This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Extensions of Kmeans-Type Algorithms: A New Clustering Framework by Integrating Intracluster Compactness and Intercluster Separation

Xiaohui Huang, Yunming Ye, and Haijun Zhang

***Abstract*— Kmeans-type clustering aims at partitioning a data set into clusters such that the objects in a cluster are compact and the objects in different clusters are well separated. However, most kmeans-type clustering algorithms rely on only intracluster compactness while overlooking intercluster separation. In this paper, a series of new clustering algorithms by extending the existing kmeans-type algorithms is proposed by integrating both intracluster compactness and intercluster separation. First, a set of new objective functions for clustering is developed. Based on these objective functions, the corresponding updating rules for the algorithms are then derived analytically. The properties and performances of these algorithms are investigated on several synthetic and real-life data sets. Experimental studies demonstrate that our proposed algorithms outperform the state-of-the-art kmeans-type clustering algorithms with respect to four metrics: accuracy, RandIndex, Fscore, and normal mutual information.**

***Index Terms*— Clustering, data mining, feature weighting, kmeans.**

I. Introduction

CLUSTERING is a basic operation in many applications in nature [1], such as gene analysis [2], image processing [3], text organization [4], and community detection [5], to name just a few. It is a method of partitioning a data set into clusters such that the objects in the same cluster are similar and the objects in different clusters are dissimilar according to certain predefined criteria [6].

There are many types of approaches [7] to solve a clustering problem: partitioning methods, hierarchical methods, density-based methods, grid-based methods, model-based methods, and so on. The kmeans-type clustering algorithms are a kind of partitioning method, which has been widely used in many real-world applications. Most of the existing kmeans-type clustering algorithms consider only the similarities among the objects in a cluster by minimizing the dispersions of the cluster. The representative ones of these algorithms include basic kmeans [8], [9], projected clustering [10], locally adaptive clustering [11], [12], automated variable weighting kmeans (Wkmeans) [6], attributes-weighting clustering algorithms (AWA) [13], entropy weighting kmeans (EWkmean) [4], feature group Wkmeans [14], and two-level variable Wkmeans [15]. Amidst them, basic kmeans, the simplest one, addresses all the selected features equally during the process of minimizing the dispersions. However, different features have different discriminative capabilities in real applications. For example, in the sentence London is the first city to have hosted the modern Games of three Olympiads, the keywords London and Olympiads have more discriminative information than the words city and modern in sport news. In literature, a large body of feature selection and weighting methods has been proposed in clustering analysis [4], [6], [10]–[15]. All these methods have the same characteristic: the features must be evaluated with the big weights if the dispersions of the features in a data set are small. In essence, the discriminative capability of a feature not only relates to the dispersion, but also associates with the distances between the centroids, i.e., intercluster separation. As a matter of fact, intercluster separation plays an important role in supervised learning methods (e.g., linear discriminative analysis) [16], [17]. Most of the existing kmeans-type algorithms, however, overlook the intercluster separation.

In this paper, we investigate the potential framework of the kmeans-type algorithms by integrating both intracluster compactness and intercluster separation. We propose three algorithms: E-kmeans, E-Wkmeans, and E-AWA, which extend basic kmeans, Wkmeans [6], and AWA [13], respectively. In addition, the convergence theorems of our proposed algorithms are given. Extensive experiments on both synthetic and real-life data sets corroborate the effectiveness of our proposed methods. The desirable features of our algorithms can be concluded as follows.

1) The generality of the extending algorithms encompasses a unified framework that considers both the intracluster compactness and the intercluster separation. Concretely, we develop a new framework for kmeans-type algorithms to include the impacts of the intracluster compactness and the intercluster separation in the clustering process.
2) The proposed framework is robust because it does not introduce new parameters to balance the intracluster

Manuscript received November 7, 2012; revised August 22, 2013 and December 1, 2013; accepted December 1, 2013. This work was supported in part by the National Natural Science Foundation of China under Grant 61073195, Grant 61272538, and Grant 61300209, in part by the National Key Technology Research and Development Program of MOST China under Grant 2012BAK17B08, in part by the Shenzhen Strategic Emerging Industries Program under Grant JCYJ20130329142551746, and in part by the Shenzhen Foundation Research Fund under Grant JCY20120613115205826.

The authors are with the Shenzhen Key Laboratory of Internet Information Collaboration, Shenzhen Graduate School, Harbin Institute of Technology, Harbin 150001, China (e-mail: hxh016@gmail.com; yeyunming@hit.edu.cn; aarhzhang@gmail.com).

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

Digital Object Identifier 10.1109/TNNLS.2013.2293795

compactness and the intercluster separation. The proposed algorithms are also more robust for the input parameter which is used to tune the weights of the features in comparison to the original kmeans-type algorithms.

3) The extending algorithms are able to produce better clustering results in comparison to the state-of-the-art algorithms in most of cases since they can utilize more information than traditional kmeans-type algorithms such that our approaches have capability to deliver discriminative powers of different features.

The main contributions of this paper are twofold:

1) we propose a new framework of kmeans-type algorithm by combining the dispersions of the clusters which reflect the compactness of the intracluster and the distances between the centroids of the clusters indicating the separation between clusters;
2) we give the complete proof of convergence of the extending algorithms based on the new framework.

The remaining sections of this paper are organized as follows: a brief overview of related works on various kmeans-type algorithms is presented in Section II. Section III introduces the extensions of the kmeans-type algorithms. Experiments on both synthetic and real data sets are presented in Section IV. We discuss the features of our algorithms in Section V and conclude this paper in Section VI.

II. Related Work

In this section, we give a brief survey of kmeans-type clustering from three aspects: 1) no Wkmeans-type algorithms; 2) vector Wkmeans-type algorithms; and 3) matrix Wkmeans-type algorithms. For detailed surveys of kmeans family algorithms, readers may refer to [18] and [19].

A. No Wkmeans-Type Algorithm

Kmeans-type clustering algorithms aim at finding a partition such that the sum of the squared distances between the empirical means of the clusters and the objects in the clusters is minimized.

1) No Wkmeans-Type Algorithm Without Intercluster Separation: Let $X = \{X_1, X_2, \dots, X_n\}$ be a set of n objects. Object $X_i = \{x_{i1}, x_{i2}, \dots, x_{im}\}$ is characterized by a set of m features (dimensions). The membership matrix U is a $n \times k$ binary matrix, where $u_{ip} = 1$ indicates that object i is allocated to cluster p, otherwise, it is not allocated to cluster p. $Z = \{Z_1, Z_2, \dots, Z_k\}$ is a set of k vectors representing the centroids of k clusters. The basic kmeans relies on minimizing an objective function [6]

$$P(U, Z) = \sum_{p=1}^{k} \sum_{i=1}^{n} \sum_{j=1}^{m} u_{ip} (x_{ij} - z_{pj})^2 \tag{1}$$

subject to

$$u_{ip} \in \{0, 1\}. \tag{2}$$

U and Z can be solved by optimizing the objective function. The detailed process of optimization can be referred to [9].

Basic kmeans algorithm has been extended in many ways. Steinbach *et al.* proposed a hierarchical divisive version of kmeans, called bisecting kmeans [20], which recursively partitions objects into two clusters at each step until the number of clusters is k. Bradley *et al.* [21] presented a fast scalable and singlepass version of kmeans that does not require all the data to be feed in the memory at the same time. Since the kmeans-type algorithms are sensitive to the choice of initial centroids and usually get stuck at local optima, many methods [22], [23] are proposed to overcome this problem. Another problem of kmeans algorithms is to require tuning of parameter K. X-means [24] automatically finds K by optimizing a criterion such as Akaike information criterion (AIC) or Bayesian information criterion (BIC).

2) No Wkmeans-Type Algorithm With Intercluster Separation: To obtain the best k (the number of clusters), some validity indexes [25] which integrate both intracluster compactness and intercluster separation are used in the clustering process. Yang *et al.* [26] and Wu *et al.* [27] proposed a fuzzy compactness and separation (FCS) algorithms which calculates the distances between the centroids of the cluster and the global centroids as the intercluster separation. The promising results are obtained since FCS is more robust to noises and outliers than traditional fuzzy kmeans clustering.

B. Vector Wkmeans-Type Algorithm

A major problem of no Wkmeans-type algorithms lies in treating all features equally in the clustering process. In practice, an interesting clustering structure usually occurs in a subspace defined by a subset of all the features. Therefore, many studies attempt to weight features with various methods [6], [28]–[31].

1) Vector Wkmeans-Type Algorithm Without Intercluster Separation: Automated variable Wkmeans [6] is a typical vector weighting clustering algorithm, which can be formulated as

$$P(U, W, Z) = \sum_{p=1}^{k} \sum_{i=1}^{n} u_{ip} \sum_{j=1}^{m} w_j^{\beta} (x_{ij} - z_{pj})^2 \tag{3}$$

subject to

$$u_{ip} \in \{0, 1\}, \sum_{p=1}^{k} u_{ip} = 1, \sum_{j=1}^{m} w_j = 1, 0 \le w_j \le 1 \tag{4}$$

where W is a weighting vector for the features. The details of the solution for U, Z, and W are given in [6].

De Sarbo *et al.* first introduced a feature selection method, SYNCLUS [28], which partitions features into several groups and uses weights for feature groups in the clustering process. The algorithm needs a large amount of computational cost [6]. It may not be applicable for large data sets. Inspired by the idea of two-level weighting strategy [28], Chen *et al.* proposed a two-level variable Wkmeans [15] based on Wkmeans [6].

2) Vector Wkmeans-Type Algorithm With Intercluster Separation: De Soete [29], [30] proposed an approach to optimize feature weights for ultrametric and additive tree fitting. This approach calculates the distances between all pairs

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

of objects and finds the optimal weight for each feature. However, this approach requires high-computational cost [6] since the hierarchical clustering method used to solve the feature selection problem in this approach needs high-computational cost. To decrease the computational cost, Makarenkov and Legendre [31] extended De Soete's approach to optimize feature weighting method for kmeans clustering. Usually, these algorithms are able to gain promising results to the data sets involving error-perturbed features or outliers.

C. Matrix Wkmeans-Type Algorithm

Matrix Wkmeans-type algorithms seek to group objects into clusters in different subsets of features for different clusters. It includes two tasks: 1) identification of the subsets of features where clusters can be found and 2) discovery of the clusters from different subsets of features.

1) Matrix Wkmeans-Type Algorithm Without Intercluster Separation: Aggarwal *et al.* [10] proposed the PROjected CLUStering (PROCLUS) algorithm which is able to find a subset of features for each cluster. Using PROCLUS, a user, however, needs to specify the average number of cluster features. Different to PROCLUS, feature weighting has been studied extensively in recent years [3], [4], [13], [14], [32], [33]. Therein, AWA [13] is a typical matrix weighting clustering algorithm, which can be formulated as

$$P(U, W, Z) = \sum_{p=1}^{k} \sum_{i=1}^{n} u_{ip} \sum_{j=1}^{m} w_{pj}^{\beta} (x_{ij} - z_{pj})^2 \quad (5)$$

subject to

$$u_{ip} \in \{0, 1\}, \sum_{p=1}^{k} u_{ip} = 1, \sum_{j=1}^{m} w_{pj} = 1, 0 \leq w_{pj} \leq 1 \quad (6)$$

where W is a weighting matrix, each row in which denotes a weight vector of the features in a cluster. The process of minimizing the objective function to solve U, Z, and W can be referred to [13].

Based on AWA [13], Jing *et al.* proposed an EWkmeans [4] which minimizes the intracluster compactness and maximizes the negative weight entropy to stimulate more features contributing to the identification of a cluster. In a later study, Chen *et al.* [14] proposed a two-level matrix Wkmeans algorithm and Ahmad and Dey [34] developed a matrix kmeans-type clustering algorithm of mixed numerical and categorical data sets based on EWkmeans [4]. Domeniconi *et al.* [11], [35] and Al-Razgan and Domeniconi [12] discovered clusters in subspaces spanned by different combinations of features via local weights of features. However, Jing *et al.* [4] pointed out that the objective functions in their methods are not differentiable while minimizing the objective functions.

2) Matrix Wkmeans-Type Algorithm With Intercluster Separation: Friedman and Meulman [36] published the clustering objects on subsets of features algorithm for matrix weighting clustering which involves the calculation of the distances between all pairs of objects at each iterative step. This results in a high-computational complexity $O(tn^2m)$ where n, m, and t is the number of objects, features, and iterations, respectively. Combining the FCS method [27] and EWKmeans [4], Deng *et al.* proposed an enhanced soft subspace clustering (ESSC) [3] algorithm that is able to use both intracluster compactness and intercluster separation. ESSC is able to effectively reduce the effect of the features on which the centroids of the clusters are close to the global centroid. However, negative values may be produced in the membership matrix if the balancing parameter is large. In addition, ESSC has three manual input parameters. In practice, it is difficult to find a group of appropriate values for the parameters.

D. Characteristics of Our Extending Kmeans-Type Algorithms

The main feature of our proposed framework lies in the fusion of the information of intercluster separation in a clustering process. At present, most traditional kmeans-type algorithms (e.g., basic kmeans, Wkmeans [6], and AWA [13]) only utilize the intracluster compactness. On the contrary, our proposed framework synthesizes both the intracluster compactness and the intercluster separation.

On the other hand, some existing algorithms have also introduced the intercluster separation into their models as we mentioned in Sections II-A.2, II-B.2, and II-C.2. The schemes of using the intercluster separation in these algorithms can be summarized into two classes: 1) calculating the distances between all pairs of objects which belong to different clusters [29]–[31], [36] and 2) calculating the distance between the centroid of each cluster and the global centroid [3], [26], [27]. Both schemes are able to help the algorithms improve the clustering results. However, the objective functions involved in the algorithms using the way of class 1 are not differentiable [4]. A subtraction framework embedded in the objective functions, which are differentiable in class 2, is usually used to integrate the intercluster separation by the existing algorithms [3], [26], [27]. However, a new parameter is usually required in the subtraction framework to balance the intracluster compactness and the intercluster separation. In practice, it is difficult to seek an appropriate value for this parameter. In our extending algorithms, to guarantee that the objective function in our proposed framework is differentiable, we calculate the distances between the centroids of the clusters with the global centroid as the intercluster separation. Consequently, we propose a division framework to integrate both the intracluster compactness and the intercluster separation.

III. EXTENDING MODEL OF KMEANS-TYPE ALGORITHM

A. Motivation

From the analysis of the related works, most of existing clustering methods consider only intracluster compactness. Take the AWA [13] as an example, the weights of features are updated according to the dispersions of the cluster. It means, in the same cluster, the features of small dispersions must be evaluated with big weights and the features of big dispersions must be evaluated with small weights. However, this does not work well in certain circumstances. For example, we have three clusters: C1 (London Olympic game), C2 (London riots), and C3 (Beijing Olympic game), as shown in Table I, each row

TABLE I

EXAMPLE OF THREE CLUSTERS USED TO ILLUSTRATE OUR MOTIVATION

Cluster	DocID	Document
C1	1	London, the capital of England, held the Olympic game in 2012.
	2	2012 London olympic game is the third olympic game held in England.
C2	5	2011 London riots in England is called “Blackberry riots”.
	4	Blackberry riots is the largest riots in London, England in recent years.
C3	5	Beijing, the capital of China, held the Olympic game in 2008.
	6	2008 Beijing olympic game is the first olympic game held in China.

TABLE II

TERM FREQUENCIES OF THE EXAMPLE IN TABLE I

Cluster	Beijing	China	London	England	olympic	game	Blackberry	riots
C1	0	0	1	1	1	1	0	0
	0	0	1	1	2	2	0	0
C2	0	0	1	1	0	0	1	2
	0	0	1	1	0	0	1	2
C3	1	1	0	0	1	1	0	0
	1	1	0	0	2	2	0	0

in which represents a document. The distribution of keyword frequencies is shown in Table II. From the table we can see that the dispersions of features are similar in all clusters. The traditional Wkmeans will evaluate the similar weight to each feature. But we can observe, comparing C1 with C2, the features: Olympic, game, Blackberry, and riots, have more discriminative capabilities. Comparing C1 with C3, the features: London, England, Beijing, and China have more discriminative capabilities. Thus, it may be ineffective to evaluate the weights of the features using only the dispersions of a data set. Under this condition, the intercluster separation can play an important role in distinguishing the importance of different features. In this paper, we focus on the extending kmeans-type algorithms by integrating both the intracluster compactness and the intercluster separation. Intuitively, it is ideal to compare all the pairs of objects or centroids to utilize the intercluster separation. In comparison with previous Wkmeans methods (e.g., Wkmeans and AWA), we may have a new objective function with the subtraction structure

$$P(U, W, Z) = \sum_{p=1}^{k} \sum_{i=1}^{n} u_{ip} \sum_{\substack{q=1 \\ q \neq p}}^{k} \sum_{i'=1}^{n} u_{i'q} \sum_{j=1}^{m} w_{pqj}^{\beta} D_{ii'pqj} \tag{7}$$

subject to

$$\sum_{j=1}^{m} w_{pqj} = 1 \tag{8}$$

$$D_{ii'pqj} = (x_{ij} - z_{pj})^2 + (x_{i'j} - z_{qj})^2 - \eta(x_{ij} - x_{i'j})^2. \tag{9}$$

This function aims to compare all pairs of objects in different clusters. In this objective function, each cluster has $k - 1$ weighting vectors, which represent the discriminative capabilities of the features while comparing to the other $k - 1$ clusters. However, it is not solvable for the membership matrix U in this objective function. Instead of comparing to each pair of objects, we compare each pair of centroids to maximize the distances of different clusters in the objective function with the subtraction structure

$$P(U, W, Z) = \sum_{p=1}^{k} \sum_{\substack{q=1 \\ q \neq p}}^{k} \sum_{j=1}^{m} w_{pqj}^{\beta} \sum_{i=1}^{n} u_{ip} D_{pqj} \tag{10}$$

$$D_{pqj} = (x_{ij} - z_{pj})^2 - \eta(z_{pj} - z_{qj})^2. \tag{11}$$

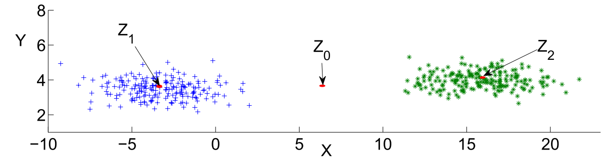


Fig. 1. Illustrative example of the effect of intercluster separation (Z_0 is the global centroid and Z_1, Z_2 are the centroids of cluster 1, cluster 2, respectively).

The membership matrix U can be solved in theory. However, it is difficult to seek an appropriate value for balancing parameter η. Negative values in weight are often produced if η is large. Different to functions (7) and (10), we may develop another objective function

$$P(U, W, Z) = \sum_{p=1}^{k} \sum_{\substack{q=1 \\ q \neq p}}^{k} \sum_{j=1}^{m} w_{pqj}^{\beta} \sum_{i=1}^{n} u_{ip} \frac{(x_{ij} - z_{pj})^2}{(z_{pj} - z_{qj})^2}. \tag{12}$$

This function employs the division structure similar to LDA [16], [17]. However, it is technically difficult to derive the centroid matrix Z in (12).

To make the centroid Z and membership matrix U solvable in the process of optimizing the objective functions, we can use the distances between the centroids of the clusters and the global centroid to approximate the distances of all pairs of centroids in the objective function, as shown in (12). We believe that this approximation is reasonable, because making the centroid of a cluster away from the global centroid is approximately equivalent to make the cluster away from the other clusters in most of cases. For example, in Fig. 1, making cluster 1 away from the global centroid Z_0 is equal to make cluster 1 away from cluster 2 to some extent. Thus, it may attain the purpose of maximizing the distances of the clusters. It is worth noting that our extending algorithms try to maximize the distances between the centroids of the clusters and the global centroid while keeping to minimize the intracluster compactness which also has impact on the weight assignment. Thus, the errors produced by the approximation may be reduced. In addition, many existing algorithms [3], [26], [27] maximize the intercluster separation by comparing the centroids of the clusters and the global centroid. The distances between the centroids of the clusters and the global centroid are also widely used as the intercluster separation in classification algorithms [16], [17], [37]. Under this framework, in the following sections, we demonstrate the detailed derivative process of three extending algorithms: 1) E-kmeans; 2) E-Wkmeans; and 3) E-AWA.

B. Extension of Basic Kmeans (E-Kmeans)

Basic kmeans is a typical clustering algorithm which has been widely used in various data analysis. However, it considers only the distances between centroids and objects, i.e., intracluster compactness. To utilize intercluster separation, we introduce the global centroid of a data set. Different to the basic kmeans, our proposed algorithm, E-kmeans, is expected to minimize the distances between objects and the centroid of the cluster that the objects belong to, while maximizing the distances between centroids of clusters and the global centroid.

To integrate intracluster compactness and intercluster separation, we modify the objective function, as shown in (1), into

$$P(U,Z)=\sum_{p=1}^{k}\sum_{i=1}^{n}u_{ip}\sum_{j=1}^{m}\frac{(x_{ij}-z_{pj})^2}{(z_{pj}-z_{0j})^2} \tag{13}$$

subject to

$$u_{ip}\in\{0,1\},\ \sum_{p=1}^{k}u_{ip}=1. \tag{14}$$

z_{0j} is the jth feature of the global centroid z_0 of a data set. We calculate z_{0j} as

$$z_{0j}=\frac{\sum_{i=1}^{n}x_{ij}}{n}. \tag{15}$$

We can minimize (13) by iteratively solving the following two problems.

1) Problem P1: fix $Z=\hat{Z}$, and solve the reduced problem $P(U,\hat{Z})$.
2) Problem P2: fix $U=\hat{U}$, and solve the reduced problem $P(\hat{U},Z)$.

The problem P1 is solved by

$$u_{ip}=\begin{cases}1, & \text{if } \sum_{j=1}^{m}\frac{(x_{ij}-z_{pj})^2}{(z_{pj}-z_{0j})^2}\le\sum_{j=1}^{m}\frac{(x_{ij}-z_{p'j})^2}{(z_{p'j}-z_{0j})^2}\\ 0, & \text{otherwise}\end{cases} \tag{16}$$

where $1\le p'\le k$, $p'\ne p$. If $(z_{pj}-z_{0j})^2=0$, we remove the jth feature at this iteration while calculating the membership U and the value of the objective function $P(U,Z)$. The proof procedure of minimizing objective function, as shown in (13), to solve U is given in [9]. The solution to problem P2 is given by the following theorem.

Theorem 3.1: Let $U=\hat{U}$ be fixed, P2 is minimized iff

$$z_{pj}=\begin{cases}z_{0j}, & \text{if } \sum_{i=1}^{n}u_{ip}(x_{ij}-z_{0j})=0\\ \frac{\sum_{i=1}^{n}u_{ip}(x_{ij}-z_{0j})x_{ij}}{\sum_{i=1}^{n}u_{ip}(x_{ij}-z_{0j})}, & \text{otherwise.}\end{cases} \tag{17}$$

Proof: For minimizing objective function (13), we derive the gradient of $z_{p,j}$ as

$$\frac{\partial P(\hat{U},Z)}{\partial z_{pj}}=\sum_{i=1}^{n}u_{ip}\frac{z_{pj}(x_{ij}-z_{0j})-x_{ij}(x_{ij}-z_{0j})}{(z_{pj}-z_{0j})^3}. \tag{18}$$

If $z_{pj}-z_{0j}\ne 0$, we set (18) to zero, we have

$$z_{pj}=\frac{\sum_{i=1}^{n}u_{ip}(x_{ij}-z_{0j})x_{ij}}{\sum_{i=1}^{n}u_{ip}(x_{ij}-z_{0j})}. \tag{19}$$

If $\sum_{i=1}^{n}u_{ip}(x_{ij}-z_{0j})=0$, we set $z_{pj}=z_{0j}$. The overall procedure of E-kmeans can be described as Algorithm 1. It is noted that, since the objective function shown in (13) is strictly decreasing, when we optimize U and Z, Algorithm 1 can guarantee that the objective function converges to local minimum.

Algorithm 1 E-kmeans

Input: $X=\{X_1,X_2,\ldots,X_n\}$, k.
Output: U, Z.
Initialize: Randomly choose an initial $Z^0=Z_1,Z_2,\ldots,Z_k$.
repeat
  Fixed Z, solve the membership matrix U with (16);
  Fixed U, solve the centroids Z with (17);
until Convergence.

C. Extension of Wkmeans (E-Wkmeans)

As mentioned before, basic kmeans and E-kmeans treat all the features equally. However, features may have different discriminative powers in real-world applications. Motivated by this, Huang *et al.* proposed the Wkmeans [6] algorithm which evaluates the importance of the features according to the dispersions of a data set. In this section, we propose the E-Wkmeans algorithm, which is able to consider the dispersions of a data set and the distances between the centroids of the clusters and the global centroid simultaneously while updating the feature weights.

Let $W=\{w_1,w_2,\ldots,w_m\}$ be the weights for m features and β be a parameter for tuning weight w_j, we extend (3) into

$$P(U,W,Z)=\sum_{p=1}^{k}\sum_{i=1}^{n}u_{ip}\left[\sum_{j=1}^{m}w_j^{\beta}\frac{(x_{ij}-z_{pj})^2}{(z_{pj}-z_{0j})^2}\right] \tag{20}$$

subject to

$$u_{ip}\in\{0,1\},\sum_{p=1}^{k}u_{ip}=1,\sum_{j=1}^{m}w_j=1,0\le w_j\le 1. \tag{21}$$

Similar to solve (13), we can minimize (20) by iteratively solving the following three problems.

1) Problem P1: fix $Z=\hat{Z}$ and $W=\hat{W}$, and solve the reduced problem $P(U,\hat{Z},\hat{W})$.
2) Problem P2: fix $U=\hat{U}$ and $W=\hat{W}$, and solve the reduced problem $P(\hat{U},Z,\hat{W})$.
3) Problem P3: fix $U=\hat{U}$ and $Z=\hat{Z}$, and solve the reduced problem $P(\hat{U},\hat{Z},W)$.

Problem P1 is solved by

$$u_{ip}=\begin{cases}1, & \text{if } \sum_{j=1}^{m}w_j^{\beta}\frac{(x_{ij}-z_{pj})^2}{(z_{pj}-z_{0j})^2}\le\sum_{j=1}^{m}w_j^{\beta}\frac{(x_{ij}-z_{p'j})^2}{(z_{p'j}-z_{0j})^2}\\ 0, & \text{otherwise}\end{cases} \tag{22}$$

where $1\le p'\le k, p'\ne p$. Problem P2 is solved by (17) and the solution to problem P3 is given in the following Theorem 3.2.

Theorem 3.2: Let $U=\hat{U}$ and $Z=\hat{Z}$ be fixed, $P(\hat{U},\hat{Z},W)$ is minimized iff

$$w_j=\begin{cases}0, D_j=0 \text{ or } z_{pj}=z_{0j}\\ \frac{1}{\sum_{t=1}^{m}\left(\frac{D_j}{D_t}\right)^{1/(\beta-1)}}, & \text{otherwise}\end{cases} \tag{23}$$

Algorithm 2 E-Wkmeans

Input: $X = \{X_1, X_2, \ldots, X_n\}, k$.
Output: U, Z, W.
Initialize: Randomly choose an initial $Z^0 = Z_1, Z_2, \ldots, Z_k$ and weight $W = \{w_1, w_2, \ldots, w_m\}$.
repeat
  Fixed Z, W, solve the membership matrix U with (22);
  Fixed U, W, solve the centroids Z with (17);
  Fixed U, Z, solve the weight W with (23);
until Convergence.

where

$$D_j = \sum_{p=1}^{k}\sum_{i=1}^{n} u_{ip}\frac{(x_{ij}-z_{pj})^2}{(z_{pj}-z_{0j})^2}. \tag{24}$$

Proof: We consider the relaxed minimization of $P(\hat{U}, \hat{Z}, W)$ via a Lagrange multiplier obtained by ignoring the constraint $\sum_{j=1}^{m} w_j = 1$. Let α be the multiplier and $\Psi(W, \alpha)$ be the Lagrangian

$$\Psi(W,\alpha) = \sum_{j=1}^{m} w_j^{\beta} D_j - \alpha\left(\sum_{j=1}^{m} w_j - 1\right). \tag{25}$$

By setting the gradient of the function (25) with respects to w_j and α to zero, we obtain the equations

$$\frac{\partial \Psi(W,\alpha)}{\partial w_j} = \beta w_j^{\beta-1} D_j - \alpha = 0 \tag{26}$$

$$\frac{\partial \Psi(W,\alpha)}{\partial \alpha} = \sum_{j=1}^{m} w_j - 1 = 0. \tag{27}$$

From (26), we obtain

$$w_j = \left(\frac{\alpha}{\beta D_j}\right)^{1/(\beta-1)}. \tag{28}$$

Substituting (28) into (27), we have

$$\alpha^{1/(\beta-1)} = 1 \Bigg/ \left[\sum_{j=1}^{m}\left(\frac{1}{\beta D_j}\right)^{1/(\beta-1)}\right]. \tag{29}$$

Substituting (29) into (28), we have

$$w_j = \frac{1}{\sum_{t=1}^{m}\left(\frac{D_j}{D_t}\right)^{1/(\beta-1)}}. \tag{30}$$

The overall procedure of E-Wkmeans can be described as Algorithm 2. Given a data partition, the goal of feature weight aims to assign a larger weight to a feature that has a smaller intracluster compactness and larger intercluster separation. The parameter β are used to control the distribution of the weight W.

When $\beta = 0$, E-Wkmeans is equivalent to E-kmeans, because $w_j^{\beta} = 1$ regardless of the value of w_j.

When $\beta = 1$, w_j is equal to 1 for the smallest D_j shown in (24), and the weights of other feature is 0. That means, it chooses only one feature for clustering. It is unreasonable for the high-dimensional data clustering.

When $0 < \beta < 1$, the objective function (20) cannot converge to the minimization.

When $\beta < 0$, the larger the value of D_j is, the larger value of w_j we can get. This is not applicable to feature selection. The aim of feature selection is to evaluate bigger weights to smaller D_j, i.e., we should evaluate bigger weights to the features that have small intracluster compactness and large intercluster separation. $\beta < 0$ cannot satisfy the demand of feature selection.

When $\beta > 1$, the larger the value of D_j is, the smaller value of w_j we can get. This is able to satisfy all the demand of the algorithms and the objective function shown in (20) is strictly decreasing when we optimize U, Z, and W, it is able to converge to local minimum.

D. Extension of AWA (E-AWA)

In Wkmeans and E-Wkmeans, the same feature in different clusters has the same weight. The same feature in different clusters, however, has different weights in most real-world applications. In this subsection, we focus on developing an algorithm to solve this problem under the condition of utilizing both intracluster compactness and intercluster separation.

Let $W = \{W_1, W_2, \ldots, W_k\}$ be a weight matrix for k clusters. $W_p = \{w_{p1}, w_{p2}, \ldots, w_{pm}\}$ denotes the feature weights in cluster p, we extend (5) into

$$P(U,W,Z) = \sum_{p=1}^{k}\sum_{i=1}^{n} u_{ip}\left[\sum_{j=1}^{m} w_{pj}^{\beta}\frac{(x_{ij}-z_{pj})^2}{(z_{pj}-z_{0j})^2}\right] \tag{31}$$

subject to

$$u_{ip} \in \{0,1\}, \sum_{p=1}^{k} u_{ip} = 1, \sum_{j=1}^{m} w_{pj} = 1, 0 \le w_{pj} \le 1. \tag{32}$$

Similar to solve (13), we can minimize (20) by iteratively solving the following three problems.

1) Problem P1: fix $Z = \hat{Z}$ and $W = \hat{W}$, and solve the reduced problem $P(U, \hat{Z}, \hat{W})$.
2) Problem P2: fix $U = \hat{U}$ and $W = \hat{W}$, and solve the reduced problem $P(\hat{U}, Z, \hat{W})$.
3) Problem P3: fix $U = \hat{U}$ and $Z = \hat{Z}$, and solve the reduced problem $P(\hat{U}, \hat{Z}, W)$.

Problem P1 is solved by

$$u_{ip} = \begin{cases} 1, \text{ if } \sum_{j=1}^{m} w_{pj}^{\beta}\frac{(x_{ij}-z_{pj})^2}{(z_{pj}-z_{0j})^2} \le \sum_{j=1}^{m} w_{p'j}^{\beta}\frac{(x_{ij}-z_{p'j})^2}{(z_{p'j}-z_{0j})^2} \\ 0, \text{ otherwise} \end{cases} \tag{33}$$

where $1 \le p' \le k$, $p' \ne p$. Problem P2 is solved by (17) and the problem P3 is solved by

$$w_{pj} = \begin{cases} 0, \text{ if } (z_{pj}-z_{0j})^2 = 0 \\ \frac{1}{m_i}, \text{ if } D_{pj} = 0 \text{ and } z_{pj} \ne z_{0j} \\ \quad m_i = |\{t : D_{pt} = 0 \text{ and } (z_{pt}-z_{0t})^2 \ne 0\}| \\ 0, \text{ if } D_{pj} \ne 0, \text{ but } D_{pt} = 0, \text{ for some } t \\ \quad \frac{1}{\sum_{t=1}^{m}\left(\frac{D_{pj}}{D_{pt}}\right)^{1/(\beta-1)}} \ \forall 1 \le t \le m, \text{ otherwise} \end{cases} \tag{34}$$

Algorithm 3 E-AWA

Input: $X = \{X_1, X_2, \ldots, X_n\}, k$.
Output: U, Z, W.
Initialize: Randomly choose an initial $Z^0 = Z_1, Z_2, \ldots, Z_k$ and weight $W = \{w_{pj}\}$.
repeat
  Fixed Z, W, solve the membership matrix U with (33);
  Fixed U, W solve the centroids Z with (17);
  Fixed U, Z solve the weight W with (34);
until Convergence.

where

$$D_{pj} = \sum_{i=1}^{n} u_{ip} \frac{(x_{ij} - z_{pj})^2}{(z_{pj} - z_{0j})^2}. \tag{35}$$

The convergence of proof process is similar to that of E-Wkmeans. The procedure of E-AWA can be described as Algorithm 3. Similar to the E-Wkmeans, the input parameter β is used to control the distribution of the weight W and β should be evaluated the value greater than 1. Since objective function is strictly decreasing in each step when optimizing U, Z, and W, Algorithm 3 can assure that the objective function converges to local minimum.

E. Relationship Amidst Algorithms

E-kmeans, E-Wkmeans, and E-AWA are the extensions of basic kmeans, Wkmeans, and AWA, respectively. Basic kmeans, Wkmeans, and AWA employ only the intracluster compactness while updating the membership matrix and weights. However, the extending algorithms take the intercluster separation into account.

From another perspective, E-kmeans does not weight the features, i.e., all the features are treated equally. E-Wkmeans weights the features with a vector, which means, each feature has a weight representing the importance of the feature in the entire data set. E-AWA weights the features with a matrix, i.e., each cluster has a weighting vector representing the subspace of the cluster. When $\beta = 0$, E-Wkmeans and Wkmeans degenerate to E-kmeans and basic kmeans, respectively. Since $\beta = 0$, $w_j^\beta = 1$ regardless of the value of w_j. Thus, the features are treated equally while updating the membership matrix U. Likewise, when $\beta = 0$, E-AWA and AWA degenerate to the basic kmeans and E-kmeans. When $\beta = 0$, E-Wkmeans and E-AWA have the same clustering result, whilst Wkmean and AWA have the same clustering result. When the weights of the same feature in different clusters are equal, E-AWA and AWA are equivalent to E-Wkmeans and Wkmeans, respectively. However, this case rarely happens in real-world data sets.

F. Computational Complexity

Similar to the basic kmeans, Wkmean, and AWA, the extending algorithms are also iterative algorithms. The computational complexity of basic kmeans is $O(tknm)$, where t is the iterative times; k, n, and m are the number of the clusters, objects and features, respectively. E-kmeans as well as basic kmeans has two computational steps: 1) updating the membership matrix and 2) updating the centroids. The complexities of updating centroids and updating membership matrix of E-kmeans are $O(knm + nm)$ and $O(knm + km)$, respectively. Therefore, the complexity of the overall E-kmeans algorithm is $O(tknm)$.

In comparison to the E-kmeans, the E-Wkmeans and E-AWA have another step: updating the weights. The complexity of updating the weights of E-Wkmeans and E-AWA is $O(knm+km)$. Therefore, the overall computational complexities of E-Wkmean and E-AWA are also $O(tknm)$. In summary, compared with the original algorithms, our extending algorithms need extra $O(km)$ computational time to calculate the distances between the centroids of the clusters and the global centroid while updating member matrix and weights, and we need extra $O(nm)$ to calculate distances between the objects and the global centroid to update the centroids of the clusters. However, it does not change the computational complexities of the algorithms in overall. Basic kmeans, E-kmeans, Wkmeans, E-kmeans, AWA, and E-AWA have the same computational complexity $O(tknm)$.

IV. Experiments

A. Experimental Setup

In experiments, the performances of proposed approaches are extensively evaluated on two synthetic data sets and nine real-life data sets. The benchmark clustering algorithms—basic kmeans (kmeans), BiSecting kmeans (BSkmeans) [20], automated variable Wkmeans [6], AWA [13], EWkmeans [4] as well as ESSC [3] are chosen for the performance comparison with the proposed algorithms. Among these algorithms, kmeans, E-kmeans, and BSkmeans have no input parameter; Wkmeans, AWA, E-Wkmeans, and E-AWA have a parameter β to tune the weights of the features. In our experiments, we choose $\beta = 8$ according to the empirical study of parameter β in Sections IV-B.1 and IV-C.1. EWkmeans has parameter γ, which controls the strength of the incentive for clustering on more features. In the experiments, we set $\gamma = 5$ according to [4]. ESSC has three parameters λ, γ, and η, where λ is the fuzzy index of fuzzy membership, γ and η are used to control the influences of entropy and balance the weights between intracluster compactness and intercluster separation, respectively. We have chosen the empirical values $\lambda = 1.2$, $\gamma = 5$, and $\eta = 0.1$ according to [3]. Since ESSC [3] is a fuzzy kmeans algorithms and each object corresponds to a membership vector which indicates the degree that the object belongs to the corresponding clusters, we assign the object to the cluster corresponding to the maximal value in the membership vector for simplification to compare the performance. In the experiments, we implement all the algorithms with MATLAB and run the algorithms in a workstation with Intel(R)Xeon(R) 2.4 Hz CPU, 8 GB RAM.

In this paper, four evaluation metrics including accuracy (Acc), RandIndex (RI), Fscore, and normal mutual information (NMI) are used to evaluate the results of the algorithms. Acc represents the percentage of the objects that are correctly recovered in a clustering result and RI [3], [6] considers the percentage of the pairs of objects which cluster correctly. The computational processes of Acc and RI can be referred in [6]. Fscore [4] is able to leverage the information of precision and recall. NMI [3], [4] is a popular measure of clustering quality, which is more reliable to measure the imbalanced data sets (i.e., most of objects are from one cluster and only a few objects belong to other clusters) in comparison to the other three metrics. The computational processes of Fscore and NMI can be referred to [4].

It is well known that the kmeans-type clustering process produces local optimal solution. The final result depends on the locations of initial cluster centroids. In Wkmeans clustering algorithms, the initial weights also affect the final clustering result. To compare the performance between the extending algorithms and the existing algorithms, the same set of centroids which randomly generated are used to initialize the different algorithms and all the weighting algorithms are initialized with the same weights. Finally, we calculate the average Acc, RI, Fscore, and NMI produced by the algorithms after running 100 times.

B. Synthetic Data Set

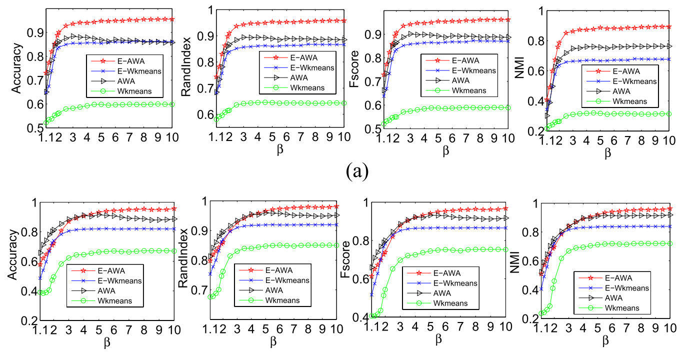
In this subsection, two synthetic data sets are constructed to investigate the performances of the proposed algorithms. To validate the effect of intercluster separation, different features on the two data sets are designed to different discriminative capabilities on an intercluster perspective. The centroids and standard deviations of synthetic data sets are given in Table III. Each cluster contains eight features and 100 objects. The synthetic data sets are generated by three steps.

1) Generating the centroid for each cluster. For making different features have different discriminative capabilities from an intercluster separation perspective, we generate three 8-D vectors as the centroids of synthetic data set 1 (Synthetic 1) where the first three features are well separated and the other features are very close to each other when comparing cluster 1 with cluster 2. However, comparing cluster 1 with cluster 3, the second three features are well separated and the other features are close to each other. The last two features are noisy features as they are very close to each other for all the clusters. The generating procedure of the centroids of the synthetic data set 2 (Synthetic 2) is similar to that of Synthetic 1. Synthetic 2 has five clusters. Based on the centroids of Synthetic 1, we generate another two 6-D vectors as the first six features of the centroids of the last two clusters. The values of the first three, the fourth, the fifth, and the sixth features of the centroids of cluster 4 are similar to the values of the corresponding features of centroids of cluster 2, 3, 3, and 2, respectively. The values of the first two, the second two, the fifth, and the sixth features of the centroids of cluster 5 are similar to the values of the corresponding features of centroids of cluster 3, 2, 3, and 1, respectively. Then, we generate five 2-D vectors as the centroids of the last two features of the five clusters.

TABLE III

CENTROIDS AND THE STANDARD DEVIATIONS OF SYNTHETIC DATA SETS

Data Set	Cluster centroid	Standard deviation
Synthetic1	0.503 0.325 0.728 0.103 0.814 0.613 0.510 0.893	0.225 0.115 0.020 0.132 0.294 0.257 0.064 0.715
	0.238 0.841 0.367 0.099 0.805 0.621 0.499 0.897	0.146 0.180 0.923 0.863 0.977 0.751 0.767 0.642
	0.498 0.335 0.701 0.556 0.312 0.972 0.507 0.901	0.598 0.655 0.653 0.121 0.140 0.428 0.671 0.419
Synthetic2	0.503 0.325 0.728 0.103 0.814 0.913 0.110 0.098	0.048 0.174 0.050 0.223 0.193 0.044 0.013 0.256
	0.238 0.841 0.367 0.099 0.805 0.021 0.099 0.097	0.018 0.070 0.254 0.222 0.202 0.004 0.285 0.221
	0.498 0.335 0.701 0.556 0.312 0.672 0.107 0.101	0.180 0.251 0.109 0.060 0.086 0.069 0.079 0.263
	0.240 0.838 0.365 0.560 0.310 0.074 0.109 0.103	0.246 0.046 0.137 0.004 0.288 0.192 0.238 0.211
	0.501 0.322 0.363 0.097 0.307 0.930 0.105 0.105	0.234 0.248 0.105 0.217 0.094 0.224 0.207 0.009



(a)

(b)

Fig. 2. Effects with various β on synthetic data sets. (a) Synthetic 1. (b) Synthetic 2.

2) Generating the standard deviations for each cluster. For the Synthetic 1, we generate randomly an 8-D vector of which the values are between 0 and 1 as the standard deviations for each cluster. For the Synthetic 2, we generate randomly an 8-D vector of which the values are between 0 and 0.3 as the standard deviations for each cluster. Because, we want to observe the effect of our proposed algorithms applying to the data sets which have different separability. Normally, the data sets with small standard deviations are separated easier than the data sets with large the standard deviations under the condition of similar centroids.

3) Generating the objects of the data sets. We generate 100 points for each cluster using normal distribution with the centroids derived by the first step and the standard deviations derived the second step.

1) Parametric Study: Parameter β, which is used in the algorithms: Wkmeans, E-Wkmeans, AWA, and E-AWA, is an important factor to tune the weights of the features. In this subsection, we give the empirical study of β for its effects on the results, as shown in Fig. 2. The clustering results are shown from $\beta = 1.1$ until the results do not change or begin to reduce by increasing the value of β. We have used the increment of the value of 0.2 for β in this scope in our experiments because the performance is sensitive to the change of the values of β in the range of $(1, 2]$. From the value of $\beta = 2$, we have used the increment of the value of 0.5 for β since the performances are more stable in this range.

Fig. 2 shows the changing trend of the average results produced by the compared algorithms after running 100 times on the two synthetic data sets. From the results, we can observe that the performances of the extending algorithms, E-AWA and E-Wkmeans, are consistently better than AWA and Wkmeans, respectively, across all the evaluation metrics.

TABLE IV

RESULTS ON SYNTHETIC DATA SETS (THE STANDARD DEVIATION IN BRACKET)

Data Set	Algorithm	Acc	RI	Fscore	NMI
Synthetic1	BSkmeans	0.6384(±0.08)	0.6651(±0.04)	0.6436(±0.06)	0.3064(±0.06)
	EWkmeans	0.7731(±0.15)	0.8096(±0.12)	0.8036(±0.13)	0.6563(±0.21)
	ESSC	0.9441(±0.11)	0.9457(±0.09)	0.9457(±0.10)	0.8784(±0.16)
	kmeans	0.6103(±0.06)	0.6361(±0.04)	0.6048(±0.05)	0.2955(±0.04)
	E-kmeans	0.8663(±0.13)	0.8642(±0.08)	0.8771(±0.10)	0.6643(±0.14)
	Wkmeans	0.5975(±0.04)	0.6421(±0.04)	0.5884(±0.03)	0.3123(±0.06)
	E-Wkmeans	0.8625(±0.14)	0.8671(±0.09)	0.8714(±0.12)	0.6782(±0.17)
	AWA	0.8600(±0.17)	0.8862(±0.10)	0.8879(±0.12)	0.7617(±0.13)
	E-AWA	**0.9550(±0.10)**	**0.9572(±0.07)**	**0.9613(±0.07)**	**0.8908(±0.12)**
Synthetic2	BSkmeans	0.9271(±0.02)	0.9466(±0.01)	0.9271(±0.02)	0.8452(±0.02)
	EWkmeans	0.6954(±0.16)	0.8567(±0.08)	0.7760(±0.11)	0.7713(±0.10)
	ESSC	0.7070(±0.09)	0.8685(±0.03)	0.7695(±0.06)	0.7792(±0.06)
	kmeans	0.7186(±0.16)	0.8726(±0.07)	0.7923(±0.11)	0.7536(±0.08)
	E-kmeans	0.8377(±0.15)	0.9304(±0.05)	0.8766(±0.11)	0.8446(±0.09)
	Wkmeans	0.6688(±0.17)	0.8496(±0.08)	0.7516(±0.12)	0.7186(±0.11)
	E-Wkmeans	0.8225(±0.16)	0.9200(±0.06)	0.8678(±0.11)	0.8398(±0.10)
	AWA	0.8858(±0.14)	0.9500(±0.06)	0.9159(±0.10)	0.9137(±0.09)
	E-AWA	**0.9565(±0.09)**	**0.9805(±0.03)**	**0.9659(±0.06)**	**0.9570(±0.05)**

[1] Note: The results in the table are produced by using β=8.

AWA and E-AWA perform better than Wkmeans and E-Wkmeans. E-AWA performs best in all the algorithms on both data sets. We can also observe that the performances tend to be constant when β is greater than three. Since the relatively good results can be obtained when $\beta = 8$, in the later study, we choose $\beta = 8$ in the experiments.

2) Results and Analysis: The average Acc, RI, Fscore, NMI, and the standard deviations produced by the compared algorithms after running 100 times are summarized in Table IV for the two synthetic data sets by using $\beta = 8$. In view of the overall experiment, the best performance is delivered by E-AWA with respects to all the evaluation criteria. In addition, E-Wkmeans and E-kmeans perform better than Wkmeans and basic kmeans, respectively.

For Synthetic 1, we observe from Table IV that the performances of the E-AWA, E-Wkmeans, and E-kmeans are better than that of AWA, Wkmeans, and kmeans, respectively. In comparison to kmeans, Wkmeans, and AWA and E-kmeans, E-Wkmeans, and E-AWA are able to deliver over 25%, 26%, and 9% Acc improvement, respectively, on Synthetic 1. In the three pairs of algorithms (basic kmeans and E-kmeans, Wkmeans and E-Wkmeans, and AWA and E-AWA), AWA and E-AWA perform the best. We believe that this is caused by the same features in different clusters playing different roles in the data set. For example, feature 1 in cluster 2 plays more important than that in cluster 1 and cluster 3, because the dispersion of feature 1 in cluster 2 is small and the centroid of feature 1 in cluster 2 is far from the global centroid. ESSC is the second best algorithm for this data set. This result indicates that ESSC is able to utilize effectively the intercluster separation as our extending algorithms do.

In comparison to Synthetic 1 and Synthetic 2 is a more complex data set. We can observe the results of the Synthetic 2 in the Table IV that the performances of the E-AWA, E-Wkmeans, and E-kmeans also perform better than AWA, Wkmeans, and kmeans, respectively. Compared with kmeans, Wkmeans, and AWA and E-kmeans, E-Wkmeans, and E-AWA produces 11%, 15%, and 7% Acc improvement, respectively. Likewise, the AWA and E-AWA perform the best in three pairs of algorithms, even better than the result of Synthetic 1. The result of Synthetic 2 implies that AWA and E-AWA are more

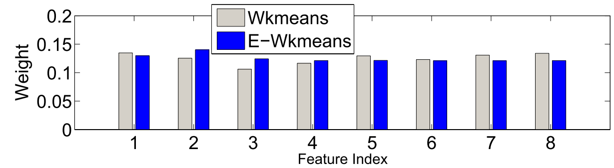


Fig. 3. Comparison of the feature weights on Synthetic 1 ($\beta = 8$).

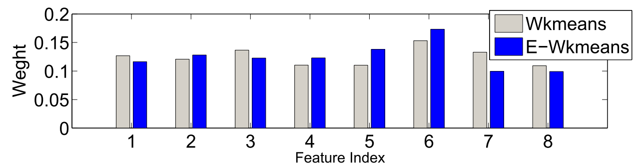


Fig. 4. Comparison of the feature weights on Synthetic 2 ($\beta = 8$).

applicable to the complex data set. The performance of ESSC in Synthetic 2 is not as promising as that in Synthetic 1. We believe that ESSC may perform worse for a complex data set due to the presence of a number of parameters which are difficult to select for a better performance. We can see the Table IV, the standard deviations of the results produced by the extending algorithms are similar to that produced by the original algorithms. In overall, the results produced by the algorithms of no input parameter have smaller standard deviations than that produced by the algorithms with one or more parameters.

In summary, our proposed algorithms, E-AWA, E-Wkmeans, and E-kmeans perform better than the original kmeans-type approaches, AWA, Wkmeans, and kmeans, respectively, on the synthetic data sets. We believe that this performance gain is contributed to consider the intercluster separation in the clustering process.

3) Feature Selection: In this subsection, we study the effect of feature weight with different kmeans-type algorithms. In a clustering process, both Wkmeans and E-Wkmeans produce a weight for each feature, which represents the contribution of this feature in the entire data set. Figs. 3 and 4 show the comparison of feature weights of Wkmeans and E-Wkmeans on two synthetic data sets. From the Table III, we can observe that the centroids of feature 7 and feature 8 in all the clusters are very close to each other. Therefore, we can consider feature 7 and feature 8 as noisy features from the viewpoint of intercluster separation. From Figs. 3 and 4, we observe that E-Wkmeans can reduce the weight of noisy features (i.e., feature 7 and feature 8) and increase the weights of features which are relatively far away from each other as opposed to Wkmeans.

On the other hand, both E-AWA and AWA produce a weight for each feature in each cluster, which represents the contribution of a feature in a cluster. From Figs. 5 and 6, we observe that E-AWA can also reduce the weight of noisy features (i.e., feature 7 and feature 8) and increase the weights of the features which are far from the centroid of other clusters. For example, feature 3 of cluster 1 in Synthetic 1, this feature has small dispersion and is far away from the centroid of other clusters. E-AWA is able to increase the weight of this feature in a significant rate comparing to AWA. And for

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

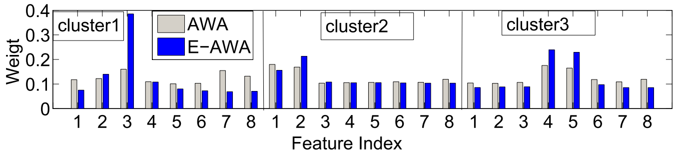


Fig. 5. Comparison of the feature weights on Synthetic 1 ($\beta = 8$).

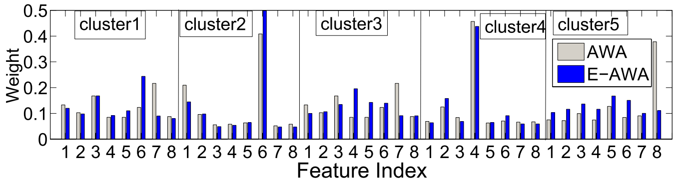


Fig. 6. Comparison of the feature weights on Synthetic 2 ($\beta = 8$).

TABLE V

AVERAGE ITERATIONS AND THE RUNNING TIME ON SYNTHETIC DATA SETS

Data Set	E-AWA	AWA	E-Wkmeans	Wkmeans	E-kmeans	kmeans
Synthetic1	9.47(6.18)	9.30(5.20)	11.2(5.73)	13.1(5.32)	11.4(3.83)	14.6(3.83)
Synthetic2	10.5(14.6)	10.4(13.1)	9.82(11.4)	12.3(11.5)	9.56(7.81)	10.9(6.85)

[1] Note: The results in the table are produced by using β=8.
[2] Note: The values in brackets are the running time in seconds produced by running the algorithms with the same centroids in 100 runs.

feature 8 of cluster 5 in Synthetic 2, the AWA evaluates the big weight due to the small dispersion. As a matter of fact, this feature has small discriminative capability because the mean of this feature in all the clusters are similar. E-AWA reduces the weight of this feature in cluster 5 comparing to AWA. From the analysis of the synthetic data sets, we can see that the more appropriate weights of the features can be obtained by integrating the information of the intercluster separation. It is worth noting that the weights of features are influenced simultaneously by parameter β, the intracluster compactness and the intercluster separation. For Wkmeans, AWA, E-Wkmeans, and E-AWA, the variance of the weights of the different features will reduce with the increase of the values of β, i.e., when β is large, the weights of different features will tend to be similar.

4) Convergence Speed: In this paper, we import the intercluster separation into the objection functions of kmeans-type algorithms. Intuitively, the factor from intercluster separation may speed up the convergence process of the clustering. In this subsection, we investigate the convergence speed of the Wkmeans-type algorithms with respects to the iterations and the running time. The stopping criterion of the algorithms relies on the condition that the membership matrix U is no longer changing. Table V lists the average iterations and the running time of the algorithms initialized by the same centroids in 100 runs with $\beta = 8$. From this table, we can observe that the iterations of the extending algorithms are similar to those of the corresponding original algorithms. However, the extending algorithms spend slightly more running time in comparison to the original algorithms. This additional time cost comes from the calculation of the distances between the centroids of the clusters and the global centroid at every iteration.

TABLE VI

PROPERTIES OF REAL-LIFE DATA SETS

Data Set	No. of features	No. of clusters	No. of objects
Wine	13	3	178
WDBC	30	2	569
Vertebral2	6	2	310
Vertebral3	6	3	310
Robot	90	4	88
Cloud	10	2	2048
LandsatSatellite	33	6	6435
Glass	9	2	214
Parkinsons	22	2	195

C. Real-Life Data Set

To further investigate the performance of the extending algorithms in real-life data sets, we have evaluated our algorithms in nine data sets reported in Machine Learning Repository (http://archive.ics.uci.edu/ml/). The properties of these data sets are described in Table VI.

1) Parametric Study: We show the average Acc, RI, Fscore, and NMI produced by Wkmeans, E-Wkmeans, AWA, and E-AWA after running 100 times from $\beta = 1.1$ until the clustering results do not change or begin to reduce by increasing the value of β, as shown in Fig. 7. From this figure, we can observe that E-AWA and E-Wkmeans outperform AWA and Wkmeans, respectively, for most values of β across the data sets Vertebral 2, Vertebral 3, Robot, Cloud, LandsatSatellite, Glass, and Parkinson. The algorithms, AWA and E-AWA, perform better than Wkmeans and E-Wkmeans, respectively, on data sets Vertebral 2, Vertebral 3, Cloud, LandsatSatellite, and Parkinson. On Robot and Glass, E-Wkmeans outperform the other algorithms. We can observe that the results of most algorithms are unstable when the values of β is between 1 and 3 and trend to stability after the value of β is greater than 3. This observation is similar to the results of the synthetic data sets. Moreover, we can observe that the performances of our extending algorithms are more stable than the performances of the original algorithms with the change of the values of β in most of data sets, especially, when $1 < \beta \leq 3$. This indicates that the intracluster separation can help to improve clustering results no matter what value β is assigned when it is greater than 1 in most of cases.

However, we can see that Wkmeans performs better than E-Wkmeans on data set Wine. Likewise, AWA achieves better results than E-AWA on data set WDBC. We believe that performance degradation on Wine and WDBC may be caused by the resulting errors in the process of approximation when we use the distances between the centroids of the clusters and the global centroid to approximate the distances among all pairs of centroids. When the centroid of a cluster on all the features or some important features are very close to the corresponding features of the global centroid, but the centroids of the clusters on these features are not close to each other, the approximation may introduce errors. For example, we can observe from Table VII, the values of the centroids on the sixth, seventh, eighth, and 12th features in the second clusters are very close to the values of the corresponding features in the global centroid. Thus, the small weights will be assigned to these features. However, the distances among the centroids of the clusters on these features are not very close to each other and the dispersions of the data set on these features are

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

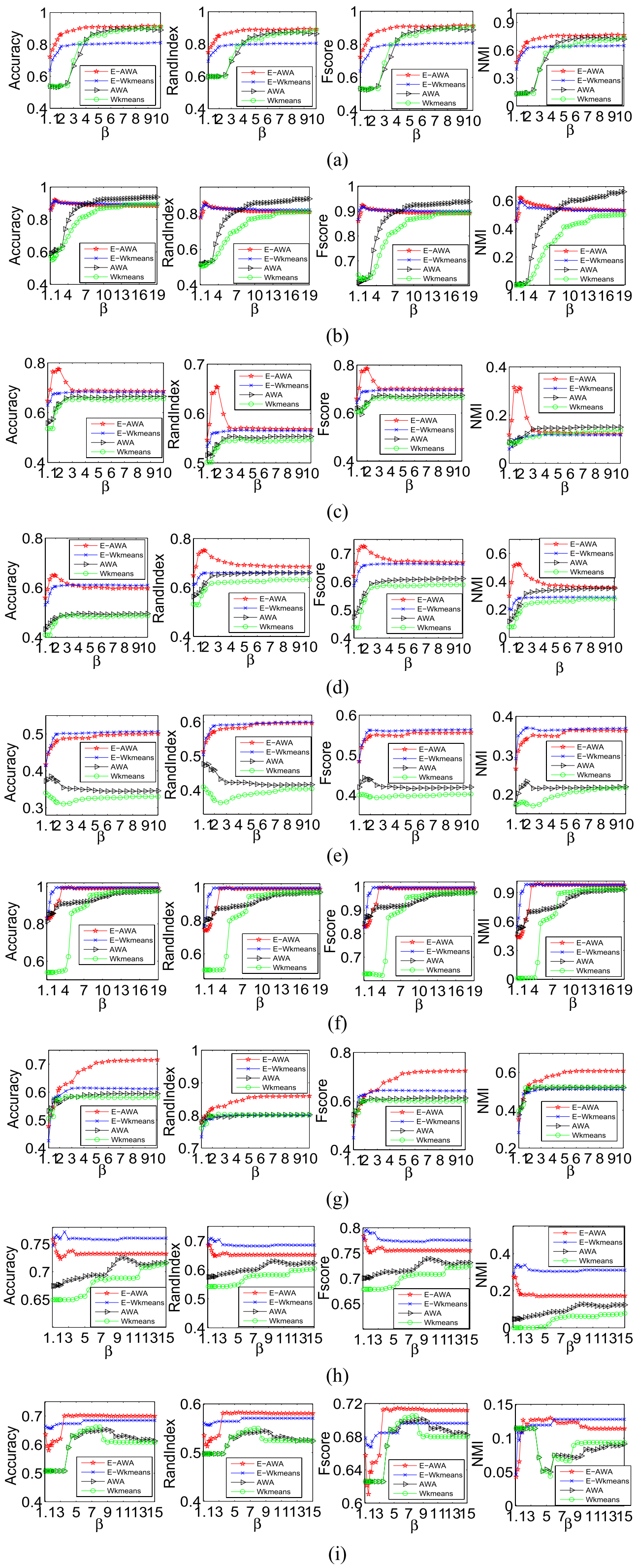


Fig. 7. Effects with various β on real data sets. (a) Wine. (b) WDBC. (c) Vertebral 2. (d) Vertebral 3. (e) Robot. (f) Cloud. (g) LandsatSatellite. (h) Glass. (i) Parkinson.

small, which indicates that these features should be assigned by big weights. Thus, it may produce the errors when assigning the objects into the second clusters. It is noteworthy that our extending algorithms have also included the intracluster compactness. It is able to minimize the distances between the centroid of a cluster and the objects that belong to the cluster such that the weight assignment can be accomplished in a principled manner. Thus, the problem caused by the approximation process may be relieved to some extent. From the experimental results, our proposed algorithms outperform the original algorithms in most of cases.

TABLE VII

CHARACTERISTICS OF DATA SET WINE

	Feature 1 to 13												
centroid of cluster1	13.7	2.010	2.455	17.0	106.	2.840	2.982	0.290	1.899	5.528	1.062	3.157	1115.
centroid of cluster2	12.2	1.932	2.244	20.2	94.5	**2.258**	**2.080**	**0.363**	1.630	3.086	1.056	**2.785**	519.0
centroid of cluster3	13.1	3.333	2.437	21.4	99.3	1.678	0.781	0.447	1.153	7.396	0.682	1.683	629.8
Global centroid	13.0	2.336	2.366	19.5	99.7	**2.295**	**2.029**	**0.361**	1.590	5.058	0.957	**2.611**	746.0
Global dispersion	0.26	0.896	0.061	7.60	169.	**0.179**	**0.247**	**0.011**	0.233	2.576	0.022	**0.149**	2.9E4

2) Results and Analysis: The average Acc, RI, Fscore, NMI, and standard deviations produced by the compared algorithms after running 100 times are summarized in Table VIII on nine real-life data sets by using $\beta = 8$ according to the study of Section IV-C.1. On the data sets Robot and Cloud, the extending algorithms: E-kmeans, E-Wkmeans, and E-AWA, outperform kmeans, Wkmeans, and AWA, respectively, across all the evaluation metrics. For example, comparing with AWA, E-AWA obtains 15% and 7% Acc improvement on Robot and Cloud, respectively. Compared with kmeans, Wkmeans, and AWA and E-kmeans, E-Wkmeans, and E-AWA achieve 6%, 24%, and 7% NMI improvement on data set Glass and 7%, 6%, and 5% NMI improvement on data set Parkinson, respectively. However, it is noticed that the best clustering performance as indicated by NMI, is not always consistent with that indicated by Acc, RI, and Fscore. This is caused by the imbalanced properties of Glass and Parkinson. Both of two data sets include two clusters. The numbers of objects in two clusters are 163 and 51, respectively, in Glass. And the numbers of objects in two clusters are 47 and 147, respectively, in Parkinson. Under an extreme condition, if all the objects are assigned to the same cluster, we can obtain high values of Acc, RI, and Fscore, but NMI is 0. Therefore, NMI is a more reliable metric for the imbalanced data sets. For data sets Vertebral 2 and Vertebral 3, our extending algorithms achieve comparable results with the original algorithms when $\beta = 8$. The best results on the two data sets are gained by using $1 < \beta < 3$ from the study of parameter β in Fig. 7 of Section IV-C.1. When $\beta = 1.9$, E-AWA is able to obtain 0.7739 and 0.6539 Acc on Vertebral 2 and Vertebral 3, respectively. These results are significantly better than the results produced by other algorithms. For data set LandsatSatellite, E-AWA achieves 12% Acc, 5% RI, 11% Fscore, and 9% NMI improvement compared with AWA. For data set Wine, E-AWA and E-Kmeans obtain 1% to 3% Acc, RI, Fscore, and NMI improvement compared with AWA and kmeans. Moreover, from Table VIII, we can see that the results produced by our extending algorithms have slightly bigger standard deviations than that produced by the original algorithms. This may suggest that the algorithms which consider twofold factors, i.e., intercluster compactness and intercluster separation, may have relatively lower stability than the algorithms which consider only one factor.

TABLE VIII

RESULTS ON REAL-LIFE DATA SETS (THE STANDARD DEVIATION IN BRACKET)

Metric	Algorithm	Wine	WDBC	Vertebral2	Vertebral3	Robot	Cloud	LandsatSatellite	Glass	Parkinson
Acc	BSkmeans	0.7182(±0.01)	0.8541(±0.01)	0.6698(±0.01)	**0.6253(±0.01)**	0.3737(±0.03)	0.7472(±0.01)	0.5564(±0.07)	**0.8057(±0.06)**	0.7274(±0.01)
	EWkmeans	0.4244(±0.03)	0.7882(±0.05)	0.6698(±0.01)	0.5673(±0.03)	0.3701(±0.02)	0.8381(±0.17)	0.5926(±0.03)	0.7628(±0.02)	**0.7528(±0.01)**
	ESSC	0.6460(±0.09)	0.8119(±0.04)	0.6426(±0.08)	0.5913(±0.09)	0.3485(±0.03)	0.7511(±0.14)	0.3919(±0.07)	0.7266(±0.12)	0.6817(±0.09)
	kmeans	0.6986(±0.02)	0.8541(±0.01)	0.6698(±0.01)	0.5672(±0.03)	0.3514(±0.03)	0.7472(±0.01)	0.5806(±0.02)	**0.8057(±0.06)**	0.7274(±0.01)
	E-kmeans	0.8668(±0.05)	0.8777(±0.06)	**0.6906(±0.01)**	0.6055(±0.08)	**0.5197(±0.07)**	0.9648(±0.05)	0.6296(±0.03)	0.7065(±0.10)	0.6990(±0.03)
	Wkmeans	0.8998(±0.01)	0.8405(±0.01)	0.6537(±0.01)	0.4859(±0.03)	0.3299(±0.01)	0.9529(±0.12)	0.5797(±0.02)	0.6870(±0.08)	0.6615(±0.01)
	E-Wkmeans	0.8069(±0.07)	0.8984(±0.06)	0.6824(±0.01)	0.6093(±0.07)	0.5059(±0.07)	**0.9945(±0.04)**	0.6121(±0.04)	0.7578(±0.16)	0.6846(±0.04)
	AWA	0.8969(±0.02)	**0.8951(±0.07)**	0.6651(±0.01)	0.4945(±0.03)	0.3449(±0.04)	0.9235(±0.13)	0.5946(±0.02)	0.7037(±0.11)	0.6491(±0.01)
	E-AWA	**0.9107(±0.04)**	0.8897(±0.08)	0.6867(±0.01)	0.5985(±0.07)	0.4994(±0.08)	0.9907(±0.06)	**0.7132(±0.06)**	0.7323(±0.15)	0.7016(±0.03)
RI	BSkmeans	0.7270(±0.01)	0.7504(±0.01)	0.5562(±0.01)	0.6139(±0.01)	0.5189(±0.06)	0.6220(±0.01)	0.8147(±0.02)	**0.6934(±0.08)**	0.6015(±0.01)
	EWkmeans	0.3694(±0.03)	0.6719(±0.05)	0.5562(±0.01)	0.6731(±0.01)	0.3884(±0.08)	0.7910(±0.17)	0.8040(±0.01)	0.6372(±0.02)	**0.6259(±0.01)**
	ESSC	0.6445(±0.09)	0.6974(±0.04)	0.5530(±0.04)	0.6524(±0.09)	0.4413(±0.07)	0.6701(±0.17)	0.5232(±0.14)	0.6298(±0.10)	0.5834(±0.07)
	kmeans	0.7178(±0.01)	0.7504(±0.01)	0.5562(±0.01)	0.6730(±0.01)	0.4492(±0.06)	0.6220(±0.01)	0.8007(±0.01)	**0.6934(±0.08)**	0.6015(±0.01)
	E-kmeans	0.8478(±0.03)	0.7930(±0.06)	**0.5714(±0.01)**	0.6654(±0.03)	**0.6135(±0.10)**	0.9375(±0.06)	0.8073(±0.01)	0.6032(±0.06)	0.5798(±0.03)
	Wkmeans	0.8732(±0.01)	0.7315(±0.01)	0.5459(±0.01)	0.6314(±0.01)	0.4035(±0.04)	0.9434(±0.15)	0.8009(±0.01)	0.5820(±0.08)	0.5499(±0.01)
	E-Wkmeans	0.8021(±0.04)	**0.8266(±0.07)**	0.5652(±0.01)	0.6605(±0.03)	0.5983(±0.10)	**0.9940(±0.05)**	0.7999(±0.01)	0.6822(±0.14)	0.5706(±0.03)
	AWA	0.8705(±0.02)	0.8243(±0.07)	0.5532(±0.01)	0.6599(±0.02)	0.4157(±0.07)	0.8958(±0.16)	0.8020(±0.01)	0.6091(±0.10)	0.5421(±0.01)
	E-AWA	**0.8893(±0.04)**	0.8171(±0.09)	0.5686(±0.01)	**0.6861(±0.04)**	0.5962(±0.13)	0.9901(±0.06)	**0.8586(±0.02)**	0.6519(±0.11)	0.5817(±0.03)
Fscore	BSkmeans	0.7238(±0.01)	0.8443(±0.01)	0.6731(±0.01)	**0.6751(±0.01)**	0.4709(±0.04)	0.7299(±0.01)	0.6002(±0.06)	**0.7893(±0.07)**	0.7371(±0.01)
	EWkmeans	0.5142(±0.01)	0.7972(±0.04)	0.6731(±0.01)	0.6561(±0.02)	0.4367(±0.03)	0.8612(±0.13)	0.6126(±0.03)	0.7524(±0.02)	**0.7449(±0.01)**
	ESSC	0.6573(±0.09)	0.8136(±0.04)	0.6704(±0.06)	0.6386(±0.08)	0.4092(±0.02)	0.7485(±0.14)	0.4490(±0.06)	0.7486(±0.08)	0.7021(±0.07)
	kmeans	0.7126(±0.01)	0.8443(±0.01)	0.6731(±0.01)	0.6559(±0.02)	0.4313(±0.04)	0.7299(±0.01)	0.6003(±0.02)	**0.7893(±0.07)**	0.7371(±0.01)
	E-kmeans	0.8653(±0.05)	0.8822(±0.05)	**0.7032(±0.01)**	0.6607(±0.05)	**0.5602(±0.06)**	0.9663(±0.04)	0.6541(±0.02)	0.7275(±0.09)	0.7105(±0.02)
	Wkmeans	0.8978(±0.01)	0.8379(±0.01)	0.6623(±0.01)	0.5896(±0.02)	0.4006(±0.01)	0.9549(±0.12)	0.5994(±0.02)	0.7086(±0.06)	0.7053(±0.01)
	E-Wkmeans	0.8044(±0.06)	**0.9025(±0.05)**	0.6952(±0.01)	0.6622(±0.04)	0.5627(±0.06)	**0.9962(±0.03)**	0.6421(±0.03)	0.7729(±0.14)	0.6961(±0.03)
	AWA	0.8936(±0.02)	0.8979(±0.06)	0.6734(±0.01)	0.6103(±0.02)	0.4173(±0.03)	0.9213(±0.14)	0.6130(±0.02)	0.7228(±0.09)	0.6996(±0.01)
	E-AWA	**0.9088(±0.05)**	0.8958(±0.06)	0.7001(±0.01)	0.6701(±0.05)	0.5553(±0.08)	0.9932(±0.04)	**0.7223(±0.05)**	0.7552(±0.10)	0.7135(±0.02)
NMI	BSkmeans	0.3972(±0.01)	0.4672(±0.01)	**0.2542(±0.01)**	0.3865(±0.01)	0.3370(±0.06)	0.3416(±0.01)	0.5173(±0.05)	0.1756(±0.17)	0.0698(±0.05)
	EWkmeans	0.0885(±0.08)	0.3383(±0.11)	0.2542(±0.01)	0.4184(±0.01)	0.1849(±0.11)	0.5092(±0.29)	0.5232(±0.02)	0.0253(±0.06)	0.0022(±0.01)
	ESSC	0.3715(±0.12)	0.3745(±0.08)	0.1506(±0.08)	0.3482(±0.10)	0.2091(±0.05)	0.3717(±0.33)	0.3256(±0.07)	0.1589(±0.12)	0.0433(±0.09)
	kmeans	0.4285(±0.01)	0.4672(±0.01)	**0.2542(±0.01)**	**0.4186(±0.01)**	0.2650(±0.06)	0.3416(±0.01)	0.5210(±0.01)	0.1756(±0.17)	0.0698(±0.05)
	E-kmeans	0.6995(±0.06)	0.4824(±0.11)	0.1286(±0.04)	0.2970(±0.02)	**0.3565(±0.10)**	0.8427(±0.13)	0.5231(±0.02)	0.2361(±0.14)	**0.1393(±0.09)**
	Wkmeans	0.7062(±0.01)	0.3440(±0.01)	0.1320(±0.01)	0.2734(±0.02)	0.2147(±0.04)	0.8913(±0.29)	0.5213(±0.02)	0.0635(±0.14)	0.0665(±0.01)
	E-Wkmeans	0.6459(±0.07)	0.5430(±0.13)	0.1177(±0.04)	0.2873(±0.02)	0.3682(±0.11)	**0.9846(±0.09)**	0.5102(±0.03)	**0.3051(±0.23)**	0.1279(±0.08)
	AWA	0.7326(±0.03)	0.5362(±0.09)	0.1496(±0.01)	0.3484(±0.04)	0.2169(±0.05)	0.7573(±0.31)	0.5190(±0.01)	0.1000(±0.13)	0.0729(±0.01)
	E-AWA	**0.7593(±0.06)**	**0.5440(±0.16)**	0.1222(±0.05)	0.3582(±0.08)	0.3639(±0.15)	0.9813(±0.13)	**0.6070(±0.04)**	0.1727(±0.10)	0.1239(±0.08)

[1] Note: The results in the table are produced by using β=8.

TABLE IX

FREQUENCIES THAT THE EXTENDING ALGORITHMS PRODUCE BETTER RESULTS THAN THE ORIGINAL ALGORITHMS INITIALIZED BY THE SAME CENTROIDS IN 100 RUNS

Metric	Accuracy							NMI	
Data set	A	B	C	D	E	F	G	H	I
Fre(E-kmeans, kmeans)	97	94	98	56	94	99	90	**42**	69
Fre(E-Wkmeans, Wkmeans)	**0**	94	100	94	100	**12**	74	94	64
Fre(E-AWA, AWA)	71	63	94	87	93	98	95	67	64

[1] Note: A, B, C, D, E, F, G, H and I represent the data sets: Wine, WDBC, Vertebral2, Vertebral3, Robot, Cloud, LandsatSatellite, Glass and Parkinson, respectively.

[2] Note: Fre(alg1, alg2) represents the number of running times that algorithm alg1 produces better results than algorithm alg2 initialized by the same 100 controids.

TABLE X

AVERAGE ITERATIONS AND THE RUNNING TIME ON REAL-LIFE DATA SETS

Data Set	E-AWA	AWA	E-Wkmeans	Wkmeans	E-kmeans	kmeans
Wine	9.8(2.91)	6.3(1.64)	8.1(2.97)	5.3(1.63)	7.7(1.78)	7.3(1.22)
WDBC	7.5(8.14)	8.4(7.34)	7.7(13.7)	11 (15.2)	8.0(5.02)	6.2(2.75)
Vertebral2	5.4(2.62)	6.5(2.91)	5.9(2.60)	7.7(2.66)	5.7(1.62)	9.1(1.85)
Vertebral3	10 (6.08)	10 (5.61)	8.2(4.31)	9.3(3.85)	9.2(3.21)	9.2(2.46)
Robot	4.9(9.63)	3.3(6.06)	5.5(4.63)	3.6(2.69)	4.9(1.88)	4.4(0.97)
Cloud	6.3(18.1)	6.8(16.7)	7.1(29.3)	12 (38.0)	8.1(14.1)	10 (14.1)
LandsatSatellite	28.8(499)	38.7(521)	16.6(381)	40.4(699)	17.7(245)	59 (605)
Glass	5.3(1.38)	4.8(1.03)	5.9(1.98)	5.5(1.51)	8.7(1.70)	5.3(0.78)
Parkinson	3.9(1.26)	7.7(1.81)	3.7(1.67)	11 (3.46)	2.9(0.74)	4.0(0.60)

[1] Note: The results in the table are produced by using β=8.

[2] Note: The values in brackets are the running time in seconds produced by running the algorithms with the same centroids in 100 runs.

Table IX shows the number of running times that the extending algorithms produce better results than the original algorithms initialized by the same centroids in 100 runs. Due to the imbalanced property of data sets Glass and Parkinson, we show the comparative results on metric NMI for the two data sets. We can see from Table IX that the extending algorithms can produce better results than the original algorithms in most of cases, if we initialize the algorithms with the same centroids. It is worth noting that Wkmeans produces more times of better results than E-Wkmeans on Cloud. However, from the Table VIII, the results produced by Wkmeans have larger standard deviations, i.e., partial results produced by Wkmeans are inferior. Moreover, the average results produced by E-Wkmeans are better than those produced by Wkmeans on Cloud. It is also noteworthy that Wkmeans significantly outperforms E-Wkmeans on data set Wine. That may be caused by the resulting errors when we use the distances between the centroids of the clusters and the global centroid to approximate the distances among all pairs of centroids. We give a detailed analysis in Section IV-C.1. In summary, E-AWA, E-Wkmeans, and E-kmeans are able to obtain performance improvement by maximizing the intercluster separation in contrast to the original algorithms in most of cases. Table IX suggests that our extending algorithms have high possibilities to obtain better results in comparison to the original algorithms with the same initial centroids.

3) Convergence Speed: In this subsection, we study the effect of the intercluster separation on the convergence speed of Wkmeans-type algorithms on real-life data sets with respects to the iterations and the running time. Table X lists the average iterations and the running time of the algorithms initialized by the same centroids in 100 runs with $\beta = 8$ on the real data sets. From this table, we can observe that the iteration of E-Wkmeans is slightly less than that of Wkmeans in most of the data sets. We can also see from the table that the extending algorithms spend slightly more running time as opposed to the original algorithms under the condition of similar or less iterations on some data sets. This is caused by the extra computational cost that the extending algorithms must spend to calculate the distances between the centroids of the clusters and the global centroid at every iteration. For LandsatSatellite, the iterations of E-AWA, E-Wkmeans, and E-kmeans reduce 26%, 60%, and 70% in comparison to those of AWA, Wkmeans, and kmeans, respectively. Correspondingly, the running time of E-AWA, E-Wkmeans, and E-kmeans reduces 4.2%, 45%, and 59% as opposed to those of AWA, Wkmeans, and kmeans on LandsatSatellite, respectively.

V. Discussion

From the results in Section IV, E-AWA, E-Wkmeans, and E-kmeans outperform AWA, Wkmeans, and kmeans, respectively, in terms of various evaluation measures: Acc, RI, Fscore, and NMI in most of cases. Therein, AWA performs better than kmeans and Wkmeans, and E-AWA performs better than E-kmeans and E-Wkmeans in most of data sets. That suggests the information of the intercluster separation can help to improve the clustering results by maximizing the distances among the clusters. E-AWA performs the best in all the compared algorithms. Due to the parameter problem, ESSC does not perform well in comparison to our proposed algorithms in most of cases.

According to the comparison of the weights of the features, the extending algorithms can reduce the weights of the features, the centroids of which are very close to each other, and increase the weight of the features, the centroids of which are far away from each other. Therefore, our extending algorithms can effectively improve the performance of feature weighting such that they perform well for clustering.

In contrast to ESSC, the extending algorithms utilize only one parameter β as used in Wkmeans and AWA. Thus, our algorithms are more applicable for complex data sets in practice. We can observe from the experiments that the performances of E-AWA and E-Wkmeans are more smooth than those of AWA and Wkmeans, respectively, with various values of β, especially, when $1 < \beta < 3$. Therefore, our proposed algorithms are more robust than the original algorithms in overall.

The extending algorithms have the same computational complexities compared with basic kmeans algorithms. Since clustering using kmeans-type algorithm is an iterative process, the computational time also depends on the total number of iterations. From the empirical study, we can observe that the total number of iterations of the extending algorithms is similar to or less than that of the original algorithms. However, the extending algorithms must spend extra time to calculate the distances between the centroids of the clusters and the global centroid at each iteration. The extending algorithms may spend slightly more time in comparison to the original algorithms on some data sets.

From the experiments and analysis, we can find that our proposed algorithms have a limitation: when the centroid of certain cluster is very close to the global centroid and the centroids among the clusters are not close to each other, maximizing the distances between the centroids of clusters in place of maximizing the distances among the clusters may produce errors and the performances of our extending algorithms may decrease to some extent. However, since our extending algorithms consider both the intracluster compactness and the intercluster separation, our extending algorithms are able to obtain better results in comparison to the original algorithms in most of cases. It is also worth pointing out that the current extending algorithms are difficult to apply for categorical data sets. This is caused by the possibility that the denominators of the objective functions become zeros, i.e., division-by-zero problem.

VI. Conclusion

In this paper, we have presented three extensions of kmeans-type algorithms by integrating both intracluster compactness and intercluster separation. This paper involves the following aspects: 1) three new objective functions are proposed based on basic kmeans, Wkmeans, and AWA; 2) the corresponding updating rules are derived and the convergence is proved in theory; and 3) extensive experiments are carried out to evaluate the performances of E-kmeans, E-Wkmeans, and E-AWA algorithms based on four evaluation metrics: Acc, RI, Fscore, and NMI. The results demonstrate that the extending algorithms are more effective than the existing algorithms. In particular, E-AWA delivers the best performance in comparison to other algorithms in most of cases.

In the future work, we plan to further extend our algorithms to categorical data sets by developing new objective functions to overcome the division-by-zero problem. It will be of great importance in applying our algorithms to more real data sets. We also plan to investigate the potential of our proposed algorithms for other applications, such as gene data clustering, image clustering, community discovery, and so on.

Acknowledgment

The authors would like to thank the associate editor and anonymous referees for their helpful comments.

References

[1] M. R. Anderberg, *Cluster Analysis for Applications*. New York, NY, USA: Academic, 1973.
[2] M. Sardana and R. Agrawal, "A comparative study of clustering methods for relevant gene selection in microarray data," in *Advances in Computer Science, Engineering & Applications*. New York, NY, USA: Springer-Verlag, 2012, pp. 789–797.
[3] Z. Deng, K. Choi, F. Chung, and S. Wang, "Enhanced soft subspace clustering integrating within-cluster and between-cluster information," *Pattern Recognit.*, vol. 43, no. 3, pp. 767–781, 2010.
[4] L. Jing, M. Ng, and J. Huang, "An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 8, pp. 1026–1041, Aug. 2007.
[5] L. Tang, H. Liu, and J. Zhang, "Identifying evolving groups in dynamic multi-mode networks," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 1, pp. 72–85, Jan. 2012.
[6] J. Huang, M. Ng, H. Rong, and Z. Li, "Automated variable weighting in k-means type clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 5, pp. 657–668, May 2005.
[7] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*. San Mateo, CA, USA: Morgan Kaufmann, 2011.
[8] J. Bezdek, "A convergence theorem for the fuzzy ISODATA clustering algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-2, no. 1, pp. 1–8, Jan. 1980.
[9] S. Selim and M. Ismail, "K-means-type algorithms: A generalized convergence theorem and characterization of local optimality," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-6, no. 1, pp. 81–87, Jan. 1984.

[10] C. Aggarwal, J. Wolf, P. Yu, C. Procopiuc, and J. Park, "Fast algorithms for projected clustering," *ACM SIGMOD Rec.*, vol. 28, no. 2, pp. 61–72, 1999.

[11] C. Domeniconi, D. Papadopoulos, D. Gunopulos, and S. Ma, "Subspace clustering of high dimensional data," in *Proc. SIAM Int. Conf. Data Mining*, 2004, pp. 517–521.

[12] M. Al-Razgan and C. Domeniconi, "Weighted clustering ensembles," in *Proc. SIAM Int. Conf. Data Mining*, 2006, pp. 258–269.

[13] E. Chan, W. Ching, M. Ng, and J. Huang, "An optimization algorithm for clustering using weighted dissimilarity measures," *Pattern Recognit.*, vol. 37, no. 5, pp. 943–952, 2004.

[14] X. Chen, Y. Ye, X. Xu, and J. Zhexue Huang, "A feature group weighting method for subspace clustering of high-dimensional data," *Pattern Recognit.*, vol. 45, no. 1, pp. 434–446, 2012.

[15] X. Chen, X. Xu, J. Huang, and Y. Ye, "TW-k-means: Automated two-level variable weighting clustering algorithm for multi-view data," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 4, pp. 932–944, Apr. 2013.

[16] R. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Human Genet.*, vol. 7, no. 2, pp. 179–188, 1936.

[17] D. Cai, X. He, and J. Han, "Semi-supervised discriminant analysis," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–7.

[18] A. Jain, "Data clustering: 50 years beyond k-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[19] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, May 2005.

[20] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," in *Proc. KDD Workshop Text Mining*, vol. 400. 2000, pp. 525–526.

[21] P. Bradley, U. Fayyad, and C. Reina, "Scaling clustering algorithms to large databases," in *Proc. 4th Int. Conf. Knowl. Discovery Data Mining*, 1998, pp. 9–15.

[22] D. Arthur and S. Vassilvitskii, "K-means^{++}: The advantages of careful seeding," in *Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2007, pp. 1027–1035.

[23] S. Shahnewaz, M. Rahman, and H. Mahmud, "A self acting initial seed selection algorithm for k-means clustering based on convex-hull," in *Informatics Engineering and Information Science*. New York, NY, USA: Springer-Verlag, 2011, pp. 641–650.

[24] D. Pelleg and A. Moore, "X-means: Extending k-means with efficient estimation of the number of clusters," in *Proc. 17th Int. Conf. Mach. Learn.*, San Francisco, CA, USA, 2000, pp. 727–734.

[25] S. Das, A. Abraham, and A. Konar, "Automatic clustering using an improved differential evolution algorithm," *IEEE Trans. Syst., Man Cybern., A, Syst. Humans*, vol. 38, no. 1, pp. 218–237, Jan. 2008.

[26] M. Yang, K. Wu, and J. Yu, "A novel fuzzy clustering algorithm," in *Proc. IEEE Int. Symp. Comput. Intell. Robot. Autom.*, vol. 2. 2003, pp. 647–652.

[27] K. Wu, J. Yu, and M. Yang, "A novel fuzzy clustering algorithm based on a fuzzy scatter matrix with optimality tests," *Pattern Recognit. Lett.*, vol. 26, no. 5, pp. 639–652, 2005.

[28] W. De Sarbo, J. Carroll, L. Clark, and P. Green, "Synthesized clustering: A method for amalgamating alternative clustering bases with differential weighting of variables," *Psychometrika*, vol. 49, no. 1, pp. 57–78, 1984.

[29] G. De Soete, "Optimal variable weighting for ultrametric and additive tree clustering," *Qual. Quantity*, vol. 20, no. 2, pp. 169–180, 1986.

[30] G. De Soete, "OVWTRE: A program for optimal variable weighting for ultrametric and additive tree fitting," *J. Classificat.*, vol. 5, no. 1, pp. 101–104, 1988.

[31] V. Makarenkov and P. Legendre, "Optimal variable weighting for ultrametric and additive trees and k-means partitioning: Methods and software," *J. Classificat.*, vol. 18, no. 2, pp. 245–271, 2001.

[32] L. Parsons, E. Haque, and H. Liu, "Subspace clustering for high dimensional data: A review," *ACM SIGKDD Explorat. Newslett.*, vol. 6, no. 1, pp. 90–105, 2004.

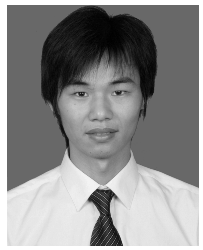
[33] H. Frigui and O. Nasraoui, "Simultaneous clustering and dynamic keyword weighting for text documents," *Survey of Text Mining*. New York, NY, USA: Springer-Verlag, 2004, pp. 45–70.

[34] A. Ahmad and L. Dey, "A k-means type clustering algorithm for subspace clustering of mixed numeric and categorical datasets," *Pattern Recognit. Lett.*, vol. 32, no. 7, pp. 1062–1069, 2011.

[35] C. Domeniconi, D. Gunopulos, S. Ma, B. Yan, M. Al-Razgan, and D. Papadopoulos, "Locally adaptive metrics for clustering high dimensional data," *Data Mining Knowl. Discovery*, vol. 14, no. 1, pp. 63–97, 2007.

[36] J. Friedman and J. Meulman, "Clustering objects on subsets of attributes," *J. R. Statist. Soc., Ser. B*, vol. 66, no. 4, pp. 815–849, 2004.

[37] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. New York, NY, USA: Wiley, 2012.

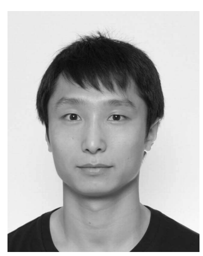


Xiaohui Huang received the B.Eng. and master's degrees from Jiangxi Normal University, Nanchang, China, in 2005 and 2008, respectively. He is currently pursuing the Ph.D. degree with the Shenzhen Graduate School, Harbin Institute of Technology, Harbin, China.

His current research interests include data mining, topic detection, and clustering algorithm.

Yunming Ye received the Ph.D. degree in computer science from Shanghai Jiao Tong University, Shanghai, China.

He is currently a Professor with the Shenzhen Graduate School, Harbin Institute of Technology, Harbin, China. His current research interests include data mining, text mining, and ensemble learning algorithms.



Haijun Zhang received the B.Eng. and master's degrees from Northeastern University, Shenyang, China, in 2004 and 2007, respectively, and the Ph.D. degree from the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, in 2010.

He was a Post-Doctoral Research Fellow with the Department of Electrical and Computer Engineering, University of Windsor, Windsor, ON, Canada, from 2010 to 2011. Since 2012, he has been with the Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, China, where he is currently an Associate Professor of computer science. His current research interests include multimedia data mining, machine learning, pattern recognition, evolutionary computing, and communication networks.