FUZZY KOHONEN CLUSTERING NETWORKS

James C. Bezdek[1], Eric Chen-Kuo Tsao and Nikhil R. Pal*
Division of Computer Science
The University of West Florida
Pensacola, FL 32514

[1]Research supported by NSF Grant IRI-9003252
*on leave from Indian Statistical Institute, Calcutta, India

ABSTRACT

Kohonen networks are well known for cluster analysis (unsupervised learning). This class of algorithms is a set of heuristic procedures that suffers from several major problems (e.g., termination is forced, convergence is not guaranteed, no model is optimized by the learning strategy, and the output is often dependent on the sequence of data). In this paper we propose a fuzzy Kohonen clustering network which integrates the Fuzzy c-Means (FCM) model into the learning rate and updating strategies of the Kohonen network. This yields an optimization problem related to FCM, and our numerical results show improved convergence as well as reduced labeling errors. We prove that the proposed scheme is equivalent to the c-Means algorithms. The new method can be viewed as a Kohonen type of FCM, but is "self-organizing" since the "size" of the update neighborhood and learning rate in the competitive layer are automatically adjusted during learning. We use Anderson's IRIS data to illustrate this method; and compare our results with the standard Kohonen approach.

Keywords: Cluster Analysis, c-Means, Fuzzy Sets, Kohonen Networks, Self-Organization, Unsupervised Learning

1. INTRODUCTION

The area of classical pattern recognition most closely related to the Kohonen self-organizing [1] algorithms is known as cluster analysis. Clustering algorithms attempt to assess the interaction among patterns by organizing the patterns into clusters such that patterns within a cluster are more similar to each other than are patterns belonging to different clusters. Treatments of many classical approaches to this problem include the texts by Kohonen [1], Bezdek [2], Duda and Hart [3], Tou and Gonzalez [4], Hartigan [5], and Dubes and Jain [6]. Kohonen's work has become particularly timely in recent years because of the widespread resurgence of interest in the theory and applications of neural network structures [7]. However, Kohonen clustering networks (KCNs) suffer from several major problems. First, KCNs are heuristic procedures, so *termination* is not based on optimizing any model of the process or its data. Secondly, the final weight vectors usually depend on the input sequence. Thirdly, different initial conditions usually yield different results. Fourthly, several parameters of the KCN algorithms, such as the learning rate, the size of update neighborhood, and the strategy to alter these two parameters during learning, must be varied from one data set to another to achieve "useful" results.

It is well known that KCN clustering is closely related to the c-Means (CM) algorithms [8]. What is unknown, however, is just what "closely related" really means. Since CM algorithms are optimization procedures, whereas KCN is not, integration of CM and KCN is one way to address several problems of KCNs while simultaneously attacking the general problem of how the two families are related. Huntsberger and Ajjimarangsee [9] first considered this approach. This note extends their ideas to a new family of algorithms we shall call the Fuzzy Kohonen Clustering Network (FKCN) algorithms. We combine the ideas of fuzzy membership values for learning rates, the parallelism of Fuzzy c-Means (FCM), and the structure and update rules of KCNs. Moreover, we prove that each step of FKCN is equivalent to FCM or Hard c-Means (HCM). In addition, FKCN is self-organizing, since the "size" of the update neighborhood is automatically adjusted during learning, and FKCN usually terminates in such a way that the FCM objective function is approximately minimized. FKCN is non-sequential, and hence is independent of the sequence of feed of the input data. The neighborhood constraint of KCN is relaxed in FKCN, but is embedded in the learning rate strategy.

The remainder of this paper is organized as follows. In the next section, since the learning rate we use is based on fuzzy membership values from FCM, FCM and HCM are briefly described. The structure and update rule of KCN are also discussed. In section 3, we present the details of the FKCN model, and prove that FKCN is equivalent to either FCM or HCM. In Section 4, a comparison between FKCN and KCN is reported using Anderson's iris data. Finally, some remarks on future research are given to conclude this paper in Section 5.

2. FUZZY C-MEANS, HARD C-MEANS AND KOHONEN CLUSTERING NETWORKS

Let c be an integer, $1< c<n$, and let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ denote a set of n feature vectors in R^p. X is numerical object data, the j-th object has vector $\mathbf{x}_j$ as it's numerical representation, and x_{jk} is the k-th characteristic (or feature) associated with object j. Given X, we say that c fuzzy subsets $\{u_i : X \rightarrow [0,1]\}$ are a fuzzy c-partition of X in case the cn values $\{u_{ik} = u_i(x_k), 1\leq k\leq n, 1\leq i\leq c\}$ satisfy three conditions:

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

$$\Sigma_i u_{ik} = 1 \text{ for all } k ; \tag{1b}$$

$$\Sigma_i u_{ik} > 0 \ \forall i. \tag{1c}$$

Here u_{ik} is interpreted as the *membership* of $\mathbf{x}_k$ in the i-th partitioning subset (cluster) of X. If *all* of the u_{ik}'s are in {1,0}, $\mathbf{U} = [u_{ik}]$ is a conventional (crisp, hard) c-partition of X. The most well known objective function for clustering in X is the classical within groups sum of squared errors function, defined as :

$$J_1(U,v ; X) = \Sigma_i\Sigma_k\, u_{ik}\, (\|x_k - v_i\|)^2 , \tag{2}$$

where $v = (v_1, v_2, ..., v_c)$ is a vector of (unknown) cluster centers (weights or prototypes), $v_i \in R^p$ for $1 \leq i \leq c$, and U is a hard or conventional c-partition of X. Optimal partitions U* of X are taken from pairs (U*, v*) that are "local minimizers" of J_1. Dunn [10] first generalized (2) for m=2, and subsequently, Bezdek [2] generalized (2) to the infinite family written as:

$$J_m(U,v;X) = \Sigma_i\Sigma_k u_{ik}^{\,m} (\|x_k - v_i\|_A)^2 , \tag{3}$$

where $m \in [1, \infty)$ is a weighting exponent on each fuzzy membership, U is a fuzzy c-partition of X, $v = (v_1, v_2, ..., v_c)$ are cluster centers in R^p, A = any positive definite (p x p) matrix, and $\|x_k - v_i\|_A = (x_k - v_i)^T A\, (x_k - v_i)$ is the distance (in the A norm) from $\mathbf{x}_k$ to v_i. Our interest lies with the cases represented by equations (2) and (3). The conditions that are necessary for J_1 and J_m follow :

Hard c-Means (HCM) Theorem [2] (U,v) may minimize $\Sigma\Sigma\, u_{ik}(\, \|x_k - v_i\|_A)^2$ only if :

$$u_{ik} = \left| \begin{array}{l} 1;\ (\, \|x_k - v_i\|_A)^2 = \min_j \{(\|x_k - v_j\|_A)^2\} \\ \\ 0; \text{ otherwise} \end{array} \right| \quad \text{for all } i,k \tag{4a}$$

$$v_i = \Sigma u_{ik} x_k / \Sigma u_{ik} \quad \text{for all } i \tag{4b}$$

Note that the HCM produces a partition U that contains hard clusters. The well known generalization of HCM is contained in the following:

Fuzzy c-Means (FCM) Theorem [2] Assume $\|x_k - v_j\|_A^{\,2} > 0, \ \forall\ j,k$ at each iteration of (5); (U,v) may minimize $\Sigma\Sigma\, u_{ik}^{\,m}(\, \|x_k - v_i\|_A)^2$ for m>1 only if :

$$u_{ik} = (\Sigma\, (\, \|x_k - v_i\|_A \ / \ \|x_k - v_j\|_A)^{2/(m\,-1)})^{-1} \quad \text{for all } i,k \tag{5a}$$

$$v_i = \Sigma (u_{ik})^m x_k / \Sigma (u_{ik})^m \quad \text{for all } i \tag{5b}$$

It has been shown that conditions (5) $\rightarrow$ (4) and $J_m \rightarrow J_1$ as $m \rightarrow 1$ from above. The FCM (HCM) algorithms are iterative procedures for approximately minimizing J_m (J_1) by Picard iteration through (5) or (4), respectively. A brief specification of these procedures follows:

FCM/HCM Algorithms [2]

CM1. Fix : $1 \le c < n$; $1 < m < \infty$ (m=1 for HCM); $\| \ \|_A$; and $\varepsilon > 0$ some small positive constant.

CM2. Initialize network weight vector $v_0 = (v_{1,0}, v_{2,0}, ..., v_{c,0}) \in R^{cp}$.

CM3. For t = 1,2, ..., t_{max}.

a. Update all (cn) memberships $[u_{ik,t}]$ with (5a) or (4a).

b. Update all (c) weight vectors $\{v_{i,t}\}$ with (5b) or (4b).

c. Compute $E_t = \|v_t - v_{t-1}\|^2 = \Sigma_i \|v_{i,t} - v_{i,t-1}\|^2$.

d. If $E_t \le \varepsilon$ stop; Else Next t.

Notes on the c-Means Algorithms

1. These are non-sequential algorithms: updates on the weights $\{v_{i,t}\}$ are performed after each pass through X. Thus, the iterate sequence $\{v_{i,t}\}$ is independent of the data labels.
2. The parameter (m) essentially controls the "amount of fuzziness" in u. As $m \to \infty$, $u_{ik,t} \to 1/c$; when $m \to {}^+1$, $u_{ik,t} \to 1$ or 0. The performance of FCM is quite dependent on a good choice for m. Although several studies have attempted to find a good way to choose m [11,12], this choice is still largely heuristic (see [13] for a physical interpretation of FCM at m=2).
3. Ties in (4a) and singularity in (5a) are resolved arbitrarily.
4. The termination criterion E_t specified in CM3.c is sometimes replaced by $\|U_t - U_{t-1}\|^2$. The important point is that this strategy is designed to stop the procedure when successive *iterates show little change*. We emphasize that termination using this scheme may or may not stop the algorithm at a "desirable" solution pair (U_t, v_t). And we specifically reserve the word termination to mean "the place at which a finite numerical procedure stops.
5. The iterate sequence $\{U_t, v_t\}$, or a subsequence of it, converges q-linearly to either a local minimum or saddle point of J_m, (U_t, v_t) such that $E_t \le \varepsilon$ is an *estimate* of such a point. We emphasize particularly the difference between the words termination and convergence.

Kohonen clustering networks (KCNs) are unsupervised schemes which find the "best" set of weights for hard clusters in an iterative, sequential manner. The structure of KCN consists of two layers: an input (fanout) layer, and an output (competitive) layer as shown in Fig.1. The $\{v_{i,t}\}$ can be interpreted as "weights" attached to the edges that connect the p input nodes to the c output nodes. The aggregate of the c weight vectors (the *network* weight vector v_t) is adjusted during learning. Given an input vector, the neurons in the output layer compete among themselves and the winner (whose weight has the minimum distance from the input) updates its weights and those of some set of predefined neighbors. The process is continued until the weight vectors "stabilize." In this method, a learning rate must be defined which decreases with time in order to force termination. The update neighborhood must be defined and is also reduced with time. Asymptotic convergence of KCN schemes to a point that possesses same *property* such as fixed point, local minimum, etc. has not been established.

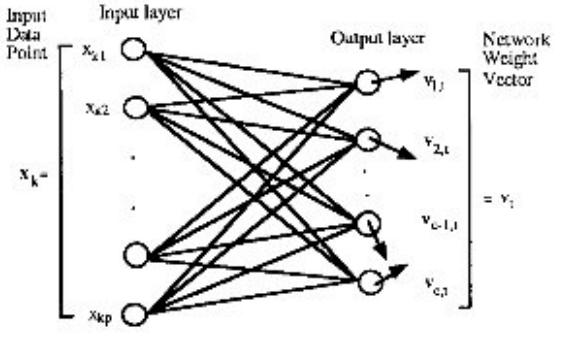


Fig. 1. The structure of a Kohonen clustering network.

A brief specification of KCN algorithms is given below :

Kohonen Clustering Network Algorithms [3]

KCN1. Fix c, $\|\ \|$, and $\varepsilon > 0$ some small positive constant.

KCN2. Initialize $v_0 = (v_{1,0}, v_{2,0}, \ldots, v_{c,0}) \in R^{cp}$, learning rate $\alpha_{ik,0} \in (1,0)$ and update neighborhood N_0 = an index set in {1,2, ..,c}.

KCN3. For $t = 1,2, \ldots, t_{max}$; For $k = 1,2, \ldots, n$:

a. Calculate $d_{ik}^2 = \|x_k - v_{i,t}\|^2$, for i = 1 to c.

b. Order the $\{d_{ik}^2\}$ in ascending order: $d_{1k}^2 \le d_{2k}^2 \le \ldots \le d_{ck}^2$.

c. Update the winner :

$$v_{1,t} = v_{1,t-1} + \alpha_{1k,t}(x_k - v_{1,t-1}) \qquad (6a)$$

d. If $|N_t| = n_t$, update the (n_t-1) nodes which are closest to x_k in b :

$$v_{i,t} = v_{i,t-1} + \alpha_{ik,t}(x_k - v_{i,t-1}),\ i=2,3, \ldots, (n_t-1) \qquad (6b)$$

e. Next k.

KCN4. Compute $E_t = \|v_t - v_{t-1}\|^2 = \Sigma_i \|v_{i,t} - v_{i,t-1}\|^2$.

KCN5. If $E_t \le \varepsilon$ stop; Else adjust learning rates $\{\alpha_{ik,t}\}$ and update neighborhood N_t; Next t.

Notes on the KCN Algorithms

1. As specified, KCN is sequential - one x_k is fed to the network, and some node weights are immediately updated. This makes the sequence $\{v_{i,t}\}$ label-dependent, a very undesirable property for any iterative algorithm.
2. We have indexed learning rate α on i,k, and t. This is because the values of α used in updating the node weights are distributed over N_t, "centered" at the winner, $v_{1,t}$, and the index of v_1 is a function of x_k. Moreover, α is usually decreased with time (t) to *force* $E_t \le \varepsilon$ (clearly E_t will $\to 0$ as $\alpha_{ik,t} \to 0$; this is an *artificial* termination strategy, in that the update rule guarantees closeness of successive iterates after sufficient time, even if they are not close to a "solution" vector.
3. Choosing the size of neighborhood, the learning rate and the strategies to operate these parameters is critical to termination of KCN. Moreover, different sets of parameters yield different results. Kohonen [1] has shown that this process converges, in the sense that the $\{v_t\} \to \{v_*\}$ as $\{\alpha\} \to 0$, but v_* is just a limit point of the iterate sequence - v_* is not optimal with respect to a model, or any well defined performance goal such as "error" rate.
4. KCN does not use or generate a partition U of the data during training. However, once the weight vectors stabilize, they can be clamped (fixed), and at this point the Kohonen's model is often used to produce a hard U using equation (4a) with $A = I_p$, the identity matrix on R^p.

3. FUZZY KOHONEN CLUSTERING NETWORKS

Recent studies have shown a close relationship between numerical results generated by KCN and FCM [9]. Huntsberger and Ajjimarangsee proposed the following modification of the KCN update rule (6):

$$v_{i,t} = v_{i,t-1} + u_{ik,t}(x_k - v_{i,t-1}), \qquad (7)$$

where $u_{ik,t}$ is calculated via (5a) for i = 1,2,...,n_t. There is no need to maintain U for (7), and in [9], neighborhood control was exerted as in KCN. Termination was forced on this scheme by ultimately shrinking the update neighborhood to the empty set. The scheme in [9] was a partial integration of KCN and FCM, which showed some interesting results. However, this was a hybrid scheme that fell short of realizing a *model* for KCN clustering; and no properties regarding termination or convergence were established. We complete the integration of FCM with KCN by defining the learning rate for Kohonen updating as :

$$\alpha_{ik,t} = (u_{ik,t})^{m_t} \quad ; \quad m_t = (m_0 - 1)/t_{max}, \qquad (8)$$

where $u_{ik,t}$ is calculated with equation (5a), but $m = m_t = (m_0 - t\Delta m)$, m_0 some positive constant greater than one; and $m_\infty = 1$. In practice, t cannot go to infinity, so we put $\Delta m = (m_0 - 1)/t_{max}$, where t_{max} is the iteration limit for Fuzzy KCN (FKCN). Defining the learning rate with (8) for $\alpha_{ik,t}$ yields the new clustering algorithm.

Fuzzy Kohonen Clustering Network (FKCN) Algorithms

FKCN1. Fix c, $\| \ \|_A$ and $\varepsilon > 0$ some small positive constant.

FKCN2. Initialize $v_0 = (v_{1,0}, v_{2,0}, \ldots, v_{c,0}) \in R^{cp}$. Choose $m_0 > 1$ and t_{max} = iterate limit.

FKCN3. For $t = 1, 2, \ldots, t_{max}$:

a. Compute all (cn) learning rates $\{\alpha_{ik,t}\}$ with (8) and (5a).

b. Update all (c) weight vectors $\{v_{i,t}\}$ with

$$v_{i,t} = v_{i,t-1} + [\sum_{k=1}^{n} \alpha_{ik,t}(x_k - v_{i,t-1})]/\sum_{s=1}^{n} \alpha_{is,t} \qquad (9)$$

c. Compute $E_t = \|v_t - v_{t-1}\|^2 = \Sigma_i \|v_{i,t} - v_{i,t-1}\|^2$.

d. If $E_t \le \varepsilon$ stop; Else Next t.

Notes on the FKCN Algorithms

1. It is well known [2] that the learning rates $\{\alpha_{ik,t}\} = \{(u_{ik,t})^{m_t}\}$ satisfy the following :

   a. $\lim_{m_t \to \infty}\{\alpha_{ik,t}\} = 1/c$ for all i and k;

   b. $\lim_{m_t \to +1}\{\alpha_{ik,t}\} = 1$ or 0 for all i and k;

   c. The learning rates $\{(u_{ik,t})^{m_t}\}$ are calculated with (5a), which, for fixed c, $\{v_{i,t}\}$ and m_t, have the following form for each x_k:

$$(u_{ik,t})^{m_t} = (a/d_{ik,t})^{(2m_t/(m_t-1))} \qquad (10)$$

   where a is a positive constant. Thus, the effect of (10) is to distribute the contribution of each x_k to the next update of the node weights inversely proportional to their distance from x_k. The "winner" in (9) is the $v_{i,t}$ closest to x_k, and it will be moved further along the line connecting $v_{i,t}$ to x_k than any of the other weight vectors. In the limit (m_t=1), the update rule reverts to hard c-Means (winner take all)-but non-sequentially. Constraint (1b) insures that each $\alpha_{ik,t} \le 1$, and this scheme thus amounts to distributing the partial updates across all c nodes for each $x_k \in X$.

2. KCN and Huntsberger's scheme [9] both manipulate the neighborhood N_t. In FKCN, $N_t = \{1, 2, \ldots, c\}$ $\forall t$, but the *effective* neighborhood does vary with t, because of note 1(a) and 1(b). Thus, for large values of m_t (near m_0), all c nodes are updated with lower individual learning rates, and as $m_t \to 1$, more and more of the unit sum is given to the "winner" node. In other words, the lateral distribution of learning rates is a function of t, which "sharpens" at the winner node (for each x_k) as $m_t \to^+ 1$.

3. FKCN is not sequential. Updates are done to all c nodes after each pass through X. Hence, FKCN is not label dependent.

4. Most importantly, for fixed m_t, FKCN updates the $\{v_{i,t}\}$ using the conditions that are necessary for FCM. Indeed, we may state as a:

   Proposition For fixed $m_t > 1$ (that is, $\Delta m = 0$) in (8), FKCN is FCM.

   Proof : Since (5a) is used to compute $\{\alpha_{ik,t}\}$ $\forall$ i and k, the explicit necessary conditions for the (next) half step of FCM are satisfied. It remains to be shown that the update rule in FKCN3.b is equivalent to (5b). Since

$$\left[\sum_{k=1}^{n} \alpha_{ik,t}(x_k - v_{i,t-1})\right] / \sum_{s=1}^{n} \alpha_{is,t}$$

$$= \sum_{k=1}^{n} \alpha_{ik,t} x_k / \sum_{s=1}^{n} \alpha_{is,t} - v_{i,t-1} \sum_{k=1}^{n} \alpha_{ik,t} / \sum_{s=1}^{n} \alpha_{is,t}$$

update rule (10) in FKCN becomes

$$v_{i,t} = (v_{i,t-1} - v_{i,t-1}) + \sum_{k=1}^{n} \alpha_{ik,t} x_k / \sum_{s=1}^{n} \alpha_{is,t} = \sum_{k=1}^{n} (u_{ik,t})^{m_t} x_k / \sum_{s=1}^{n} (u_{is,t})^{m_t}$$

which is (5b) for $m_t = m$. We see that taking $\Delta m=0$ in FKCN makes it FCM (with $m=m_0$). ■

5. Since equations (5) $\rightarrow$ (4) and $J_m \rightarrow J_1$ as $m_t \rightarrow^+ 1$, it is clear that at $m_t = 1$ with $\Delta m=0$, FKCN is hard c-Means. Given this network interpretation, we may call FKCN the Hard KCN (HKCN) algorithm when $m_0 = 1$ in FKCN. Of course, the HKCN algorithm computes the $\{\alpha_{ik,t}\}$ in FKCN3.a using (4a).
6. Although FKCN for each m_t is one step of FCM, the FKCN iterates sequence $\{v_t\}$ of node weights is *not*, even if we maintain U_t as well, the same as the FCM sequence, because m in (5) is varying as a function of time (iteration). FKCN is a true KCN type algorithm in that it does possess a well defined method for adjusting both the learning rate distribution and update neighborhood in the KCN approach as functions of time. Hence, FKCN has the "Self-organizing" structure of KCN, and at the same time, is stepwise optimal with respect to a well known and widely used fuzzy clustering model. How well does FKCN work? We illustrate some of its computational properties in the next section.

4. NUMERICAL EXAMPLE

We use Anderson's IRIS data [3] as an experimental data set. Let **X** be the IRIS data. **X** contains 50 (labeled) vectors in R^4 for each of c=3 classes of IRIS subspecies. Properties of the data are well known [3], **X** has been used in (conservatively !) at least 50 papers to illustrate various clustering (unsupervised) and classifier (supervised) designs. Typical error rates for supervised designs are 0-5 mistakes (resubstitution); and for unsupervised design , 10-15 "mistakes". Our comparison below are based on training KCN and FKCN with all of the data (until termination occurs or is forced); and then classifying each data point in **X** with the nearest prototype rule (4a) using Euclidean distance.

All clustering algorithms produce clusters that have *numerical* (not physical) labels. In KCN and FKCN, it is the c output nodes that have unidentified labels. We use the following algorithm to (re)label terminal weight vectors so that terminal prototypes correspond to the correct physical labels of the subclasses.

Relabeling Algorithm (RL Algorithm)

Given : n_i physically labeled vectors in $\mathbf{X}_i \subset R^p$; $1 \le i \le c$. Let $\Sigma_i n_i = n$.
$\{v_1, v_2, \ldots, v_c\}$ any (c) prototypes in R^p.
Find : Physical labels of the $\{v_i\}$.
Let : c_{ij} = number of points from class i closest to v_j via the rule (4a).
While : $i \le c$:
Find the maximum value in $C = [c_{ij}]$, say $c_{i^*j^*}$.
Relabel $v_{j^*} \rightarrow v_{i^*}$.
Delete row i* and column j* from C.
Wend.

We illustrate the RL algorithm as follows. Suppose, after C is constructed, we have

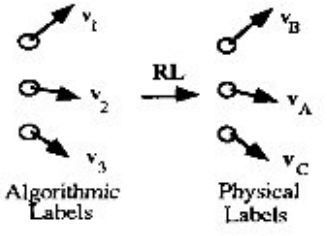
$$C = \begin{array}{c|ccc|c} & 1 & 2 & 3 & \\ \hline & 28 & 12 & 10 & A \\ & 46 & 3 & 1 & B \\ & 7 & 5 & 38 & C \end{array} , \qquad (11)$$

where {1,2,3} are the algorithmic labels on the prototypes, and {A,B,C} are the physical classes. Application of RL to (11) yields :

$$\begin{array}{c|ccc} & 1 & 2 & 3 \\ A & 28 & 12 & 10 \\ B & \mathbf{46} & 3 & 1 \\ C & 7 & 5 & 38 \end{array} \rightarrow \begin{array}{c|cc} & 2 & 3 \\ A & 12 & 10 \\ & & \\ C & 5 & 38 \end{array} \quad \text{with } 1 = B$$

$$\rightarrow \quad A \quad |12| \quad \text{with } 3 = C \text{ and } 2 = A.$$

Thus, the output node labels are transformed as in (12) :



(12)

We count "errors" by submitting **X** to (4a) using the physical labels of the output nodes. This affords a means for insuring that the prototypes really represent the "right" subclasses. In the example below, we initialized KCN and FKCN randomly, used only Euclidean distance, and fixed $\varepsilon = 0.0001$, $t_{max} = 50{,}000$ for KCN (t_{max} is a variable for FKCN, as discussed below). Generally, we can make these observations :

1. **Learning rate** α : We tried several strategies for $\alpha_{ik,t}$ with KCN. Each produced different results. Moreover, KCN never terminated unless $\alpha_{ik,t} \rightarrow 0$ (i.e., it was *forced* to stop at the predefined limit for t_{max}). On the other hand, FKCN always terminated in the sense that $E_t \leq \varepsilon$ was satisfied before $t = t_{max}$.
2. **Sequential data feed** : Since KCN is sequential, different sequences of feeding the data did alter the final results. However, FKCN accumulates Δv_t for each input and updates v_t after each pass through all the data. Thus, FKCN is parallel and is independent of the feeding sequence. It always stopped at the same set of node weights.
3. **Complexity** : Suppose t^* is the number of iterations for termination or the predefined number of iterations for stopping. Due to the sequential nature of KCN, its complexity is $O(t^*n)$, where n is the number of input data. However, because FKCN is non-sequential, its complexity is $O(t^*)$, independent of the number of input data.
4. **Termination** : We found that KCN always ran to its iterate limit ($t_{max} = 50{,}000$) unless $\alpha_{ik,t}$ was forced to zero. On the other hand, FKCN always satisfied the termination criterion in 14-40 iterations. Figure 2 shows error rates obtained by various algorithms as a function of time. In each case, we interrupted the algorithm at each t, ran (RL) on the current node weights, and counted the error rates. Note that KCN ran out to $t_{max} = 50{,}000$, whereas FKCN terminates in both cases shown at $t = 17$. Note also that FCM (m=4) and FKCN (m_0=4, m=0) generate exactly the same curve (as required by our proposition above). Finally, we observe that FKCN and FCM ran *past* the minimum error rate (t=10) to get their terminal states. This is the well-known "over-training" phenomena.

We also studied the stability of FKCN to both m_0 and Δm. Figure 3 shows FKCN error rate evolution with time for fixed m_0=4; variable $\Delta m = 0.02, 0.04, 0.06$. Evidently the trends of Figure 2 hold, and small changes in FKCN do not affect the results much (for this data).

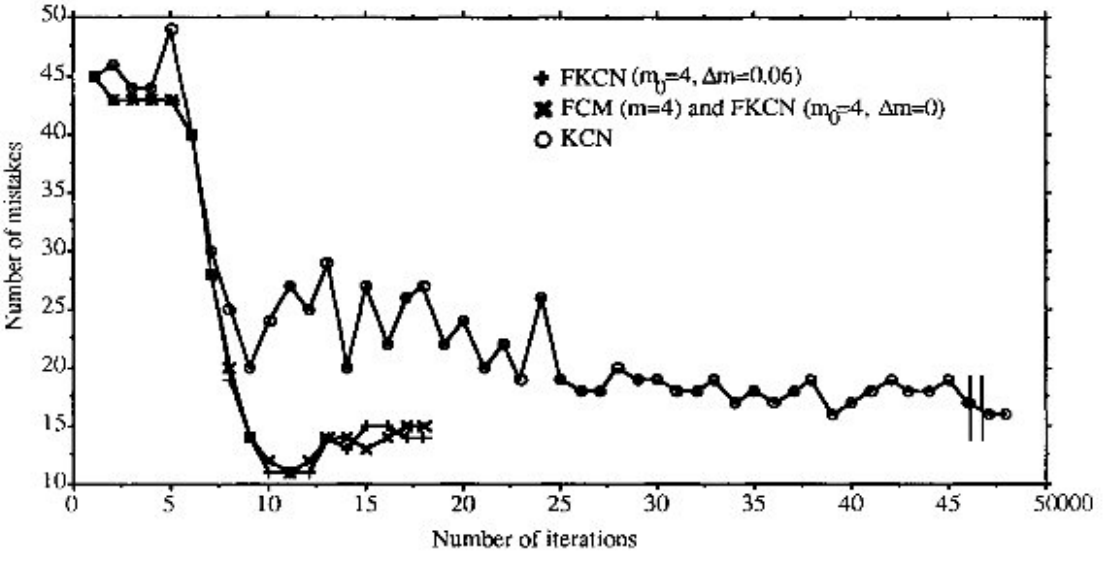


Fig. 2. Error rates with t for KCN, FKCN (m_0=4 and Δm=0.06), FKCN (m_0=4 and Δm=0) and FCM (m=4). Note the break in the horizontal axis between 45 and 50,000.

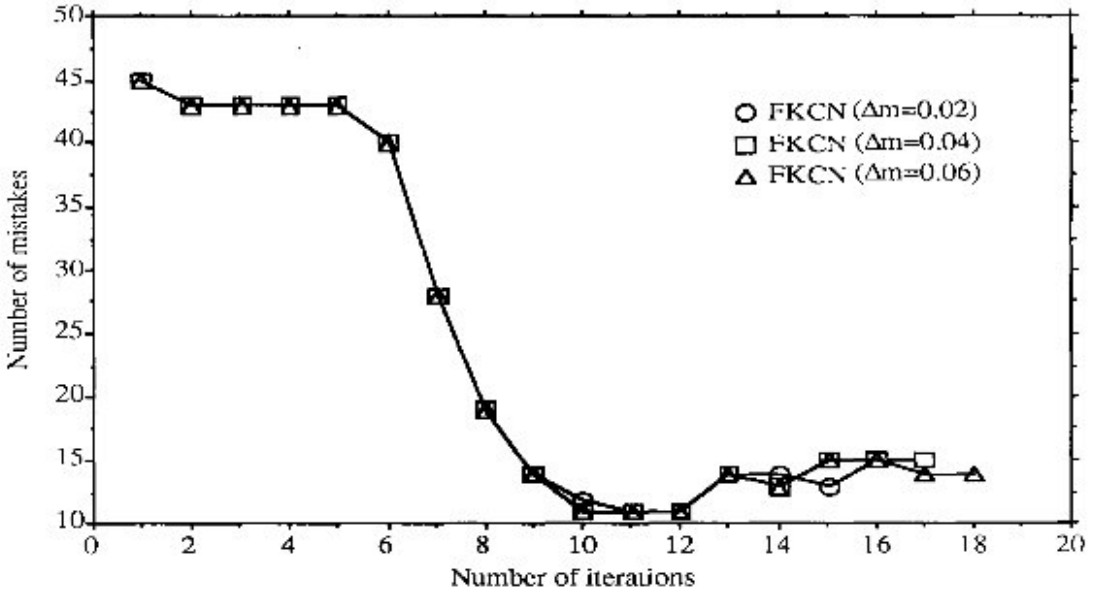


Fig. 3. FKCN error rates with t with m_0=4; and Δm = 0.02, 0.04, 0.06.

Figure 4 shows the reverse of the study in Figure 3: we hold Δm = 0.02 fixed, and vary m_0 = 3, 4, 5, 6. Again, the trend is clear.Figures 2, 3 and 4 all suggest that FKCN will terminate rapidly at a very predictable solution (10-13 errors); and that it is relatively insensitive to changes in m_0 and Δm. These results certainly encourage further study.

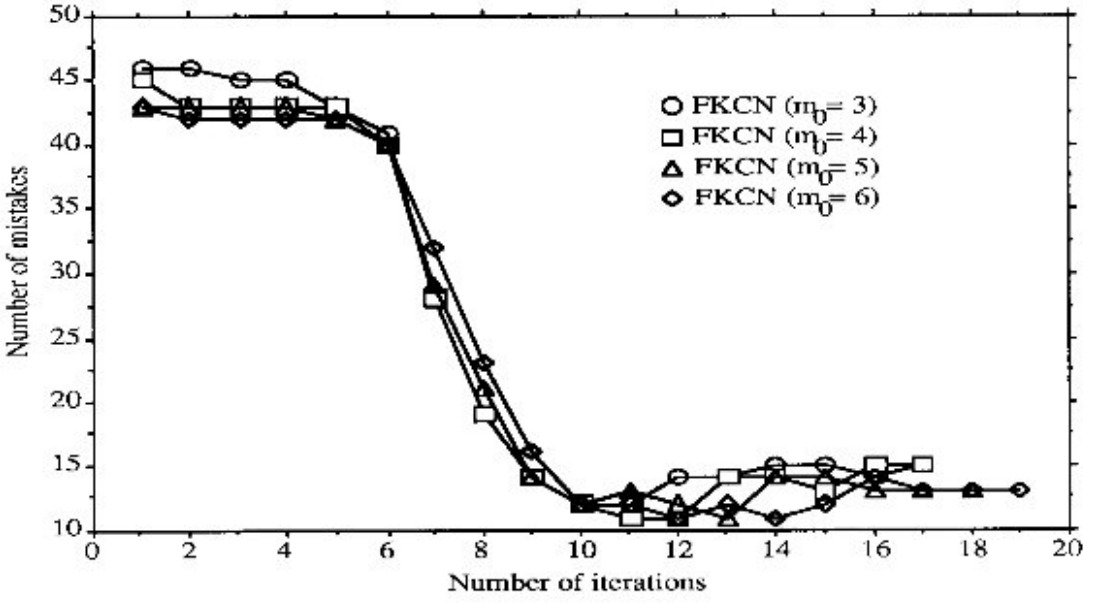


Fig. 4. FKCN error rates as a function of time with $\Delta m = 0.02$; and $m_0 = 3, 4, 5, 6$.

5. CONCLUSIONS

In this paper, a fuzzy Kohonen clustering network (FKCN) algorithm, based on the integration of Fuzzy c-Means (FCM) and Kohonen clustering network (KCN), is proposed. Our algorithm addresses some intrinsic problems of KCN. The FKCN is non-sequential, unsupervised, and uses fuzzy membership values from FCM as learning rates. This yields automatic control of both the learning rate distribution and update neighborhood. The KCN neighborhood constraint has been dropped, but is embedded in the learning rate which is manipulated by reducing m from a large value to 1. Moreover, FKCN always terminates independent of the sequence of feeding data in many less iterations. Finally, we proved that FKCN with fixed m is equivalent to FCM (and Hard c-Means (HCM) if m=1). Thus, FKCN can be considered as a Kohonen network implementation of FCM.

REFERENCES

[1] Kohonen, T. *Self-Organization and Associative Memory*, 3rd Edition, Springer-Verlag, Berlin, 1989.

[2] Bezdek, J. *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum, New York, 1981.

[3] Duda, R. and Hart, P. *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.

[4] Tou , J. and Gonzalez, R. *Pattern Recognition Principles*, Addison-Wesley, Reading, 1974.

[5] Hartigan, J. *Clustering Algorithms*, Wiley, New York, 1975.

[6] Dubes, R. and Jain, A. *Algorithms that Cluster Data*, Prentice Hall, Englewood Cliffs, 1988.

[7] Pao, Y.H. *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, Reading, 1989.

[8] Lippman, R. An Introduction to Neural Computing, *IEEE ASSP Magazine*, April, 1987, 4-22.

[9] Huntsberger, T. and Ajjimarangsee, P. Parallel Self-Organizing Feature Maps for Unsupervised Pattern Recognition, *Int'l. Jo. General Systems*, vol. 16, pp. 357-372,1989.

[10] Dunn, J. A Fuzzy Relative of the ISODATA Process and its Use in Detecting Compact, Well-Separated Clusters, J. Cybernetics, 3(3), 32-57, 1974.

[11] McBratney, A.B. and Moore, A. W. Application of Fuzzy Sets to Climatic Classification, *Ag. & Forest Meteor*, 35, 165-185, 1985.

[12] Choe, H. and Jordan, J. On the Optimal Choice of Parameters in a Fuzzy c-Means Algorithms, *Proc. First IEEE Conf. on Fuzzy Systems*, San Diego, 1992, in press.

[13] Bezdek, J, A Physical Interpretation of Fuzzy ISODATA, *IEEE Trans. SMC*, SMC-6(5), 387-389, 1976.