FINITE STATE RESIDUAL VECTOR QUANTIZATION USING TREE-STRUCTURED COMPETITIVE NEURAL NETWORK

Syed A. Rizvi and Nasser M. Nasrabadi
Department of Electrical & Computer Engineering
State University of New York at Buffalo
Amherst, NY 14260

ABSTRACT

The performance of an ordinary Vector Quantizer (VQ) can be improved by incorporating memory in the VQ scheme. A VQ scheme with finite memory known as Finite State Vector Quantization has been shown to give better performance than the ordinary VQ. The major problems with the FSVQ are the lack of accurate prediction of the current state, the state codebook design, and the amount of memory required to store all the state codebooks. This paper presents a new FSVQ scheme called Finite-State Residual Vector Quantization (FSRVQ) in which a neural network based state prediction is used. Furthermore, a novel tree-structured competitive neural network is used to jointly design the next-state and the state codebooks for the proposed FSRVQ. Simulation results show that the new scheme gives better performance with significant reduction in the memory requirement when compared to the conventional FSVQ schemes.

1. INTRODUCTION

In Vector Quantization (VQ) a block of pixels (vector) is encoded as oppose to scalar quantization in which each pixel is encoded independently [1]. For a given bit-rate, the codebook search complexity increases exponentially with the increase in block size. Therefore, an ordinary VQ uses a relatively small block size; consequently, in highly correlated data such as images there still exists a high correlation among neighboring blocks. This inter-block correlation can be exploited by incorporating memory into the VQ scheme to further improve the performance of a VQ. Several VQ schemes with memory such as Predictive Vector Quantization (PVQ) [1]-[3] and Finite-State Vector Quantization (FSVQ) [1], [4]-[8] have been proposed in the literature where the inter-block correlation is exploited.

An FSVQ has a finite number of states where with each state a distinct small codebook is associated. The current state of the encoder (and decoder) is determined by the previous state and the previously encoded vectors. The input vector is then quantized using the current state codebook. Problems with an FSVQ are the lack of accurate prediction of the current state of the encoder (and decoder), the state codebook design, and the large amount of memory required to store all the state codebooks. Inaccurate prediction of current state duplicates a large number of unnecessary (redundant) states; consequently, a huge amount of memory is required to store all the codebooks corresponding to these redundant states. Therefore, the major task in an FSVQ is the joint optimization of the next-state function (next-state codebook) and all the state codebooks, which would eliminate the redundant states.

In recent years, several improved FSVQ schemes have been proposed. The FSVQ scheme proposed by Kim [6] gives better performance than the other FSVQ schemes; however, this FSVQ scheme uses a large number of states which increases the memory requirements significantly. For example, for a supercodebook of size M, M^2 possible states are needed. In [7], an FSVQ scheme based on neural network classification of states is proposed which attempts to reduce the memory requirements of an FSVQ. Another FSVQ scheme, called Finite-state Binary Residual VQ (FS-BRVQ), has been proposed in [8] that combines the FSVQ with a binary RVQ. The codebook search complexity of an FSBRVQ is significantly lower than that of an FSVQ; however, the performance of the FSBRVQ deteriorates when compared with an FSVQ.

This paper presents a new Finite-state RVQ scheme where a neural network based state prediction is used. The prediction is based on the previously encoded blocks and the predicted block (vector) is used to identify the current state as well as generating a residual vector. This residual vector is then encoded using the current state codebook. In order to achieve joint optimization of the next-state function and state codebooks, a novel tree-structured competitive neural network is developed. The proposed scheme not only reduces the search complexity and memory requirements significantly but also gives improved performance when compared to an ordinary FSVQ.

This paper is organized as follows: in section 2, tree-structured competitive neural network is presented, and in section 3 FSRVQ scheme is presented. Experimental results for still images are presented in section 4. Section 5 concludes the paper.

2. TREE-STRUCTURED COMPETITIVE NEURAL NETWORK

Figure 1 shows a two-stage competitive neural network that is used to design a two-stage RVQ [9] where each stage (codebook) of the RVQ is represented by a single layer of the competitive neural network (weights). In Fig. 1, each arrow represents a k dimensional weight vector. The number of nodes (codevectors) in the first and the second layer are represented by N_1 and N_2, respectively. The weight vectors $\mathbf{W}_j$ and $\mathbf{V}_{jk}$ represent the codevectors for the first- and second-stage codebooks, respectively. The weight vectors connecting all the neurons from the

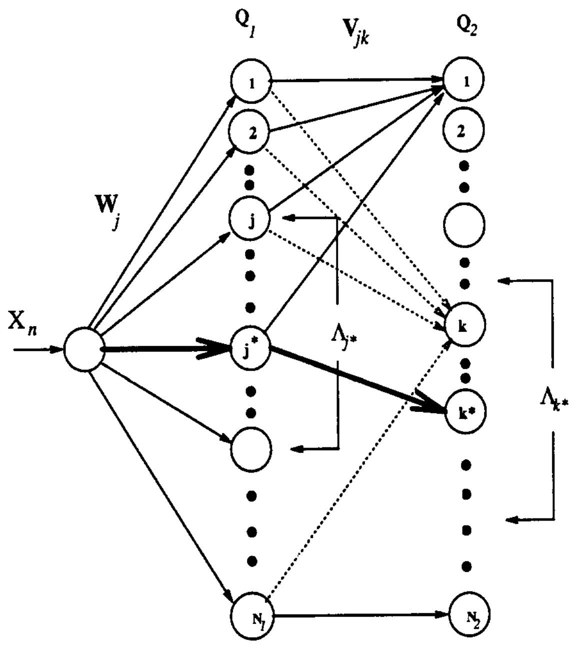

Figure 1: Two-stage Competitive Neural Networks.

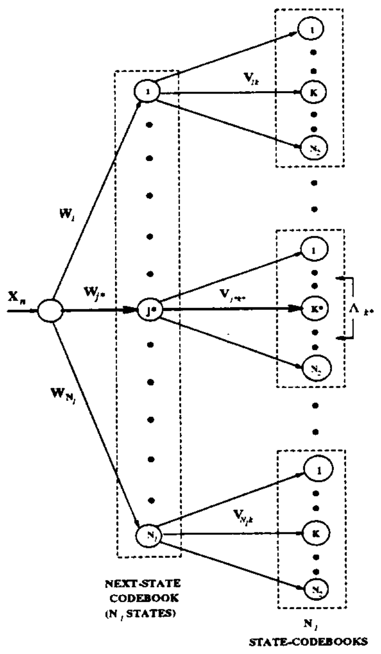

Figure 2: Tree-structured Competitive Neural Network where the first layer implements the next state function and the second layer implements the state codebooks.

first layer to a single neuron in the second layer are the same: that is, $V_{jk} = V_{lk}$ for all $j, l \in N_1$. In Fig. 1, the nodes j^* and k^* are the winning neurons in the first and second stage, respectively. The input to the first stage is a k-dimensional vector $\mathbf{X}_n \in \Re^k$ and the input to the second stage is the quantization error vector $\mathbf{e}_{(Q_1)_{j^*}}$ obtained from the first stage winning neuron j^*.

A modified multi-stage competitive neural network known as Tree-structured competitive neural network (TSCNN) is used to design the FSRVQ. TSCNN can easily be implemented by modifying a multi-layer competitive neural network. Let us reconsider the two-layer competitive neural network shown in Fig. 1. If the weights connecting all the neurons from the first layer to a single neuron in the second layer are all different i.e. $V_{jk} \neq V_{lk}$ for all $l \neq j$, $l, j \in N_1$, then a TSCNN is obtained. Figure 2 shows the equivalent diagram for a TSCNN. This particular TSCNN is used to design a Finite State RVQ (FSRVQ) where the first layer represents the next state function with N_1 states and each state having a codebook of size N_2 is represented by the second layer of the neural network.

3. FINITE STATE RESIDUAL VECTOR QUANTIZATION

This scheme incorporates a neural network predictor for the state prediction. The basic structure of the scheme is shown in Fig. 3. The scheme consists of a neural network predictor and a next-state codebook containing m codevectors corresponding to each of the m states. Each state codebook (subcodebook) consists of N codevectors (all state codebooks are assumed to have the same size). This scheme differs from the conventional FSVQ in that the state codebooks encode the residual vectors instead of the original vectors. A neural network predictor predicts the current block based on the four previously encoded blocks as shown in Fig. 3. The predicted vector $\tilde{\mathbf{X}}_n$ is then classified by performing a nearest neighbor search of the next-state codebook. The index $(\mathbf{I}_{ns})$ of the codevector closest to $\tilde{\mathbf{X}}_n$ in Euclidean distance sense represents the current state. The residual vector $\mathbf{E}_n = \mathbf{X}_n - \tilde{\mathbf{X}}_n$ is then encoded using the current state codebook. The neural network predictor is designed using the back-propagation learning algorithm [2]. The next-state codebook and state codebooks are designed using the tree-structured competitive neural network mentioned in the previous section. The summary of the design procedure is given below.

Algorithm:

Initialization: Given the training set $T = [\mathbf{X}_n; n = 1, 2, ..., M]$. Initialize the tree-structured competitive neural network by selecting the weights $\mathbf{W}_j$'s and $\mathbf{V}_{jk}$'s corresponding to the codevectors of the initial codebooks for the next-state codebook and the residual state codebooks, respectively. Choose the maximum number of iteration τ_{max} be a large number. Set the current number of iteration $\tau \to 0$. Choose initial learning rate α^τ and the size of initial neighborhood ν^τ.

STEP 1: *Compute the output of the predictor $\tilde{\mathbf{X}}_n$ based on the four previously encoded blocks as shown in Fig. 3.*

- *Compute distance D_j between the predicted vector $\tilde{\mathbf{X}}_n$ and each of the codevectors $\mathbf{W}_j$ and select the winning codevector $\mathbf{W}_{j^*}$ with minimum D_j such that*

$$||\tilde{\mathbf{X}}_n - \mathbf{W}_{j^*}|| \leq ||\tilde{\mathbf{X}}_n - \mathbf{W}_j|| \quad for \quad j \neq j^* \quad (1)$$

- *Modify $\mathbf{W}_{j^*}$'s contents and its neighboring codevectors by*

$$\mathbf{W}_j^{t+1} = \mathbf{W}_j^t + \alpha_1^\tau \left(\tilde{\mathbf{X}}_n - \mathbf{W}_j^t\right) \quad for \quad j \in \Lambda_{1s} \quad (2)$$

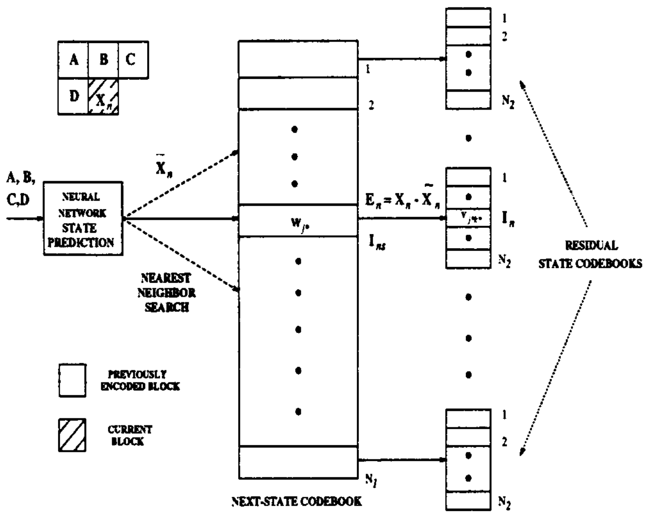

Figure 3: An FSRVQ with neural network state prediction.

where t is the time index and $\Lambda_{1s} = \{\ y_{1s} \mid max(1, j^* - \nu_1^\tau) < y_{1s} < min(N_1, j^* + \nu_1^\tau)\}$. ν_1^τ represents the topological neighborhood around the winning codevector. The learning rate $\alpha_1^\tau \in [0,1]$ is the adaptation gain term. The neighborhood and learning rate is decreased with the number of iterations as given below

$$\nu_1^\tau = \lceil \nu_1^0 \left(1 - \tfrac{\tau}{\tau_{max}}\right) \rceil$$

$$\alpha_1^\tau = \alpha_1^0 \left(1 - \tfrac{\tau}{\tau_{max}}\right)$$

where $\lceil x \rceil$ represents the smallest integer greater than x.

STEP 2: *Find the residual as*

$$\mathbf{E}_n = \mathbf{X}_n - \tilde{\mathbf{X}}_n$$

- *Compute the distance D_k between the residual error $\mathbf{E}_n$ and each of the codevectors $\mathbf{V}_{j^*k}$, and select the winning codevector $\mathbf{V}_{j^*k^*}$ with minimum D_k such that*

$$||\mathbf{E}_n - \mathbf{V}_{j^*k^*}|| \leq ||\mathbf{E}_n - \mathbf{V}_{j^*k}|| \quad for \quad k \neq k^* \qquad (3)$$

- *Modify the contents of the winning codevector and its neighboring codevectors by*

$$\mathbf{V}_{j^*k}^{t+1} = \mathbf{V}_{j^*k}^t + \alpha_2^\tau \left(\mathbf{X}_n - \tilde{\mathbf{X}}_n - \mathbf{V}_{j^*k}^t\right) \quad for \quad k \in \Lambda_{2s} \qquad (4)$$

where Λ_{2s} and α_2^τ are defined in a similar manner as defined in step 1.

- *Find the current reconstructed vector $\hat{\mathbf{X}}_n$ as*

$$\hat{\mathbf{X}}_n = \tilde{\mathbf{X}}_n + \mathbf{V}_{j^*k^*}^{t+1}. \qquad (5)$$

STEP 3: *Set $n \rightarrow n+1$. If $n < M$ go to STEP 1; otherwise goto next STEP.*

STEP 4: *Set $\tau \rightarrow \tau + 1$. If $\tau < \tau_{max}$, then set $n \rightarrow 0$ and goto STEP 1; otherwise STOP.*

4. SIMULATION RESULTS

Computer simulations were performed in order to implement the proposed FSRVQ scheme. Two FSRVQ schemes were designed with the state codebooks of size 32 and 16, respectively. Both FSRVQs use only 16 states. A block size of 4×4 was used with a total of 81920 training vectors. Two 512×512 test images Lena and Boats (test images Lena and Boats were outside the training set) were used to evaluate the performance of the proposed FSRVQ. Simulation results are presented in terms of Peak Signal to Noise Ratio (PSNR) at the average bit rates (entropy coded) of 0.25 bit per pixel (bpp) and below.

Table I shows the performance comparison of the proposed FSRVQ with that of the FSVQ schemes given in [6] and [7] for the test image Lena. In Table I the search complexity is given in terms of number of multiplications required to encode 1 pixel and the storage requirement is in terms of number of bytes required to store all the codebooks. Note that computational complexities of the next-state functions for different FSVQ schemes are not included in the comparisons. It can be seen from Table I that the proposed FSRVQ scheme outperforms the FSVQ schemes developed in [6], [7] by 0.65 and 0.5 dB, respectively. The FSVQ scheme developed in [6] uses a supercodebook of size 256 and state codebooks of size 64 each. The number of states used is 65536. The memory requirements for the proposed FSRVQ are about 512 times lower than that required by the FSVQ developed in [6]. The FSVQ scheme developed in [7] uses 1024 states and state codebooks of size 64 each. This corresponds to a memory requirement of 1056768 bytes (compare with the FSRVQ scheme that requires only 8448 byte).

Table II shows the performance comparison of the proposed FSRVQ with that of JPEG (current standard for image compression) for the test images Lena and Boats. It can be seen from the Table II that the proposed FSRVQ scheme clearly outperforms JPEG at these bit rates. Figure 4 shows the original and encoded image Lena using FSRVQ and JPEG at the bit rate of 0.18 bpp. It is interesting to note that the visual quality of the image compressed by JPEG is much worse than indicated by the PSNR. The reason for this is that at low bit rates JPEG discards most of the high frequency components which results in annoying blocking artifacts.

5. CONCLUSION

In this paper a new FSVQ scheme is introduced. The proposed scheme uses a neural network based state prediction. Furthermore, residual obtained by subtracting the predicted vector from the original vector is then encoded using the current state codebook. A novel tree-structured competitive neural network is used to jointly design the next-state and the current state codebooks. The joint optimization of next-state codebook and the state codebooks eliminates a large number of redundant states. Simulation results show that the proposed scheme outperforms the ordinary FSVQ schemes in terms of performance, search complexity, and memory requirements. Furthermore, the proposed FSRVQ clearly outperforms JPEG at low bit rates. A better rate-distortion performance can be achieved by imposing a constraint on the output entropy of the proposed FSRVQ scheme. The design of Entropy-Constrained FSRVQ is the focus of the future research.

References

[1] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression.* Boston: Kluwer Academic Publishers, 1992.

[2] N. Mohsenian, S. A. Rizvi, and N. M. Nasrabadi, "Predictive vector quantization using a neural network approach," *Optical Engineering*, vol. 32, no. 7, pp. 1503-1513, July 1993.

[3] S. A. Rizvi and N. M. Nasrabadi, "Predictive vector quantization using constrained optimization,"*IEEE Signal Processing Lett.*, vol. 1, no. 1, pp. 15-18, January 1994.

[4] N. M. Nasrabadi, C. Y. Choo, and Y. S. Feng, "Dynamic Finite-state vector quantization of digital images," *IEEE Trans. Commun.*, vol. 32, no. 5, pp. 2145-2154, May 1994.

[5] N. M. Nasrabadi and S. A. Rizvi, "Next-state functions for finite state vector quantization," in *Proc. IEEE Int. Conf. Acoust. Speech, Signal Processing,* (Adelaide, Australia), April 19-22 1994, vol. 5, pp. 617-620.

[6] T. Kim, "Side match and overlap match vector quantizers for images," *IEEE Trans. Image Processing,* vol. 1,no. 2, pp. 170-185, April 1992.

[7] C. N. Manikopoulos, "Finite state vector quantization with neural network classification of states," *IEE Proceedings-F,* vol. 140, no. 3, pp. 153-161, June 1993.

[8] F. Kossentini, M. J. T. Smith, and C. F. Barnes, "Residual VQ with state prediction: an new method for image coding," in *Proc. Visual Commun. Image Processing'91* (Boston), Nov. 11-13, 1991, pp. 383-394.

[9] S. A. Rizvi and N. M. Nasrabadi, "Residual vector quantization using multi-layer competitive neural network,"*IEEE J. Select. Areas Commun.* (to appear), Dec. 1994.

Table I: Performance comparison of the proposed FSRVQ with that of FSVQ schemes given in Ref. 6 and 7, respectively, for the test image Lena.

FSVQ Technique	Bit-rate bpp	PSNR dB	SC*	SR**
Proposed	0.23	30.65	32	8448
Ref. 6	0.25	30.00	64	4329472
Ref. 7	0.24	30.15	64	1056768

* Search complexity

** Storages requirement

Table II: Performance comparison of the proposed FSRVQ with that of JPEG for the test images Lena and Boats.

Test Image	FSRVQ		JPEG	
	Bit-rate bpp	PSNR dB	Bit-rate bpp	PSNR dB
Lena	0.23	30.65	0.23	29.94
Lena	0.18	29.68	0.18	27.33
Boats	0.20	28.55	0.20	27.44
Boats	0.25	29.53	0.25	29.06

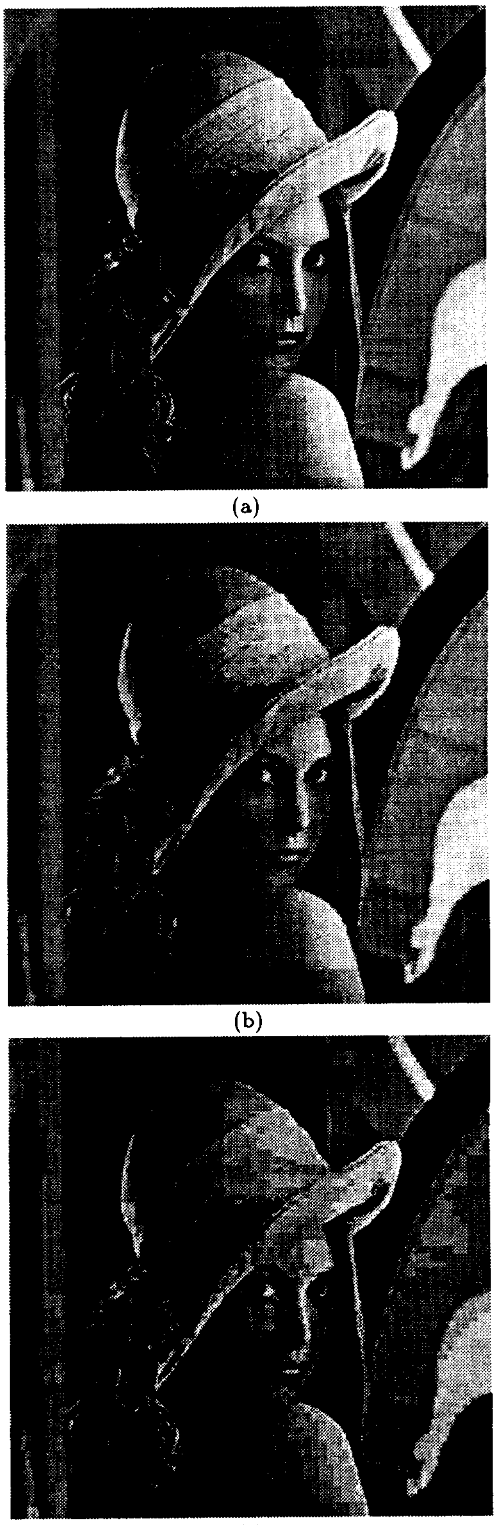

Figure 4: (a) Original image "Lena", (b) encoded image "Lena" using FSRVQ, $SNR = 29.68$ at bit-rate = 0.18 *bpp*, (c) encoded image "Lena" using JPEG, $SNR = 27.33$ at bit-rate = 0.18 *bpp*.