A PREDICTIVE RESIDUAL VQ USING MODULAR NEURAL NETWORK VECTOR PREDICTOR

Lin-Cheng Wang, Syed A. Rizvi,* and Nasser M. Nasrabadi†
Department of Electrical and Computer Engineering
State University of New York at Buffalo, Amherst, NY 14260
*Department of Applied Sciences
College of Staten Island of City University of New York, Staten Island, NY 10314
†US Army Research Laboratory, ATR Research Branch
2800 Powder Mill Road, Adelphi, MD 20783

ABSTRACT

This paper presents a predictive residual vector quantization (PRVQ) scheme using a modular neural network vector predictor. The proposed PRVQ scheme takes the advantage of the high prediction gain and the improved edge fidelity of a modular neural network vector predictor in order to implement a high performance vector quantization (VQ) scheme with low search complexity and a high perceptual quality. Simulation results show that the proposed PRVQ with modular vector predictor outperforms the equivalent PRVQ with general vector predictor (operating at the same bit rate) by more than $1dB$. Furthermore, the perceptual quality of the reconstructed image is also improved.

1. INTRODUCTION

A predictive residual vector quantization (PRVQ) scheme uses a predictor that predicts the current block from the previously encoded blocks and constructs a residual block (the difference between the original and predicted blocks) [1]. The residual block is then encoded using relatively small multistage codebooks. The prediction exploits the inter-block correlation and removes the redundancy among neighboring blocks.

Recently, we have reported on a modular neural network vector predictor (shown in Figure 3) to improve the predictive component of a PRVQ coding scheme [2]. The vector prediction technique consists of five dedicated predictors (experts), where each expert predictor is optimized for a particular class of input vectors. An input vector is classified into one of five classes based on its *directional variances*. One expert is optimized for stationary blocks, and each of the other four experts are optimized to predict horizontal, vertical, 45^o, and 135^o diagonally oriented edge-blocks, respectively. An integrating unit, called gating network, is then used to select or combine the outputs of the experts in order to form the final output of the modular neural network vector predictor. Experimental results show that the modular vector predictor gives an improvement of $1.7dB$ (in an open-loop design) when compared to a single multi-layer perceptron (MLP) predictor. The perceptual quality of the predicted images is also improved significantly.

A closed-loop approach is used for the PRVQ design. In the closed-loop approach, the modular vector predictor is first designed based on the original previous training vectors (open-loop design). The residual vector quantization (RVQ) codebooks are then iteratively optimized for by fixing the predictor in a closed-loop fashion, where the prediction is based on the previously reconstructed vectors. The generalized Lloyd algorithm (GLA) is used for designing the stage codebooks in the closed-loop design technique.

Since the gating network in the modular vector predictor is trained such that its output produces the probability that the input vector belongs to a particular class, we may choose the largest output value of the gating network as the class of the current block. Thus, the gating network *predicts* the class of the current block. This class information can be used to select one of several sets of codebooks where we may create several separate sets of codebooks for several classes of blocks. Thus, each set of codebooks is optimized for a particular class of blocks. In this way, the edge accuracy of the edge vectors can further be improved. Note that the class information is available at the decoder, thus, no class information is needed to be sent.

2. PREDICTIVE RESIDUAL VECTOR QUANTIZER WITH MODULAR PREDICTOR

Figure 1 shows the functional diagram of the proposed PRVQ scheme. The PRVQ employs a modular neural network vector predictor and a modular RVQ. Each component RVQ in the modular RVQ is referred to as *class-RVQ*. The vector prediction system consists of five dedicated predictors (experts), where each expert predictor is optimized for a particular class of input vectors. An input vector is presented to all the predictors and a gating network is then used to select or combine the outputs of the experts in order to form the final output of the modular network. The residual vector, $\mathbf{e_1}$ is quantized by using the appropriate class-RVQ.

Previous research shows that the improvement by using a modular neural network vector predictor results not only in

a higher prediction gain but also the edges in the predicted block are reproduced with high fidelity [2]. In the modular neural network vector predictor, expert predictors *cooperate* in the prediction process. When the integrating unit is well designed, the current block can be predicted with improved accuracy by its corresponding class expert predictor, since each expert predictor has been trained for predicting a particular class of input vectors. It will also enhance performance of a PRVQ, since the increase in prediction gain results in an improvement in overall coding gain.

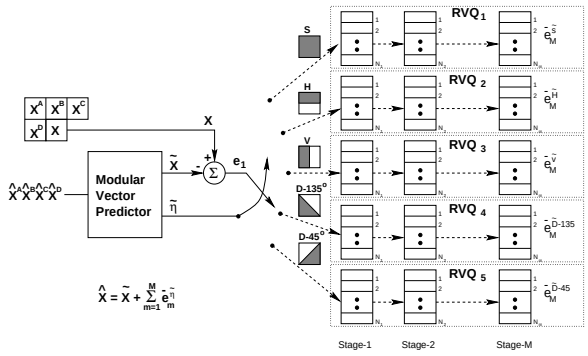

Figure 1: The proposed PRVQ scheme.

3. MODULAR NEURAL NETWORK VECTOR PREDICTOR

In the vector predictors employed in this coding scheme, the prediction of the current block at location (m, n) is estimated by using four causal neighboring blocks located at $(m-1, n-1)$, $(m-1, n)$, $(m-1, n+1)$, and $(m, n-1)$, as shown in Figure 2.

$$\tilde{\mathbf{X}} = \mathbf{\Phi}\left(\hat{\mathbf{X}}_A, \hat{\mathbf{X}}_B, \hat{\mathbf{X}}_C, \hat{\mathbf{X}}_D\right) \tag{1}$$

where $\mathbf{\Phi}(\cdot)$ is a non-linear function.

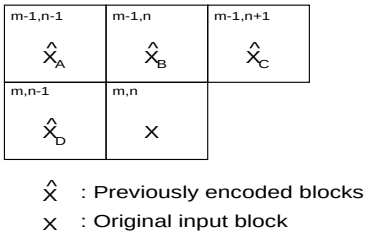

Figure 2: Geometry of block (vector) prediction.

A modular neural network consists of several *expert* neural networks (modules), where each expert is optimized to perform a particular task of an overall complex operation. An integrating unit, called *gating network*, is used to select or combine the outputs of the modules (expert networks) in order to form the final output of the modular network.

In statistics, this idea of modularity is referred to as the mixture model [4]. Mixture models have been used in a

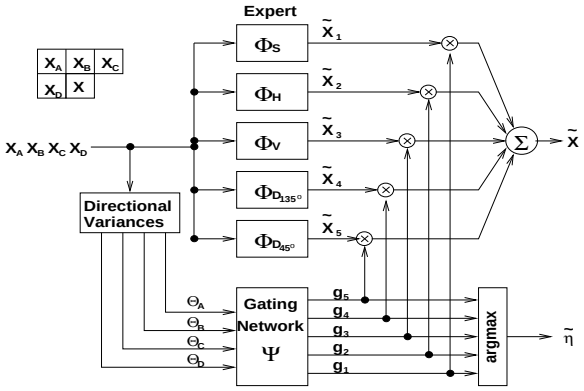

Figure 3: Modular neural network vector Predictor.

wide variety of applications where data derived from two or more categories are mixed in varying proportions [4]. A data sample y in a data set which is approximated by a mixture model by giving input x can be represented by a linear combination of outputs of several distinct functions $y_k(\cdot)$, where $k = 1, 2, \ldots, K$.

$$y = \sum_{k=1}^{K} \pi_k \, y_k(x), \tag{2}$$

where

$$0 \leq \pi_k \leq 1 \qquad k = 1, 2, \ldots, K \tag{3}$$

and

$$\sum_{k=1}^{K} \pi_k \;=\; 1. \tag{4}$$

The parameters π_k are called mixing weights and the functions $y_k(\cdot)$ are called mixing components, where $k = 1, 2, \ldots, K$. Each function $y_k(\cdot)$ is responsible for approximating data y in its corresponding category k, while the parameter π_k gives a weight to the functions $y_k(\cdot)$. If the output of one function can closely approximate a data sample, its corresponding weighting factor should be high. On the contrary, if the output of one function is far away from a data sample, its corresponding weighting factor should be low. We may consider in stochastic processes that the data is obtained by selecting function $y_k(\cdot)$ with probability π_k and then by calculating function $y_k(\cdot)$ for data y.

The architecture of a modular neural network consists of K ($K = 5$) expert networks and one gating network, shown in Figure 3. The expert networks share the same inputs $\{\mathbf{X}_A, \mathbf{X}_B, \mathbf{X}_C, \mathbf{X}_D\}$, which are the pixel values of four neighboring blocks, whereas the gating network accesses the block variances and directional variances of four neighboring blocks. We may regard the block variances and directional variances as extracted features for classification task. The task of each expert network is to predict image-blocks that belong to a particular class. The desired response of each expert network is the current block. The task of the gating network is to assign one expert network to each input. The desired response of the gating network is the class of the current block. The

final output of a modular neural network is a weighted linear combination of the expert networks given by

$$\tilde{\mathbf{X}} = \sum_{k=1}^{K} g_k \, \tilde{\mathbf{X}}_k. \tag{5}$$

where weighting factors, g_k, are given by the gating network.

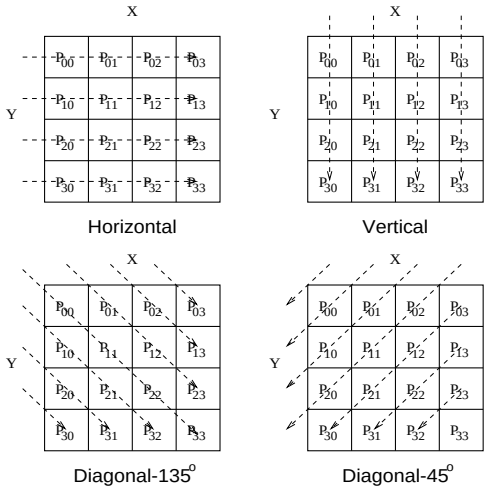

Figure 4: The directional variances calculated by the interest operator.

The first step of the partitioning algorithm is to assign a class label to each of the training blocks out of the five directional classes: stationary (S), horizontal (H), vertical (V), diagonal-135^o (D-135^o), and diagonal-45^o (D-45^o). A feature extraction technique, called the interest operator [5], is used to find the contour directions of a vector, shown in Figure 4. The interest operator finds the variances in the horizontal, vertical, and both diagonal directions for each block in the image.

All the training blocks are first classified as either stationary or edge blocks by only using the block variance σ. A simple thresholding criterion is used for this classification; that is, if the block variance does not exceed a given variance threshold λ_σ, then that block is assumed to have low *textural activity* and is labeled as a stationary block. Otherwise, the block is labeled as an edge block. The edge blocks are then further classified into one of the four directional classes. The directional class of an edge block, η, is identified by selecting the directional class with the least directional variance.

$$\eta \quad = \quad \begin{cases} \text{Stationary} & \sigma \le \lambda_\sigma \\ \arg\min_c \sigma_c & \text{otherwise,} \end{cases}$$

where c = { H, V, $D-135^o$, and $D-45^o$ }. The variance threshold λ_σ is found experimentally.

Once all of the training vectors are assigned a class label, we can now design each individual expert by using only the subset of the training data that belongs to that particular class. All of these experts are designed by minimizing the mean square error (MSE) between the desired (target) and the output vectors. It is an auto-regression problem, where the inputs of each expert are the four causal neighboring blocks { $\mathbf{X}_A$, $\mathbf{X}_B$, $\mathbf{X}_C$, $\mathbf{X}_D$ } and the desired response $\mathbf{X}$ is the current block. The predicted block, $\tilde{\mathbf{X}}_k$, is given by

$$\tilde{\mathbf{X}}_k = \mathbf{\Phi}_k(\mathbf{X}_A, \mathbf{X}_B, \mathbf{X}_C, \mathbf{X}_D), \tag{6}$$

where $\mathbf{\Phi}_k(\cdot)$ is the prediction function performed by expert network k, and k = { S, H, V, $D-135^o$, and $D-45^o$ }. We implement each expert network by using a multi-layer perceptron (MLP) neural network with one hidden layer.

The gating network is designed in order to combine the outputs of the expert predictors. The gating network is designed such that its outputs estimate the probabilities that a given block is predicted by expert networks. This is accomplished by applying a transformation, called soft max [3], at the output of the gating network. Soft max can be regarded as the smooth version of the winner-takes-all activation model. The softmax function at the output layer also implements the (soft) competition between the expert networks. The goal of training the gating network is to minimize the probability of misclassification. We also implement the gating network by using a multi-layer perceptron (MLP) neural network with one hidden layer.

4. EXPERIMENTAL RESULTS

The proposed PRVQ scheme is designed with a two-stage residual vector quantizer, using the block size of 4×4 for the peak bit rates of 0.75 bit per pixel (bpp) (two residual quantizers of size 64 each). Table 1 shows some of the preliminary results for a test image Lena, in terms of average bit rate (indices are entropy encoded) and Peak Signal to Noise Ratio (PSNR). The first scheme, *Proposed-1*, uses only one set of two-stage residual vector quantizer for all classes of residual vectors. The second scheme, *Proposed-2*, uses separate set of two-stage residual vector quantizer for each class of residual vectors. The third scheme, *Proposed-3*, uses only one-stage residual vector quantizer for the class of stationary vectors, which is recognized as an easy-to-encode class, and uses 4 separate sets of two-stage residual vector quantizers for the other four classes. It can be seen from the Table 1 that the *Proposed-1* outperforms the equivalent PRVQ scheme by $0.71dB$ at a lower bit rate (around 8% lower). The improvement is due to the high prediction gain of the modular vector predictor. The *Proposed-2* further exploits the ability of the modular vector predictor by applying separate sets of residual codebooks. It gives an improvement of $1.11dB$ over the equivalent PRVQ scheme at a similar bit rate. Moreover, the bit rate could be further reduced when the proposed PRVQ scheme uses only one-stage residual vector quantizer for the class of stationary vector and uses two-stage residual vector quantizer for the other four classes, since the class of stationary vectors is recognized as an easy-to-encoded class. This scheme gives an extra saving of $0.187bpp$ (33%), when the PSNR drops only $0.55dB$. Compared to the equivalent PRVQ scheme, the *Proposed-3* has a higher PSNR ($0.54dB$)

and a lower bit rate (0.164*bpp*). The original and reconstructed images are shown in Figure 5.

Table 1: Performance comparison in terms of average Bit-Rate and PSNR for the test image Lena. The peak bit rate is 0.75 bit/pixel (bpp). Indices are entropy encoded.

VQ Schemes	**Average Bit-Rate (entropy)**	**PSNR** *dB*
PRVQ [1]	0.5462 *bpp*	34.008
Proposed-1	0.5055 *bpp*	34.721
Proposed-2	0.5686 *bpp*	35.124
Proposed-3	0.3819 *bpp*	34.548

References

[1] S. A. Rizvi and N. M. Nasrabadi, "Predictive residual vector quantization," *IEEE Trans. Image Processing*, vol. 4, no. 11, pp.1482-1495, Nov. 1995.

[2] L.-C. Wang, S. A. Rizvi, and N. M. Nasrabadi, "A modular neural network vector predictor for predictive VQ," *IEEE Int. Conf. Image Processing 1996*, Lausanne, Switzerland, Sept. 16-19, 1996.

[3] C. M. Bishop, *Neural Networks for Pattern Recognition*, Clarendon Press, 1995.

[4] D. M. Titterington, A. F. M. Smith, and U. E. Markov, *Statistical Analysis of Finite Mixture Distributions*, John Wiley & Sons, 1985.

[5] H. Moravec, *Robot Rover Visual Navigation,* Ann Arbor, MI: U.M.I. Research Press, 1981.

(a)

(b)

(c)

(d)

Figure 5: (a) Original image "Lena," (b) reconstructed test image "Lena" using *Proposed-1*, (c) reconstructed test image "Lena" using *Proposed-2*, (d) reconstructed test image "Lena" using *Proposed-3*.