all the datasets. According to Table 7, the performance of the proposed method is obviously better than EWFCM method, both in terms of ACC and NMI criteria.

4.7. Experiment 4: effect of feature weighting on real world datasets

In order to examine the effect of feature weighting on the clustering of real world datasets, we evaluate the performance of

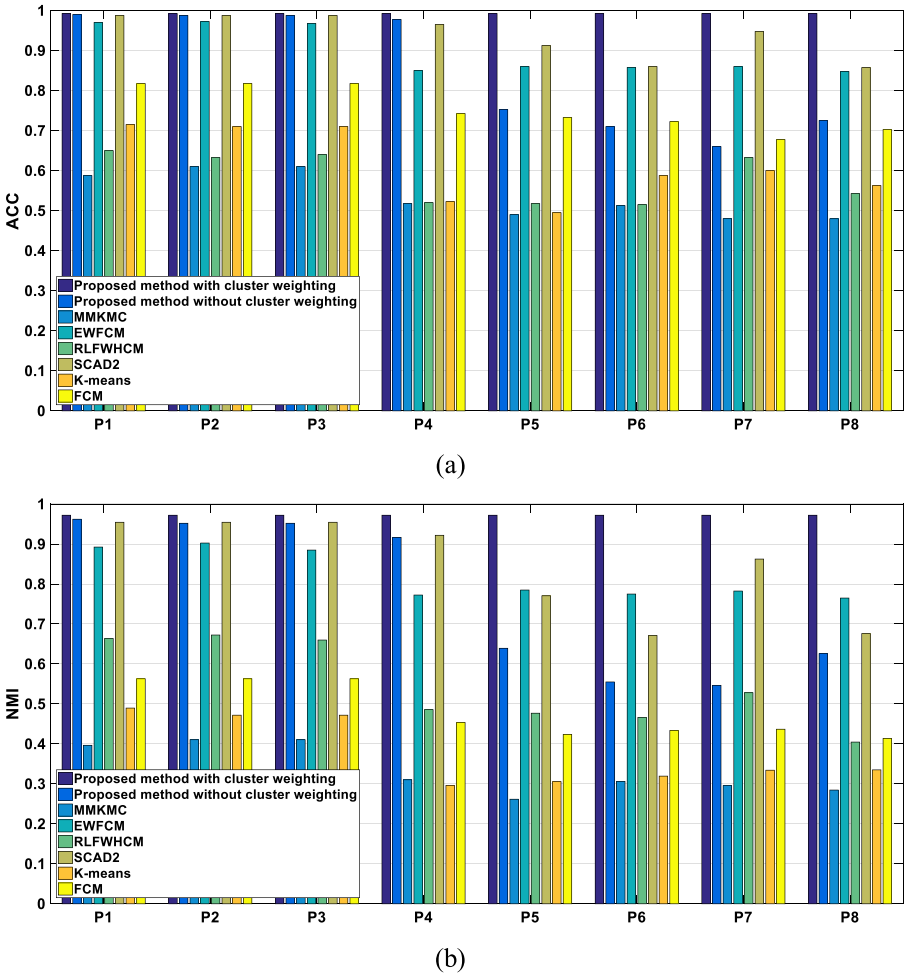

Fig. 9. The performance of different algorithms using the selected initial points: (a) ACC, and (b) NMI.

Table 6
The performances of the proposed algorithm and EWFCM on the real world datasets.

Dataset	EWFCM		k-means		FCM		CSAD2		k-means++		Ours	
	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
Ecoli	0.5351	0.5649	0.5312	0.5959	0.5101	0.5554	0.6425	0.6093	0.5479	0.6031	**0.7142**	**0.6121**
Iris	0.9666	0.8801	0.8363	0.7257	0.8933	0.7496	0.9600	0.8641	0.8533	0.7367	**0.9666**	**0.8801**
Wine	0.7022	0.4287	0.6817	0.4277	0.6853	0.4167	0.8764	0.6811	0.6893	0.4282	**0.9387**	**0.7967**
Glass	**0.5439**	**0.4263**	0.5106	0.3804	0.4904	0.3594	0.4630	0.3865	0.5304	0.4004	0.4766	0.3169
Dermatology	0.7233	0.7689	0.6819	0.7615	0.7233	0.7668	0.4740	0.4610	0.6923	0.8135	**0.7385**	**0.7748**
Zoo	0.7267	0.7154	0.6694	0.7054	0.5792	0.6672	0.7376	0.7759	0.7445	0.7362	**0.8455**	**0.8599**
Ionosphere	**0.7658**	**0.2026**	0.7039	0.1235	0.7094	0.1299	0.7094	0.1299	0.7061	0.1258	0.7094	0.1299
Breast Cancer Wisconsin (Original)	0.9613	0.7516	0.9613	0.7516	0.9560	0.7300	0.9531	0.7158	0.9607	0.7491	**0.9666**	**0.7750**
Letter Recognition	0.7618	0.4967	0.7179	0.4549	0.7804	0.4720	0.7699	0.4838	0.7172	0.4544	**0.8912**	**0.6803**
Breast Cancer Wisconsin (Diagnostic)	0.8760	0.5035	0.8541	0.4671	0.8541	0.4671	0.9086	0.5648	0.8541	0.4671	**0.9420**	**0.6825**
Pima Indians Diabetes	0.6576	0.0774	0.6601	0.0297	0.6588	0.0337	0.6757	0.0447	0.6601	0.0297	**0.6894**	**0.1052**
Heberman	**0.7712**	**0.0992**	0.5246	0.0042	0.5098	0.0025	0.7549	0.0970	0.5153	0.0005	0.7590	0.0947
Statlog (heart)	0.7114	0.2011	0.5902	0.0189	0.5018	0.1474	0.8296	0.3370	0.5896	0.0186	**0.8444**	**0.3731**
Balance	0.4987	0.1011	0.5123	0.1107	0.5023	0.0953	0.5049	0.1083	0.5163	0.1134	**0.5233**	**0.1212**
Vowel	0.5458	0.5018	0.5308	0.4853	0.4833	0.4716	0.5946	0.5538	0.5161	0.4850	**0.6451**	**0.5760**
Spectf heart	**0.7278**	0.0445	0.6250	0.0897	0.5018	0.1474	0.5056	0.1513	0.6254	0.0897	0.6254	**0.2077**

Table 7
The average value of the results of the proposed method and EWFCM on the all real world datasets.

Metric	EWFCM	k-means	FCM	CSAD2	k-means++	**Ours**
ACC	0.7172	0.6620	0.6462	0.7100	0.6699	**0.7672**
NMI	0.4227	0.3833	0.3883	0.4353	0.3907	**0.4992**

the proposed algorithm once by assigning weights to the features, and once without considering the weighting of the features. In case in which the feature weighting is not considered, the same weights are assigned to all the features in each cluster, so that $w_{km} = \frac{1}{M}$, where w_{km} represents the weight of the *mth* feature in the *kth* cluster and M indicates the total number of features. These weights are kept fixed while the algorithm is running. Fig. 10 shows the average results of 100 runs of the algorithms on each dataset.

As shown in Fig. 10, the proposed algorithm performs better while applying the weighting scheme on the features. This is more apparent due to very different levels of importance of the features in some datasets, in the case of datasets such as *Iris*, *Ecoli*, *Dermatology*, *Zoo*, *Breast Cancer Wisconsin* (Diagnostic), *Letter Recognition*, *Statlog* (heart)", *Heberman*, *Vowel*, and *Spectf heart*, where different features have very different levels of importance.

4.8. Experiment 5: robustness of the proposed method over initialization problem

In order to demonstrate the robustness of the proposed method against initialization problem and to gain an understanding of its advantages compared to other methods, similar to the experiments performed in [54], we conduct an experiment where the cluster centers obtained by proposed method are used to initialize the standard k-means and FCM algorithms. The same experiments are done for methods MMKMC [54] and k-means++ [47]. The obtained clustering results in this way allow us to determine if k-means or FCM performance can be improved when initialized by solutions of these three approaches, and also, facilitates the comparison of the tested methods under a common objective function (k-means objective function or FCM objective function). Accordingly, the robustness of different methods against initialization problem and their ability to find the proper cluster centers are evaluated very well.

The results of this experiment using k-means and FCM clustering algorithms are summarized in Tables 8 and 9 respectively. In these tables, the combination of different methods with k-means/FCM algorithms are represented using "+" sign. For example, "Ours + k-means", refers to the experiment where solution obtained by our method is used to initialize a subsequent run of k-means.

As shown in Table 8, the Ours + k-means method compared to MMKMC + k-means, performs better on *Iris*, *Ecoli*, *Zoo*, *Letter Recognition*, *Vowel, Dermatology* and *Balance* in terms of ACC, NMI and the objective function value (the lower objective function value is the better). The ACC and NMI metrics and objective function value for *Wine*, *Ionosphere*, *Spectf heart*, *Breast Cancer Wisconsin* (Diagnostic), *Breast Cancer Wisconsin* (Original), *Statlog* (heart) and *Pima Indians Diabetes* datasets are almost equal for both methods. For *Glass* and *Heberman* datasets, the objective function value for the MMKMC + k-means method is better than the Ours + k-means method. However, the ACC and NMI metrics for the Ours + k-means are much higher than the MMKMC + k-means method.

Table 8 shows that the Ours + k-means method compared to k-means++ + k-means method, performs better on *Wine*, *Iris*, *Ecoli*, *Glass*, *Zoo*, *Ionosphere*, *Spectf heart*, *Breast Cancer Wisconsin* (Diagnostic), *Letter Recognition*, *Heberman*, *Statlog* (heart), *Vowel*, *Pima Indians Diabetes, Dermatology* and *Balance* in terms of ACC, NMI and objective function value. For *Breast Cancer Wisconsin* (Original) dataset, the ACC and NMI metrics for the k-means++ + k-means method are just slightly higher than the Ours + k-means method, but the objective function value for the k-means++ + k-means method is higher than the Ours + k-means method.

As shown in Table 9, the Ours + FCM method outperforms two other methods for all the metrics, demonstrating its ability to provide better cluster centers. Of course, for some datasets, the ACC and NMI metrics are equal for the Ours + FCM method and other methods.

Considering the overall results in Tables 8 and 9, the k-means (FCM) algorithm, when initialized by our algorithm, leads to a lower objective function value for most datasets. To add to the above, Ours + k-means and Ours + FCM methods obtain higher NMI and ACC values than k-Means and FCM (see Table 6), i.e. k-means (FCM) converges to a better local minima when initialized by our method. In fact, the main difference between k-means (FCM) and Ours + k-means (Ours + FCM) is that some of k-means (FCM) restarts return solutions with excessively high objective function, while for Ours + k-means (Ours + FCM) such poor out comes do note emerge, illustrating the robustness of our method over bad initializations.

The stochastic initialization process of k-means++ + k-means (k-means++ + FCM) improves performance, as lower objective function (and equal ACC and NMI) values are achieved on most cases compared to the randomly restarted k-means (FCM). When put up against proposed method though, similar conclusions to those mentioned above for k-means and FCM can be drawn, further confirming the potential of the proposed algorithm. It is of particular interest that Ours + k-means (Ours + FCM) yields better objective function values and ACC and NMI scores on every dataset, despite k-means++ + k-means (k-means++ + FCM) carefully picking the initial centers. This definitely reveals that the centers obtained by Ours + k-means (Ours + FCM) consist good initializations for k-means (FCM).

5. Conclusion

So far, numerous researches have been conducted in designing clustering algorithms that can benefit from feature weighting. Extensive researches have also been proposed to reduce the sensitivity of the fuzzy c-means algorithm with respect to initialization. Till now, however, feature weighting in a local manner and not being sensitive with respect to initialization has not yet been applied simultaneously.

In this research, a clustering algorithm was proposed. The algorithm assigns weights to the clusters based on their intra-cluster distances, and also, it takes the importance of the features in each cluster into account (sum of the intra-cluster weighted-feature distance). The results of the experiments on a large real world datasets as well as a synthetic dataset confirmed that the proposed algorithm has a very promising performance, and it is not sensitive to the initialization of the cluster centers, even for bad initialization, and demonstrated better clustering results than the competitors. According to the nature of the algorithm designed in the proposed method and the results of extensive experiments on various datasets, it seems that the proposed method can be efficiently used as an online feature weighting and cluster weighting-based clustering method on big data clustering and co-clustering applications.

As a future direction, we are interested in improving the proposed algorithm using Kernel-based clustering approach [62] which allows the algorithm to distinguish the clusters that have linearly non-separable patterns or non-hyper-spherical shapes.

Appendix

Firstly, we present the steps to obtain the updating equation for the membership function u_{nk} from the objective function F. The objective function F is written as the Lagrange $\tilde{F}$ equation. These steps are as follows:

$$\frac{\partial \tilde{\mathrm{F}}}{\partial u_{nk}} = 0$$

$$\sum_{m=1}^{M} \alpha {u_{nk}}^{\alpha-1} w_{km}^{q} z_k^p d^2(x_{nm} - c_{km}) - \delta = 0$$

$$u_{nk} = \left[\frac{\delta}{\alpha z_k^p \sum_{m=1}^{M} w_{km}^{q} d^2(x_{nm} - c_{km})} \right]^{\frac{1}{\alpha-1}} \tag{A.1}$$

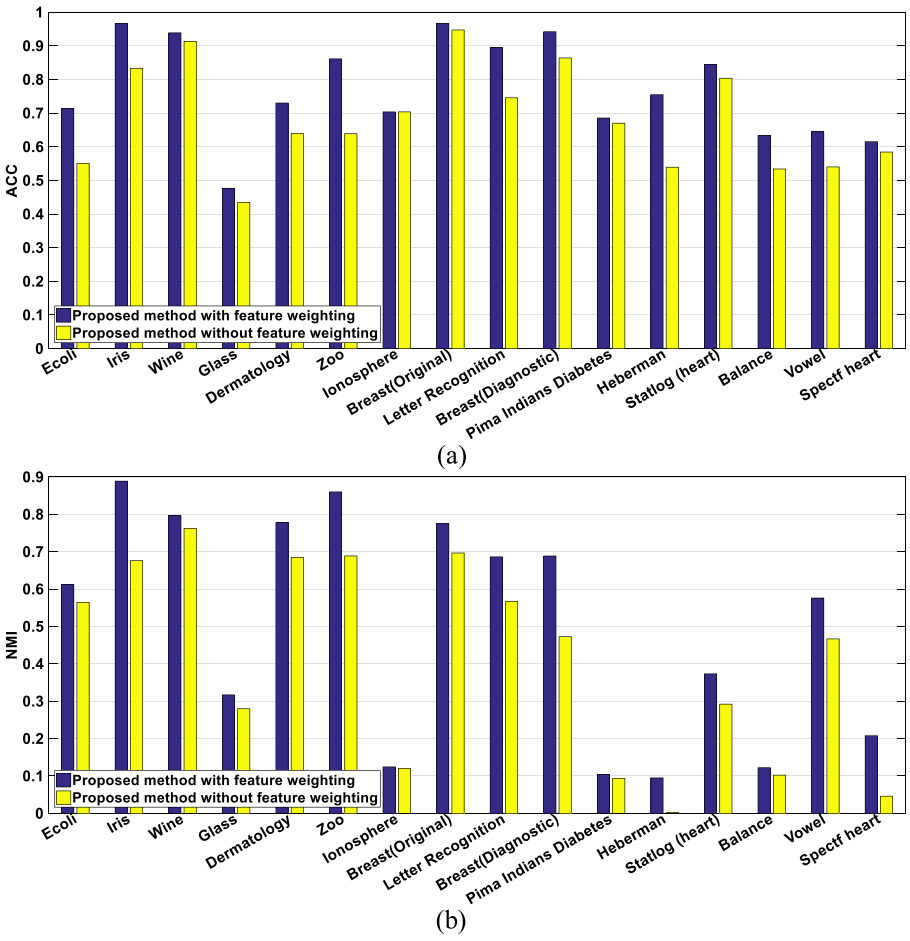

Fig. 10. Effect of feature weighting on the quality of clustering on real world datasets. (a) ACC, and (b) NMI.

Table 8
Comparative results on real world datasets when k-means is initialized by the solution returned by MMKMC, k-means++, and our algorithm.

Dataset	MMKMC + k-means			k-means++ + k-means			Ours + k-means		
	ACC	NMI	Objective function value	ACC	NMI	Objective function value	ACC	NMI	Objective function value
Ecoli	0.5494	0.5982	14.4571	0.5479	0.6031	14.9596	**0.5887**	**0.6134**	**14.3653**
Iris	0.8866	0.7419	78.9450	0.8533	0.7367	8.3933	**0.8933**	**0.7582**	**78.9408**
Wine	**0.7022**	**0.4287**	**2370689.6867**	0.6893	0.4282	2396976.2513	**0.7022**	**0.4287**	**2370689.6867**
Glass	0.5294	0.4098	**345.5294**	0.5304	0.4004	383.7175	**0.5401**	**0.4102**	370.9211
Dermatology	0.7052	0.8495	3560.7358	0.6923	0.8135	3639.6618	**0.7132**	**0.8501**	**3560.7340**
Zoo	0.7366	0.7447	126.6725	0.7445	0.7362	142.6589	**0.7910**	**0.7659**	**126.3213**
Ionosphere	**0.7122**	**0.1349**	**2419.3648**	0.7061	0.1258	2483.0288	**0.7122**	**0.1349**	**2419.3648**
Breast Cancer Wisconsin (Original)	0.9604	0.7478	**19323.1738**	**0.9607**	**0.7491**	19323.1800	0.9604	0.7478	**19323.1738**
Letter Recognition	0.7171	0.4533	102136.4622	0.7172	0.4544	102136.9617	**0.7184**	**0.4545**	**102136.2308**
Breast Cancer Wisconsin (Diagnostic)	**0.8541**	**0.4671**	**77943099.8782**	**0.8541**	**0.4671**	77983660.9087	**0.8541**	**0.4671**	**77943099.8782**
Pima Indians Diabetes	**0.6601**	**0.0297**	**5142376.4559**	**0.6601**	**0.0297**	5182937.4863	**0.6601**	**0.0297**	**5142376.4559**
Heberman	0.5205	0.0001	**30525.3585**	0.5153	0.0005	71103.8286	**0.7483**	**0.0656**	43425.3943
Statlog (heart)	**0.5925**	**0.0199**	**549314.6882**	0.5896	0.0186	589894.6605	0.5888	0.0184	549340.4078
Balance	0.5040	0.1042	3501.3653	0.5163	0.1134	3499.0082	**0.5246**	**0.1341**	**3493.1398**
Vowel	0.5110	0.4860	30939375.2076	0.5161	0.4850	31874264.5508	**0.5168**	**0.4874**	**30746149.2968**
Spectf heart	**0.6254**	**0.0897**	**826606.7537**	**0.6254**	**0.0897**	**826606.7537**	**0.6254**	**0.0897**	**826606.7537**

Now, we only need to obtain the value of δ. To obtain δ, the following steps are carried out:

$$\frac{\partial \tilde{J}}{\partial u_{nl}} = 0$$

$$\delta = \sum_{m=1}^{M} \alpha {u_{nl}}^{\alpha-1} w_{lm}^{q} z_l^p d^2(x_{nm} - c_{lm})$$

$$\delta = \alpha {u_{nl}}^{\alpha-1} z_l^p \sum_{m=1}^{M} w_{lm}^{q} d^2(x_{nm} - c_{lm}) \tag{A.2}$$

From Eqs. (1) and (2), it is concluded that:

$$u_{nk} = \left[\frac{\alpha {u_{nl}}^{\alpha-1} z_l^p \sum_{m=1}^{M} w_{lm}^{q} d^2(x_{nm} - c_{lm})}{\alpha z_k^p \sum_{m=1}^{M} w_{km}^{q} d^2(x_{nm} - c_{km})} \right]^{\frac{1}{\alpha-1}}$$

Table 9
Comparative results on real world datasets when FCM is initialized by the solution returned by MMKMC, k-means++, and our algorithm.

Dataset	MMKMC + FCM			k-means++ + FCM			Ours + FCM		
	ACC	NMI	Objective function value	ACC	NMI	Objective function value	ACC	NMI	Objective function value
Ecoli	0.5152	0.5536	5.3743	0.5107	0.5551	5.3586	**0.5164**	**0.5557**	**5.3547**
Iris	**0.8933**	**0.7496**	**60.5759**	**0.8933**	**0.74963**	**60.5759**	**0.8933**	**0.7496**	**60.5759**
Wine	**0.6853**	**0.4167**	1796082.7604	0.6853	0.4167	1796082.7603	**0.6853**	**0.4167**	**1796082.7601**
Glass	0.4854	0.3468	157.3818	0.4897	0.3572	154.7284	**0.4906**	**0.3594**	**154.1459**
Dermatology	0.7021	0.7746	1383.7241	0.7102	0.7889	1383.6792	**0.7155**	**0.7897**	**1383.6700**
Zoo	0.6336	0.6917	63.2147	0.6009	0.6650	62.7767	**0.6732**	**0.6961**	**62.6229**
Ionosphere	**0.7094**	**0.1299**	**1543.9842**	**0.7094**	**0.1299**	**1543.9842**	**0.7094**	**0.1299**	**1543.9842**
Breast Cancer Wisconsin (Original)	**0.9560**	**0.7300**	**14916.6839**	**0.9560**	**0.7300**	**14916.6839**	**0.9560**	**0.7300**	**14916.6839**
Letter Recognition	0.7804	0.4720	56180.6723	0.7804	0.4720	**56097.4084**	**0.7832**	**0.4766**	**56097.4084**
Breast Cancer Wisconsin (Diagnostic)	**0.8541**	**0.4671**	62075261.0031	**0.8541**	**0.4671**	62075261.0051	**0.8541**	**0.4671**	**62075261.0023**
Pima Indians Diabetes	**0.6588**	**0.0337**	**3986824.9489**	**0.6588**	**0.0337**	**3986824.9491**	**0.6588**	**0.0337**	**3986824.9489**
Heberman	**0.5098**	**0.0025**	**21582.6290**	**0.5098**	**0.0025**	**21582.6290**	**0.5098**	**0.0025**	**21582.6290**
Statlog (heart)	**0.5925**	**0.0214**	406868.0964	**0.5925**	**0.0214**	406868.09640	**0.5925**	**0.0214**	**406868.0963**
Balance	0.5985	0.0634	**1666.6666**	0.6003	0.0912	**1666.6666**	**0.6165**	**0.1168**	**1666.6666**
Vowel	**0.4833**	**0.4716**	17108477.6810	**0.4833**	**0.4716**	17108477.6812	**0.4833**	**0.4716**	**17108477.6807**
Spectf heart	0.5018	0.1474	580844.0823	0.5018	0.1474	572868.7894	**0.5430**	**0.1715**	**572868.7893**

$$u_{nk} = u_{nl}\left[\frac{z_l^p \sum_{m=1}^{M} w_{lm}^q d^2(x_{nm}-c_{lm})}{z_k^p \sum_{m=1}^{M} w_{km}^q d^2(x_{nm}-c_{km})}\right]^{\frac{1}{\alpha-1}}$$
$$u_{nl} = u_{nk}\left[\frac{z_k^p \sum_{m=1}^{M} w_{km}^q d^2(x_{nm}-c_{km})}{z_l^p \sum_{m=1}^{M} w_{lm}^q d^2(x_{nm}-c_{lm})}\right]^{\frac{1}{\alpha-1}}$$
$$\sum_{l=1}^{K} u_{nl} = \sum_{l=1}^{K} u_{nk}\left[\frac{z_k^p \sum_{m=1}^{M} w_{km}^q d^2(x_{nm}-c_{km})}{z_l^p \sum_{m=1}^{M} w_{lm}^q d^2(x_{nm}-c_{lm})}\right]^{\frac{1}{\alpha-1}} \quad (A.3)$$

Considering $\sum_{l=1}^{K} u_{nl} = 1$, Eq. (3) is written as follows:

$$1 = u_{nk}\sum_{l=1}^{K}\left[\frac{z_k^p \sum_{m=1}^{M} w_{km}^q d^2(x_{nm}-c_{km})}{z_l^p \sum_{m=1}^{M} w_{lm}^q d^2(x_{nm}-c_{lm})}\right]^{\frac{1}{\alpha-1}}$$
$$u_{nk} = \frac{1}{\sum_{l=1}^{K}\left[\frac{z_k^p \sum_{m=1}^{M} w_{km}^q d^2(x_{nm}-c_{km})}{z_l^p \sum_{m=1}^{M} w_{lm}^q d^2(x_{nm}-c_{lm})}\right]^{\frac{1}{\alpha-1}}}$$

Thus, the membership function (in Eq. (6) in Section 3.1) is obtained.

Now let us explain the steps for obtaining the updating function c_{km}:

$$\frac{\partial \tilde{F}}{\partial c_{km}} = 0$$
$$\sum_{n=1}^{N} 2\, {u_{nk}}^{\alpha} w_{km}^q z_k^p (-\gamma_m)(x_{nm}-c_{km})\left(\exp\left(-\gamma_m (x_{nm}-c_{km})^2\right)\right) = 0 \quad (A.4)$$

Eq. (4) is extended as follows:

$$\sum_{n=1}^{N} 2\, {u_{nk}}^{\alpha} w_{km}^q z_k^p (-\gamma_m)(x_{nm})\left(\exp\left(-\gamma_m (x_{nm}-c_{km})^2\right)\right) - \sum_{n=1}^{N} 2\, {u_{nk}}^{\alpha} w_{km}^q z_k^p (-\gamma_m)(c_{km})\left(\exp\left(-\gamma_m (x_{nm}-c_{km})^2\right)\right) = 0$$
$$2 w_{km}^q z_k^p (-\gamma_m)\sum_{n=1}^{N} {u_{nk}}^{\alpha}(x_{nm})\left(\exp\left(-\gamma_m (x_{nm}-c_{km})^2\right)\right) - 2 w_{km}^q z_k^p (-\gamma_m)(c_{km})\sum_{n=1}^{N} {u_{nk}}^{\alpha}(x_{nm})\left(\exp\left(-\gamma_m (x_{nm}-c_{km})^2\right)\right) = 0$$

$$c_{km} = \frac{\sum_{n=1}^{N} {u_{nk}}^{\alpha}\left(\exp\left(-\gamma_m (x_{nm}-c_{km})^2\right)\right)(x_{nm})}{\sum_{n=1}^{N} {u_{nk}}^{\alpha}\left(\exp\left(-\gamma_m (x_{nm}-c_{km})^2\right)\right)}$$

As a result, the updating function of cluster centers (in Eq. (7) in Section 3.1) is obtained.

With regard to w_{km}, we should say that if $Dw_{km} = 0$, and h_m is the number of features like s, which Dw_{ks} is zero for them, then w_{km} is defined as $\frac{1}{h_m}$. On the other hand, if $Dw_{km} \neq 0$, but there does not exist a feature like s, which Dw_{ks} is zero for it, then the value w_{km} is calculated as follows:

$$\frac{\partial \tilde{F}}{\partial w_{km}} = 0$$
$$\sum_{n=1}^{N} {u_{nk}}^{\alpha} q w_{km}^{q-1} z_k^p d^2(x_{nm}-c_{km}) - \psi = 0$$
$$w_{km} = \left[\frac{\psi}{q z_k^p \sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{nm}-c_{km})}\right]^{\frac{1}{q-1}} \quad (A.5)$$

Now, we only need to obtain the value ψ. To obtain the value ψ, we do as follows:

$$\frac{\partial \tilde{F}}{\partial w_{ks}} = 0$$
$$\psi = \sum_{n=1}^{N} {u_{nk}}^{\alpha} q w_{ks}^{q-1} z_k^p d^2(x_{ns}-c_{ks})$$
$$\psi = q w_{ks}^{q-1} z_k^p \sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{ns}-c_{ks}) \quad (A.6)$$

From the Eqs. (3) and (4) we conclude:

$$w_{km} = \left[\frac{q w_{ks}^{q-1} z_k^p \sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{ns}-c_{ks})}{q z_k^p \sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{nm}-c_{km})}\right]^{\frac{1}{q-1}}$$
$$w_{km} = w_{ks}\left[\frac{\sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{ns}-c_{ks})}{\sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{nm}-c_{km})}\right]^{\frac{1}{q-1}}$$
$$w_{ks} = w_{km}\left[\frac{\sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{nm}-c_{km})}{\sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{ns}-c_{ks})}\right]^{\frac{1}{q-1}}$$
$$\sum_{s=1}^{M} w_{ks} = \sum_{s=1}^{M} w_{km}\left[\frac{\sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{nm}-c_{km})}{\sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{ns}-c_{ks})}\right]^{\frac{1}{q-1}} \quad (A.7)$$

Considering $\sum_{s=1}^{M} w_{ks} = 1$, we can write Eq. (7) as follows:

$$1 = \sum_{s=1}^{M} w_{km} \left[\frac{\sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{nm} - c_{km})}{\sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{ns} - c_{ks})} \right]^{\frac{1}{q-1}}$$

$$w_{km} = \frac{1}{\sum_{s=1}^{M} \left[\frac{\sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{nm} - c_{km})}{\sum_{n=1}^{N} {u_{nk}}^{\alpha} d^2(x_{ns} - c_{ks})} \right]^{\frac{1}{q-1}}}$$

Thus, the matrix for updating the weights (in Eq. (8) in Section 3.1) is obtained.

Regarding z_k, it should be said that if $Dz_k = 0$, and g_k be the number of clusters like l, which Dz_l is zero for them, then z_k is defined as $\frac{1}{g_k}$. On the other hand, if $Dz_k \neq 0$, but there is a cluster like l, which Dz_l is zero for it, the value of z_k will be equal to zero. If there are no cluster like l, which Dz_l is zero for it, then the value z_k is calculated as follows:

$$\frac{\partial \tilde{F}}{\partial z_k} = 0$$

$$\sum_{n=1}^{N} \sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} \, \mathrm{p} \, z_k^{p-1} d^2(x_{nm} - c_{km}) - \omega = 0$$

$$z_k = \left[\frac{\omega}{\mathrm{p} \sum_{n=1}^{N} \sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} \, d^2(x_{nm} - c_{km})} \right]^{\frac{1}{p-1}} \tag{A.8}$$

Now we only need to obtain the value of ω. To obtain ω, we do as follows:

$$\frac{\partial \tilde{F}}{\partial z_l} = 0$$

$$\omega = \sum_{n=1}^{N} \sum_{k=1}^{K} {u_{nl}}^{m} w_{lm}^{q} \, \mathrm{p} \, z_l^{p-1} d^2(x_{nm} - c_{lm})$$

$$\omega = \mathrm{p} \, z_l^{p-1} \sum_{n=1}^{N} \sum_{k=1}^{K} {u_{nl}}^{m} w_{lm}^{q} d^2(x_{nm} - c_{lm}) \tag{A.9}$$

From Eqs. (6) and (7) we conclude that:

$$z_k = \left[\frac{\mathrm{p} \, z_l^{p-1} \sum_{n=1}^{N} \sum_{k=1}^{K} {u_{nl}}^{m} w_{lm}^{q} d^2(x_{nm} - c_{lm})}{\mathrm{p} \sum_{n=1}^{N} \sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} \, d^2(x_{nm} - c_{km})} \right]^{\frac{1}{p-1}}$$

$$z_k = z_l \left[\frac{\sum_{n=1}^{N} \sum_{k=1}^{K} {u_{nl}}^{m} w_{lm}^{q} d^2(x_{nm} - c_{lm})}{\sum_{n=1}^{N} \sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} \, d^2(x_{nm} - c_{km})} \right]^{\frac{1}{p-1}}$$

$$z_l = z_k \left[\frac{\sum_{n=1}^{N} \sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} \, d^2(x_{nm} - c_{km})}{\sum_{n=1}^{N} \sum_{k=1}^{K} {u_{nl}}^{m} w_{lm}^{q} d^2(x_{nm} - c_{lm})} \right]^{\frac{1}{p-1}}$$

$$\sum_{l=1}^{K} z_l = \sum_{l=1}^{K} z_k \left[\frac{\sum_{n=1}^{N} \sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} \, d^2(x_{nm} - c_{km})}{\sum_{n=1}^{N} \sum_{k=1}^{K} {u_{nl}}^{m} w_{lm}^{q} d^2(x_{nm} - c_{lm})} \right]^{\frac{1}{p-1}} \tag{A.10}$$

Considering that $\sum_{l=1}^{K} z_l = 1$, then Eq. (10) is written as follows:

$$1 = z_k \sum_{l=1}^{K} \left[\frac{\sum_{n=1}^{N} \sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} \, d^2(x_{nm} - c_{km})}{\sum_{n=1}^{N} \sum_{k=1}^{K} {u_{nl}}^{m} w_{lm}^{q} d^2(x_{nm} - c_{lm})} \right]^{\frac{1}{p-1}}$$

$$z_k = \frac{1}{\sum_{l=1}^{K} \left[\frac{\sum_{n=1}^{N} \sum_{m=1}^{M} {u_{nk}}^{\alpha} w_{km}^{q} \, d^2(x_{nm} - c_{km})}{\sum_{n=1}^{N} \sum_{k=1}^{K} {u_{nl}}^{m} w_{lm}^{q} d^2(x_{nm} - c_{lm})} \right]^{\frac{1}{p-1}}}$$

Thus, the updating matrix of the weights (in Eq. (9) in Section 3.1) is obtained.

References

[1] V.N. Vapnik, V. Vapnik, Statistical Learning Theory, Vol. 1, Wiley, New York, 1998.
[2] J. Han, J. Pei, M. Kamber, Data Mining: Concepts and Techniques, Elsevier, 2011.
[3] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, 1967, pp. 281-297.
[4] J.C. Bezdek, Objective function clustering, in: Pattern Recognition with Fuzzy Objective Function Algorithms, Springer, 1981, pp. 43–93.
[5] T. Kanungo, D.M. Mount, N.S. Netanyahu, C.D. Piatko, R. Silverman, A.Y. Wu, An efficient k-means clustering algorithm: Analysis and implementation, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2002) 881–892.
[6] L. Zhu, F.-L. Chung, S. Wang, Generalized fuzzy c-means clustering algorithm with improved fuzzy partitions, IEEE Trans. Syst. Man Cybern. B 39 (2009) 578–591.
[7] P. Huang, D. Zhang, Locality sensitive C-means clustering algorithms, Neurocomputing 73 (2010) 2935–2943.
[8] Y. Liu, F. Tian, Z. Hu, C. DeLisi, Evaluation and integration of cancer gene classifiers: identification and ranking of plausible drivers, Sci. Rep. 5 (2015).
[9] H. Frigui, O. Nasraoui, Unsupervised learning of prototypes and attribute weights, Pattern Recognit. 37 (2004) 567–581.
[10] S. Sivarathri, A. Govardhan, Experiments on hypothesis fuzzy k-means is better than k-means for clustering, Int. J. Data Mining Knowl. Manag. Process 4 (2014) 21.
[11] A. Stetco, X.-J. Zeng, J. Keane, Fuzzy C-means++: Fuzzy C-means with effective seeding initialization, Expert Syst. Appl. 42 (2015) 7541–7548.
[12] S. Askari, N. Montazerin, M.H. Fazel Zarandi, Generalized possibilistic Fuzzy C-means with novel cluster validity indices for clustering noisy data, Appl. Soft Comput. 53 (2017) 262–283.
[13] Y. Chen, H. Zhang, Y. Zheng, B. Jeon, Q.M.J. Wu, An improved anisotropic hierarchical fuzzy c-means method based on multivariate student t-distribution for brain MRI segmentation, Pattern Recognit. 60 (2016) 778–792.
[14] J. Aparajeeta, P.K. Nanda, N. Das, Modified possibilistic fuzzy C-means algorithms for segmentation of magnetic resonance image, Appl. Soft Comput. 41 (2016) 104–119.
[15] Z. Ji, J. Liu, G. Cao, Q. Sun, Q. Chen, Robust spatially constrained fuzzy c-means algorithm for brain MR image segmentation, Pattern Recognit. 47 (2014) 2454–2466.
[16] O.P. Mahela, A.G. Shaik, Recognition of power quality disturbances using S-transform based ruled decision tree and fuzzy C-means clustering classifiers, Appl. Soft Comput. 59 (2017) 243–257.
[17] E. Esme, B. Karlik, Fuzzy C-means based support vector machines classifier for perfume recognition, Appl. Soft Comput. 46 (2016) 452–458.
[18] K. Siang Tan, N.A. Mat Isa, Color image segmentation using histogram thresholding – Fuzzy C-means hybrid approach, Pattern Recognit. 44 (2011) 1–15.
[19] V. T, Performance based analysis between k-means and Fuzzy C-means clustering algorithms for connection oriented telecommunication data, Appl. Soft Comput. 19 (2014) 134–146.
[20] A. Klomsae, S. Auephanwiriyakul, N. Theera-Umpon, A string grammar fuzzy-possibilistic C-medians, Appl. Soft Comput. 57 (2017) 684–695.
[21] R. Xu, D. Wunsch, Survey of clustering algorithms, IEEE Trans. Neural Netw. 16 (2005) 645–678.
[22] J. Zhou, L. Chen, C.L.P. Chen, Y. Zhang, H.-X. Li, Fuzzy clustering with the entropy of attribute weights, Neurocomputing 198 (2016) 125–134.
[23] J.M. Peña, J.A. Lozano, P. Larrañaga, An empirical comparison of four initialization methods for the K-means algorithm, Pattern Recognit. Lett. 20 (1999) 1027–1040.
[24] M.E. Celebi, H.A. Kingravi, P.A. Vela, A comparative study of efficient initialization methods for the k-means clustering algorithm, Expert Syst. Appl. 40 (2013) 200–210.
[25] D.S. Modha, W.S. Spangler, Feature weighting in k-means clustering, Mach. Learn. 52 (2003) 217–237.
[26] L. Parsons, E. Haque, H. Liu, Subspace clustering for high dimensional data: a review, SIGKDD Explor. Newsl. 6 (2004) 90–105.
[27] J.Z. Huang, M.K. Ng, H. Rong, Z. Li, Automated variable weighting in k-means type clustering, IEEE Trans. Pattern Anal. Mach. Intell. 27 (2005) 657–668.
[28] E.Y. Chan, W.K. Ching, M.K. Ng, J.Z. Huang, An optimization algorithm for clustering using weighted dissimilarity measures, Pattern Recognit. 37 (2004) 943–952.
[29] A. Likas, N. Vlassis, J.J. Verbeek, The global k-means clustering algorithm, Pattern Recognit. 36 (2003) 451–461.
[30] A.M. Bagirov, Modified global k-means algorithm for minimum sum-of-squares clustering problems, Pattern Recognit. 41 (2008) 3192–3199.
[31] A.M. Bagirov, J. Ugon, D. Webb, Fast modified global k-means algorithm for incremental cluster construction, Pattern Recognit. 44 (2011) 866–876.

[32] K.-L. Wu, M.-S. Yang, Alternative c-means clustering algorithms, Pattern Recognit. 35 (2002) 2267–2278.
[33] X.-b. Zhi, J.-l. Fan, F. Zhao, Robust local feature weighting hard c-means clustering algorithm, Neurocomputing 134 (2014) 20–29.
[34] W.S. DeSarbo, J.D. Carroll, L.A. Clark, P.E. Green, Synthesized clustering: A method for amalgamating alternative clustering bases with differential weighting of variables, Psychometrika 49 (1984) 57–78.
[35] P.E. Green, J. Kim, F.J. Carmone, A preliminary study of optimal variable weighting in k-means clustering, J. Classification 7 (1990) 271–285.
[36] X. Wang, Y. Wang, L. Wang, Improving fuzzy c-means clustering based on feature-weight learning, Pattern Recognit. Lett. 25 (2004) 1123–1132.
[37] C.-Y. Tsai, C.-C. Chiu, Developing a feature weight self-adjustment mechanism for a K-means clustering algorithm, Comput. Statist. Data Anal. 52 (2008) 4658–4672.
[38] W.-L. Hung, M.-S. Yang, D.-H. Chen, Bootstrapping approach to feature-weight selection in fuzzy c-means algorithms with an application in color image segmentation, Pattern Recognit. Lett. 29 (2008) 1317–1325.
[39] H. Xing, X. Wang, M. Ha, A comparative experimental study of feature-weight learning approaches, in: 2011 IEEE International Conference on Systems, Man, and Cybernetics, 2011, pp. 3500-3505.
[40] H.-J. Xing, M.-H. Ha, Further improvements in feature-weighted Fuzzy C-means, Inform. Sci. 267 (2014) 1–15.
[41] H. Shen, J. Yang, S. Wang, X. Liu, Attribute weighted mercer kernel based fuzzy clustering algorithm for general non-spherical datasets, Soft Comput. 10 (2006) 1061–1073.
[42] L. Jing, M.K. Ng, J.Z. Huang, An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data, IEEE Trans. Knowl. Data Eng. 19 (2007).
[43] M.R.P. Ferreira, F. d. A. T. de Carvalho, Kernel fuzzy c-means with automatic variable weighting, Fuzzy Sets and Systems 237 (2014) 1–46.
[44] Z. Zhou, S. Zhu, Kernel-based multiobjective clustering algorithm with automatic attribute weighting, Soft Comput. 22 (2018) 3685–3709.
[45] B.A. Pimentel, R.M.C.R. de Souza, Multivariate Fuzzy C-means algorithms with weighting, Neurocomputing 174 (2016) 946–965.
[46] Z. Zhou, S. Zhu, Kernel-based multiobjective clustering algorithm with automatic attribute weighting, Soft Comput. (2017).
[47] D. Arthur, S. Vassilvitskii, k-means++: The advantages of careful seeding, in: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, 2007, pp. 1027-1035.
[48] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, S. Vassilvitskii, Scalable k-means++, Proc. VLDB Endow. 5 (2012) 622–633.
[49] A. Banerjee, J. Ghosh, Frequency-sensitive competitive learning for scalable balanced clustering on high-dimensional hyperspheres, IEEE Trans. Neural Netw. 15 (2004) 702–719.
[50] C.-D. Wang, J.-H. Lai, J.-Y. Zhu, A conscience on-line learning approach for kernel-based clustering, in: Data Mining (ICDM), 2010 IEEE 10th International Conference on, 2010, pp. 531-540.
[51] R. Duwairi, M. Abu-Rahmeh, A novel approach for initializing the spherical K-means clustering algorithm, Simul. Model. Pract. Theory 54 (2015) 49–63.
[52] P.S. Bradley, U.M. Fayyad, Refining initial points for K-Means clustering, in: ICML, 1998, pp. 91-99.
[53] S. Ting, J. Dy, A deterministic method for initializing K-means clustering, in: 16th IEEE International Conference on Tools with Artificial Intelligence, 2004, pp. 784-786.
[54] G. Tzortzis, A. Likas, The minmax k-means clustering algorithm, Pattern Recognit. 47 (2014) 2505–2516.
[55] G. Tzortzis, A. Likas, The global kernel k-means clustering algorithm, in: Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, 2008, pp. 1977-1984.
[56] G.F. Tzortzis, A.C. Likas, The global kernel $k $-means algorithm for clustering in feature space, IEEE Trans. Neural Netw. 20 (2009) 1181–1194.
[57] Y. Li, M. Dong, J. Hua, Localized feature selection for clustering, Pattern Recognit. Lett. 29 (2008) 10–18.
[58] http://archive.ics.uci.edu/ml/index.php.
[59] J.C. Bezdek, A convergence theorem for the fuzzy ISODATA clustering algorithms, IEEE Trans. Pattern Anal. Mach. Intell. (1980) 1–8.
[60] A. Strehl, J. Ghosh, Cluster ensembles—a knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. 3 (2002) 583–617.
[61] F.J. Valverde-Albacete, C. Peláez-Moreno, 100% classification accuracy considered harmful: The normalized information transfer factor explains the accuracy paradox, PLoS One 9 (2014) e84217.
[62] M. Filippone, F. Camastra, F. Masulli, S. Rovetta, A survey of kernel and spectral methods for clustering, Pattern Recognit. 41 (2008) 176–190.