

Fuzzy Clustering with Multiple Kernels

Naouel Baili
Multimedia Research Lab
CECS Department
University of Louisville, USA
Email: n0bail02@louisville.edu

Hichem Frigui
Multimedia Research Lab
CECS Department
University of Louisville, USA
Email: h.frigui@louisville.edu

Abstract—In this paper, the kernel fuzzy c-means clustering algorithm is extended to an adaptive cluster model which maps data points to a high dimensional feature space through an optimal convex combination of homogenous kernels with respect to each cluster. This generalized model, called Fuzzy C-Means with Multiple Kernels (FCM-MK), strives to find a good partitioning of the data into meaningful clusters and the optimal kernel-induced feature map in a completely unsupervised way. It constructs the kernel from a number of Gaussian kernels and learns a resolution specific weight for each kernel function in each cluster. This allows better characterization and adaptability to each individual cluster. The effectiveness of the proposed algorithm is demonstrated for several toy and real data sets.

Index Terms—Fuzzy Clustering, Multiple Kernels, Resolution Weights.

I. INTRODUCTION

Clustering methods have been used extensively in computer vision and pattern recognition. Fuzzy clustering, where an object can belong to multiple clusters with a certain degree of membership, has been shown to be more effective than crisp clustering where a point can be assigned to only one cluster. This is particularly useful when the boundaries among the clusters are not well separated and ambiguous. Moreover, the memberships may help in discovering more hidden relations between a given object and the disclosed clusters. The Fuzzy C-Means (FCM) is one of the most popular fuzzy clustering algorithms [1]. The basic FCM uses the squared-norm to measure similarity between prototypes and data points, and is suitable for identifying spherical clusters. Many extensions of the FCM have been proposed to cluster more general data set. Most of those algorithms are realized by replacing the squared-norm in the object function of FCM with other similarity measures [1][2]. Others, such as the Kernel-based fuzzy c-means (KFCM) [3] adopts a kernel-induced metric in the data space to replace the original Euclidean norm. By replacing the inner product with an appropriate kernel function, one can implicitly perform a nonlinear mapping to a high dimensional feature space without increasing the number of parameters. This kernel approach has been successfully applied to many learning systems [4], such as Support Vector Machines (SVMs), kernel principal component analysis and kernel fisher discriminant analysis [5].

Kernel-based clustering relies on a kernel function to project data samples into a high-dimensional kernel-induced feature space. A good choice of the kernel function is therefore

imperative to the success of the clustering. However, one of the central problems with kernel methods in general is that it is often unclear which kernel is the most suitable for a particular task [6][7][8]. Thus, instead of using a single fixed kernel, recent developments in SVM and other supervised kernel methods have shown encouraging results in constructing the kernel from a number of homogeneous or even heterogeneous kernels [7][9] [10][11]. This provides extra flexibility and also allows domain knowledge from possibly different information sources to be incorporated to the base kernels. However, previous work in this so-called multiple kernel learning approach have all been focused on the supervised and semi-supervised learning settings. Therefore, how to efficiently learn and adopt multiple kernels in unsupervised learning, or fuzzy clustering in particular, is still an interesting yet unexplored research topic.

In this paper, we propose a new fuzzy clustering with multiple kernels algorithm (FCM-MK). FCM-MK strives to find a good partitioning of the data into meaningful clusters and the optimal kernel-induced feature map in a completely unsupervised way. FCM-MK is a generalization of the KFCM algorithm and uses a new optimization criterion to learn the optimal convex combination of homogenous kernels with respect to each cluster. It constructs the kernel from a number of Gaussian kernels and learns a resolution-specific weight for each kernel function in each cluster. This allows better characterization and adaptability to each individual cluster.

The organization of this paper is as follows. In section II, we give a brief overview of the Kernel-based learning algorithms. In section III, we describe the proposed fuzzy c-means with multiple kernels algorithm. Experiments are discussed in section IV and conclusions are provided in section V.

II. RELATED WORK

Kernel-based learning algorithms [12][13], are based on Cover's theorem. By nonlinearly transforming a set of complex and nonlinearly separable patterns into a higher-dimensional feature space, it is possible to separate these patterns linearly [14]. The difficulty of the curse of dimensionality can be overcome by the kernel trick, arising from Mercer's theorem [14]. By designing and calculating an inner-product kernel, we can avoid the time-consuming, sometimes even infeasible process to explicitly describe the nonlinear mapping $\Phi : X \mapsto \mathcal{F}$ from the input space X to a high dimensional feature space $\mathcal{F}$ and

compute the corresponding points in the transformed space. Computing the Euclidean distances in $\mathcal{F}$ without explicit knowledge of Φ is possible using the so called distance kernel trick:

$$\begin{aligned}\|\Phi(x_i) - \Phi(x_j)\|^2 &= (\Phi(x_i) - \Phi(x_j)) \cdot (\Phi(x_i) - \Phi(x_j)) \\ &= \Phi(x_i) \cdot \Phi(x_i) + \Phi(x_j) \cdot \Phi(x_j) - 2\Phi(x_i) \cdot \Phi(x_j) \\ &= K(x_i, x_i) + K(x_j, x_j) - 2K(x_i, x_j)\end{aligned}\quad (1)$$

Thus, the computation of the distances in the feature space is just a function of the input vectors. In fact, every algorithm in which input vectors appear only in dot products with other input vectors can be kernelized [15].

In (1), $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$ is the Mercer Kernel. It is a symmetric function $K : X \times X \rightarrow \mathbb{R}$ and satisfies

$$\sum_{i=1}^N \sum_{j=1}^N c_i c_j K(x_i, x_j) \geq 0 \quad \forall n \geq 2, \quad (2)$$

where $c_r \in \mathbb{R}$, and $r = 1, \dots, N$. Examples of Mercer kernels include [16]

- Linear

$$K^{(l)}(x_i, x_j) = x_i \cdot x_j \quad (3)$$

- Polynomial of degree p

$$K^{(p)}(x_i, x_j) = (1 + x_i \cdot x_j)^p, \quad p \in \mathbb{N} \quad (4)$$

- Gaussian

$$K^{(g)}(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right), \quad \sigma \in \mathbb{R} \quad (5)$$

Kernel-based clustering algorithms have the following main advantages.

- 1) They are more likely to obtain a linearly separable hyperplane in the high-dimensional, or even in an infinite feature space;
- 2) They can identify clusters with arbitrary shapes;
- 3) Kernel-based clustering algorithms, like support vector clustering (SVC), have the capability of dealing with noise and outliers;
- 4) For SVC, there is no requirement for prior knowledge to determine the system topological structure. In [17], the kernel matrix can provide the means to estimate the number of clusters.

The kernelized metric Fuzzy C-Means [18] is one of the most common kernel-based clustering algorithm. It minimizes the following objective function.

$$J^\phi = \sum_{i=1}^K \sum_{j=1}^N u_{ij}^m \|\phi(x_j) - \phi(v_i)\|^2, \quad (6)$$

subject to

$$u_{ij} \in [0, 1] \quad \text{and} \quad \sum_{i=1}^K u_{ij} = 1 \quad \forall j \quad (7)$$

In (6), u_{ij} denotes the membership of x_j in cluster i , v_i is the center of cluster i in the input space, and ϕ is the mapping

from the input space $\mathcal{X}$ to the feature space $\mathcal{F}$. Minimization of the function in (6) has been proposed only in the case of a Gaussian kernel. The reason is that the derivative with respect to the v_i in this case allows to use the kernel trick:

$$\frac{\partial K(x_j, v_i)}{\partial v_i} = \frac{(x_j, v_i)}{\sigma^2} K(x_j, v_i) \quad (8)$$

It can be shown [18] that the update equation for the membership is

$$u_{ij}^{-1} = \sum_{h=1}^K \left(\frac{1 - K(x_j, v_i)}{1 - K(x_j, v_h)} \right)^{1/(m-1)}, \quad (9)$$

and for the codevectors

$$v_i = \frac{\sum_{j=1}^N u_{ij}^m K(x_j, v_i) x_j}{\sum_{j=1}^N u_{ij}^m K(x_j, v_i)}. \quad (10)$$

A critical issue related to kernel-based clustering is the selection of an "optimal" kernel for the problem at hand. In fact, the performance of a kernel-based clustering depends critically on the selection of the kernel function, and on the setting of the involved parameters. The kernel function in use must conform with the learning objectives in order to obtain meaningful results. While solutions to estimate the optimal kernel function and its parameters have been proposed in a supervised setting [19][20][21][22], the problem presents open challenges when no labeled data are provided.

III. FUZZY C-MEANS WITH MULTIPLE KERNELS

Given N data points, $x_j \in \mathbb{R}^d$, for $j = 1, \dots, N$. Then, x_j is transformed via S mappings $\Phi_l(x_j) \mapsto \mathbb{R}^{d'_l}$ for $l = 1, \dots, S$ from the input space into S feature spaces $(\Phi_1(x_j), \dots, \Phi_S(x_j))$ where d'_l denotes the dimensionality of the l^{th} feature space. By using the closure properties of kernel functions [23], we construct a new cluster dependent similarity based on kernel K^i , between object j and cluster i . In particular, we define K^i as a linear combination of S Gaussian kernels $K_1, \dots, K_S$, with spread parameters $\sigma_1, \dots, \sigma_S$, respectively. That is,

$$\begin{aligned}K^{(i)}(x_j, v_i) &= \sum_{l=1}^S w_{il} K_l(x_j, v_i) \\ &= \sum_{l=1}^S \frac{w_{il}}{\sigma_l} \exp\left(-\frac{\|x_j - v_i\|^2}{2\sigma_l^2}\right)\end{aligned}\quad (11)$$

In (11), $W = [w_{il}]$, where $w_{il} \in [0, 1]$ is a resolution-specific weight for the kernel matrix K_l with respect to cluster i . A low value of w_{il} indicates that the bandwidth of kernel K_l is not relevant for the density estimation of cluster i , and that this matrix should not have a significant impact on the creation of this cluster. Similarly, a high value of w_{il} indicates that the bandwidth of kernel K_l is highly relevant for the density estimation of cluster i , and that this matrix should be the main factor in the creation of this cluster.

We normalize the kernel $K^{(i)}$ to construct a new kernel $\tilde{K}^i$,

where $\tilde{K}^i(x_j, x_j) = 1$, for $j = 1, \dots, N$. This normalized kernel is defined as

$$\begin{aligned}\tilde{K}^{(i)}(x_j, v_i) &= \frac{K^{(i)}(x_j, v_i)}{\sqrt{K^{(i)}(x_j, x_j) \cdot K^{(i)}(v_i, v_i)}} \\ &= \sum_{l=1}^S \frac{w_{il}}{\sigma_l} \exp\left(-\frac{\|x_j - v_i\|^2}{2\sigma_l^2}\right) \times \frac{1}{\sum_{l=1}^S \frac{w_{il}}{\sigma_l}}\end{aligned}\quad (12)$$

After kernel normalization, the distance in (1) becomes

$$\|\tilde{\Phi}^{(i)}(x_j) - \tilde{\Phi}^{(i)}(v_i)\|^2 = 2 - 2\tilde{K}^{(i)}(x_j, v_i) \quad (13)$$

Substituting the distance in (13) into the kernelized metric Fuzzy C-Means objective function (6), we define the objective function of the proposed FCM-MK algorithm as

$$\begin{aligned}J(U, V, W) &= 2 \sum_{i=1}^C \sum_{j=1}^N u_{ij}^m \\ &\left(1 - \sum_{l=1}^S \frac{w_{il}}{\sigma_l} \exp\left(-\frac{\|x_j - v_i\|^2}{2\sigma_l^2}\right) \times \frac{1}{\sum_{l=1}^S \frac{w_{il}}{\sigma_l}}\right)\end{aligned}\quad (14)$$

subject to

$$u_{ij} \in [0, 1], \text{ and } \sum_{i=1}^C u_{ij} = 1, \text{ for } j = 1, \dots, N; \quad (15)$$

and

$$w_{il} \in [0, 1], \text{ and } \sum_{l=1}^S w_{il} = 1, \text{ for } i = 1, \dots, C. \quad (16)$$

In (14), C and N represent the number of clusters and the number of data points respectively, u_{ij} is the fuzzy membership of point x_j in cluster i , v_i are the centers or prototypes of the clusters, and $m \in (1, \infty)$ denotes the fuzzifier.

The goal of the FCM-MK is to identify the resolution-specific weight w_{il} , the membership values u_{ij} and the cluster prototypes v_i by optimizing (14). In order to compute the optimal values of w_{il} , u_{ij} and v_i , we use an alternating optimization method, where we will alternate the optimization of w_{il} , u_{ij} and of v_i .

To optimize (14) with respect to the memberships, we first rewrite (14) as

$$J(U, V, W) = \sum_{i=1}^C \sum_{j=1}^N u_{ij}^m \text{dist}_{ij}^2 \quad (17)$$

where

$$\text{dist}_{ij}^2 = \left(2 - 2 \sum_{l=1}^S \frac{w_{il}}{\sigma_l} \frac{\exp\left(-\frac{\|x_j - v_i\|^2}{2\sigma_l^2}\right)}{\sum_{l=1}^S \frac{w_{il}}{\sigma_l}}\right) \quad (18)$$

Note that in (18), dist_{ij}^2 is not a function of the fuzzy memberships u_{ij} .

Then, to optimize (17) with respect to u_{ij} subject to (15), we use the Lagrange multiplier technique and obtain

$$J = \sum_{i=1}^C \sum_{j=1}^N u_{ij}^m \text{dist}_{ij}^2 - \sum_{j=1}^N \lambda_j \left(\sum_{i=1}^C u_{ij} - 1\right). \quad (19)$$

By setting the gradient of J to zero, we obtain the following update equation for the membership

$$u_{ij} = \frac{1}{\sum_{t=1}^C (\text{dist}_{ij}^2 / \text{dist}_{tj}^2)^{\frac{1}{m-1}}} \quad (20)$$

where dist_{ij}^2 is as defined in (18).

To optimize (14) with respect to the cluster prototypes v_i , we take the first derivative of J with respect to v_i and set it to zero. This yields the following update equation for the prototypes

$$v_i = \frac{\sum_{j=1}^N u_{ij}^m \bar{K}^{(i)}(x_j, v_i) x_j}{\sum_{j=1}^N u_{ij}^m \bar{K}^{(i)}(x_j, v_i)} \quad (21)$$

where

$$\bar{K}^{(i)}(x_j, v_i) = \left(\sum_{l=1}^S \frac{w_{il}}{\sigma_l^3} \exp\left(-\frac{\|x_j - v_i\|^2}{2\sigma_l^2}\right) \times \frac{1}{\sum_{l=1}^S \frac{w_{il}}{\sigma_l}}\right) \quad (22)$$

The optimization of (14) with respect to the resolution-specific weights has no closed form solution. Thus, we use the gradient descent method and update w_{il} iteratively using

$$w_{il}^{(new)} = w_{il}^{(old)} - \rho \frac{\partial J}{\partial w_{il}} \quad (23)$$

where ρ is a scalar parameter that determines the learning rate. In our approach, ρ is optimized via a line-search method. It can be shown that the gradient of J with respect to w_{il} is given by

$$\begin{aligned}\frac{\partial J}{\partial w_{il}} &= -2 \sum_{j=1}^N \frac{u_{ij}^m}{\sigma_l} \left(\frac{\exp\left(-\frac{\|x_j - v_i\|^2}{2\sigma_l^2}\right)}{\sum_{t=1}^S \frac{w_{it}}{\sigma_t}} - \frac{\sum_{t=1}^S \frac{w_{it}}{\sigma_t} \exp\left(-\frac{\|x_j - v_i\|^2}{2\sigma_t^2}\right)}{\left(\sum_{t=1}^S \frac{w_{it}}{\sigma_t}\right)^2} \right) \\ &= -2 \sum_{j=1}^N \frac{u_{ij}^m}{\sigma_l \sum_{t=1}^S \frac{w_{it}}{\sigma_t}} \left(K_l(x_j, v_i) - \tilde{K}^{(i)}(x_j, v_i) \right)\end{aligned}\quad (24)$$

The resulting FCM-MK algorithm is summarized in Algorithm 1.

The time complexity of the first step of the algorithm is $\mathcal{O}(NCdS)$ where N is the number of samples or data points, C is the number of clusters, d is the number of features or dimensions and S is the number of Gaussian kernels. In the second step of FCM-MK to update the prototypes, the time complexity is also $\mathcal{O}(NCdS)$. The time spend in the membership update is $\mathcal{O}(NC^2)$. We use the gradient descent technique to update the resolution-specific weight. If we note by τ_{gd} the number of iterations for the gradient descent

Algorithm 1 Fuzzy C-Means with Multiple Kernels

Fix the number of clusters C , fuzzification parameter $m > 1$, stopping criterion ϵ , maximum number of iterations q_{max} and iteration counter $q = 1$;
Initialize the fuzzy partition matrix $\mathbf{U}$;
Initialize cluster prototypes $\mathbf{V}$;
Pick the Gaussian parameters $\sigma_1, \dots, \sigma_S$ and initialize $w_{il}^{(0)} = 1/S$
repeat
1. Compute total similarities $\tilde{K}^{(i)}$ using (12);
2. Update cluster prototypes using (21);
3. Update fuzzy membership using (20);
repeat
4.1 Update $w_{il}^{(q-1)} = w_{il}^{(q)}$
4.2 Compute the gradients $\frac{\partial J}{\partial w_{il}}$ using (24)
4.3 Update resolution-specific weights using (23);
until $\|w_{il}^{(q)} - w_{il}^{(q-1)}\| < \epsilon$ or $q = q_{max}$
until (fuzzy memberships do not change);

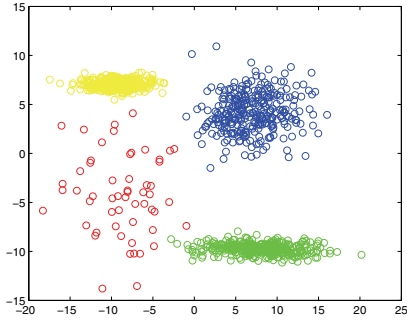


Fig. 1. Data set with 4 clusters of different shapes and densities

to converge, the time complexity of step 6 is $\mathcal{O}(\tau_{gd}NdS)$. Therefore, the total computational complexity of FCM-MK is

$$\mathcal{O}(\tau(NCdS + NC^2) + \tau\tau_{gd}NdS) \quad (25)$$

where τ is the number of iterations.

IV. EXPERIMENTAL RESULTS

To illustrate the ability of FCM-MK to learn appropriate local density fitting functions and cluster the data simultaneously, we use it to categorize a synthetic data set (refer to figure 1), that includes categories with unbalanced sizes and densities, and two real-world data sets selected from the UCI repository (digits and ionosphere). For the digits data, we focus on the pair 3 vs 8 that is difficult to differentiate [24]. We summarize all of these in Table I.

For each data set, we show the effectiveness of the proposed algorithm by evaluating the resolution specific weights assigned to each kernel induced-similarity. A low value of w_{il} indicates that the bandwidth σ_l of kernel K_l is not relevant for the density fitting of cluster i , and that the distance matrix computed with this kernel should not have significant impact on the creation of this cluster. Similarly, a high value of w_{il}

TABLE I
DESCRIPTION OF THE DATA SETS

Data	Size	Dimension	Class
Synthetic data set	1078	2	4
ionosphere	351	34	2
digits3v8	345	64	2
digits0689	692	64	4

TABLE II
RESOLUTION WEIGHTS LEARNED BY FCM-MK FOR THE DATA IN FIGURE 1

	$\sigma_1 = 0.5$	$\sigma_2 = 1$	$\sigma_3 = 3$	$\sigma_4 = 5$	True (σ_x, σ_y)
Cluster1	0.203	0.294	0.387	0.115	(3.20, 1.99)
Cluster2	0.106	0.298	0.395	0.201	(3.80, 4.09)
Cluster3	0.212	0.284	0.358	0.145	(3.77, 0.54)
Cluster4	0.296	0.390	0.202	0.114	(2.10, 0.42)

indicates that the bandwidth σ_l of kernel K_l is highly relevant for fitting the points in cluster i , and that this matrix should be the main factor in the creation of this cluster. We demonstrate the effectiveness of FCM-MK by comparing the clustering results to the following algorithms:

- 1) FCM with the Euclidean norm;
- 2) Kernel metric FCM (KFCM) where K is a gaussian kernel with S different values of σ , $\sigma_1, \dots, \sigma_S$;
- 3) KFCM with a kernel K constructed as the sum of the S Gaussian kernels, $K(x_j, v_i) = \sum_{l=1}^S \exp(-\frac{\|x_j - v_i\|^2}{2\sigma_l^2})$

For all algorithms, we use the same initialization, the same number of clusters, and the same fuzzifier. In all experiments, we choose the number of kernels $S = 4$ and $\sigma = 0.1 * D, 0.2 * D, 0.3 * D, 0.4 * D$ where $D = \left[\sum_{k=1}^d (\max(X^k) - \min(X^k))^2 \right]^{\frac{1}{2}}$ and X^k is the k^{th} attribute of the data set.

To assess the performance of the different clustering algorithms and compare them, we assume that the ground truth is known and we use some relative cluster evaluation measures [25]: the accuracy rate (Q_{RR}), the Jaccard coefficient (Q_{JC}), the Folkes-Mallows index (Q_{FMI}) and the Hubert index (Q_{HI}).

The FCM algorithm cannot categorize the data in figure 1 as it can be seen in figure 2a. This is due to the fact that FCM is designed to seek compact spherical clusters. However, the geometry of this data is more complex. The clusters are close to each others and have different densities. Similarly, the KFCM algorithm, using a single bandwidth, was not able to categorize this data correctly as it can be seen from figures 2b-2e. The KFCM algorithm, using a kernel constructed as the average of the 4 Gaussian kernels K_l with bandwidth σ_l , also cannot categorize the data. This is because one global bandwidth cannot take into account the variations of the different clusters.

On the other hand, the resolution weights learned by our approach (see table II) make it possible to identify the four clusters correctly. In table II, cluster4 has an ellipsoidal shape with a true standard deviation of 2.10 in the horizontal

TABLE III
RESOLUTION WEIGHTS LEARNED BY FCM-MK FOR DIGITS3v8

	$\sigma_1 = 5$	$\sigma_2 = 10$	$\sigma_3 = 15$	$\sigma_4 = 20$	True σ
digit3	0.198	0.356	0.256	0.190	7.89
digit8	0.202	0.395	0.236	0.167	7.25

TABLE IV
RESOLUTION WEIGHTS LEARNED BY FCM-MK FOR IONOSPHERE

	$\sigma_1 = 0.5$	$\sigma_2 = 1$	$\sigma_3 = 1.5$	$\sigma_4 = 2$	True σ
Cluster1	0.174	0.266	0.361	0.199	1.41
Cluster2	0.172	0.202	0.258	0.368	1.59

direction and 0.42 in the vertical direction. The geometry of cluster4 is captured by our algorithm. In fact, higher weights were assigned to $\sigma_l = 0.5$ and $\sigma_l = 1$ which reflects the existence of some points with small variance along the horizontal direction and some points with larger variance along the vertical direction. Similarly, the other three clusters were assigned different weights. The clustering performances of the different algorithms on this data are reflected on the cluster evaluation measures in table VI. As it can be seen, the statistics provided by FCM-MK are higher than the statistics provided by the other algorithms.

For the digits0689 data set, the FCM-MK assigned high weight to $\sigma_l = 10$ for all clusters (refer to table V). This is an indication that the clusters within this data have similar distributions and one global σ is sufficient for this data. This explains the comparable behavior of FCM-MK with KFCM with kernel K_2 as reported in table IX.

On the other hand, for the ionosphere data set, FCM-MK assigned different weights to the different clusters as reported in table IV. This indicates that the clusters within this data have different distributions. Thus, the statistics provided by FCM-MK are higher than the statistics of the other competing algorithms.

V. CONCLUSIONS

In this paper, we proposed a new fuzzy clustering algorithm with multiple kernels. The proposed FCM-MK algorithm uses a fixed set of kernels with different resolutions that cover the spectrum of the entire data. Data points are mapped to a high dimensional feature space through a convex combination of these kernels. The kernel weights are adapted to the different clusters to reflect their distributions. The FCM-MK algorithm optimizes one objective function to identify the optimal clusters' prototypes, their fuzzy membership degrees, and the optimal convex combination of the homogeneous kernels in an unsupervised way.

The effectiveness of the proposed algorithm is demonstrated and compared to similar algorithms using synthetic and real data sets. We showed that when the data has clusters with similar densities, the FCM-MK performance is comparable to the kernel FCM with one kernel. However, for data sets that have clusters with different densities, the proposed FCM-MK learns different kernel weights and outperforms the kernel FCM.

TABLE V
RESOLUTION WEIGHTS LEARNED BY FCM-MK FOR DIGITS0689

	$\sigma_1 = 5$	$\sigma_2 = 10$	$\sigma_3 = 15$	$\sigma_4 = 20$	True σ
digit0	0.281	0.384	0.261	0.074	6.45
digit6	0.174	0.377	0.343	0.106	7.55
digit8	0.203	0.414	0.335	0.048	7.25
digit9	0.144	0.412	0.352	0.102	9.25

In the current implementation, the FCM-MK algorithm uses fuzzy memberships that are constrained to sum to one. This makes the algorithm sensitive to noise and outliers. In fact, noise points can lead the algorithm to learn kernel weights that do not reflect the overall distribution of the clusters. One way to address this limitation is to adapt the FCM-MK to use possibilistic membership functions. We are currently investigating this alternative.

ACKNOWLEDGMENT

This work was supported in part by U.S. Army Research Office Grants Number W911NF-08-0255 and by a grant from the Kentucky Science and Engineering Foundation as per Grant Agreement No. KSEF-2079-RDE-013 with the Kentucky Science and Technology Corporation.

REFERENCES

- [1] J. Bezdek, *Pattern recognition with fuzzy objective function algorithms*. New York: Plenum, 1981.
- [2] K. Wu and M. Yang, "Alternative c-means clustering algorithms," *Pattern Recognition*, vol. 35, pp. 2267–2278, 2002.
- [3] W. X. Z. Wu and J. Yu, "Fuzzy c-means clustering algorithm based on kernel method," *Computational Intelligence and Multimedia Applications*, 2003.
- [4] D. Graves and W. Pedrycz, "Kernel-based fuzzy clustering and fuzzy clustering: A comparative experimental study," *Fuzzy Sets and Systems*, vol. 161, no. 4, pp. 522 – 543, 2010.
- [5] N. Cristianini and J. Taylor, "An introduction to SVMs and other kernel-based learning methods," *Cambridge Univ. Press*, 2000.
- [6] O. Bousquet and D. Herrmann, "On the complexity of learning the kernel matrix," *NIPS*, 2003.
- [7] L. G. P. B. G. Lanckriet, N. Cristianini and M. Jordan, "Learning the kernel matrix with semidefinite programming," *JMLR*, vol. 5, pp. pp. 27–72, 2006.
- [8] A. S. C. Ong and R. Williamson, "Learning the kernel with hyperkernels," *JMLR*, vol. 6, pp. pp. 1043–1071, 2005.
- [9] G. L. F. Bach and M. Jordan, "Multiple kernel learning, conic duality, and the smo algorithm," *ICML*, 2004.
- [10] S. C. A. Rakotomamonjy, F. Bach and Y. Grandvalet, "More efficiency in multiple kernel learning," *ICML*, 2007.
- [11] S. J. J. Ye and J. Chen, "Learning the kernel matrix in discriminant analysis via quadratically constrained quadratic programming," *SIGKDD*, 2007.
- [12] G. R. K. T. K. Muller, S. Mika and B. Scholkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Network*, vol. 12, no. 2, pp. pp. 181–201, 2001.
- [13] B. Scholkopf and A. Smola, *Learning with kernels: Support vector machines, regularization, optimization and beyond*. Cambridge, MA: MIT Press, 2002.
- [14] S. Haykin, *Neural networks: S comprehensive foundation*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1999.
- [15] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.
- [16] V. Vapnik, *The Nature of Statistical Learning Theory*. Inc., New York, NY, USA, 1995.
- [17] M. Girolami, "Mercer kernel based clustering in feature space," *IEEE Trans. Neural Netw.*, vol. 13, no. 3, pp. pp. 780–784, 2002.

- [18] W. X. Z.D. Wu and J. Yu., "Fuzzy c-means clustering algorithm based on kernel method," *Computational Intelligence and Multimedia Applications*, 2003.
- [19] J. S.-T. N. Cristianini and A. Elisseeff, "On kernel-target alignment," *Neural Information Processing Systems (NIPS)*, 2001.
- [20] O. Chapelle and V. Vapnik, "Choosing mutiple parameters for support vector machines," *Machine Learning*, vol. 46, no. 1, pp. pp. 131–159, 2002.
- [21] W. L. W. Wang, Z. Xu and X. Zhang, "Determination of the spread parameter in the gaussian kernel for classification and regression," *Neurocomputing*, vol. 55, no. 3, p. pp. 645, 2002.
- [22] W. C. J. Huang, P.C. Yuen and J. Lai, "Kernel subspace lda with optimized kernel parameters on face rcognition," *The sixth IEEE International Conference on Automatic Face and Gesture Recognition*, 2004.
- [23] J. Taylor and N. Cristianini, *Kernel methods for pattern analysis*. Cambrige University, 2004.
- [24] A. Asuncion and D. Newman, "Uci machine learning repository," 2007.
- [25] C. H. H. Frigui and F. C.-H. Rhee, "Clustering and aggregation of relational data with applications to image database categorization," *Pattern Recognition*, vol. 40, no. 11, pp. 3053–3068, 2007.

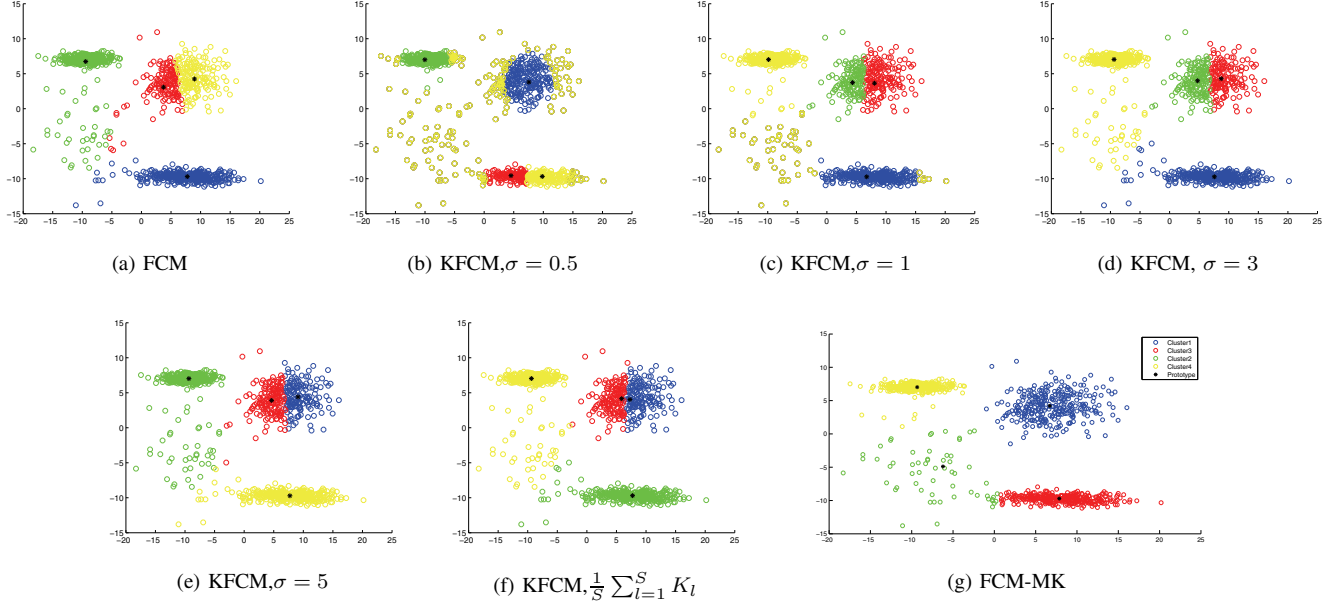


Fig. 2. Comparison of the partitions of the data set in figure 1 generated by the different algorithms

TABLE VI
COMPARISON OF THREE DIFFERENT ALGORITHMS ON THE DATA SET IN FIGURE 1

	FCM	KFCM					FCM-MK
		K_1	K_2	K_3	K_4	$\frac{1}{S} \sum_{l=1}^S K_l$	
Q_{RR}	84.9%	73.9%	86.7%	83.6%	82.7%	81.2%	98.9%
Q_{JC}	0.535	0.163	0.178	0.295	0.416	0.193	0.640
Q_{FMI}	0.687	0.281	0.303	0.456	0.588	0.325	0.781
Q_{HI}	0.569	0.007	0.035	0.241	0.423	0.065	0.691

TABLE VII
COMPARISON OF THREE DIFFERENT ALGORITHMS ON DIGITS3V8

	FCM	KFCM					FCM-MK
		K_1	K_2	K_3	K_4	$\frac{1}{S} \sum_{l=1}^S K_l$	
Q_{RR}	94.6%	90%	94.6%	94.4%	91.7%	88.7%	97.9%
Q_{JC}	0.189	0.189	0.189	0.189	0.19	0.189	0.396
Q_{FMI}	0.342	0.341	0.342	0.342	0.343	0.341	0.608
Q_{HI}	0	0	0	0	0.003	0	0.431

TABLE VIII
COMPARISON OF THREE DIFFERENT ALGORITHMS ON IONOSPHERE

	FCM	KFCM					FCM-MK
		K_1	K_2	K_3	K_4	$\frac{1}{S} \sum_{l=1}^S K_l$	
Q_{RR}	68%	86%	77%	72%	67%	62%	91%
Q_{JC}	0.297	0.326	0.322	0.317	0.298	0.294	0.396
Q_{FMI}	0.461	0.495	0.489	0.485	0.462	0.457	0.56
Q_{HI}	0.012	0.066	0.014	0.05	0.014	0.006	0.102

TABLE IX
COMPARISON OF THREE DIFFERENT ALGORITHMS ON DIGITS0689

	FCM	KFCM					FCM-MK
		K_1	K_2	K_3	K_4	$\frac{1}{S} \sum_{l=1}^S K_l$	
Q_{RR}	42.3%	93.1%	96.6%	94.1%	84.5%	82%	97.7%
Q_{JC}	0.119	0.255	0.257	0.255	0.207	0.160	0.399
Q_{FMI}	0.215	0.317	0.379	0.318	0.234	0.227	0.577
Q_{HI}	0	0.004	0.011	0.006	0.008	0.001	0.462