161812

N93-29528

IMAGE SEGMENTATION USING FUZZY LVQ CLUSTERING NETWORKS

Eric Chen-Kuo Tsao, James C. Bezdek* and Nikhil R. Pal
Division of Computer Science
The University of West Florida
Pensacola, FL 32514
*Research Supported by NSF Grant IRI-9003252

ABSTRACT

In this note we formulate image segmentation as a clustering problem. Feature vectors extracted from a raw image are clustered into subregions, thereby segmenting the image. A fuzzy generalization of Kohonen learning vector quantization (LVQ) which integrates the Fuzzy c-Means (FCM) model with the learning rate and updating strategies of the LVQ is used for this task. This network, which segments images in an unsupervised manner, is thus related to the FCM optimization problem. Numerical examples on photographic and magnetic resonance images are given to illustrate this approach to image segmentation.

1. INTRODUCTION

Image segmentation divides an image into regions with uniform and homogeneous attributes such as gray tone or texture [1]. Roughly speaking, conventional segmentation algorithms can be divided into two classes: region-based schemes, wherein areas of images with homogeneous properties are found, which in turn gives region boundaries [2-4]; and edge-based schemes, where local discontinuities are detected first, and then connected to form longer, hopefully complete, boundaries [5]. Image segmentation should result in regions that cover semantically distinct visual entities and is a crucial step for subsequent recognition or interpretation tasks.

Several image segmentation methods based on Markov Random Fields (MRFs) have been proposed. The basic idea is to model spatial interaction of the image features by a MRF which is a probability distribution defined over a discrete random field. Hongo *et al.* [6] proposed a "multiple level multiple resolution MRF" to detect the edges which was an extension of the work of Geman and Geman [7]. This model incorporates *a priori* knowledge about global structures in images, but can be implemented in a local (and parallel) mode. Three algorithms (simulated annealing, iterative conditional modes, and maximization of posterior marginals) are compared in [8]; all use MRF models to include prior contextual information. Most of these approaches use an energy function to guide image segmentation and numerical schemes for minimization of the energy functional. However, the search procedure for a global minimum (optimal solution) is usually time consuming. Moreover, edge-based segmentation schemes usually need a linking procedure to connect broken edges in order to make image subregions that have closed boundaries. Recently, several attempts to apply computational neural network architectures to image segmentation have been made. For example, edge detection has been formulated in the context of an energy-minimizing model by eliminating weak boundaries and small segments [9]; and also as a fuzzy feed-forward computational neural network problem [10]. A neural network system capable of detecting potential edges in various orientations that uses simulated and mean field annealing is discussed in [11].

In this note we propose using a new family of clustering algorithms called Fuzzy Learning vector Quantization (FLVQ) for image segmentation. FLVQ is a partial integration of Fuzzy c-Means (FCM) and Kohonen clustering networks (LVQs). The block diagram of the process is shown in Fig. 1. Unlabeled feature vectors (one for each pixel) are first extracted from an image. Then FLVQ clusters these feature vectors to get cluster centers. Each cluster center is regarded as a prototype (or vector quantizer) of some subregion of the image. Finally, each pixel feature vector is compared to the cluster centers, and is assigned a constant value corresponding to the closest cluster center. Note that the number of constant values is the same as the number of clusters.

Figure 1. FLVQ Image Segmentation: Overall Architecture.

The remainder of this paper is organized as follows. In the next section, we briefly review the FCM, LVQ and FLVQ algorithms. In Section 3, experimental segmentation results on photographic and Magnetic Resonance images are reported. Section 4 contains a discussion, conclusions, and some ideas for future research.

2. KOHONEN CLUSTERING NETWORKS

Many classical clustering algorithms can be found in the texts of Duda and Hart [12], Hartigan [13], and Jain and Dubes [14]. In [15] Lippman suggested that Kohonen's learning vector quantization (LVQ) [16] is closely related to the sequential Hard c-Means (HCM) algorithm. Fuzzy c-Means (FCM) is a well known generalization of HCM [17,18]. Since HCM/FCM are optimization procedures, whereas LVQ is not, integration of FCM and LVQ is one way to address several problems of LVQs while simultaneously attacking the general problem of how the two families are related. Huntsberger and Ajjimarangsee [19] first considered this approach, and their idea was extended in [20] to the FLVQ algorithms described below.

Let c be an integer, $1< c<n$, and let $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$ denote a set of n feature vectors in $\Re^p$. 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 constrained fuzzy c-partition of X in case the cn values $\{u_{ik} = u_i(\mathbf{x}_k),\ 1\le k\le n,\ 1\le i\le c\}$ satisfy three conditions:

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

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

$$0 < \Sigma_i\, u_{ik} < n \ \forall i. \tag{1c}$$

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}, $U = [u_{ik}]$ 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 :

$$J_1(U,\mathbf{v} ; X) = \Sigma_i\Sigma_k\, u_{ik}\, ||\mathbf{x}_k-\mathbf{v}_i||^2 , \tag{2}$$

where $\mathbf{v} = (\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_c)$ is a vector of (unknown) cluster centers (weights, prototypes, or vector quantizers), $\mathbf{v}_i \in \Re^p$ for $1 \le i \le c$, and U is a hard or conventional c-partition of X. Optimal partitions U^* of X are taken from pairs $(U^*, \mathbf{v}^*)$ that are "local minimizers" of J_1. Dunn [18] first generalized (2) for m=2, and subsequently, Bezdek [17] generalized (2) to the infinite family written as:

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

where $m \in [1, \infty)$ is a weighting exponent on each fuzzy membership, U is a fuzzy c-partition of X, $\mathbf{v} = (\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_c)$ are cluster centers in $\Re^p$, **A**= any positive definite (p x p) matrix, and $||\mathbf{x}_k-\mathbf{v}_i||_A = (\mathbf{x}_k-\mathbf{v}_i)^T A\, (\mathbf{x}_k-\mathbf{v}_i)$ is the distance (in the **A** norm) from $\mathbf{x}_k$ to $\mathbf{v}_i$. Conditions that are necessary for

extrema of J_1 and J_m follow : **Hard c-Means (HCM) Theorem [17]** (U,v) may minimize $\Sigma\Sigma\, u_{ik}(\,\|\mathbf{x}_k - \mathbf{v}_i\|_A)^2$ only if :

$$u_{ik} = \begin{cases} 1; & (\|\mathbf{x}_k - \mathbf{v}_i\|_A)^2 = \min_j\{(\|\mathbf{x}_k - \mathbf{v}_j\|_A)^2\} \\ 0; & \text{otherwise} \end{cases} \quad (4a)$$

$$\mathbf{v}_i = \sum u_{ik}\mathbf{x}_k \,/\, \sum u_{ik} \quad (4b)$$

In the context of image segmentation, equation (4a) will be used to assign each (pixel) vector $\mathbf{x}_k$ to its closest prototype $\mathbf{v}_i$; this is the essence of our segmentation scheme. 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 [17]** Assume $\|\mathbf{x}_k - \mathbf{v}_j\|_A^2 > 0$, $\forall j,k$ at each iteration of (5); (U,v) may minimize $\Sigma\Sigma\, u_{ik}^m(\,\|\mathbf{x}_k - \mathbf{v}_i\|_A)^2$ for m>1 only if :

$$u_{ik} = (\sum(\|\mathbf{x}_k - \mathbf{v}_i\|_A / \|\mathbf{x}_k - \mathbf{v}_j\|_A)^{2/(m-1)})^{-1} \quad (5a)$$

$$\mathbf{v}_i = \sum(u_{ik})^m \mathbf{x}_k \,/\, \sum(u_{ik})^m \quad (5b)$$

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. C-Means algorithms are non-sequential algorithms: updates on the weights $\{\mathbf{v}_{i,t}\}$ are performed after each pass through X. Thus, iterate sequence $\{\mathbf{v}_{i,t}\}$ is independent of the sequence of the data labels. The parameter (m) essentially controls the "amount of fuzziness" in U. As $m \rightarrow \infty$, $u_{ik,t} \rightarrow 1/c$; when $m \rightarrow^{+} 1$, $u_{ik,t} \rightarrow 1$ or 0.

Kohonen clustering networks (LVQs) are unsupervised schemes which find the "best" set of prototypes (for hard clusters) in an iterative, sequential manner. The structure of LVQ consists of two layers: an input (fanout) layer, and an output (competitive) layer as shown in Fig. 2. The edges that connect the p input nodes to the c output nodes do not have "weights" attached to them, as, for example, in a feed forward network architecture. Instead, each output node has a prototype (vector quantizer) attached to it, and it is this set of network weight vectors that are adjusted during learning. A formal description of LVQ is given below. There are other versions of LVQ; this one is usually regarded as the "standard" form.

The LVQ Clustering Algorithm [16]

LVQ1. Given unlabeled data set $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n\} \subset \Re^p$. Fix c, T, and $\varepsilon > 0$.

LVQ2. Initialize $\mathbf{V}_0 = (\mathbf{v}_{1,0}, \ldots, \mathbf{v}_{c,0}) \in \Re^{cp}$, and learning rate $\alpha_0 \in (1,0)$.

LVQ3. For t = 1,2, ..., T;

For k = 1,2, ..., n:

a. Find $\left\|\mathbf{x}_k - \mathbf{v}_{i,t-1}\right\| = \min_{1\le j\le c}\left\{\left\|\mathbf{x}_k - \mathbf{v}_{j,t-1}\right\|\right\}$. (6)

b. Update the winner : $\mathbf{v}_{i,t} = \mathbf{v}_{i,t-1} + \alpha_t(\mathbf{x}_k - \mathbf{v}_{i,t-1})$ (7)

Next k.

d. Apply the 1-NP (nearest prototype) rule to the data :

$$u_{LVQ_\alpha} = \begin{cases} 1; & \left\|\mathbf{x}_k - \mathbf{v}_i\right\| \le \left\|\mathbf{x}_k - \mathbf{v}_j\right\|,\ 1 \le j \le c, j \ne i \\ 0; & \textit{otherwise} \end{cases}, \quad 1\le i\le c \text{ and } 1\le k\le n. \quad (8)$$

e. Compute $E_t = \left\| \mathbf{V}_t - \mathbf{V}_{t-1} \right\|_1 = \sum_{r=1}^{c} \left\| \mathbf{v}_{r,t} - \mathbf{v}_{r,t-1} \right\|_1 = \sum_{k=1}^{n} \sum_{r=1}^{c} \left| v_{rk,t} - v_{rk,t-1} \right|$.

f. If $E_t \leq \varepsilon$ stop; Else adjust learning rate α_t;

Next t

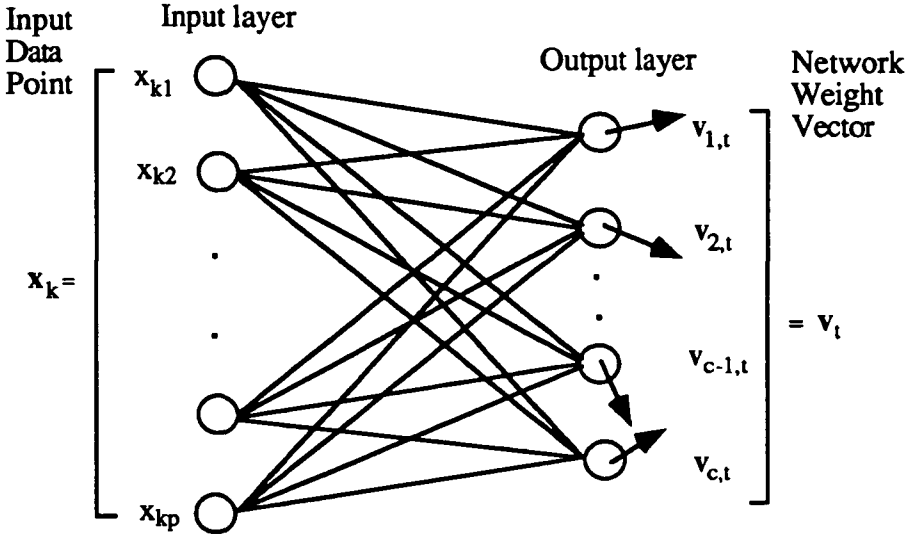

Figure 2. The structure of a Kohonen clustering network.

The numbers $U_{LVQ} = \left[u_{LVQ_{ik}} \right]$ at (8) are a cxn matrix that almost always (constraint (1c) may not be satisfied) define a hard c-partition of X using the 1-NP classifier assignment rule at (4). Our inclusion of computation of the hard 1-NP c-partition of X at the end of each pass through the data (step LVQ3.d) is **not** part of the LVQ algorithm - that is, the LVQ iterate sequence does not depend on cycling through U's. Ordinarily this computation is done once, non-iteratively, outside and after termination of LVQ. Note that LVQ uses the Euclidean distance in step LVQ3.a. This choice corresponds roughly to the update rule shown in (7), since $\nabla_{\mathbf{v}}(\|\mathbf{x} - \mathbf{v}\|_I^2) = -2I(\mathbf{x} - \mathbf{v}) = -2(\mathbf{x} - \mathbf{v})$. The origin of this rule assumes that each $\mathbf{x} \in \Re^p$ is distributed according to a probability density function $f(\mathbf{x})$. LVQ's objective is to find a set of $\mathbf{v}_i$'s to minimize the expected value of the square of the discretization error :

$$E\left(\left\| \mathbf{x} - \mathbf{v}_i \right\|^2 \right) = \int \int \ldots \int_{\Re^p} \left\| \mathbf{x} - \mathbf{v}_i \right\|^2 f(\mathbf{x}) d\mathbf{x} \qquad (9)$$

In this expression $\mathbf{v}_i$ is the winning prototype for each $\mathbf{x}$, and will of course vary as $\mathbf{x}$ ranges over $\Re^p$. A sample function of this optimization problem is $e = \left\| \mathbf{x} - \mathbf{v}_i \right\|^2$. An optimal set of $\mathbf{v}_i$'s can be approximated by applying local gradient descent to a finite set of samples drawn from f. The extant theory for this scheme is contained in [21], which states that LVQ converges in the sense that the prototypes $\mathbf{V}_t = (\mathbf{v}_{1,t}, \mathbf{v}_{2,t}, \ldots, \mathbf{v}_{c,t})$ generated by the LVQ iterate sequence converge, i.e., $\{\mathbf{V}_t\} \xrightarrow{t \to \infty} \hat{\mathbf{V}}$, provided two conditions are met by the sequence $\{\alpha_t\}$ of learning rates used in (7) :

$$\sum_{t=0}^{\infty} \alpha_t = \infty \qquad : \qquad \text{and} \qquad \sum_{t=0}^{\infty} \alpha_t^2 < \infty \quad . \tag{10}$$

One choice for the learning rates that satisfies these conditions is the harmonic sequence $\alpha_t = 1/t$ for $t \geq 1$; $\alpha_0 \in (0,1)$. Kohonen has shown that (under some assumptions) steepest descent optimization of the average expected error function (9) is possible, and leads to update rule (7). The update scheme at (7) has the simple geometric interpretation shown in Figure 3.

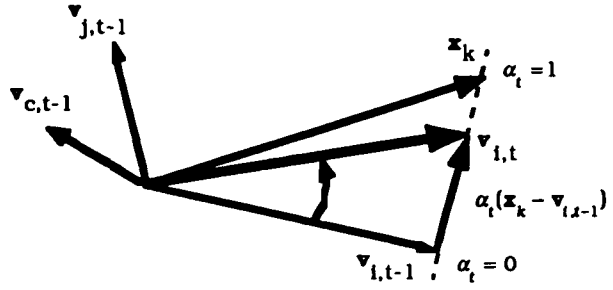

Figure 3. Updating the winning LVQ Prototype.

The winning prototype at iteration t, $\mathbf{v}_{i,t-1}$, is simply rotated towards the current data point by moving along the vector $(\mathbf{x}_k - \mathbf{v}_{i,t-1})$ which connects it to $\mathbf{x}_k$. The amount of shift depends on the value of a "learning rate" parameter α_t, which varies from 0 to 1. As seen in Figure 3, there is no update if α_t=0, and when α_t=1, $\mathbf{v}_{i,t}$ becomes $\mathbf{x}_k$ ($\mathbf{v}_{i,t}$ is just a convex combination of $\mathbf{x}_k$ and $\mathbf{v}_{i,t-1}$). This process continues until termination via LVQ3.f, when the terminal prototypes yield a "best" hard c-partition of X via (8).

Comments on LVQ : **(1)** Kohonen in [21] mentions that LVQ converges to a unique limit if and only if conditions (10) are satisfied. However, nothing was said about what sort or *type* of points the final weight vectors produced by LVQ are. Since LVQ does not model a well defined property of clusters (in fact, LVQ does not maintain a partition of the data at all), the fact that $\{\mathbf{V}_t\} \xrightarrow{t \to \infty} \hat{\mathbf{V}}$ does not insure that the limit vector $\hat{\mathbf{V}}$ is a good set of prototypes in the sense of representation of clusters or clustering tendencies. **(2)** The termination strategy at LVQ3.e is based on small successive changes in the cluster centers. This method of algorithmic control offers the best set of centroids for compact representation (quantization) of the data in each cluster. However, LVQ seldom terminates in less than, say, 20,000 iterates unless $\alpha_t \to 0$: this forces it to stop because successive iterates are necessarily close. **(3)** LVQ often runs to its iterate limit, and sometimes passes the optimal (clustering) solution in terms of minimal apparent label error rate. This is called the "over-training" phenomenon in the neural network literature.

Huntsberger and Ajjimarangsee [19] combined the 1-NP rule at (4) with Self-Organizing Feature Maps (SOFMs) to develop clustering algorithms. Algorithm 1 in [19] is the SOFM algorithm with an additional layer of neurons that does not participate in weight updating. After the self-organizing network terminates, the additional layer, for each input, finds the weight vector (prototype) closest to it and assigns the input data point to that class. A second algorithm in their paper used the necessary conditions for FCM to assign a membership value in [0,1] to each data point for each of the c classes. Specifically, Huntsberger and Ajjimarangsee suggested fuzzification of SOFM by replacing the learning rates $\{\alpha_{ik,t}\}$ usually found in rules such as (7) with fuzzy membership values $\{u_{ik,t}\}$ computed with the FCM formula [17]:

$$\alpha_{ik,t} = u_{ik,t} = \left(\sum_{j=1}^{c} \frac{D_{ikA,t}}{D_{jkA,t}} \right)^{\frac{-2}{m-1}} \quad , \qquad m>1, \tag{11}$$

where $D_{ikA,t} = \left\| \mathbf{x}_k - \mathbf{v}_{i,t} \right\|_A$. Numerical results reported in Huntsberger and Ajjimarangsee suggest that in many cases their algorithms and standard LVQ produce very similar answers. Their scheme was a partial integration of LVQ with FCM that showed some interesting results. However, it fell short of realizing a *model* for fuzzy LVQ clustering; and no properties regarding terminal points or convergence were established. Moreover, since the objective of LVQ is to find cluster centers (prototypes) in $\Re^p$, the need for and use of the topological ordering idea of (images of) the weight vectors in display space is not well justified. Consequently, the approach taken in [19] seems to mix two objectives, feature mapping and clustering, and the overall methodology is difficult to interpret in either sense.

Integration of FCM with LVQ can be more fully realized by defining the learning rate for Kohonen updating as :

$$\alpha_{ik,t} = (u_{ik,t})^{m_t} = \left(\sum_{j=1}^{c} \frac{D_{ik,t}}{D_{jk,t}} \right)^{\frac{-2m_t}{m_t - 1}}, \qquad \text{where} \qquad (12a)$$

$$m_t = m_0 + t[(m_f - m_0)/T] = m_0 + t\Delta m \; ; \qquad m_f, m_0 \geq 1; \qquad t=1,2,\ldots T. \qquad (12b)$$

m_t replaces the (fixed) parameter m in (11). This results in three families of *Fuzzy LVQ* or FLVQ algorithms, the cases arising by different treatments of parameter m_t. In particular, for $t \in \{1,2,\ldots,T\}$, we have three cases depending on choice of the initial (m_0) and final (m_f) values of m:

1. $m_0 > m_f \Rightarrow \{m_t\} \downarrow m_f$: Descending FLVQ (13a)
2. $m_0 < m_f \Rightarrow \{m_t\} \uparrow m_f$: Ascending FLVQ (13b)
3. $m_0 = m_f \Rightarrow m_t \equiv m_0 \equiv m$: FLVQ $\equiv$ FCM (13c)

Cases 1 and 3 are discussed at length in [20]. Equation (13c) asserts that when $m_0 = m_f$, FLVQ reverts to FCM; this results from defining the learning rates via (12a), and using them in the update rule for the prototypes shown in FLVQ3.b below. We provide a formal description of FLVQ :

Fuzzy LVQ (FLVQ) [20]

FLVQ1. Given unlabeled data set $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$. Fix c, T, $\| \; \|_A$ and $\varepsilon > 0$.

FLVQ2. Initialize $\mathbf{v}_0 = (\mathbf{v}_{1,0}, \ldots, \mathbf{v}_{c,0}) \in \Re^{cp}$. Choose $m_0, m_f \geq 1$.

FLVQ3. For t = 1, 2, ..., T.

a. Compute all (cn) learning rates $\{\alpha_{ik,t}\}$ with (12).

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

$$\mathbf{v}_{i,t} = \mathbf{v}_{i,t-1} + \sum_{k=1}^{n} \alpha_{ik,t}(\mathbf{x}_k - \mathbf{v}_{i,t-1}) / \sum_{s=1}^{n} \alpha_{is,t}$$

c. Compute $E_t = \left\| \mathbf{v}_t - \mathbf{v}_{t-1} \right\| = \sum_{i=1}^{c} \left\| \mathbf{v}_{i,t} - \mathbf{v}_{i,t-1} \right\|$.

d. If $E_t \leq \varepsilon$ stop; Else

Next t.

Observe that FLVQ is *not* a direct fuzzy generalization of LVQ because it does not revert to LVQ in case all of the $u_{ik,t}$'s are either 0 or 1 (the crisp case). Instead, if $m_0 = m_f = 1$, FCM reverts to HCM, and the HCM prototype update formula, which *is* driven by finding unique winners, as in LVQ, is a different formula than (7). Nonetheless, FLVQ is perhaps the closest possible link between LVQ and

c-Means type algorithms. For fixed c, $\{\mathbf{v}_{i,t}\}$ and m_t, the learning rates $\alpha_{ik,t} = (u_{ik,t})^{m_t}$ at (12a) satisfy the following :

$$\alpha_{ik,t} = (u_{ik,t})^{m_t} = \left(\frac{\kappa}{D_{ik,t}}\right)^{\frac{2m_t}{m_t-1}} \quad , \qquad (14)$$

where κ is a positive constant. Apparently the contribution of $\mathbf{x}_k$ to the next update of the node weights is inversely proportional to their distances from it. The "winner" is the $\mathbf{v}_{i,t-1}$ closest to $\mathbf{x}_k$, and it will be moved further along the line connecting $\mathbf{v}_{i,t-1}$ to $\mathbf{x}_k$ than any of the other weight vectors. Since $\sum u_{ik,t} = 1 \Rightarrow \sum \alpha_{ik,t} \leq 1$, this amounts to distributing partial updates across all c nodes for each $\mathbf{x}_k \in X$. This is in sharp contrast to LVQ, where only the winner is updated for each data point.

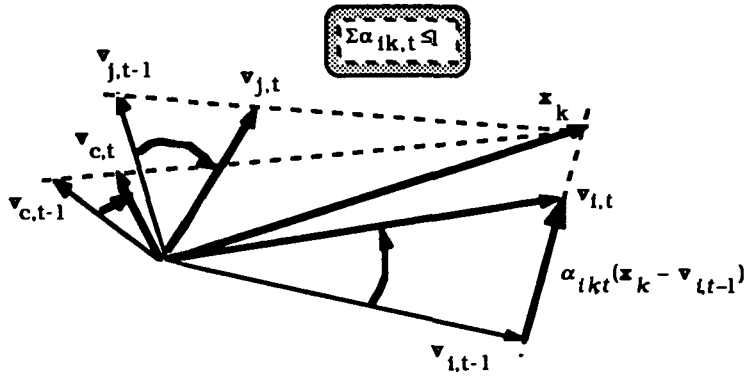

Figure 4. Updating Feature Space Prototypes in FLVQ Clustering Nets.

Figure 4 illustrates the update geometry of FLVQ; note that *every* node is (potentially) updated at every iteration, and the sum of the learning rates is always less than or equal to 1, an added constraint on the overall movement of the c prototypes at each t. In *descending* FLVQ (13a), for large values of m_t (near m_0), all c nodes are updated with lower individual learning rates, and as $m_t \rightarrow m_f$, more and more of the update is given to the "winner" node. In other words, the lateral distribution of learning rates is a function of t, which in the descending case "sharpens" at the winner node (for each $\mathbf{x}_k$) as $m_t \rightarrow m_f$.

Comments on FLVQ : (1) In contradistinction to Huntsberger and Ajjimarangsee's approach, there is no need to choose an update neighborhood . Neighborhood control is automatic, and depends entirely on the relative geometry of the data and their prototypes. **(2)** Reduction of the learning coefficient with distance (either topological or in $\Re^p$) from the winner node is not required. Instead, reduction is done automatically and adaptively by the learning rules. **(3)**The greater the mismatch to the winner (i.e., the higher the quantization error), the *smaller* the impact to the weight vectors associated with other nodes (recall (14)). **(4)**The learning process attempts to minimize a well-defined objective function (stepwise). This procedure depends on generation of a fuzzy c-partition of the data, so it is an iterative clustering model - indeed, stepwise, it is exactly fuzzy c-means [20]. **(5)** Our termination strategy is based on small successive changes in the cluster centers. This method of algorithmic control offers the best set of centroids for compact representation (quantization) of the data in each cluster.

3. EXPERIMENTAL RESULTS

In this section we discuss an application of *ascending* FLVQ to segmentation of intensity and Magnetic Resonance (MR) images. Success of a clustering technique as a tool for image segmentation depends largely on the choice of useful feature vectors. We first discuss the application of FLVQ to segmentation of light intensity images, and then to MR images. For a digital intensity image, every pixel is usually represented by a feature vector derived from pixel statistics like the mean, standard deviation, edginess and so on, computed over a small neighborhood (window) about the pixel under consideration. In this note we illustrate FLVQ using very simple feature vectors obtained by arranging pixels into an array. For a pxp (p is odd) neighborhood, the p^2 pixels will be arranged into a linear array in a systematic manner, starting from the top left corner of the window. For example, for a 3x3 window the feature vector corresponding to pixel (i,j), with clockwise traversal of pixels from location (i-1,j-1), takes the following form :

$$\mathbf{x}_{ij} = [f(i-1,j-1), f(i-1,j), f(i-1,j+1), f(i,j+1), f(i+1,j+1), f(i+1,j), f(i+1,j-1), f(i,j-1), f(i,j)]^T$$

This kind of feature vector has the advantage that it accounts for spatial details of local gray levels and requires no computation for feature vector generation. This choice has disadvantages as well. For example, permutation of the same set of gray values over a window will generate different feature vectors, which may in turn lead to different results. This problem can be circumvented by sorting the gray values. In this investigation, we implemented the FLVQ algorithm with three different window sizes; 1x1, 3x3 and 5x5. The computing protocols are summarized in Table 1.

	norm	c	m_0	Δm	t_{max}	ε	iterations
Fig. 5(b)	Euclidean	6	1.05	0.2	200	0.05	17
Fig. 5(c)	Euclidean	6	1.05	0.2	200	0.05	24
Fig. 5(d)	Euclidean	6	1.05	0.2	200	0.05	29
Fig. 6(c)	Euclidean	6	1.05	0.2	200	0.05	22

Table 1. Protocols for the Computational Experiments

Fig. 5(a) An intensity image

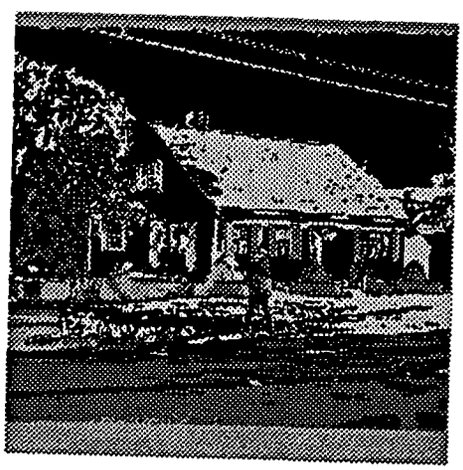

Fig. 5(b) Segmentation result using 1x1

Figure 5(a) depicts the intensity image of a house. Figures 5 (b), (c) and (d) represent segmentation results produced by window of sizes 1x1, 3x3 and 5x5, respectively. Note that Figure 5(b) is too detailed, in a sense "noisy". This is so because the 1x1 window does not take into account the spatial distribution of gray levels, but only histogram information; in fact, in some sense, this is

equivalent to histogram thresholding. Comparison of Figure 5(c) with 5(b) reveals that the roof and the walls of the house are better segmented by the 3x3 window. On the other hand, Figure 5(d) contains more compact segmented regions; even the textured tree is segmented as compact homogeneous regions. This shows that too small a window may result in too many details, while too large a window may smooth out much relevant information. Probably a reasonably good compormise is a neighborhood of size 3x3.

Fig. 5(c) Segmentation result using 3x3

Fig. 5(d) Segmentation result using 5x5

If q images are correlated in the sense that they are perfectly registered because they are taken in different bands, pixel vectors of size q can be erected at each spatial site by simply aggregating the intensity across bands. This amounts to a multichannel version of the 1x1 window. Magnetic Resonance Imagery, e.g. typically generates 3 bands, namely, T1 relaxation (spin lattice), T2 relaxation (transverse), and ρ (proton density). At pixel site (i,j), MRI data can thus result in 3 dimensional pixel vectors, say $\mathbf{x}_{ij} = (T1_{ij}, T2_{ij}, \rho_{ij})$. This $\mathbf{x}_{ij}$ can then be used a feature vector for segmentation of the MR image. Figures 6(a) and 6(b) show two bands (ρ and T2) of one physical slice of an human head. Fig. 6(c) depicts the segmentation obtained using FLVQ with the parameters shown in the last row of Table 1. It is well-known that comparison of image segmentation algorithms is not an easy task [8]. However, one of the most important criteria for performance evaluation is whether the algorithm can outline the desired or important components in the image. For instance, in Fig. 6(c), our segmentation delineates the white and gray matter tissue regions quite well.

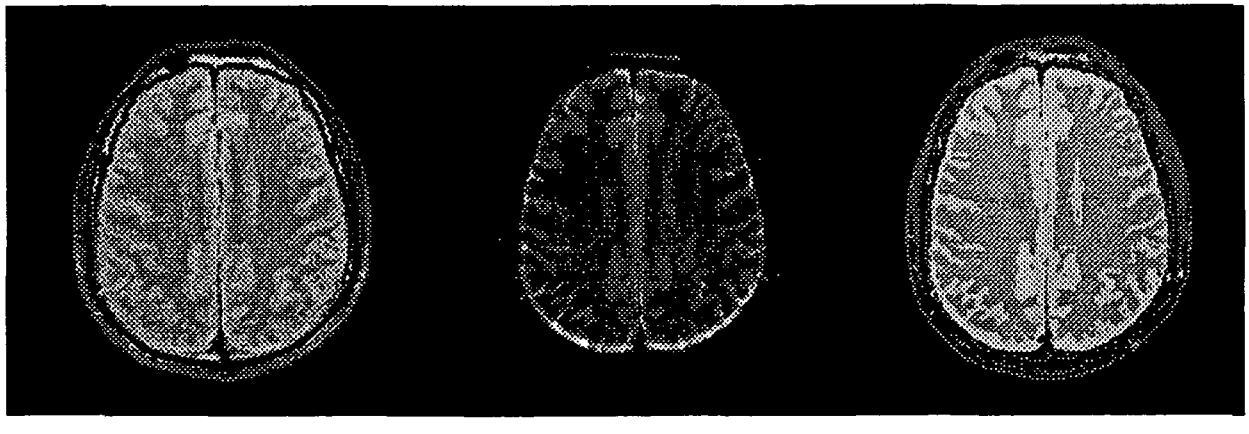

Fig. 6(a) ρ MR data

Fig. 6(b) T2 MR data

Fig. 6(c) FLVQ Segmentation

4. CONCLUDING REMARKS

In this paper a family of Fuzzy generalization of LVQ (FLVQ) algorithms based on the integration of Fuzzy c-Means and Kohonen clustering networks have been used for image segmentation. FLVQ 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. Light intensity and MR images have been segmented using various feature extraction strategies; our results seem encouraging, but much remains to be done.

REFERENCES

[1] R. M. Haralick and L. G. Shapiro, "Survey: image segmentation techniques," *Computer Vision, Graphics, and Image Processing*, vol.29, pp.100-132, 1985.

[2] B. Bhanu and B. A. Parvin, "Segmentation of natural scenes," *Pattern Recognition*, vol.20, pp.487-496, 1987.

[3] J. R. Beveridge, J. Griffith, R. R. Kohler, A. R. Hanson, and E. M. Riseman, "Segmentation image using localized histogram and region merging," *International Journal of Computer Vision*, vol.2, pp.311-347, 1989.

[4] T. Pavlidis and Y.-T. Liow, "Integrating region growing and edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.12, pp.225-233, 1990.

[5] W. A. Perkins, "Area segmentation of image using edges points," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.2, pp.8-15, 1980.

[6] S. Hongo, M. Kawato, T. Inui, and S. Miyake, "Contour extraction of image on parallel computer-local, parallel and stochastic algorithm which learns energy parameters," *Proceedings of the International Joint Conference on Neural Networks*, vol.1, pp. 161-168, 1990.

[7] S. Geman and D. Geman, Stochastic relaxation, "Gibbs distribution and the Bayesian restoration of images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.6, pp. 721-741, 1984.

[8] R. C. Dubes, A. K. Jain, S. G. Nadabar and C. C. Chen, "MRF model-based algorithms for image segmentation,"*Proceedings of the International Conference on Pattern Recognition*, pp. 808-814, 1990.

[9] J. Shah, "Parameter estimation, multiscale representation and algorithms for energy-minimizing segmentation," *Proceedings of the International Conference on Pattern Recognition*, pp. 815-819,1990.

[10] D. Kerr and J. C. Bezdek, "Edge Detection Using Neural Netwoks," SPIE, 1992.

[11] C. Cortes and J. A. Hertz, "A network system for image segmentation," *Proceedings of the International Joint Conference on Neural Networks*, vol. 1, pp. 121-125, 1989.

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

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

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

[15] R. Lippman, "An introduction to neural computing," *IEEE ASSP Magazine*, April, pp. 4-22, 1987.

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

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

[18] J. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact, well-separated clusters," *J. Cybernetics*, vol. 3, pp. 32-57, 1974.

[19] T. Huntsberger and P. Ajjimarangsee, "Parallel self-organizing feature maps for unsupervised pattern recognition," *Int'l. Jo. General Systems*, vol. 16, pp. 357-372, 1989.

[20] J. Bezdek, E. C-K. Tsao and N. Pal, "Fuzzy Kohonen Clustering Networks," *Proceeding of the IEEE International Conference on Fuzzy Systems*, pp.1035 -1043, San Diego 1992.

[21] T. Kohonen, Self-organizing maps : optimization approach, Artificial neural networks, elsevier Sc. Pub., (Eds. T. Kohonen, K. Makisara, O. Simula and J. Kangas), 981-990, 1991.