A New Side-Match Finite-State Vector Quantization Using Neural Networks for Image Coding¹

Yu-Len Huang and Ruey-Feng Chang

Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan 62107, Republic of China
E-mail: rfchang@cs.ccu.edu.tw

Received March 17, 1998; accepted March 9, 2000

The side-match finite-state vector quantization (SMVQ) schemes improve performance over the vector quantization by exploiting the neighboring vector correlations within the image. In this paper, we propose a neural network side-match finite-state vector quantization (NN-SMVQ) scheme that combines the techniques of neural network prediction and the SMVQ coding method. In our coding scheme, the multilayer perceptron network is used to improve the accuracy of side-match prediction by utilizing the property of the neural network nonlinear prediction. The NN-SMVQ scheme not only has the advantages of the SMVQ scheme but also improves the coded image quality. Experimental results are given and comparisons are made using our NN-SMVQ coding scheme and some other coding techniques. In the experiments, our NN-SMVQ coding scheme achieves the better visual quality about edge region and the best PSNR performance at nearly the same bit rate. This new NN-SMVQ scheme is also simple and efficient for the hardware design. Moreover, the new scheme does not adversely affect other useful functions provided by the conventional SMVQ scheme. © 2002 Elsevier Science (USA)

Key Words: vector quantization; SMVQ; multilayer feedforward neural network; multilayer perceptron; MLP; still image coding.

I. INTRODUCTION

Vector quantization (VQ) has been found to be an efficient method for still image coding [1–4]. The image to be encoded is first partitioned into nonoverlapping blocks to yield a set of vectors. The input vectors are individually quantized to the closest codeword in the codebook, which is generated by using some clustering algorithms from a set of training images. Image compression is achieved by using the indices of codewords for the information of the encoded images. Image decompression is done by utilizing the indices as addresses to

¹ This work was supported by the National Science Council, Taiwan, Republic of China, under Grant NSC86-2213-E-194-021.

the corresponding codewords in the decoder's codebook. This process can be implemented by table lookup techniques. Thus, the hardware design of the decoder is very simple.

Although the ordinary VQ scheme exploits the statistical redundancy within a vector and yields an acceptable performance at low bit rates, the finite-state vector quantization (FSVQ) schemes can improve performance over the ordinary VQ by exploiting the correlation between neighboring vectors within an image [5–12]. The encoding state of the current input vector is decided by the previous encoded neighboring vectors. The codebook size of the state codebook is much smaller than that of the master codebook. Hence, the encoding time can be reduced, and the image quality can be maintained at the lower bite rate. Moreover, side-match finite-state vector quantization (SMVQ) [13] is a popular class of FSVQs. The basic idea of SMVQ is to predict the current encoding vector by exploiting the correlations of encoded upper and left vectors for the current input vector. A subset of codewords will be selected from the master codebook by using the SMVQ selection function. Usually the closest codeword for the current input vector can be found in the selected codewords. In this work, when the correlations between the neighboring vectors are not high, the codewords selected by the SMVQ will be not close enough to the current input vector and achieve a large distortion in the image coding process.

Recently, artificial neural network (ANN) techniques have been applied to solve complex problems in the field of image coding. A number of studies for improving performance of VQ schemes by using neural networks have been proposed [14–18]. In these approaches, the multilayer feedforward neural network (multilayer perceptron, MLP) is widely used because it can possess the ability of extracting higher-order statistics by adding one or more hidden layers. Moreover, the error back-propagation algorithm reported by Rumelhart *et al.* [19] and Hirose *et al.* [20] is usually used to construct the MLP network. In this paper, we employ this neural network model as a vector predictor to predict the current encoding vector in the SMVQ scheme. For an ordinary SMVQ, the side-match selection of codewords usually uses a linear prediction function to predict the current input vector. The linear prediction function often produces a wrong prediction when an SMVQ is used to encode the natural images. Our new SMVQ scheme can exploit the nonlinearity property of the neural network to predict the input vectors more correctly. The multilayer neural network with a back-propagation learning algorithm is a reliable choice for our new SMVQ scheme because of the high capability of training and the efficiency of computing.

The rest of this paper is organized as follows. We discuss the VQ and SMVQ technique in Section II. Section III reviews the main features of multilayer neural networks. Section IV describes the construction of the neural side-match predictor by using the back-propagation learning algorithm. Further, Section V presents the structure and encoding method of our neural network side-mach finite state vector quantization (NN-SMVQ) scheme. Experimental results are given in Section VI for the images inside and outside the training set. Finally, conclusions are drawn in Section VII.

II. VECTOR QUANTIZATION AND SMVQ

An ordinary vector quantization is defined as a mapping from a k -dimensional Euclidean space R^k to a finite subset of R^k . This finite set $C = \{\hat{\mathbf{x}}_i; i = 1, \dots, N\}$ is called a codebook, where N is the size of the codebook. Each vector $\hat{\mathbf{x}}_i = (\hat{x}_0, \dots, \hat{x}_{k-1})$ in codebook C is called a codeword. First, the VQ encoder assigns each input vector $\mathbf{x} = (x_0, \dots, x_{k-1}) \in R^k$ to an index i , which points to the closest codeword $\hat{\mathbf{x}}_i$ in the codebook. Then the index i will

be transmitted to the VQ decoder. Finally, the VQ decoder decompresses the image by using the transmitted indices to find the corresponding codewords from the codebook. The compression ratio of an ordinary VQ coder can be controlled by choosing the different size of codebook when encoding the images. The distortion between the input vector $\mathbf{x}$ and its corresponding codeword $\hat{\mathbf{x}}_i$ is measured by the squared Euclidean distortion measure, i.e.,

$$d(\mathbf{x}, \hat{\mathbf{x}}) = \|\mathbf{x} - \hat{\mathbf{x}}\|^2 = \sum_{j=0}^{k-1} (x_j - \hat{x}_j)^2. \quad (1)$$

The codebook used by the VQ encoder has been generated by using the iterative clustering algorithm such as the LBG algorithm, due to Linde, Buzo, and Gray [21].

FSVQ exploits the redundancies between the neighboring vectors of the current input vector in order to reduce the bit rate. An FSVQ encoder establishes a uniquely defined state by using the information of previously encoded vectors. This coding state may be described by a state variable $s \in S = \{s_i: i = 1, \dots, M\}$, where M is the total number of states. Hence, an FSVQ can be defined as a mapping from $R^k \times S$ to a subset of a master codebook $MC = \{\mathbf{x}_i: i = 1, \dots, N\}$. For each state s_i , FSVQ encoder selects N_{sc} codewords by a selection function from the master codebook MC as the state codebook SC_s . For encoding an input vector $\mathbf{x}$, the encoder finds the current state s and then searches the state codebook SC_s to find its corresponding codeword. In the decompression phase, the decoder will find the same current state s and the corresponding codeword in the same state codebook SC_s by the transmitted index.

Furthermore, SMVQs are a popular class of FSVQ. The SMVQ tries to make the intensity (gray level) transition across the boundaries of the neighboring vectors as smooth as possible. That is, SMVQ uses the sides of upper and left neighboring vectors to establish the state codebook SC_s for the current input vector $\mathbf{x}$. The corresponding state codebook SC_s of the current input vector $\mathbf{x}$ is defined to be the N_{sc} codewords of the master codebook MC . In the ordinary SMVQ, a linear prediction function is used to select codewords of the state codebook SC_s that match the upper and left encoded vectors for the current input vector $\mathbf{x}$. We notice that the linear prediction function of the conventional SMVQ will select the wrong codewords from the master codebook when the correlations between the neighboring vectors are not high. For this work, we utilize the neural network as the predictor of the current input vector to achieve the better accuracy of side-match prediction in our new SMVQ.

III. MULTILAYER FEEDFORWARD NEURAL NETWORKS

The multilayer feedforward neural networks are an important class of neural networks. These neural networks have been applied successfully to solve some difficult and diverse problems. In general, there are one or more hidden layers in a multilayer feedforward neural network model, and the function of the hidden layer neurons is to arbitrate between the input and the output of neural networks. The structural graph of the neural network is shown in Fig. 1. First, the input vector is fed into the source nodes in the input layer of the neural network. The neurons of the input layer constitute the input signals applied to the neurons of the hidden layer. The output signals of the hidden layer are used as inputs to the next hidden layer. Finally, the output layer produces the output results and terminates the neural computing process.

FIG. 1. Structural graph of multilayer feedforward neural network.

Among the algorithms used to design the multilayer feedforward neural networks, the back-propagation algorithm is the most popular because it is very efficient for neural network learning. Typically, there are two different phases in the back-propagation algorithm: the forward phase and the backward phase. In the forward phase, the input signals are computed and passed through the neural network layer by layer. Then the neurons in output layer produce the output signals of the neural network. At this point, the error signals can be generated by comparing the output response of the neural network with the desired response. In the backward phase of the back-propagation algorithm, some free parameters used in the neural network are able to adjust by referring to the error signals. This work can be used to minimize the distortion of the neural network.

We notice that the multilayer neural network model has the properties of high learning capability and efficiency. Thus, the neural network is used to predict the current input vector for our new SMVQ scheme. The implementation of the multilayer feedforward neural network model is described in Section IV in detail.

IV. SIDE-MATCH VECTOR PREDICT BY USING NEURAL NETWORK LEARNING ALGORITHM

In this paper, a multilayer feedforward neural network model with a back-propagation learning algorithm is used to predict the current input vector for the selection function of our new NN-SMVQ scheme. The on-line implementation of the back-propagation learning algorithm is reiteratively executed from the training vectors and then produces the synaptic weight vectors, which are used for the neural network predictor. The back-propagation learning algorithm is described in the following steps.

Step 1. Initialization of the learning procedure. Set all initial synaptic weight vectors and the learning rate parameter η for the L layers feedforward neural network, and select a terminating error threshold value τ , which is used to stop the back-propagation learning process. This step also prepares the training samples for the following steps. Training vectors used for this learning algorithm be formed as $[\mathbf{x}, \mathbf{z}]$, where $\mathbf{x}$ is the ks -dimensional side vector applied to the neurons of the input layer and $\mathbf{z}$ denotes the kd -dimensional desired output vector for the output layer.

Step 2. Forward computation. Compute the output values of the neural network layer by layer. The internal output signal $h_j^{(l)}(n)$ for neuron j in layer l at iteration n th is defined as

$$h_j^{(l)}(n) = \sum_{i=0}^{ks} w_{ji}^{(l)}(n) p_i^{(l-1)}(n), \quad (2)$$

where $p_i^{(l-1)}(n)$ is the output signal of neuron i in the previous layer $l - 1$ and $w_{ji}^{(l)}(n)$ is the synaptic weight between neuron j in layer l and neuron i in layer $l - 1$. In this paper, a logistic function for the sigmoidal nonlinearity is used. The output signal of neuron j is defined as

$$p_j^{(l)}(n) = \begin{cases} x_j(n) & \text{if neuron } j \text{ is in the input layer (i.e., } l = 1), \\ o_j(n) & \text{if neuron } j \text{ is in the output layer (i.e., } l = L), \\ \varphi(h_j^{(l)}(n)) & \text{otherwise,} \end{cases} \quad (3)$$

where $x_j(n)$ is the j th element of the input vector $\mathbf{x}(n)$ and the sigmoidal activation function $\varphi(h)$ is defined as

$$\varphi(h) = \frac{1}{1 + \exp(-h)}. \quad (4)$$

Thus, the error signal $e_j(n)$ will be computed by

$$e_j(n) = z_j(n) - o_j(n), \quad (5)$$

where $z_j(n)$ is the j th element of the desired output vector $\mathbf{z}(n)$.

Step 3. Backward computation. Compute the vectors of local gradient δ' 's of the neural network layer by layer on the backward direction. The local gradient vectors can point to the required changes in the synaptic weights. The local gradient vector is defined as

$$\delta_j^{(l)}(n) = \begin{cases} e_j(n) o_j(n) [1 - o_j(n)] & \text{for neuron } j \text{ in output layer } l = L, \\ h_j(n) [1 - h_j(n)] \sum_{i=1}^m \delta_i^{(l)}(n) w_{ij}^{(l+1)}(n) & \text{for neuron } j \text{ in hidden layer } l, \end{cases} \quad (6)$$

where m is the total number of neurons in the layer l . Then the connection synaptic weights of the neural network between layer l and layer i are modified according to

$$\Delta w_{ji}^{(l)} = \eta \delta_j^{(l)}(n) h_i^{(l-1)}(n). \quad (7)$$

Step 4. Iteration of the learning procedure. The neural network learning algorithm will iteratively execute until the stopping criterion is satisfied. The back-propagation learning algorithm is terminated when the value of the average distortion function $\text{SE}_{\text{av}}(\mathbf{w}(n)) - \text{SE}_{\text{av}}(\mathbf{w}(n - 1))$ is smaller than the predefined threshold τ , where $\text{SE}_{\text{av}}(\mathbf{w}(n))$ is the average squared error for the training samples at the weight vector $\mathbf{w}(f)$ in iteration n .

Eventually, products the final synaptic weight vectors. By adding the final synaptic weight vectors into a multilayer feedforward neural network, this neural network is used to predict

the current input vector for the selection function in our NN-SMVQ scheme. Note that both the encoder and decoder of the NN-SMVQ have the same neural network module with $\mathbf{w}_{\text{final}}$, which was computed by the learning algorithm previously.

V. NEURAL NETWORK SIDE-MATCH FINITE-STATE VECTOR QUANTIZATION

In the conventional SMVQ, the selection function uses the sides of the upper and left neighboring blocks to generate the state codebook, SC_s for the current input vector. Figure 2 shows the side components that can be used for the side-match vector prediction in the case of block size 4×4 . The state space S is defined as $\text{MC} \times \text{MC} = (\mathbf{v} \times \mathbf{h} \mid \mathbf{v} \text{ and } \mathbf{h} \text{ are the vertical and horizontal state variables, respectively})$, where $\text{MC} = \{\mathbf{x}_i: i = 1, \dots, N\}$ is the master codebook with space size N . The variable $\mathbf{v}$ is used to observe the correlation of the upper neighboring block in the vertical direction, and the variable $\mathbf{h}$ is used to keep the correlation of the left neighboring block in the horizontal direction. Thus, the encoding state s of the current input vector can be defined by the vertical state $\mathbf{v}$ and horizontal state $\mathbf{h}$. The best match for sides of the upper block and left block to the current input vector $\mathbf{x}$ is used to define the corresponding state codebook SC_s . The vertical side-match distortion d_v and vertical side-match distortion d_h of a codeword $\mathbf{y}$ in the master codebook MC are defined as the following equations, respectively,

$$d_v(\mathbf{y}) = \sum_{i=1}^m (vy_i - u_i)^2 \quad \text{and} \quad d_h(\mathbf{y}) = \sum_{j=1}^n (hy_j - l_j)^2, \quad (8)$$

where n is the width and m is the height of a block, $\mathbf{vy}$ is the first vertical row vector of $\mathbf{y}$, and $\mathbf{hy}$ is the leftmost horizontal column vector. Eventually, the total side-match distortion d_{total} of a codeword $\mathbf{y}$ is defined as

$$d_{\text{total}}(\mathbf{y}) = d_v(\mathbf{y}) + d_h(\mathbf{y}). \quad (9)$$

FIG. 2. Boundary components used for the side-match vector prediction in SMVQ (4×4 block).

FIG. 3. Boundary components used for NN-SMVQ (4×4 block).

The SMVQ selection function selects the N_{sc} closest codewords with the total side-match distortion d_{total} from the master codebook MC as the state codebook SC_s for the corresponding state s . Then the SMVQ finds the closest codeword $\hat{\mathbf{x}}$ in the state codebook SC_s for the current input vector $\mathbf{x}$.

The neural network SMVQ scheme proposed in this paper uses a multilayer feedforward neural network to predict the current input vector. At first, the final synaptic weight vectors $\mathbf{w}_{final}$ are produced by the back-propagation learning algorithm. Then a multilayer neural network model with the final synaptic weight vectors $\mathbf{w}_{final}$ is used as the neural network side-match predictor in our NN-SMVQ. The side-match predictor utilizes the information of the boundary components from three neighboring blocks (upper block $\mathbf{u}$, upper-left block $\mathbf{ul}$, and left block $\mathbf{l}$) to generate the state codebook SC_s for the current input vector, as shown in Fig. 3.

For each input vector $\mathbf{x}$, the ks -dimensional side-match vector $\mathbf{i} = (i_1, i_2, i_3, \dots, i_{ks})$ corresponding to $\mathbf{x}$ is fed into the neural network predictor as the input signals. Then a kd -dimension output vector $\mathbf{o} = (o_1, o_2, o_3, \dots, o_{kd})$ is produced by the neural network computation layer by layer. The neural output vector $\mathbf{o}$ is used to observe the correlations of the upper neighboring block in the vertical direction, the left-upper neighboring block in the diagonal direction, and the left neighboring block in the horizontal direction. Consequently, The output vector for the current input vector $\mathbf{x}$ is used to define the corresponding state codebook SC_s . The neural network side-match distortion d_n of a codeword $\mathbf{y}$ in the master codebook MC is defined as

$$d_n(\mathbf{y}) = \sum_{i=1}^{kd} (uly_i - o_i)^2, \quad (10)$$

where $\mathbf{uly}$ denotes the union of the first vertical row vector and the leftmost horizontal column of $\mathbf{y}$.

The NN-SMVQ encoder selects the N_{sc} closest codewords with the neural side-match distortion d_n from the master codebook MC as the state codebook SC_s for the corresponding state s . The encoder finds the closest codeword $\hat{\mathbf{x}}$ in the state codebook SC_s for the current

TABLE 1

The PSNR Values, Bit Rates, and Number of Vectors with Full Searching (N_f) of the Ordinary VQ, SMVQ, and NN-SMVQ Schemes for the Images in the Training Set

Images	VQ		SMVQ			NNSMVQ		
	PSNR	Bit rate	PSNR	Bit rate	N_f	PSNR	Bit rate	N_f
Peppers	28.37	0.3125	30.40	0.3088	1712	31.99	0.3091	1722
Airplane	28.04	0.3125	29.19	0.3054	1615	30.75	0.3053	1612
Sailboat	26.59	0.3125	27.02	0.3119	1804	28.70	0.3125	1820
Boat	27.05	0.3125	28.15	0.3116	1794	29.35	0.3117	1796
Tiffany	28.86	0.3125	32.14	0.2931	1254	33.38	0.2945	1296

input vector $\mathbf{x}$. In order to avoid producing the wrong states for the input vectors and retain the image quality, the encoder checks whether the selected codeword is good enough. The NN-SMVQ encoder attempts to find a better codeword in the master codebook MC and transmits the index mp of the new codeword $\hat{\mathbf{x}}'$ to the decoder, if the distortion $d(\mathbf{x}, \hat{\mathbf{x}})$ is larger than a preset distortion threshold T_f . Otherwise, the index sp of the codeword $\hat{\mathbf{x}}$ in the state codebook SC_s will be transmitted.

The architecture of the NN-SMVQ is simple, redressing easily, and suitable for hardware design. The quality of the reconstructed image can be controlled by allowing an update of the new synaptic weight vectors for the neural networks in both the encoder and the decoder. Note that the neural network in the NN-SMVQ redresses synaptic weight vectors without modifying the other functions of the coding scheme.

VI. SIMULATIONS AND RESULTS

In the simulations, we use a two-layer multilayer feedforward neural network as the nonlinear side-match predictor. For the back-propagation learning algorithm, the initial synaptic weights are produced by a random number generation function at a floating interval $[0-0.00001]$. The learning rate parameter $\eta = 0.03$ is chosen to produce the final synaptic weight vectors because of the tradeoff between the prediction performance and computational speed. The master codebook MC is generated by the LBG algorithm from the training set of five different images. The test images are monochrome images of size 512×512 with 256 gray levels. To evaluate the coder's performance numerically, the peak signal-to-noise ratio (PSNR) between the original image and the encoded image has been calculated, where the PSNR is defined as

$$\text{PSNR} = 10 \log_{10} \frac{255^2}{\text{MSE}} \text{ dB}. \quad (11)$$

TABLE 2

The PSNR Values, Bit Rates, and Number of Vectors with Full Searching (N_f) of the Ordinary VQ, SMVQ, and NN-SMVQ Schemes for Images Outside the Training Set

Images	VQ		SMVQ			NNSMVQ		
	PSNR	Bit rate	PSNR	Bit rate	N_f	PSNR	Bit rate	N_f
Lenna	28.22	0.3125	29.53	0.3098	1743	31.16	0.3086	1707
Family	27.01	0.3125	27.68	0.3121	1809	28.94	0.3122	1813
Zelda	30.25	0.3125	32.54	0.3064	1643	33.72	0.3057	1594

FIG. 4. Results of the image Peppers: (a) original image, (b) VQ at 0.3125 bpp, (c) SMVQ at 0.3088 bpp, and (d) NN-SMVQ scheme at 0.3091 bpp.

Note that the mean squared error for an $n \times n$ images is defined as

$$\text{MSE} = \left(\frac{1}{n}\right)^2 \sum_{i=1}^n \sum_{j=1}^n (x_{ij} - \hat{x}_{ij})^2, \quad (12)$$

where x_{ij} and $\hat{x}_{ij}$ denote the original and quantized gray levels, respectively.

We compare the three coding schemes, the ordinary VQ, the conventional SMVQ, and our new NN-SMVQ in the simulations. In the ordinary VQ, the size of the input blocks is 4×4 and the codebook size is 32. In the conventional SMVQ and our NN-SMVQ schemes, the master codebook size N_m is 512, the state codebook size N_{sc} is 15, and the dimension of the input vectors is 16. The ordinary VQ is a fixed bit-rate coding scheme. Hence, we try a number of thresholds T_f for each image in our simulations in order to compare the PSNR values of the images that are encoded by these three coding schemes at nearly the same bit

FIG. 5. Results of the image Lena: (a) original image, (b) VQ at 0.3125 bpp, (c) SMVQ at 0.3098 bpp, and (d) NN-SMVQ scheme at 0.3086 bpp.

rate. Note that we can adjust the threshold T_f to obtain the desired image quality and bit rate.

Table 1 shows the PSNR values, the bit rates, and the number of vectors with full searching of simulation results for the images in the training set. In our experiments, the improvement over the ordinary VQ is up to 3.62 dB and the improvement over the SMVQ is up to 1.59 dB for the image Lena. Table 2 shows the PSNR values, the bit rates, and the number of vectors with full searching of simulation results for the images outside the training set. The improvement over the ordinary VQ is up to 2.94 dB and the improvement over the SMVQ is up to 1.63 dB for the image Lena. Moreover, we also compare the number of full search vectors for the reconstructed image Lena of both SMVQ and NN-SMVQ at nearly the same PSNR value. By using the conventional SMVQ scheme, all 1743 full search vectors are used to encode the image Lena and the PSNR value is 29.5 dB. For the same quality

FIG. 6. Results of the image Lena: (a) magnified portion of the reconstructed image Fig. 5c and (b) magnified portion of the reconstructed image Fig. 5d.

encoded image, only 612 full search vectors are need to encode the image by using the NN-SMVQ. That is, the NN-SMVQ can achieve a better accuracy of side-match prediction than the conventional SMVQ.

The images Peppers and Lena in Fig. 4a and Fig. 5a are the original 512×512 monochrome images with 8 bpp. Figures 4b–4d show the encoded results for the image Peppers, which is in the training set. Figure 4b shows the encoded image by using the ordinary VQ scheme at 0.3125 bpp. Figure 4c is the encoded result of the SMVQ scheme at 0.3088 bpp. The NN-SMVQ yields the result shown in Fig. 4d, which is encoded at 0.3091 bpp. Figures 5b–5d show the encoded results for the image Lena, which is outside the training set. Figure 5b shows the encoded image by using an ordinary VQ scheme at 0.3125 bpp. Figure 5c is the encoded result of the SMVQ and the bit rate is 0.3098 bpp. Figure 5d shows the encoded result of the NN-SMVQ for the image Lena and the average bit rate is 0.3086 bpp. To show the differences clearly, the enlarged portion in the processed images Lena is given in Fig. 6a and Fig. 6b. It is can be seen easily that our proposed scheme can obtain the better image quality and the better visual quality about a fine region. Table 3 shows the improvement

TABLE 3

The PSNR Improvement of the NN-SMVQ Scheme over the Ordinary VQ and SMVQ Schemes at Nearly the Same Bit Rate 0.31 bpp

	Images	Over VQ	Over SMVQ
Inside	Peppers	3.62	1.59
	Airplane	2.71	1.56
	Sailboat	2.11	1.68
	Boat	2.30	1.20
	Tiffany	4.52	1.24
Outside	Lenna	2.94	1.63
	Family	1.93	1.26
	Zelda	3.47	1.18

of PSNR values over the ordinary VQ and SMVQ of simulation results at nearly the same bit rate.

VII. CONCLUSIONS

The SMVQ scheme exploits the statistical correlations between neighboring blocks to obtain good encoded results. However, SMVQ uses only the linear correlations but ignores the important nonlinear correlations in natural images. In this paper, we have proposed a new SMVQ scheme for image coding. The NN-SMVQ scheme can combine the advantages of neural networks and SMVQ schemes. This encoding scheme adopts a multilayer neural network to predict all input vectors more accurately. The back-propagation learning algorithm is used to construct the synaptic weights for the neural network predictor of the NN-SMVQ scheme. The output signals of the neural network predictor will assist in selecting the state for the input vectors. The multilayer neural network can be implemented easily by using VLSI techniques. Thus, the hardware design for the NN-SMVQ scheme is simple and efficient. Moreover, in comparison to the ordinary VQ and SMVQ schemes, the NN-SMVQ scheme can obtain better image quality and better visual quality about the edge region. A better PSNR performance is achieved within the low bit rate region 0.29 to 0.32 bpp. From the experimental results, we find that NN-SMVQ is superior to the conventional VQ and SMVQ.

REFERENCES

1. R. M. Gray, Vector quantization, *IEEE Acoust. Speech Signal Process.* Apr. 1984, 4–29.
2. M. Goldberg, P. R. Boucher, and S. Shlien, Image compression using adaptive vector quantization, *IEEE Trans. Commun.* **34**, 1986, 180–187.
3. N. M. Nasrabadi and R. A. King, Image coding using vector quantization: A review, *IEEE Trans. Commun.* **36**, 1988, 957–971.
4. H. M. Hang and B. G. Haskell, Interpolative vector quantization of color images, *IEEE Trans. Commun.* **36**, 1988, 465–470.
5. J. Foster, R. M. Gray, and M. O. Dunham, Finite-state vector quantization of waveform coding, *IEEE Trans. Inform. Theory* **31**, 1985, 348–359.
6. R. Aravind and A. Gersho, Low-rate image coding with finite-state vector quantization, in *Proc. ICASSP*, 1986, pp. 137–140.
7. R. F. Chang, W. T. Chen, and J. S. Wang, A fast finite-state algorithm for vector quantizer design, *IEEE Trans. Signal Process.* **40**, 1992, 221–225.
8. W. T. Chen, R. F. Chang, and J. S. Wang, Image sequence coding using finite-state vector quantization, *IEEE Trans. Circuits Syst. Video Technol.* **2**, 1992, 15–24.
9. C. S. Kim, J. Bruder, M. J. T. Smith, and R. M. Mersereau, Subband coding of color images using finite state vector quantization, in *Proc. ICASSP*, 1988, pp. 753–756.
10. M. O. Dunham and R. M. Gray, An algorithm for the design of label-transition finite state vector quantizers, *IEEE Trans. Commun.* **33**, 1985, 83–89.
11. R. F. Chang and W. T. Chen, Image coding using variable-rate side-mach finite-state vector quantization, *IEEE Trans. Image Process.* **2**, 1993, 104–108.
12. N. M. Nasrabadi and Y. Feng, A dynamic finite-state vector quantization scheme, in *Proc. ICASSP*, 1990, pp. 2261–2264.
13. T. Kim, Side match and overlap match vector quantizers for images, *IEEE Trans. Image Proc.* **1**, 1992, 170–185.
14. N. M. Nasrabadi and Y. Feng, Vector quantization of images based upon the Kohonen self-organizing feature maps, in *Proc. IEEE Int. Conf. Neural Network*, 1988, pp. 1101–1108.

15. Y. Zhou, R. Chellappa, A. Vaid, and B. K. Jenkins, Image restoration using neural network, *IEEE Trans. Acoust. Speech Signal Process.* **36**, 1988, 1141–1151.
16. F. H. Wu and K. Ganesan, Comparative study of algorithm for VQ design using conventional and neural-net based approaches, in *Proc. ICASSP*, 1989, pp. 751–754.
17. A. K. Krishnamurthy, S. C. Ahalt, D. E. Melton, and P. Chen, Neural networks for vector quantization of speech and images, *IEEE Trans. Commun.* **8**, 1990, 1449–1457.
18. N. Mohsenian, S. A. Rizvi, and N. M. Nasrabadi, Predictive vector quantization using a neural networks approach, *Opt. Engng.* **32**, 1993, 1503–1513.
19. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning representation by back-propagation errors, *Nature* **323**, 1986, 533–536.
20. Y. Hirose, K. Yamashita, and S. Hijiva, Back-propagation algorithm which varies the number of hidden units, *Neural Networks* **4**, 1991, 61–66.
21. Y. Linde, A. Buzo, and R. M. Gray, An algorithm for vector quantizer design, *IEEE Trans. Commun.* **28**, 1980, 84–95.

YU-LEN HUANG was born in Chiayi, Taiwan, on May 22, 1970. He received the B.S. in computer science from Tung Hai University, Taichung, Taiwan, Republic of China, in 1990, and the M.S. and Ph.D. in computer science and information engineering from Nation Chung Cheng University, Chiayi, Taiwan, Republic of China, in 1994 and 1999. He is currently an assistant professor in the Department of Information Management, Chaoyang University of Technology, Taichung County, Taiwan, Republic of China. His research interests include image processing, medical imaging, multimedia/video communication, and neural networks.

RUEY-FENG CHANG was born in Taichung, Taiwan, on August 25, 1962. He received the B.S. in electrical engineering from National Cheng Kung University, Tainan, Taiwan, Republic of China, in 1984, and the M.S. in computer and decision sciences and the Ph.D. in computer science from National Tsing Hua University, Hsinchu, Taiwan, Republic of China, in 1988 and 1992, respectively. Since 1992, he has been with the Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan, Republic of China, and he is now a professor. His research interests include image/video processing and retrieval, medical computer-aided diagnosis system, and multimedia systems and communication. Dr. Chang is a member of IEEE, ACM, SPIE, and Phi Tau Phi.