**Proceedings of the 2002 International Conference on Control and Automation Xiamen, China, Jun. 2002**

**MAL08-2**

# Fuzzy Clustering Using Kernel Method

Daoqiang Zhang Songcan Chen
Department of Computer Science and Engineering
Nanjing University of Aeronautics & Astronautics,
Nanjing 210016,China
* E-mail: daoqz@mail.com

**Abstract**

Classical fuzzy C -means (FCM) clustering is performed in the input space, given the desired number of clusters. Although it has proven effective for spherical data, it fails when the data structure of input patterns is non-spherical and complex. In this paper, we present a novel kernel-based fuzzy C-means clustering algorithm (KFCM). Its basic idea is to transform implicitly the input data into a higher dimensional feature space via a nonlinear map, which increases greatly possibility of linear separability of the patterns in the feature space, then perform FCM in the feature space. Another good attribute of KFCM is that it can automatically estimate the number of clusters in the dataset. The experimental results show that the proposed method has better performance in the Ring dataset.

## 1. Introduction

Clustering analysis is a useful technique for exploring the underlying structures of a given data set and is being applied in a wide variety of engineering and scientific disciplines such as medicine, psychology, biology, sociology, pattern recognition, image processing and data mining [1,2]. Generally speaking, clustering is the process of grouping the data into classes or clusters so that objects within a cluster have high similarity in comparison to one another, but are very dissimilar to objects in other clusters [2]. There are many methods concerning clustering, each has its own merits and disadvantages. Among these, maybe the most widely used approach is the C-means algorithm and its variations.

Fuzzy C-means (FCM) [4] clustering algorithm is the soft extension of the traditional hard C-means. It considers each cluster as a fuzzy set, while a membership function measures the possibility that each training vector belongs to a cluster. As a result, each training vector may be assigned to multiple clusters. Thus it can overcome in some degree the drawback of dependence on initial partitioning cluster values in hard C-means. However, just like the C-means algorithm, FCM is effective only in clustering those crisp, spherical, and non-overlapping data. When dealing with non-spherical shape and much overlapped data, such as the Ring dataset (see Fig.3 (a)), FCM cannot always work well (see Fig.3 (b)-3c)). In this paper, we use the kernel method [3][5] to construct the nonlinear version of FCM, and propose a kernel-based fuzzy C-means clustering algorithm (KFCM). The basic ideas of KFCM is to first map the input data into a feature space with higher dimension via a nonlinear transform and then perform FCM in that feature space. Thus the original complex and nonlinearly separable data structure in input space may become simple and linearly separable in the feature space after the nonlinear transform (see Fig.1). So we desire to be able to get better performance. Another merit of KFCM is, Unlike the FCM which needs the desired number of clusters in advance, it can adaptively determine the

number of clusters in the data under some criteria. The experimental results show that KFCM has best performance in the test for the Ring dataset.

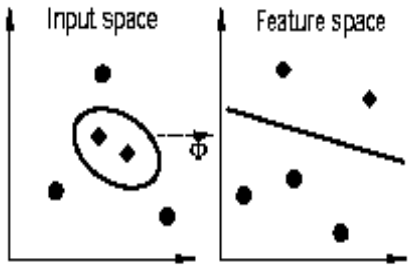



Fig.1 The idea of kernel method

## 2. Review of classical FCM

Before we discuss the kernel FCM algorithm, let us review the FCM algorithm briefly. Let $X = \{x_i, i = 1,2,..., n\}$ denote a set of $n$ input vectors, $X \subseteq R^p$ . $c$ is the desirednumber of clusters. $v_i, i = 1,2,..., c$ ,is the $i$th vector of cluster centres, $v_i \in R^p$ .And $u_{ik}, i$ =1,2,…, $c$ , $k$ = 1,2,…, $n$ , is the membership of $x_k$ in the $i$ th partitioning subset of $X$ ,satisfying

$$0 \le u_{ik} \le 1 \quad \text{for all } i, k; \tag{1a}$$

$$\sum_{i=1}^{c} u_{ik} = 1 \quad \text{for all } k; \tag{1b}$$

$$\sum_{k=1}^{n} u_{ik} > 0 \quad \forall i. \tag{1c}$$

The FCM selects the following objective function, defined as

$$J_m = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^m \| x_k - v_i \|^2 \tag{2}$$

where $m > 1$ is a weighting exponent on each fuzzy membership. According to (1b) and (2), by using an alternative iteration optimization procedure, the FCM can find the appropriate $u_{ik}$ and $v_i$ so as to make $J_m$ minimization for $m > 1$ . Once the membership $u_{ik}$ are got for all $i, k$ , we can label the input vector $x_k$ with $j_k$ , which is determined from the following equation

$$j_k = \arg_i \min u_{ik} \quad \text{for all } k. \tag{3}$$

## 3. The proposed kernel FCM algorithm

From the equation (2) , we can see that classical FCM algorithm is based on the input space sum-of-squares clustering criterion. If the separation boundaries between clusters are nonlinear then FCM will unsatisfactorily work . To solve this problem we adopt the strategy of nonlinearly mapping the input space into a higher dimensional feature space and then performing linear FCM within the feature space.

Assume we define the nonlinear map as $\Phi : x \to \Phi(x) \in F$ ,where $x \in X$ . $X$ denotes the input space, and $F$ denotes the feature space with higher dimension. Note that the cluster centre in the feature space can now be denoted by the following form

$$v_i = \sum_{k=1}^{n} \beta_{ik} \Phi(x_k) \quad \text{for all } i. \tag{4}$$

where $\beta_{ik}$ is the coefficients which will be calculated later. So similar to Equation (2) in section 2, we select the following objective equation to be optimized

$$J_m = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^m \| \Phi(x_k) - \sum_{l=1}^{n} \beta_{il} \Phi(x_l) \|^2 . \tag{5}$$

We rewrite the norm in equation (5) as the following

$$\| \Phi(x_k) - \sum_{l=1}^{n} \beta_{il} \Phi(x_l) \|^2 = \Phi(x_k)^T \Phi(x_k)$$

$$-2\sum_{l=1}^{n}\beta_{il}\Phi(x_k)^T\Phi(x_l)+\sum_{l=1}^{n}\sum_{j=1}^{n}\beta_{il}\Phi(x_k)^T\beta_{ij}\Phi(x_j). \quad (6)$$

It's interesting to see that equation (6) takes as the form of a series of dot products in feature space. And these dot products can easily be computed through Mercer kernel representations in the input space $K(x,y)=\Phi(x)^T\Phi(y)$. Thus the equation (6) can be formulated as follows

$$\| \Phi(x_k)-\sum_{l=1}^{n}\beta_{il}\Phi(x_l) \|^2$$

$$= K_{kk}-2\beta_i^T K_k+\beta_i^T K\beta_i . \quad (7)$$

where

$\beta_i=(\beta_{i1},\beta_{i2},...,\beta_{in})^T$, $i$ =1,2,..., $c$ ; $K_k$ = $(K_{k1},K_{k2},...,K_{kn})^T$ and $K=(K_1,K_2,...,K_n)$ ,for $k$ =1,2,..., $n$ . We can calculate $K_{ij}$ in the following way

$$K_{ij}=K(x_i,x_j) \quad \text{for all } i,j=1,2,...,n . \quad (8)$$

where $K$ is also named the kernel function. While any function satisfying the Mercer condition can be used as the kernel function, the best known is polynomial kernels, radial basis functions and Neural Network type kernels.

Substituting for the norm in equation (5) with equation (7), we get

$$J_m=\sum_{i=1}^{c}\sum_{k=1}^{n}u_{ik}^m(K_{kk}-2\beta_i^T K_k+\beta_i^T K\beta_i). \quad (9)$$

Minimizing $J_m$ with $m>1$ under the constraint of equation (1b), we have the expressions of $u_{ik}$ and $\beta_i$ as follows

$$u_{ik}=\frac{(1/(K_{kk}-2\beta_i^T K_k+\beta_i^T K\beta_i)^{1/(m-1)}}{\sum_{j=1}^{c}(K_{kk}-2\beta_i^T K_k+\beta_i^T K\beta_i)^{1/(m-1)}} \quad \text{all } i,k ; \quad (10a)$$

$$\beta_i=\frac{\sum_{k=1}^{n}u_{ik}^m K^{-1}K_k}{\sum_{k=1}^{n}u_{ik}^m} \quad \text{for all } i . \quad (10b)$$

The below is the full description of our proposed kernel FCM algorithm (KFCM).

Step1: Fix $m>1$ and $\varepsilon>0$ for some positive constant.

Step2: Calculate the number of clusters $c$ using the method in section 4.

Step3: Initialize the coefficient vectors $\beta_i^0=(\beta_{i1}^0,\beta_{i2}^0,...,\beta_{in}^0)^T$, for all $i$=1,2,..., $c$ .

Step4: Calculate the kernel matrix and its inverse using the expression (8) with one of the following kernel functions

$$K(x,y)=(x^T y+1)^d \quad (11a)$$

$$K(x,y)=\exp(-\frac{\| x-y \|^2}{2\sigma^2}) \quad (11b)$$

$$K(x,y)=\tanh((x^T y)+b) \quad (11c)$$

Step5: For $t$ =1,2,…, $t_{\max}$

a. Update all memberships $u_{ik}^t$ with (10a).

b. Update all $\beta_i^t$ with (10b).

c. Compute

$$E^t=\max_{i,k} | u_{ik}^t-u_{ik}^{t-1} | .$$

d. If $E^t\le\varepsilon$ , stop; else next t.

Once the membership $u_{ik}$ are got, for all $i,k$ , we can label the input vector $x_k$ using the equation (3). And it is important to note

that the proposed algorithm does not need to be given the number of clusters in advance. Instead, it can automatically compute the number of clusters in the data. In the next section, we'll discuss in detail the method used in KFCM to estimate the number of clusters.

## 4. Estimation of the number of clusters

In the KFCM algorithm, it is the kernel matrix $K$, got by expression (8), that was mostly processed, which can be easily seen from the expression (10a) and (10b). In addition, the kernel matrix can be used to determine the number of clusters in the data set. In [6], Girolami first discussed that interesting phenomenon. His idea is very simple: as each element of the kernel matrix defines a dot-product distance in the feature space , the matrix will have a block diagonal structure when there are definite clusters within the data sets. And we can use this block diagonal structure to determine the number of clusters. Although Girolami's method is first used in the C-means clustering , we found it also can be applied into our algorithm. An illustration for this method can be seen in Fig.2. Where Fig.2 (a) is the original well-separated spherical data set. Fig.2 (b) is the plot of kernel matrix in the KFCM algorithm clearly showing the inherent block structure. Fig.2 (c) shows the most significant eigenvalues of the kernel matrix in the KFCM algorithm. Thus through counting the number of significant eigenvalues of kernel matrix, we can get the number of clusters. And in this exam the value is five.

## 5. Experimental results

In this section, we show some experimental results from using the FCM and our proposed KFCM algorithm. And we use the method in section 4 to calculate the number of clusters. We first use the spherical dataset in Fig.2 (a), and the FCM and KFCM algorithms can both separate the five clusters completely. Whereas for the input data in Fig.3 (a), the results shown in Fig.3 (b)-3 (c) indicate the FCM algorithm is unable to correctly perform clustering for this input patterns. On the other hand, our proposed algorithm can successfully separate the two clusters, as shown in Fig.4 (a)-4 (b).

Finally, we tested the real Iris data (150 input patterns with 3 clusters). Fig.5 shows the process of estimation of the number of clusters. And from Fig.5 (b), we estimate the number of clusters in the Iris data at three. By using the FCM and KFCM algorithm respectively, we get 1-3 less errors with KFCM than with FCM. But the distinction is not so apparent as in the Ring data, chiefly because the Iris dataset is only a little overlapped and thus can nearly be separated by the ordinary methods, such as FCM.

## 6. Reference


[1]. H. Jiawei and K. Micheline, Data Ming: Concepts and Techniques, Academic Press, 2000

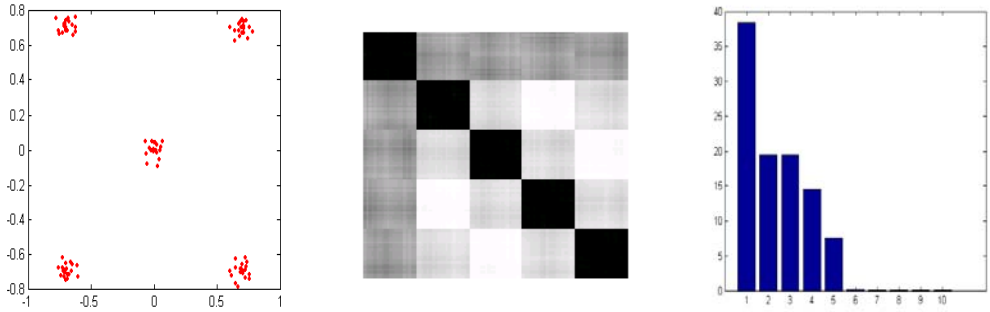
[2]. M. C. Su and C. H. Chou, A modified version of the k-means algorithm with a distance based on cluster symmetry, IEEE Trans. Pattern Anal. Machine Intell.,vol23, no.6,pp674-680,2001

[3]. K. I. Kim, S. H. Park, and H. J. Kim, Kernel principal component analysis for texture classification, IEEE Trans. Signal Proc. Letts. Vol8, no.2 , pp39-41, 2001

[4]. E.C.K.Tsao, J.C.Bezdek,and N.R.Pal, Fuzzy kohonen clustering networks, Pattern recognization, vol.27, no.5, pp.757-764, 1994

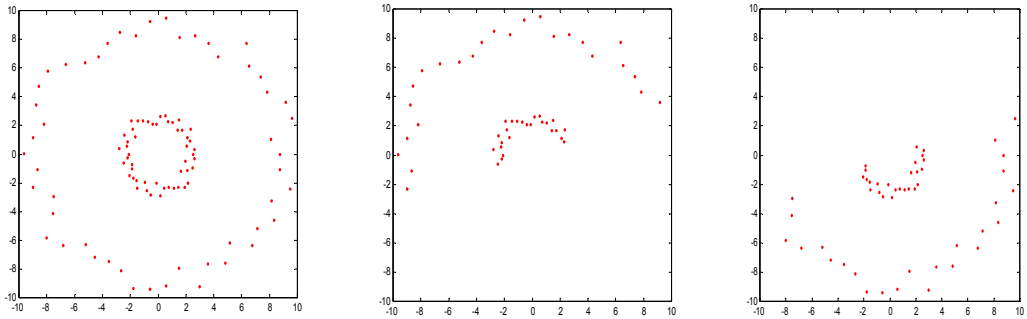
[5]. B.Scholkopf,A.Smola,and K.-R.Muller, Nonlinear component analysis as kernel eigenvalue problem, Neural Computation, vol.10,no.5,pp.1299-1319,1998

[6]. M. Girolami, Mercer kernel based clustering in feature space, IEEE Trans. Neural Net.(in press),2002.



(a) (b) (c)

Fig.2 Illustration of method used to determine the number of clusters in the KFCM algorithm



(a) (b) (c)

Fig.3 (a) Original Ring data (b) cluster 1 by FCM (c) cluster 2 by FCM.

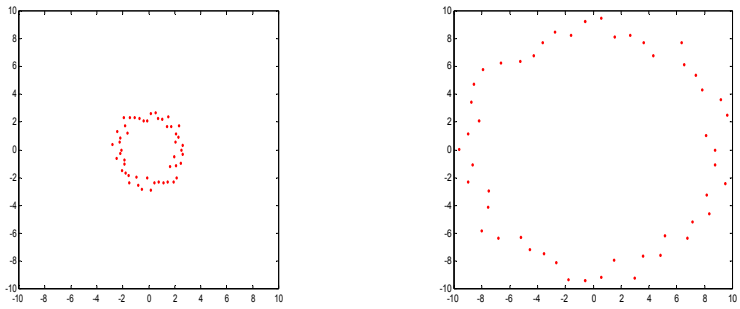


Fig.4 (a) cluster 1 by KFCM (b) cluster 2 by KFCM For the Ring dataset.

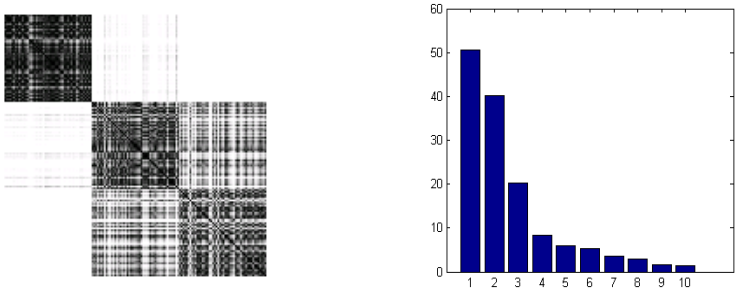


Fig.5 (a) plot of kernel matrix (b) the most significant eigenvalues of kernel matrix for Iris dataset.