AN ADAPTIVE INITIALIZATION TECHNIQUE FOR COLOR QUANTIZATION BY SELF ORGANIZING FEATURE MAP

Chip-Hong Chang, Rui Xiao, and Thambipillai Srikanthan
Center for High Performance Embedded Systems, Nanyang Technological University,
Blk N4, Nanyang Avenue, Singapore 639798

ABSTRACT

Unsupervised learning network such as Self Organizing Feature Map (SOFM) has been applied successfully to color classification for image compression and pattern recognition. Like other vector quantization algorithms, the reconstruction quality and adaptation rate of the SOFM are sensitive to the neuron initialization. In this paper, we propose an efficient new initialization method, whereby an excess number of neurons is defined and the neurons are adaptively pruned, merged and splitted within their lattice according to the spatial distribution of the input color pixels. Comparisons with conventional gray scale initialization using subsampling and butterfly jumping sequences show the proposed method obtains good initial code vectors that can accelerate the convergence of the SOFM and improve the reconstructed image quality significantly.

1. INTRODUCTION

Color is one of the most significant attributes of an object on which humans and intelligent vision systems rely to perform its discrimination. Reduction of the image colors aids segmentation, compression, display and transmission of images [1, 2, 8-10]. The frequently used techniques for color reduction are based on nearest color merging, which are in essence, a codebook based approach [3-6]. A color codebook is a table consists of references to a limited number of colors. By nearest color merging, each pixel in the image is mapped to a color in the codebook that matches closest to its true color based on some distortion metric. The optimal selection of a smaller number of representative colors to form a codebook for an image of higher color resolution is called color quantization [1, 2, 8-10].

A number of approaches have been suggested for the design of an optimal quantizer to seek the codebook that minimizes the average distortion over all possible codebooks [1-10]. Among which, the use of Kohonen's self-organizing feature map (SOFM) [2, 7, 8, 10] has become popular due to its inherent massively parallel computing structure and the ability to adapt to the input data through unsupervised learning. The competitive learning algorithms used by the SOFM are gradient descent in nature; hence the network convergence and performance are sensitive to the initialization of the neurons. In this paper, we propose a new initialization method for upgrading the performance SOFM or a broader class of learning vector quantization algorithms. In our method, a list of nodes corresponds to an oversize codebook is defined and the cluster membership count of each node is updated along with the input data. The number of nodes is reduced to the size of the desired codebook by pruning, merging and splitting operations. In the process, the weight vectors of the nodes are modified to capture the spatial distribution of the input data. Experimental results indicate significant performance improvement when the SOFM initialized by the weight vectors of the nodes is used for color quantization.

2. LVQ AND GLA ALGORITHMS

Learning vector quantization (LVQ) is the term used to classify unsupervised learning algorithms associated with a competitive neural network [4, 5]. Kohonen's SOFM is a special subset of LVQ that incorporates a winner-take-all strategy with an architecture amenable to VLSI implementation. If we consider vector quantization as a constrained minimization problem of partitioning M feature vectors to $N < M$ codebook vectors, the optimization procedure in SOFM is in principle comparable to an iterative refinement clustering algorithm known as generalized Lloyd algorithm (GLA) [3-6]. In both algorithms, a gradient descent approach is commonly used to minimize the generalized mean distortion, D defined as:

$$D = \frac{1}{M}\sum_{i=1}^{M}\left(\frac{1}{N}\sum_{j=1}^{N}\left\|x_i - v_j\right\|^r\right)^{\frac{1}{r}} \tag{1}$$

where $x_i \in \mathbb{R}^n, 1 \le i \le M$, $v_j \in \{v_1, v_2, \cdots, v_N\} \subset \mathbb{R}^n$ and $r \in \mathbb{R} - \{0\}$. In most practical application, $r = 2$.

Let $v_c = v_c(t)$ be the closest codebook vector to the input vector, $x = x(t)$ in the n-dimension Euclidean space, $\mathbb{R}^n$. The following steepest decent gradient sequence in discrete time formalism asymptotically minimize the distortion of (1) based on square-error criterion ($r = 2$):

$$\begin{aligned} v_c(t+1) &= v_c(t) + \alpha(t)\left\|x(t) - v_c(t)\right\| \\ v_i(t+1) &= v_i(t) \quad for\ i \neq c \end{aligned} \tag{2}$$

where $\alpha(t)$ is a monotonically decreasing sequence of scalar-value gain coefficients, $0 < \alpha(t) < 1$.

For a given partition of the feature space, each codebook vector must be the centroid of the input vectors that are mapped to it. The way the centroid condition is satisfied is perhaps the most significant difference between LVQ and GLA algorithms. LVQ begins with an initial set of N topologically connected neurons whose values represent the codebook vectors, where N is the desired number of codebook vectors. Each time an input vector is presented, the neurons compete for the opportunity to

update their values. Only the winner and its topologically related neighbors will reduce its distortion relative to the input vector according to (2). On the other hand, GLA algorithm decouples the processes of nearest neighbors' assignment and centroid updating. An initial set of N or fewer codebook vectors are chosen arbitrarily or based on some evenly spaced points in the input vector space. Each input vector is first assigned to its best-matched codebook vector through an exhaustive search. Then, the codebook vector is modified to minimize its distortion with respect to the set of input vectors assigned to it. The two-step process iterates until the overall distortion of (1) changes by a small fraction between two iterations. Some GLA algorithms begin with a single codebook vector and use a splitting technique [3, 6] to expand the codebook size until the desired number of codebook vectors is reached.

3. NEW INITIALIZATION TECHNIQUE

The reliance on alternating projections makes LVQ and GLA algorithms susceptible to the local minimum problem due to inappropriately chosen initialization. It has been observed that both the convergence rate and reconstruction quality based on the converged codebook depends heavily on the initial codebook vectors [6]. To improve the performance of codebook-based quantization, we propose an adaptive initialization (AI) method as oppose to fixed and randomized initializations for LVQ algorithm. For ease of exposition, we illustrate the concept by a typical application of color quantization based on Kohonen SOFM.

We first define some notations. The input vector is represented by $x_i = [r, g, b]$ where $r, g, b \in \{0, 1, \ldots, 255\}$ correspond to the red, green and blue tristimulus intensities of the color value. The RGB colorspace is partitioned into a three-dimensional lattice structure with $Q^3 = Q \times Q \times Q$ nodes uniformly distributed on the grid intersections. Each node, v has as attributes a weight vector, w_v, a topology vector ς_v and a membership counter, f_v. $Q^3 = \beta N$ where N is the final codebook size and β a scaling factor to modulate the disparity between pixel clusters associated with adjacent nodes.

The topology vector of each node is assigned according to its coordinate values along the R, G and B axes in the 3D RGB colorspace, i.e, $\varsigma = [i, j, k]$ where $0 \le i, j, k < Q$. Initially, each node is allocated a weight vector proportional to its topology vector. For a node v with $\varsigma_v = [i, j, k]$, $w_v = [\text{round}(255i/(Q-1)), \text{round}(255j/(Q-1)), \text{round}(255k/(Q-1))]$, where round($a$) produces an integer b such that $|b-a| \le 0.5$. The membership counters of all nodes are initialized to 0. The membership counters are updated through one pass of all the input training pixels, $x_i = [r_i, g_i, b_i]$, $i = 1, 2, \ldots, M$. The counter, f_v is incremented if an input vector falls within the cube of length $256/(Q-1)$ centered at node v. It is easy to determine the nearest node to the input vector as the topology vector of the best match node, c can be computed from the input vector by:

$$\varsigma_c = \left[round\left(\frac{r_i(Q-1)}{255}\right), round\left(\frac{g_i(Q-1)}{255}\right), round\left(\frac{b_i(Q-1)}{255}\right)\right] \quad (3)$$

Fig. 1 shows an example of the lattice structure of $3 \times 3 \times 3$ nodes ($Q = 3$) in the RGB space and an input pixel $x = [48, 198, 130]$ under the shaded cubic region of influence centered at the node with $\varsigma = [0, 2, 1]$. It should be noted that the nearest node to the input pixel can be identified and its membership counter incremented without the need for an exhaustive search. If a decision tree or a link list is used to arrange the nodes, the nodes can be sorted in ascending order of membership count as the input pixels are presented.

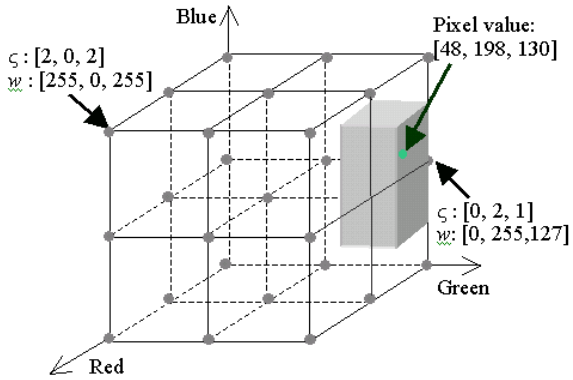

Figure 1. A lattice structure of 27 nodes.

After one pass of M input training pixels, a list of nodes sorted in order of its membership count is obtained. Three operations, namely delete, merge and split are applied to reduce the total number of nodes down to the desired codebook size N. Nodes with $f = 0$ are deleted from the list. Merge operation combines a low membership node with an adjacent high membership node, thus reducing the number of nodes by one. Split operation inserts a node between two adjacent high membership nodes. A node u is said to be the neighbor of node v if the topological adjacency, $\|\varsigma_v - \varsigma_u\| \le 1$ is satisfied. Consider the two adjacent nodes u and v with $w_v = [w_{v1}, w_{v2}, w_{v3}]$ and $w_u = [w_{u1}, w_{u2}, w_{u3}]$. If $f_v < f_u$, upon merging, node v will be deleted from the list and node u replaced by u' with the membership count and weight vector given by:

$$f_{u'} = f_u + f_v$$
$$w_{u'} = \frac{w_u f_u + w_v f_v}{f_u + f_v} \quad (4)$$

Fig. 2 illustrates the merge operation where node v is a low membership node and node u is the densest node among the six neighborhood of v.

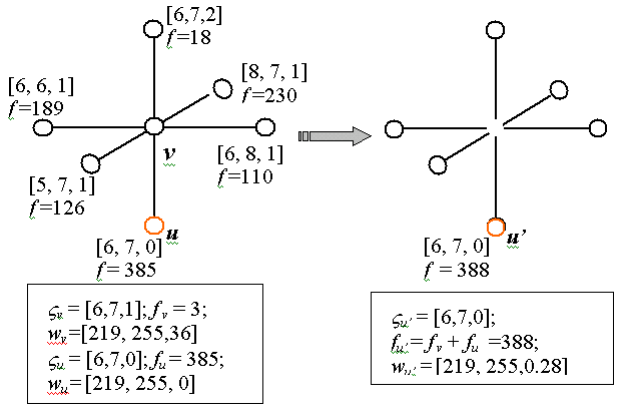

Figure 2. Merge operation.

In split operation, a new node t is generated between two adjacent nodes u and v chosen for splitting. The membership count and the weight vector of the newly generated node, t are given by:

$$f_t = f_v' = f_u' = \frac{1}{3}(f_u + f_v) \tag{5}$$

$$w_t = \frac{w_u f_u + w_v f_v}{f_u + f_v}$$

The membership counts of u and v are modified to that of t but their weight vectors remain unchanged. The effect of splitting is exemplified in Fig. 3. It should be noted that the topology vector of any new node generated from the merge and split operation can be calculated from its weight vector by (3).

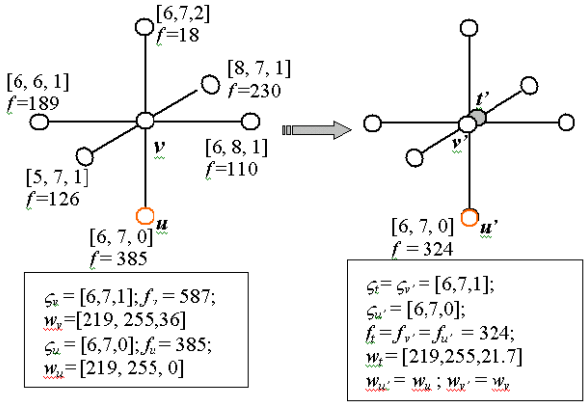

Figure 3. Split operation.

Assume a sorted list, L of Q nodes is obtained after one pass of the input pixels, the procedure to reduce the cardinality of L to N is described as follows:

(1) Delete all nodes with zero membership count.

(2) If $|L| > N$, select a node $v \in L$ such that $f_v = \min_{i \in L} f_i$. Select a node u from the neighborhood of v with the highest membership count, apply a merge operation between nodes u and v. Delete nodes u and v and insert the merged node u' with its attributes determined by (4) into the sorted list, i.e., $L = L - \{u, v\} + \{u'\}$. If there is no node with adjacency ≤ 1 to node v, delete node v if $f_v < \tau$, where τ is a predefined threshold value to differentiate between noises and clusters of minority colors that can be preserved.

(3) If $|L| < N$, select a node $v \in L$ such that $f_v = \max_{i \in L} f_i$. Select a node u from the neighborhood of v with the highest membership count, apply a split operation to generate a new node u' and modified the attributes of all nodes involved according to (5). Insert u' into the sorted list, i.e., $L = L + \{u'\}$. If there is no node with adjacency ≤ 1 to node v, the next higher membership node of L is sought and the same split operation is executed.

(4) Repeat Steps (2) and (3) until $|L| = N$.

The weight vectors of the final list of nodes will be used to initialize the N neurons of the SOFM. In order to discriminate in favor of the vital colors while preserving minority colors when the neuron number exceeds the number of major colors, τ is set equal to the membership counter value of the Nth node in the list sorted in descending order of f.

4. PERFORMANCE EVALUATION

To show the effectiveness of our new adaptive initialization method (AI) in helping the SOFM to achieve better reconstructed quality as well as faster convergence in color quantization, a set of experiments are carried out on different sizes of SOFM initialized by our proposed AI ($\beta = 8$) and the widely adopted gray scale initialization (GSI) [7, 8, 10]. Three 24-bit true color images, Lena, Pepper and Baboon taken from the USC image database are used to train the network. All the images are of size 512×512 pixels.

The test images are fed into the initialized SOFM in two ways: a raster scan sequence with subsampling (SUB) and a butterfly jumping sequence (BF) from [8]. By subsampling, the test image is divided into blocks of 8×8 pixels and one pixel per block is fed into the SOFM in a raster scan order. The next pixel from each block is fed when all pixels of the same intra-block coordinate have exhausted. The BF sequence keeps maximally distinct data fed in the neural network. The purpose of testing with different input sequences is to evaluate if our AI method can also help to desensitize the performance of SOFM from the ordering of data input.

For (2), we use the l_1 norm and the learning rate function suggested by [8], which can be re-expressed in our context as:

$$\alpha(t) = \alpha(0) k^{t \bmod s} \tag{6}$$

where $\alpha(0)$ and k are both set to 0.85, and s is the sweep number defined as the ratio of the image size to the block size (skip factor) of the subsampling or butterfly sequence. In our case, s = 512×512/(8×8) = 4096.

The Peak Signal to Noise Ratios (PSNRs) of the images reconstructed from the codebooks generated by the SOFM initialized by AI and GSI methods are compared. The PSNR is defined as:

$$PSNR = 10 \log \frac{3 \times 255^2}{MSE} \tag{7}$$

$$MSE = \frac{\sum_{i=0}^{M-1} (x_i - x_i')^2}{M}$$

where x_i and x_i' are the pixel values of the original and the reconstructed images, respectively and M is the total number of pixels. The fractional drop of distortion [4], $\Delta D < 0.001$ in two consecutive training epochs is adopted as the convergence criterion, where each epoch consists of 8s pixels. The fractional drop of distortion, ΔD is given by:

$$\Delta D = \frac{|MSE_{i-1} - MSE_i|}{MSE_i}, \; i = t \bmod 8s \tag{8}$$

The PSNR results of the three test images reconstructed from the codebooks generated by AI and GSI initialized SOFMs are plotted in Fig. 4 for SUB input sequence and Fig. 5 for BF sequence, respectively. The codebook size ranges from 16 to 256 and the PSNRs shown are taken after the networks have converged.

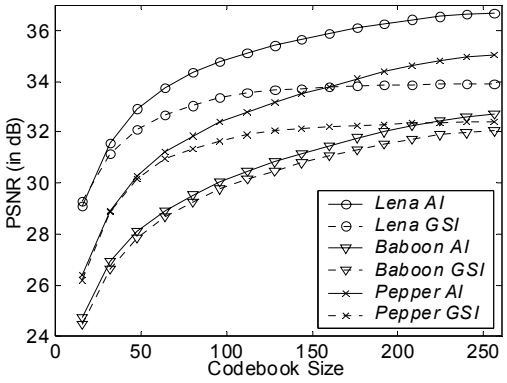

Figure 4. Comparison of PSNRs for Lena, Baboon and Pepper images for AI and GSI methods in subsampling SOFM.

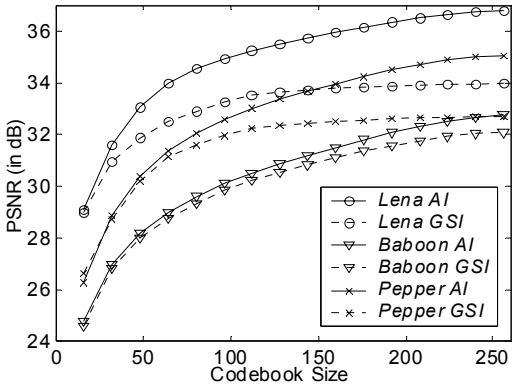

Figure 5. Comparison of PSNRs for Lena, Baboon and Pepper images for AI and GSI methods in BF SOFM.

It is evident that the proposed new initialization technique (AI) produces better codebooks with higher PSNRs for SOFM with SUB and BF input sequences, indicating the effectiveness of the initialization method is relatively unaffected by the order of input data. The improvement in the reconstruction quality is significant when the network size is large, particularly for smooth images like Lena and Pepper. A conspicuous 3dB performance gain is achieved over the GSI SOFM for these images when the codebook size exceeds 150. In general, the AI SOFM converges an epoch (equivalent to 1/8 of the total training time) ahead of the GSI SOFM.

(a) by AI method (b) by GSI method

Figure 6. Subjective evaluation of the reconstructed Lena in 256-neuron SOFM with SUB input sequence.

The reconstructed Lena using AI and GSI SOFM of 256 neurons are shown in Fig. 6 for a subjective evaluation. In regions of slow color transition, abrupt color changes are noticeable in the image reconstructed from codebook generated by GSI SOFM whereas the AI SOFM preserves the subtle color features of those regions.

5. CONCLUSION

A simple and efficient method (AI) for codebook initialization has been proposed. The method is most suitable for augmenting the performance of LVQ and GLA algorithms in color classification applications. It uses the M input data to update the density information of βN evenly distributed cluster centroids over the feature space in $O(M)$ computational complexity. The number of clusters is subsequently reduced to N codebook vectors by pruning useless clusters, and merging and splitting adjacent clusters. The overall computational complexity is $O(M + \beta N)$ which is much lower than $O(MN)$ complexity of GLA initialization method of [6] since $\beta << M$. Simulation results comparing the performance of SOFM initialized by the new AI scheme and the conventional gray scale initialization have demonstrated the effectiveness of our new scheme.

6. REFERENCES

[1] J.P. Braquelaire and L. Brun, "Comparison and optimization of methods of color image quantization," *IEEE Trans. Image Process*., vol. 6, no. 7, pp. 1048-1052, 1997.

[2] C.H. Chang, R. Xiao and T. Srikanthan, "A MSB-biased self-organizing feature map for still color image compression," in *Proc. IEEE Asia Pacific Conf. on Ciruits and Syst.*, 2002.

[3] W. Equitz, "A new vector quantization clustering algorithm," *IEEE Trans. Acoustics, Speech and Signal Proc.*, vol. 37, no. 10, pp. 1568-1575, 1989.

[4] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic, 1992.

[5] N.B. Karayiannis, "Soft learning vector quantization and clustering algorithms based on ordered weighted aggregation operators," *IEEE Trans. On Neural Networks*, vol. 11, no. 5, pp. 1093-1105, 2000.

[6] I. Katsavounidis, C.-C. J. Kuo and Z. Zhang, " A New Initialization technique for generalized Lloyd iteration," *IEEE Signal Process. Lett.*, vol. 1, no. 10, pp. 144-146, 1994.

[7] T. Kohonen, "The self-organizing map," *Proc. of the IEEE*, vol. 78, no. 9, pp. 1464-1479, 1990.

[8] S.C. Pei and Y.S. Lo, "Color image compression and limited display using self-organization Kohonen map," *IEEE Trans. Circuits Syst. Video Techno*., vol. 8, no. 2, pp. 191-205, 1998.

[9] G. Sharma and H. J. Trusell, "Digital color imaging," *IEEE Trans. Image Process*., vol. 6, no. 7, pp. 901-932, 1997.

[10] R. Xiao, C.H. Chang and T. Srikanthan, "On the initialization and training methods for Kohonen self-organizing feature maps in color image quantization," in *Proc. 1st Int. Workshop on Electronic Design, Test and Applications*, Christchurch, New Zealand, pp. 321-325, 2002.