GMKIT2-FCM: A Genetic-based Improved Multiple Kernel Interval Type-2 Fuzzy C-Means Clustering

Dzung Dinh Nguyen, Long Thanh Ngo and Long The Pham
Department of Information Systems,
Le Quy Don Technical University
Hanoi, Vietnam
Email: dinhdung1082@gmail.com, ngotlong@gmail.com

***Abstract*—This paper deals with a Genetic Multiple Kernel Interval Type 2 Fuzzy C-means clustering (GMKIT2-FCM), which automatically find the optimal number of clusters and determine the coefficients of the multiple kernel. The proposed GMKIT2-FCM algorithm provides us a new flexible vehicle to fuse different data information in the classification problems. That is, different information represented by different kernels is combined in the kernel space to produce a new kernel. The proposed algorithm contains two main stages. The first, a heuristic method based on Genetic algorithm (GA) and the average multiple kernel interval type 2 fuzzy c-means clustering (MKIT2-FCM) is adopted to automatically determine the optimal number of clusters and the initial the centroids. Then the results of the first stage are used in combination with GA and MKIT2-FCM to adjust the coefficients of multiple kernel to achieve better results. The experiments are done based on well-known datasets with the statistics show that the algorithm generates good quality of clustering problems.**

***Index Terms*—Genetic Algorithm, Type-2 fuzzy sets, type-2 fuzzy c-means clustering, Multiple kernel-based clustering, Genetic Multiple kernel clustering.**

I. Introduction

Clustering is a mathematical tool used to detect any structures or patterns in the data set, in which objects within the cluster level data show certain similarities. Clustering algorithms have different shapes from simple clustering as k-means and various improvements [2], [3], development of family of fuzzy c-mean clustering (FCM) [7].

To overcome the drawbacks of conventional FCM technique, Kernel fuzzy c-means clustering (KFCM) algorithm has been proposed. The limitation of the standard FCM algorithm is based on the Euclidian norm distance in the observation space, it has been shown while it is effective for spherical clusters it does not perform well for more general clusters [15],which is eliminated in KFCM by adapting a new kernel induced metric in the data space [14], which maps the original inputs into a much higher dimensional Hilbert space by some transform function.

However, for such kernel-based methods, a crucial step is the combination or selection of the best kernels among an extensive range of possibilities. This step is often heavily influenced by prior knowledge about the data and by the patterns we expect to discover [21]. Unfortunately, it is unclear which kernels are more suitable for a particular task [22], [23].

The problem is aggravated for many real-world clustering applications, in which there are multiple potentially useful cues.

Thus,it is necessary to aggregate features from different sources into a single aggregated feature. However, these features are often not equally relevant to clustering; some are irrelevant, and some are less important than others [15]. As most clustering methods do not embed a feature selection capability, such feature imbalances often necessitate an additional process of feature selection, or feature fusion, before clustering. Instead of a single fixed kernel, multiple kernels may be used. Recent developments in multiple kernel learning have shown that the construction of the multiple kernel fuzzy c-means (MKFC) algorithm simultaneously finds the the best degrees of membership and the optimal kernel weights for a non-negative combination of a set of kernels.

We also embed the feature weight computation into the clustering procedure. The incorporation of multiple kernels and the automatic adjustment of kernel weights renders MKFCM more immune to unreliable features or kernels. It also makes combining kernels more practical since appropriate weights are assigned automatically. Effective kernels or features tend to contribute more to the clustering and therefore improve results.

Recently, type-2 fuzzy sets are extensions of original fuzzy sets, have the advantage of handling uncertainty, which have been developed and applied to many different problems [4], [5] including data clustering problems. In addition, Interval type-2 fuzzy c-means clustering algorithm (IT2FCM) [1] has developed a step in the clustering method in which FOU (footprint of uncertainty) is created for the fuzzier m using two parameters for handling of uncertainty, making clustering more efficiently

A Multiple kernel interval type 2 fuzzy c-mean clustering (MKIT2-FCM) [26] has been proposed to handle the uncertainty better than MKFCM, which are more appropriate when clusters have significant overlap. It uses a linear composite of multiple kernels based on the IT2FCM algorithm.

However, the general MKFCM and MKIT2-FCM also have difficulties in determining the number of clusters, choosing the initial centroids of the clusters and updating the coefficients of the combined kernel (multiple kernel).

Thus, we proposed a Genetic Multiple kernel Interval

978-1-4673-6469-0/13/$31.00 © 2013

type 2 Fuzzy C-means(GMKIT2-FCM) which uses the cluster validity measure proposed by Ramze Rezaee [6] to evaluate the clustering results and by minimizing this validity measure by GA through adjusting the initial centroids of the clusters, the number of clusters and the linear coefficients of the combined kernel, the proposed algorithm will automatically find the optimal number of clusters and gain the better results. The proposed algorithm contains two main stages. The first, a heuristic method based on Genetic algorithm (GA) and the average multiple kernel interval type 2 fuzzy c-means clustering (MKIT2-FCM) is adopted to automatically determine the optimal number of clusters and the initial the centroids. Then the results of the first stage are used in combination with GA and MKIT2-FCM to adjust the coefficients of multiple kernel to achieve better results. Experiments are implemented based various datasets of classification to show the advantage of the proposed approach.

II. Preliminaries

A. The Kernel Technique[25]

The key idea of the kernel technique is to invert the chain of arguments, i.e., choose a kernel k rather than a mapping before applying a learning algorithm. Of course, not any symmetric function k can serve as a kernel. Suppose our input space χ has a finite number of elements, i,e., $\chi = \{x_1, .., x_r\}$. Then, the $r \times r$ kernel matrix $\mathbf{K}$ with $\mathbf{K}_{ij} = k(x_i, x_j)$ is by definition a symmetric matrix and $k(x_i, x_j)$ is kernel value between $x_i, x_j \in \chi$. The necessary and sufficient conditions of $k : \chi \times \chi \to \mathbb{R}$ be a kernel are given by Mercers theorem.

Theorem 2.1: The function $k : \chi \times \chi \to \mathbb{R}$ is a Mercer kernel if, and only if, for each $r \in \mathbb{N}$ and $x = (x_1, x_2..., x_r) \in \chi^r$ the $r \times r$ matrix $K = (k(x_i, x_j))_{i,j=1}^{r}$ is positive semi definite.

There exist simple rules for designing kernels on the basis of given kernel functions.

Theorem 2.2: Functions of kernels. Let $k_1 : \chi \times \chi \to \mathbb{R}$ and $k_2 : \chi \times \chi \to \mathbb{R}$ be any two Mercer kernels. Then, the functions $k : \chi \times \chi \to \mathbb{R}$ and $x, \tilde{x} \in \chi$ given by

1.$k(x, \tilde{x}) = k_1(x, \tilde{x}) + k_2(x, \tilde{x})$

2.$k(x, \tilde{x}) = c.k_1(x, \tilde{x}) \;\; for\ all\ c \in \mathbb{R}^+$

3.$k(x, \tilde{x}) = k_1(x, \tilde{x}) + c\ for\ all\ c \in \mathbb{R}^+$

4. $k(x, \tilde{x}) = k_1(x, \tilde{x}).k_2(x, \tilde{x})$

5. $k(x, \tilde{x}) = f(x).f(\tilde{x})\, foranyfunctionf : \chi \to \mathbb{R}\ are\ also\ Mercer\ kernels.$

Theorem 2.3: Let $k_1 : \chi \times \chi \to \mathbb{R}$ be any Mercer kernel. Then, the functions $k : \chi \times \chi \to \mathbb{R}$ given by

1. $k(x, \tilde{x}) = (k_1(x, \tilde{x}) + \theta_1)^{\theta_2}, for\ all\ \theta_1 \in \mathbb{R}^+ and\ \theta_2 \in \mathbb{N}$.

2. $k(x, \tilde{x}) = \exp\left(\frac{k_1(x,\tilde{x})}{\sigma^2}\right) \;\; for\ all\ \sigma \in \mathbb{R}^+$

3.$k(x, \tilde{x}) = \exp\left(-\frac{k_1(x,x) - 2k_1(x,\tilde{x}) + k_1(\tilde{x},\tilde{x})}{2\sigma^2}\right), for\ all\ \sigma \in \mathbb{R}^+$

4.$k(x, \tilde{x}) = \frac{k_1(x,\tilde{x})}{\sqrt{k_1(x,x).k_1(\tilde{x},\tilde{x})}}$

are also Mercer kernels.

B. Kernel Interval Type-2 Fuzzy C-mean Clustering

As an enhancement of classical IT2FCM, the Kernel Interval Type 2 Fuzzy C-mean Clustering (KIT2FCM) use a nonlinear map defined as $\phi : x \to \phi(x) \in H, x \in X \in R^d$. Where X denotes the data set or the feature space and H is a Hilbert space (usually called kernel space). In the new kernel space, the data demonstrate simpler structures or patterns. According to clustering algorithms, the data in the new space show clusters that are more spherical and therefore can be clustered more easily by IT2FCM algorithms [13]. Generally, the transform function ϕ is not given out explicitly, but the kernel function is given and it is defined as $k : \theta \times \theta \to R$

$$k(x, y) = \phi(x)\,\phi(y)^T \;\; \forall x, y \in R \tag{1}$$

There are two major forms of kernel interval type 2 clustering (KIT2FCM). The first one comes with prototypes constructed in the feature space, referred as KIT2FCM-F (with F standing for the feature space). In the second category, abbreviated as KIT2FCM-K (with K standing for the kernel space), the prototypes are retained in the kernel space and thus the prototypes must be approximated in the feature space by computing an inverse mapping from kernel space to feature space.

Similar to IT2FCM, the KIT2FCM algorithm use two fuzzifiers m_1 and m_2 to handle the uncertainty

Upper/lower degrees of membership, $\overline{u}_{ij}$ and $\underline{u}_{ij}$ are determined as follows:

$$\overline{u}_{ij} = \begin{cases} \dfrac{1}{\sum_{l=1}^{c}\left(\frac{d_{\phi ij}}{d_{\phi lj}}\right)^{2/(m_1-1)}} & if\ \dfrac{1}{\sum_{l=1}^{c}\left(\frac{d_{\phi ij}}{d_{\phi lj}}\right)} < \dfrac{1}{c} \\ \dfrac{1}{\sum_{l=1}^{c}\left(\frac{d_{\phi ij}}{d_{\phi lj}}\right)^{2/(m_2-1)}} & otherwise \end{cases} \tag{2}$$

$$\underline{u}_{ij} = \begin{cases} \dfrac{1}{\sum_{l=1}^{c}\left(\frac{d_{\phi ij}}{d_{\phi lj}}\right)^{2/(m_1-1)}} & if\ \dfrac{1}{\sum_{l=1}^{c}\left(\frac{d_{\phi ij}}{d_{\phi lj}}\right)} \geq \dfrac{1}{c} \\ \dfrac{1}{\sum_{l=1}^{c}\left(\frac{d_{\phi ij}}{d_{\phi lj}}\right)^{2/(m_2-1)}} & otherwise \end{cases} \tag{3}$$

in which $i = \overline{1, c}$, $j = \overline{1, n}$, $d_{\phi ij} = \|\phi(x_j) - \phi(v_i)\|$. If we use the Gaussian kernel then $k(x, x) = 1$ and $\|\phi(x_j) - \phi(v_i)\|^2 = 2(1 - k(x_j, v_i))$.

The derivation of the prototypes depends on the specific selection of the kernel function. The calculation of the prototypes v_i for i =1,2,...,c with the Gaussian kernel and degree of membership $u_{ij} = \frac{\overline{u}_{ij} + \underline{u}_{ij}}{2}$ and m is a type 1 fuzzifier (m usually is 2) as follows:

$$v_i = \frac{\sum_{j=1}^{n} u_{ij}^m k(x_j, v_i)\, x_j}{\sum_{j=1}^{N} u_{ij}^m k(x_j, v_i)} \tag{4}$$

Because each pattern has membership interval as the upper $\overline{u}$ and the lower $\underline{u}$, each centroid of cluster is represented by the interval between v^L and v^R.

The iterative algorithm for finding centroids

- Step 1: Find $\overline{u}_{ij}, \underline{u}_{ij}$, by the equations (2)-(3).
- Step 2: Set m = arbitrary and $m \geq 1$; Compute $v'_j = (v'_{j1}, ..., v'_{jM})$ by (4) with $u_{ij} = \frac{(\overline{u}_{ij} + \underline{u}_{ij})}{2}$.
  Sort N patterns on each of M features in ascending order.
- Step 3: Find index k such that: $x_{kl} \leq v'_{jl} \leq x_{(k+1)l}$ with $k = 1, .., N$ and $l = 1, .., M$.
  Update u_{ij}: If $i \leq k$ then $u_{ij} = \underline{u}_{ij}$ else $u_{ij} = \overline{u}_{ij}$.
- Step 4: Compute v''_j by (4) and Compare v'_{jl} with v''_{jl}. If $v'_{jl} = v''_{jl}$ then $v_R = v'_j$ else set $v'_{jl} = v''_{jl}$ and back to Step 3.

In Case, to define v_L:

- In step 3 we modify: Update u_{ij}: If $i \leq k$ then $u_{ij} = \overline{u}_{ij}$ else $u_{ij} = \underline{u}_{ij}$. And in step 4 replace V_R with v_L.

After obtaining v_i^R, v_i^L, type-reduction is applied to get centroid of clusters as follows:

Compute the mean of centroid , v_j as:

$$v_i = (v_i^R + v_i^L)/2 \tag{5}$$

For membership grades:

$$u_i(x_k) = (u_i^R(x_k) + u_i^L(x_k))/2, j = 1, ..., C \tag{6}$$

in which

$$u_i^L = \sum_{l=1}^{M} u_{il}/M, u_{il} = \begin{cases} \overline{u}_i(x_k) & \text{if } x_{il} \text{ uses } \overline{u}_i(x_k) \text{ for } v_i^L \\ \underline{u}_i(x_k) & otherwise \end{cases} \tag{7}$$

$$u_i^R = \sum_{l=1}^{M} u_{il}/M, u_{il} = \begin{cases} \overline{u}_i(x_k) & \text{if } x_{il} \text{ uses } \overline{u}_i(x_k) \text{ for } v_i^R \\ \underline{u}_i(x_k) & otherwise \end{cases} \tag{8}$$

Next, The defuzzification for KIT2FCM is made as if $u_i(x_k) > u_j(x_k)$ for $j = 1, ..., C$ and $i \neq j$ then x_k is assigned to cluster i.

C. Multiple kernel Interval Type 2 Fuzzy C-mean Clustering

A MKIT2-FCM is extended by combining different kernels to obtain better results. The general framework of MKIT2-FCM aims to minimize the objective function as the KIT2-FCM, i.e.,

$$\begin{aligned} J_{m_1}(U, V) &= \sum_{i=1}^{c} \sum_{j=1}^{n} {u_{ij}}^{m_1} \| \phi_{com}(x_i) - v_i \|^2 \\ J_{m_2}(U, V) &= \sum_{i=1}^{c} \sum_{j=1}^{n} {u_{ij}}^{m_2} \| \phi_{com}(x_i) - v_i \|^2 \end{aligned} \tag{9}$$

Where ϕ_{com} is the transformation defined by the combined kernels.

$$k_{com}(x, y) = < \phi_{com}(x), \phi_{com}(y) > \tag{10}$$

The composite kernel k_{com} is defined as a combination of multiple kernels. For example, two simple composite kernels are $k = k_1 + \alpha * k_2$ or $k = k_1 * k_2$ A linearly combined kernel function is

$$k_{com} = {w_1}^b k_1 + {w_2}^b k_2 + ... + {w_l}^b k_l \tag{11}$$

Where $b > 1$ is a coefficient kernel. The regulation on weights, $w_1, w_2, ..., w_l$,is $\sum_{i=1}^{l} w_i = 1$. When the number of parameters in the combined kernel is small, the parameters can be adjusted by trial and error.

Some learning algorithms that adjust the weights w_i automatically in typical kernel learning methods like multiple-kernel regressions and classifications [17], [18] have been studied. Here, we propose a similar algorithm for MKIT2-FCM using linearly combined kernels. The typical kernels are defined on $\mathbb{R}^p \times \mathbb{R}^p$ are: Gaussian kernel $k(x_i, x_j) = exp(-|x_i - x_j|^2/r^2)$ and Polynomial kernel $k(x_i, x_j) = (x_i * x_j + d)^2$

Upper/lower degrees of membership, $\overline{u}_{ij}$ and $\underline{u}_{ij}$ are determined by the equations 2 and 3 with d_ϕ is replaced by $d_{\phi_{com}}$

$$\begin{aligned} & d_{\phi_{com}}(x_j, v_i)^2 = \| \phi_{com}(x_j) - v_i \|^2 \\ & = k_{com}(x_j, x_j) + \frac{2 \sum_{h=1}^{n} (u_{ih})^m k_{com}(x_j, x_h)}{\sum_{h=1}^{n} (u_{ih})^m} + \\ & \frac{\sum_{h=1}^{n} \sum_{l=1}^{n} (u_{ih})^m (u_{il})^m k_{com}(x_h, x_l)}{\left(\sum_{h=1}^{n} (u_{ih})^m\right)^2} \end{aligned} \tag{12}$$

Multiple kernel interval type 2 fuzzy c-means algorithm

Given a set of n data points $X = \{x_i\}_{i=1}^{n}$, a set of kernel functions $\{k_i\}_{i=1}^{l}$, parameters m_1, m_2 and the desired number of clusters c. Output a membership matrix $U = \{u_{ic}\}_{i,c=1}^{n}, c$ and weights $\{w_i\}_{i=1}^{l}$ for the kernels.

- Step 1: Initialize centroid matrix $V^0 = \{v_i\}_{i=1}^{c}$ by choosing random from dataset and the membership matrix U^0 follow the equation:

$$u_{ij} = \frac{1}{\sum_{l=1}^{c} \left(\frac{d_{ij}}{d_{lj}}\right)^{2/(m-1)}} \tag{13}$$

  Where m is a constant, $m > 1$ and $d_{ij} = d(x_j - v_i) = \|x_j - v_i\|$
- Step 2: Repeat:
  +, Calculate Interval membership values $\overline{u}_{ij}$ and $\underline{u}_{ij}$ followed Eq(2,3) and Eq(12).
  +, Update the centroid matrix followed the iterative algorithm for finding centroids in KIT2FCM and Eq(5).
  +,Update the membership matrix by Eq(6)
  +,Assign data x_j to cluster c_i if data $(u_j(x_i) > u_k(x_i))$, $k = 1, .., c$ and $j \neq k$.
- Step 3: Termination criteria satisfied or maximum iterations reached Return U and V else back to step 2.

III. Genetic Multiple kernel Interval Type-2 FCM Clustering

The algorithm contain two main step: The first step: Using the average of the coefficients of the multiple kernel with $w_i = \frac{1}{l}$ to find the initial centroids and the number of the clusters. The second step: we use the results of the first step GA to improve the coefficients of the multiple kernel.

A. The clustering validity measure

The cluster validity measure used in this paper is the one proposed by Ramze Rezaee [6]. It aims at minimizing the validity index given by the function

$$V_{CWB} = \alpha Scat(C) + Dis(C) \tag{14}$$

The term $Scat(C)$ of V_{CWB} is the average of the scattering within the clusters for C number of clusters, which is defined as

$$Scat(C) = \frac{\frac{1}{C}\sum_{i=1}^{C} ||\sigma(v_i)||}{||\sigma(X)||} \tag{15}$$

in which $||X|| = (X^T.X)^{1/2}$ and

$$\sigma(v_i) = \frac{1}{n}\sum_{k=1}^{n} u_{ik}(x_k - v_i)^2 \tag{16}$$

$$\sigma(X) = \frac{1}{n}\sum_{k=1}^{n} (x_k - \bar{x})^2 \tag{17}$$

with $\bar{x} = \sum_{k=1}^{n} x_k/n$ and $x_k \in X$. The $Scat(C)$ term is used to measure the compactness of the clusters. A small value for this term show a compact partition. The $Dis(C)$ term is the total separation between the clusters which is defined as follows:

$$Dis(C) = \frac{D_{max}}{D_{min}} \sum_{k=1}^{C} (\sum_{z=1}^{C} \parallel v_k - v_z \parallel)^{-1} \tag{18}$$

where $D_{max} = maximum(\parallel v_i - v_j \parallel)$ and $D_{min} = minimum(\parallel v_i - v_j \parallel), \forall i, j \in 1, ..., C$. Lastly, α is a weighting factor, given as

$$\alpha = Dis(C_{max}) \tag{19}$$

This validity measure serves the dual purpose of minimizing cluster spread, measure the separation of the clusters and it is influenced by the geometry of the cluster centroids.

B. Genetic Multiple kernel IT2-FCM Clustering

In GA applications, the unknown parameters are encoded in the form of strings, so-called chromosomes. A chromosome is encoded with binary, integer or real numbers.

This algorithm consist of two main stages:

Stage 1: Automatically find the optimal number of clusters and the initial centroids. In this stage, we use the average MKIT2-FCM with $w_0 = w_i = 1/l$ where l is the number of the kernels.

$$k_{com} = {w_0}^b k_1 + {w_0}^b k_2 + ... + {w_0}^b k_l \tag{20}$$

The parameter $b >$ is chosen according to experience.

Detail algorithm in the stage 1 contains six following steps:

Step 1: Initialize the chromosomes and fitness function f.

A chromosome is encoded with a unit which represents a potential cluster centroid. Because the typical data are usually represented by the positive real values. In this research, a chromosome is encoded with a unit of positive real values. For example, a chromosome CH_i contain $v_1, v_2, ..., v_c$ where v_i is a positive real value.

Without assigning the number of clusters, a variable string length is used. Invalid (non-existing) clusters are represented with negative integer "-1". It means if the prototype $v_i = -1$ then this cluster unit is invalid and we consider that this cluster doesn't exist and the number of the valid clusters is the number of the clusters. The values of the chromosomes are changed in an iterative process to determine the correct number of clusters (the number of valid units in the chromosomes) and the actual cluster centroids for a given clustering problem.

The length of the chromosome, C, is equivalent to the number of clusters in the clustering problem. C takes value in the range [C_{min}, C_{max}], where C_{min} is usually assigned to 2 and C_{max} describes the maximum chromosome length, which means the maximum number of possible cluster centroids and C_{max} must be selected according to experience.

The goal for achieving a proper clustering is to minimize the V_{CWB}. Thus, the fitness function for chromosome j is defined as $1/V^j_{CWB}$, which is equivalent to the clustering with the smallest inner-cluster scatter and the largest cluster separation. Therefore, the fitness function is defined as

$$f = \frac{1}{V_{CWB}} \tag{21}$$

Randomly generate CH chromosomes from kernel space. Such a chromosome belongs to the so-called parent generation. One (arbitrary)chromosome of the parent generation is of size C. Each chromosome of the population is a potential solution by MKIT2-FCM algorithm with number of clusters C.

Calculate the fitness value f_i of the chromosome CH_i following the equation 21. And store the best chromosomes by $f_0 = max(f_i)$ where $i = 1..ch$ and $CH_0 = CH_i$

Specify the parameters of GA are crossover probability μ_c, mutation probability μ_m and the termination criterion: the error σ and the maximum number of generations G.

Step 2:

Choose chromosomes for the next generation by using fitness value associate a probability of selection with each individual chromosome. Roulette wheel selection is applied, a proportional selection algorithm where the number of copies of a chromosome, that go into the mating pool for subsequent operations, is proportional to its fitness. If f_i is the fitness of individual CH_i in the population, its probability of being selected is as follows:

$$ch_i = \frac{f_i}{\sum_{i=1}^{CH} f_i} \tag{22}$$

where CH is the number of individuals in the population.

Step 3:

Using the crossover operation is to create two new individual chromosomes from two existing chromosomes selected randomly from the current population. In this research, the one-point crossover with a fixed crossover probability of μ_c is used; For the one-point crossover, two chromosomes are randomly chosen from the population. Assuming the length of the chromosome to be C, this process randomly chooses

a point between 1 and $C-1$ and swaps the content of the two chromosomes beyond the crossover point to obtain the offspring. A crossover between a pair of chromosomes is affected only if they satisfy the crossover probability.

Step 4:

Applying the mutation operator, all the chromosomes in the population are checked unit by unit and according to a fixed probability μ_m. All values of a specific unit may be randomly changed. In this paper, a number λ in the range $[0,1]$ is generated with uniform distribution. If the value at a gene position is v, after mutation it becomes

$$v = v + \lambda * v, if \quad v > 0$$
$$v = v + \lambda, if \quad v = 0$$

Step 5:

Reinsertion of new individuals to the population. In this step, we do a MKIT2-FCM loop to cluster the processing data.

After MKIT2-FCM step, the fitness value f_i of the chromosome CH_i can be calculated following the equation 21.

Choose the biggest fitness value f_c to compare with f_0, calculate the error $\sigma_c = f_c - f_0$.

If f_c greater than f_0 then $f_0 = f_c$ and $CH_0 = CH_c$

Step 6:Termination criterion

If the number of Iterations ¡ G and $\sigma < \sigma_c$ Back to Step 3 else output Chromosome CH_0 with the maximum fitness value f_0.

Stage 2: Automatically adjust the coefficients w_i of the multiple kernel to achieve the better clustering results.

After stage 1, we have the best chromosome CH_0 with the optimal number of clusters C_{opt} and the initial centroids of the clusters V $(v_1..v_{c_{opt}})$.

Similarity to the Stage 1, Stage 2 also use GA to get the better results through the fitness 21.

Detail algorithm in stage 2 consists of six main steps:

Step 1:

Initialize the CH chromosomes and fitness function f. A chromosome is encoded with a unit which represents the coefficients of the combined kernel. For example, a chromosome CH_i contain $w_1, w_2, ..., w_l$ where $w_i \in range[0..1]$ and the size of CH_i is l which is the number of the kernels. The exponential coefficient b is chosen according to the experiments or experience

Randomly generate CH chromosomes from range [0..1]. Each chromosome of the population is a potential solution by MKIT2-FCM algorithm with number of clusters C_{opt} and the initial centroids V $(v_1..v_{c_{opt}})$.

Calculate the fitness value f_i of the chromosome CH_i following the equation 21. And store the best chromosomes by $f_0 = max(f_i)$ where $i = 1..ch$ and $CH_0 = CH_i$

Similarity to Stage 1, specify the parameters of GA are crossover probability μ_c, mutation probability μ_m and the termination criterion: the error σ and the maximum number of generations G.

From Step 2 to Step 6: Do similar work as step 2 to step 6 in the Stage 1. There is only a bit difference in Step 6, if we achieve the better clustering result, we need to update the centroids V $(v_1..v_{c_{opt}})$.

After two stages, we achieve the best clustering results with the number of cluster C_{opt}, the centroids of the clusters V $(v_1..v_{c_{opt}})$ and the coefficients of the multiple kernel (the combined kernels)$w_1..w_l$ with the fitness value f_0.

IV. Experiments

To assess the results of clustering: Experiments are implemented on the well-known datasets which are IRIS, Wisconsin Diagnostic Breast Cancer (WDBC), Wine [27] which the number of clusters has already known.

The parameters of the GMKIT2-FCM are set: The size of the population, CH, is taken 50, selection is Roulette Wheel, crossover rate, $\mu_c = 0.9$ and mutation rate, $\mu_m = 0.01$ [8]. Over the experimental results, the algorithm uses the terminating condition with the number of iterations G is set to 50 or the difference between these two fitness values (error) σ lies below 0.00001. Because the experiments are well-known with the number of clusters smaller than 11, we choose the value of chromosome length C is 11.

For each data, we use the number of kernels corresponding to data attributes. For examples, With Iris data, two kernels corresponding to the attribute Sepal and petal are used to build the combined kernel, similar to Wine data and WDBC data, they have thirteen kernels and thirty kernels, respectively. While only one kernel for all attributes of experimental data is used with KIT2-FCM.

Table I show the optimal number of clusters for these experimental data by the proposed algorithm GMKIT2-FCM with the validity index V_{CWB}. We see that the number of clusters obtained by the proposed algorithm with datasets Iris, WDBC and Wine respectively 3, 2 and 3 corresponding to the validity index V_{CWB} reached the minimum value.

The results of the proposed algorithm will be compared with the results of the KIT2-FCM, MKIT2-FCM with the number of the clusters is optimal obtained by GMKIT2-FCM and the initial centroids are randomly selected.

In addition, we measured the results on the basis of several validity indexes to assess the performance of the algorithms.

The optimal number of clusters C_{opt} is input of KIT2-FCM and MKIT2-FCM to cluster and compute validity indexes. We performed the different validity indexes such as the Bezdeks partition coefficient (PC-I), the Dunns separation index (Dunn-I), the Davies-Bouldins index (DB-I), and the Separation index (S-I), Xie and Beni's index (XB-I), Classification Entropy index (CE-I) [12], Turi's index (T-I) [9], Yuangang Tang's Index (YT-I) [11], CBW index (CBW-I).

The various validity indices are calculated with the optimal number of clusters that are shown in the Table II.

Because, the validity indexes are proposed to evaluate the quality of clustering. The better algorithm has smaller T-I, DB-I, XB-I, S-I, CE-I, YT-I, CWB-I and larger PC-I. The summarized results in Table II show that the GMKIT2-FCM (the proposed algorithm) have a better performance or higher quality clustering than the other typical algorithm such as

TABLE I
THE VALIDITY INDICES V_{CWB} ON THE PROPOSED ALGORITHM GMKIT2-FCM. THE OPTIMAL CLUSTERS C OBTAINED IS 3 FOR IRIS DATA AND WINE DATA, $C = 2$ FOR WDBC

Data	2	3	4	5	6	7	8	9	10	11
Iris	0.1523	**0.1504**	0.1652	0.2131	0.2016	0.3211	0.2101	0.4534	0.3722	0.2223
WDBC	**0.0012**	0.0015	0.0017	0.0018	0.0020	0.0019	0.0019	0.0021	0.0021	0.0023
Wine	0.0031	**0.0022**	0.0023	0.0037	0.0045	0.0046	0.0042	0.0044	0.00577	0.0051

TABLE II
THE VARIOUS VALIDITY INDICES ON WDBC DATA, WINE DATA AND IRIS DATA.

Validity Index	WDBC data ($C = 2$)			Wine data ($C = 3$)			Iris data ($C = 3$)		
	KIT2FCM	MKIT2FCM	GMKIT2FCM	KIT2FCM	MKIT2FCM	GMKIT2FCM	KIT2FCM	MKIT2FCM	GMKIT2FCM
T-I	0.8220	0.5021	**0.4351**	0.3975	0.2146	**0.1358**	0.3656	0.3571	**0.0145**
DB-I	4.1480	1.1650	**1.0222**	4.6594	0.6872	**0.5216**	2.9797	1.3240	**1.1224**
XB-I	0.0049	0.0025	**0.0018**	0.0105	0.0056	**0.0039**	0.0104	0.0103	**0.0039**
S-I	0.0487	0.0024	**0.0001**	0.1311	0.0732	**0.0008**	0.0139	0.0115	**0.0006**
CE-I	0.1609	0.1522	**0.0502**	0.3623	0.1813	**0.0705**	0.2101	0.0903	**0.0702**
PC-I	0.8709	0.9119	**0.9594**	0.7093	0.8163	**0.9375**	0.7806	0.9106	**0.9823**
YT-I	7.343	6.8627	**1.0001**	7.4300	6.2507	**2.0032**	5.4263	4.6144	**1.5501**
CWB-I	0.0018	0.0015	**0.0012**	0.0064	0.0031	**0.0022**	0.7467	0.3073	**0.1504**

KIT2-FCM and MKIT2-FCM. Besides, GMKIT2-FCM can automatically obtains the optimal number of clusters.

V. CONCLUSION

One of the priori inputs traditionally needed for clustering problems is the number of clusters in the data set. In many cases, however, the number of classes is not available. This paper proposed a GMKT2-FCM which automatically find the optimal number of clusters and determine the coefficients of the multiple kernel. The proposed algorithm provides us a new flexible approach to fuse different data information in the clustering problems. That is, different information represented by different kernels is combined in the kernel space to produce a new kernel and the ability to adjust the coefficients of the combined kernels to achieve the better results.

ACKNOWLEDGMENT

This paper is sponsored by Vietnam's National Foundation for Science and Technology Development (NAFOSTED)

REFERENCES

[1] C. Hwang, F. C. H. Rhee, Uncertain Fuzzy clustering: Interval Type-2 Fuzzy Approach to C-Means, IEEE Tran. on Fuzzy Systems, Vol. 15, 107-120, 2007.
[2] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, A. Y. Wu, An Efficient k-Means Clustering Algorithm: Analysis and Implementation, IEEE Trans. On Pattern Analysis and Machine Intelligence, Vol 24(7), 881-893, 2002.
[3] K.R. Zalik, An efcient k-means clustering algorithm, Pattern Recognition Letters Vol. 29, 1385 - 1391, 2008.
[4] J.M. Mendel and R.I. John, Type-2 fuzzy set made simple, IEEE Trans.Fuzzy Syst., vol. 10(2), 117 - 127, 2002.
[5] N. Karnik, J.M. Mendel, Operations on Type-2 Fuzzy Sets, Fuzzy Sets and Systems, 122, 327-348, 2001.
[6] M.R. Rezaee, B.P.F. Lelieveldt, J.H.C. Reiber, A new cluster validity index for the fuzzy c-mean, Pattern Recognition Letter Vol.19, 237 - 246, 1998.
[7] J. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms. New York: Plenum, 1981.
[8] M. Srinivas, Lalit M. Patnaik: Genetic Algorithms: A Survey. IEEE Computer, vol.27(6):17-26, 1994.
[9] S.J.Roberts,R. Everson, I.Rezek, Maximum certainty data partitioning, Pattern Recognition Vol.33:833-839, 2000.
[10] R.H.Turi, Clustering-Based Color Image Segmentation, PhD Thesis, Monash University, Australia, 2001.
[11] Y.Tang, F.Sun, Improved Validation Index for Fuzzy Clustering, American Control Conference, 2005.
[12] W. Wang,Y. Zhang, On fuzzy cluster validity indices, Fuzzy Sets and Systems 158,2095-2117, 2007.
[13] L. Chen, C. L. Philip Chen, A Multiple-Kernel Fuzzy C-Means Algorithm for Image Segmentation, IEEE Trans. on Sys. Man, and Cybernetics, Vol.41(5):1263-74, 2011.
[14] D. Graves, W. Pedrycz, Fuzzy C-Means, Gustafson-Kessel FCM, and Kernel-based FCM: a comparative study, Advances in Soft Computing Vol.41:140-149, 2007.
[15] H. Shen, J. Yang, S. Wang, X. Liu, Attribute weighted mercer kernel based fuzzy clustering algorithm for general non-spherical datasets, Soft Computing, vol. 10(11):1061-1073, 2006.
[16] S. Zhou, J. Gan, Mercer kernel fuzzy c-means algorithm and prototypes of clusters, Proc. of Conf. on Internat. Data Engineering and Automated Learning, Vol. 3177, 613-618, 2004.
[17] S. Sonnenburg, G. Ratsch, C. Schafer, B. Scholkopf, Large scale multiple kernel learning, J. of Machine Learning Research, vol.7:1531-1565, 2006.
[18] F. R. Bach, G. R. G. Lanckriet, M. I. Jordan, Multiple kernel learning, conic duality, and the SMO algorithm, Proc. 21st ICML, pp. 41-48, 2004.
[19] M. Girolami, Mercer kernel-based clustering in feature space, IEEE Transactions on Neural Networks, vol. 13(3):780784, 2002.
[20] D.Q.Zhang,S.C.Chen, Clustering incomplete data using kernel-based fuzzy c-means algorithm, Neural Processing Let.,Vol.18(3):155-162, 2003.
[21] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis. Cambridge University Press, 2004.
[22] B. Zhao, J. Kwok, C. Zhang, Multiple kernel clustering, The 9th SIAM International Conference on Data Mining, p.638-649, 2009.
[23] D. Graves, W. Pedrycz, Kernel-based fuzzy clustering and fuzzy clustering: A comparative experimental study, Fuzzy Sets and Systems, vol. 161(4):522-543, 2010.
[24] W. L. Cai., S. C. Chen., and D. Q. Zhang., Fast and Robust Fuzzy C-Means Clustering Algorithms Incorporating Local Information for image Segmentation, Pattern Recognition, vol. 40(3), pp. 825-838, 2007.
[25] R. Herbrich, Learning Kernel Classifiers, MIT Press, Cambridge, 2002.
[26] D.D.Nguyen, L.T.Ngo, L.T.Pham, Multiple Kernel Interval Type-2 Fuzzy C-Means Clustering, IEEE Conf. on Fuzzy Systems, 2013, submitted.
[27] Datasets: http://www.ics.uci.edu/ mlearn/mlrepository.html