Classification Using Vector Quantization

Karen L. Oehler Pamela C. Cosman Robert M. Gray Jack May*

Information Systems Laboratory
Department of Electrical Engineering
Stanford University
Stanford, CA 94305-4055

*ESL Incorporated
MS 408
495 Java Drive
Sunnyvale, CA 94086-3510

Abstract

We describe a simple technique for combining vector quantization and low level classification of images. The goal is to classify automatically certain simple features in an image as part of the compression process in order to enhance their appearance in the reconstructed image. Images in the training sequence are divided into blocks and each block is classified into a particular class by a human observer. This a priori knowledge is then used when designing the codebook so that both small average distortion and accurate implicit classification are achieved. The codebook can also be designed to have different average distortions for the different classes. The technique is a variation on a variable rate tree-structured vector quantizer which is grown by splitting a single terminal node at each iteration. The splitting criterion selection allows tradeoffs among compression rate, distortion and misclassification rate.

1 Introduction

Because digitized images require tremendous memory storage and transmission bandwidth, image compression is often a necessity. Vector quantization (VQ) has for several years been used for image compression [1, 2, 3, 4]. We describe a simple technique for combining vector quantization and low level classification of images in order to enhance their appearance in the reconstructed image. The cluster analysis and nearest neighbor techniques used for vector quantization have also long been used for traditional pattern classification [5]. The two applications differ primarily in the measures of algorithm quality and cost and in the fact that pattern classification often implies an *a priori* set of desired patterns while in quantization the design algorithm can choose the templates to best approximate the possible inputs. Because of the similarity of the two methods, it is natural to consider combinations that both compress and classify. One possible use of this combination is to highlight regions in the reconstructed image belonging to a specific class, in order to draw attention to certain features. Another use is to develop codebooks which have improved representations for those classes that are deemed important.

2 Codebook structure

A tree-structured vector quantizer (TSVQ) is used because it has the advantage of greatly reduced search complexity over full search VQ. Traditionally, a balanced TSVQ is designed one layer at a time and implements a fixed rate code [6]. Makhoul *et al.* [7] introduced an alternative design algorithm that grows the tree one node at a time by splitting the node that contributes most to the overall distortion of the coder. This method results in an unbalanced tree because the node that is split can be at any depth. Because unbalanced trees generally produce lower distortion than balanced trees, we consider unbalanced TSVQ.

Another unbalanced tree design algorithm is presented in [8]. Here, a technique from classification and regression tree design is extended to variable rate coding. In [9], decision trees are grown by splitting one node at a time. An "impurity function" measures the lack of homogeneity of a particular node in a tree T. The "goodness" of a candidate split of node t is defined as the decrease in node impurity that it effects and is given by

$$\Delta i(s,t) = i(t) - p_L i(t_L) - p_R i(t_R). \qquad (1)$$

Here, s is a binary test (a nearest neighbor selection in a TSVQ), $i(t)$ is the impurity measured at node t of tree T (average distortion in a TSVQ), p_L is the proportion of the samples in node t that go to the left child t_L, and p_R is the proportion that go to the right child t_R.

For a TSVQ with a squared error distortion, the

1058-6393/91 $01.00 © 1991 IEEE

impurity function becomes

$$i(t) = d(t) = \frac{1}{\|\mathcal{T}_t\|} \sum_{k:x_k \in \mathcal{T}_t} d(\mathbf{x}_k, \hat{\mathbf{x}}_t) \qquad (2)$$

where $\mathcal{T}_t$ is the set of all training vectors $\mathbf{x}_k$ mapping into node t, $\|\mathcal{T}_t\|$ is the number of vectors in $\mathcal{T}_t$, $\hat{\mathbf{x}}_t$ is the reproduction codevector associated with node t, and $d(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^t(\mathbf{x} - \mathbf{y})$. In growing an unbalanced TSVQ, the terminal node measuring the highest $\Delta d(s,t)$ is always split. Let D and R stand for the distortion and rate, respectively, measured by T, and let D' and R' stand for the distortion and rate of the tree after t is split into t_L and t_R. Let $\Delta D = D' - D$ and $\Delta R = R' - R$ be the change in the distortion and rate, respectively, due to splitting t. The splitting criterion from [8] is

Criterion Split the node that provides the largest ratio of decrease in distortion to increase in rate, i.e. split the node with the largest value of

$$-\frac{\Delta D}{\Delta R} = d(t) - p_L d(t_L) - p_R d(t_R) = \Delta d(s,t). \qquad (3)$$

It was found that such trees outperform balanced trees of equivalent bit rates by up to 3.5 dB at high rates on a magnetic resonance image database [8]. Observe that average squared error is simply one example of a node impurity measure that can be used to grow TSVQs. We will here consider other measures.

3 Hand-labeled classification

The problem chosen here is to highlight areas in images that are deemed to belong to a specific category. In aerial images, image sublocks that are suspected of being "manmade" as opposed to "natural" are highlighted to attract attention of human observers. The classification of the training set of aerial photographs is done by hand labeling those features (man-made and natural regions) that are to be recognized in subsequent images. This *a priori* knowledge is used when designing the TSVQ codebook so that both small average distortion and accurate implicit classification are achieved. Hence, the TSVQ encoder is designed to classify image subblocks while compressing the image. Several codebook design methods are possible with differing compression and classification abilities on the test images. The splitting criterion selection allows tradeoffs among compression rate, distortion, and misclassification rate.

The new approach taken here is that the classification is done simultaneously as the image is encoded. One codebook search is sufficient to both encode a subblock and to classify it into one of the previously determined categories. Stored with each codeword in the codebook is a class label representing the best class prediction for image subblocks that will be represented by that codeword. Thus, once the encoder selects the most appropriate codeword, the preliminary classification of the subblock at hand is simply a matter of memory lookup; no other computations are required. In effect, we are getting this classification knowledge "for free." A simple means of incorporating classification into a TSVQ is to classify each terminal node according to a majority vote of the *a priori* class assignments of the training vectors that were represented by that node after encoding. This makes sense if the costs of misclassifying the image subblocks of the various classes are equal. However, if the error costs are unbalanced, then a more sophisticated weighting function would be appropriate.

It is important to point out similarities and differences between this approach and Classified Vector Quantizers (CVQ) [10]. Both approaches divide the training sequence vectors among different categories or classes. While the CVQ algorithm constructs a VQ codebook for each class, the algorithm described here constructs only one VQ codebook using the class information. In [10], the key goal was to code edges with higher visual quality and to reduce the complexity of the overall encoder by creating many independent subcoders. Here the key goal is to recognize and "highlight" specific classes; complexity is not reduced by the classification information, but the encoder is already low complexity due to its tree structure.

3.1 Alternative splitting criteria

The codebooks are constructed by splitting one node at a time. The splitting criterion, i.e. the methodology for deciding which node to split next, can have a strong effect on the quality of the codebook. Three different criteria are investigated:

Criterion 1 Ignore the classification information and split the node with the largest goodness of split as defined in Equation 3 where the distortion measure is mean-squared error. This design yields an ordinary VQ for comparison [8].

Criterion 2 Split the node that has the greatest percentage of misclassified training vectors, i.e. split

the node with the largest value of

$$\lambda_2 = \frac{\text{(Number of misclassified vectors)}}{\text{(Total number of vectors)}}. \quad (4)$$

This corresponds to measuring impurity by the Hamming distance between the node class, c_t, and the hand-labeled class, $c(x)$. Thus, if $d_H(x,y) = 0$ if $x = y$ and $d_H(x,y) = 1$ if $x \neq y$, then the impurity of node t is given by

$$d(t) = d_H(t) \equiv \frac{1}{\|\mathcal{T}_t\|} \sum_{k:x_k \in \mathcal{T}_t} d_H(c(\mathbf{x}_k), c_t) \quad (5)$$

from Equation 2. The node with the highest such impurity is split. Thus the encoder nearest neighbor mapping and the centroid reproduction levels are chosen to minimize squared error, but the tree is grown to reduce classification error.

Criterion 3 Split the node that has the greatest number of misclassified training vectors. This is equivalent to splitting the node with the largest impurity defined by the partial Hamming distortion $d(t) = \|\mathcal{T}_t\| d_H(t)$.

Note that the latter two splitting criteria are not formatted in the traditional "goodness" of split criterion as are criterion 1 and [8], but are more similar to the splitting criterion used in [7]. Because the classification of each node is based on majority rule, splitting criteria based on classification error can lead to a desirable split having a zero "goodness" of split.

3.2 Experimental Results

The training set consisted of 5 images provided by ESL, Inc. The images were 512 × 512 pixels of 8 bit grayscale consisting of aerial photography of the San Francisco Bay area. Each 16×16 pixel subblock in the training set was assigned to be either "man-made" or "natural" based on the perceptions of a human observer. While the training vectors were classified in 16×16 vectors to simplify the task of the human classifier (even using the 16×16 subblocks, over 5000 decisions had to be made), the codebook construction and image encoding were carried out using 4×4 pixel vectors. This coarse resolution is partially responsible for the relatively low classification abilities demonstrated with the data. Classification ability is defined as the percentage of vectors classified correctly by the TSVQ, compared to the classification standard created by the human observer.

Sets of codebooks having either the same rate or the same number of nodes were constructed to allow for comparison. Results for codebooks at a rate of 0.5 bpp are shown in Table 1. In general, the first splitting criterion provided the lowest mean squared error in the encoded image at the expense of relatively poor classification ability. The latter two splitting methods provided poorer encoded images (much more blocky in appearance) but better classification ability. Criterion 3 classified more vectors as man-made than criterion 2. For a given number of nodes, criterion 2 produced a higher rate code than 1, and 3 higher than 2. Likewise, for a given rate the tree structure produced by criterion 1 has substantially more nodes than criterion 2, and 2 more nodes than 3. Choosing the splitting criterion involves a tradeoff between compression and classification quality; or similarly, a tradeoff between rate and memory requirements (memory being proportional to the number of nodes.)

Images outside the training sequence were encoded with the resulting codebooks. Because the simultane-

Figure 1: Original aerial image

Criterion:	1	2	3
Training PSNR (dB)	26.7	25.4	25.1
Training classification ability	0.72	0.74	0.70
Test PSNR (dB)	23.4	22.8	22.7
Test classification ability	0.71	0.74	0.75

Table 1: PSNR and classification ability using TSVQ codebooks grown using various splitting criteria for both the training sequence and the test image.

Figure 2: Image compressed at 0.5 bpp using compression/classification encoder designed using Criterion 2.

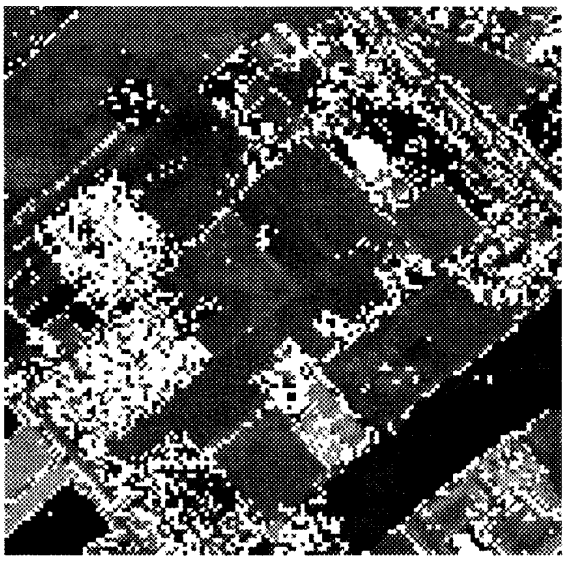

Figure 4: Natural sublocks using encoder designed using Criterion 2. Man-made subblocks are replaced by solid white subblocks.

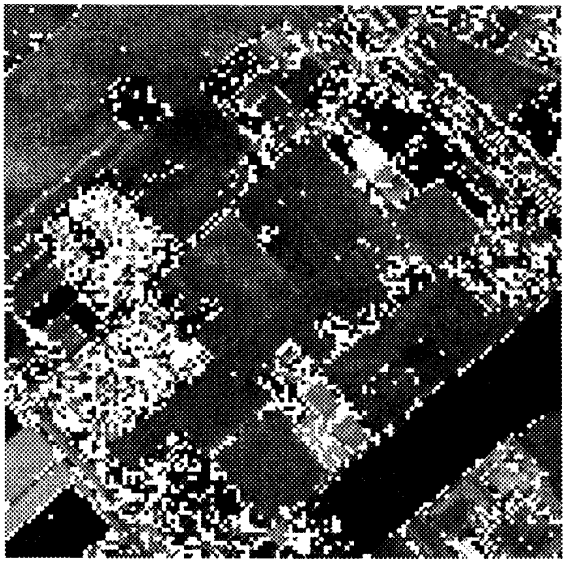

Figure 3: Natural sublocks using encoder designed using Criterion 1. Man-made subblocks are replaced by solid white subblocks.

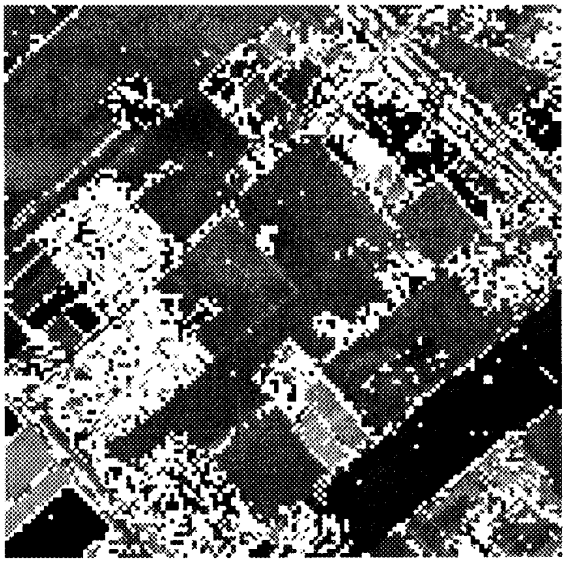

Figure 5: Natural sublocks using encoder designed using Criterion 3. Man-made subblocks are replaced by solid white subblocks.

ous display of the compression and classification results is difficult in a grayscale format, the results of compression and classification are shown separately. Figure 1 shows the original test image (not in the training set) that was used to evaluate the different codebooks. Figure 2 shows the image after compression at 0.5 bpp using the codebook constructed with Criterion 2 (results from the other criteria are similar.) Figures 3, 4 and 5 demonstrate classification from codebooks constructed with Criteria 1, 2, and 3 respectively. The images show the subblocks that the encoder classified as natural, whereas the man-made subblocks are replaced by solid white subblocks. Although this information is not as amenable to grayscale display, the images shown here reflect the accuracy of the classification encoder. The classification ability was modest; at 0.5 bpp the best classification encoder still had 25% misclassification rate on the training sequence, that is, one in four decisions disagreed with the hand-labeled classes. However, the resulting error rate was partly due to the fact that the hand-labeling of the training and test images was performed at one-fourth the resolution that the classifier was trying to achieve. This resolution mismatch not only degrades the training sequence but also produces an imperfect "gold standard" classification for the test image. Since the classification ability is computed with respect to the hand-labeling of the training and test images, it is also affected by the human observer's perception and consistency limitations.

Ideally, the compressed images would be viewed on a color monitor so that classification information can be indicated by color superimposed on the grayscale compressed image. Such a contrast makes the natural and man-made features of the image easier for a human viewer to differentiate.

3.3 Comparison with other classification methods

The classification results were compared with results obtained using the CART™ algorithm, described in [9]. CART uses training data to construct a decision tree for classification. Here, we considered decision trees constructed as a series of binary splits, where each split is conducted along one coordinate in the training data. The training vectors were preprocessed to obtain new vectors consisting of the minimum, maximum, average and standard deviation of pixel values within the vector. CART trees developed with these preprocessed vectors showed better classification ability than trees developed with the original training vectors. This was due to the fact that splitting along a single coordinate in the image vector is not as efficient as splitting along a function of several coordinates. A classification tree was trained using preprocessed vectors from the same 5 images as in Section 3.2, and was applied to the same test image. A compressed version of the test image (compressed to 0.5 bpp using the "standard" TSVQ tree built with Criterion 1) was also classified using the CART tree. The results are shown in Table 2. This preprocessing allowed CART to classify test images very well, considering the limitations on the training sequence. However, there was notable degradation in classification ability on the compressed image.

	CART
Test classification ability	0.81
Comp. Test classification ability	0.72

Table 2: Classification ability using CART algorithm for both the test image and a 0.5 bpp compressed version of the test image.

On the test image, CART performed better than the vector quantizer classifier; however, CART performed worse on the compressed test image. In general, the two classification methods are comparable.

4 Automatic classification based on input-weighted distortion measures

In the previous experiment, we used the classified training vectors to grow the tree in such a way that it attempts to keep vectors of the same class together. Instead of, or in addition to, this classification purpose, classified training vectors can also be used for an enhancement purpose: the tree can be grown so that it does a better job of encoding certain classes. We consider two examples of this type of enhancement. Both examples involve an automatic classification of the training vectors: in one case a classification by brightness, and in the other, by texture. A weight is associated with each class, and the tree is grown with a weighted distortion measure that causes the tree to have "growth spurts" for certain types of inputs that have been declared *a priori* to be important, and to become "stunted" for inputs that are less important. Further information on this approach can be found in [11].

The most common distortion measure for growing and pruning TSVQs is the squared error. We now consider the splitting criterion of Equation 3 with instead

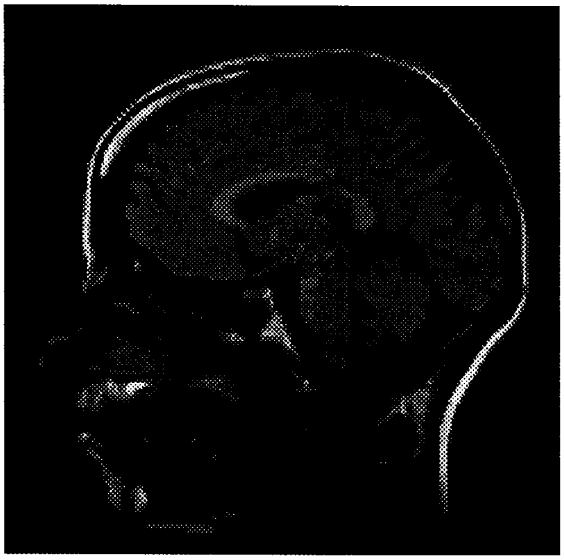

Figure 6: Compressed image at 0.75 bpp: regular distortion measure.

Figure 7: Compressed image at 0.75 bpp: weighted distortion measure.

an input-weighted distortion measure of the form

$$d(\mathbf{x},\mathbf{y}) = (\mathbf{x}-\mathbf{y})^t \mathbf{W}_x (\mathbf{x}-\mathbf{y}), \qquad (6)$$

where $\mathbf{W}_x$ is a strictly positive definite weighting matrix that depends on the input $\mathbf{x}$. In our examples, $\mathbf{W}_x$ will equal $w_x\mathbf{I}$, where $\mathbf{I}$ is the identity matrix, and w_x is a weight assigned to each training vector $\mathbf{x}$ according to its classification. Training vectors which are more important according to some criterion will be assigned higher weights. Thus nodes having high concentrations of training vectors from important classes will have an inflated ΔD and therefore will split earlier and more often than they would in the unweighted case.

4.1 Classification based on intensity

A classification based on intensity, in which bright training vectors are weighted more heavily than dark ones, is appropriate for MR brain scans because the bright parts of the image correspond typically to what is medically most important in the image. A training sequence consisting of 8 MR brain scans was blocked into 2×2 pixel blocks, and each vector was automatically assigned a weight proportional to its energy. A tree was grown to 2 bpp using the weighted distortion measure for splitting, and pruned back to 0.75 bpp. The tree was evaluated on images not in the training sequence. The same training sequence without weights assigned was used to grow an unweighted unbalanced tree according to the original greedy growing algorithm. The compressed images at bit rates of 0.75 bpp produced by the two different trees are shown in Figures 6 and 7. The image made from the weighted tree looks better in the bright regions (e.g., the cortex and cerebellum) which generally correspond to the diagnostically important part of these images.

4.2 Classification based on texture

Due to pattern masking, the human visual system is generally less sensitive to noise in highly active or textured regions of an image, and we used our class-weighted distortion measure to effect a redistribution of quantization noise into regions of more texture. The training sequence consisted of six USC database images, blocked into 4×4 vectors. A weight was assigned to each training vector according to how many of the pairs of adjacent pixels had differences exceeding some threshold. Highly textured vectors having many pairs exceeding the threshold were assigned low weights, and highly homogeneous vectors were assigned large weights. A tree was grown to 2 bpp using the weighted distortion for the splitting criterion, and pruned back to 0.54 bpp. The test image showed clouds, a lake, and

trees. A comparison of compressed images encoded from the weighted and unweighted trees showed that the one from the weighted tree had less distortion in the cloud regions, where distortion is most noticeable, and it had more in the areas of trees, where the high texture masks the noise.

5 Conclusions

We have used classification information in conjunction with a greedy growing algorithm to grow unbalanced TSVQ. Hand-labeling of aerial images was used to grow TSVQ which combine moderate classification ability with encoding, useful in highlighting the reconstructed images. The classification schemes used so far have classified each training vector without regard to its context in the training image. It might be possible to develop classifiers which incorporate more contextual information. Classified training sequences were also used to redirect quantization noise to different parts of compressed images, by using a class-weighted distortion measure. A weighting based on brightness for medical images and one based on texture for outdoor scenes both produced weighted trees that significantly improved the perceptual compressed image quality.

Acknowledgements

This work was supported in part by ESL, Incorporated, the National Science Foundation under Grant MIP-9016974, the National Institutes of Health under Grant CA49697-02, and a National Science Foundation Graduate Fellowship.

References

[1] R. M. Gray. Vector quantization. *IEEE ASSP Magazine*, 1,No. 2:4–29, April 1984.

[2] N. M. Nasrabadi and R. A. King. Image coding using vector quantization: A review. *IEEE Transactions on Communications*, COM-36:957–971, August 1988.

[3] H. Abut, editor. *Vector Quantization.* IEEE Reprint Collection. IEEE Press, Piscataway, New Jersey, May 1990.

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

[5] M. R. Anderberg. *Cluster Analysis for Applications.* Academic Press, San Diego, 1973.

[6] A. Buzo, A. H. Gray, Jr., R. M. Gray, and J. D. Markel. Speech coding based upon vector quantization. *IEEE Trans. Acoust. Speech Signal Process.*, ASSP-28:562–574, October 1980.

[7] J. Makhoul, S. Roucos, and H. Gish. Vector quantization in speech coding. *Proc. IEEE*, 73. No. 11:1551–1587, November 1985.

[8] E. A. Riskin and R. M. Gray. A greedy tree growing algorithm for the design of variable rate vector quantizers. *IEEE Trans. Signal Process.*, November 1991. To appear.

[9] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth, Belmont,California, 1984.

[10] B. Ramamurthi and A. Gersho. Classified vector quantization of images. *IEEE Trans. Comm.*, COM-34:1105–1115, November 1986.

[11] P. C. Cosman, K. L. Oehler, A. A. Heaton, and R. M. Gray. Tree-structured vector quantization with input-weighted distortion measures. In *Proceedings Visual Communications and Image Processing '91*, Boston, Massachusetts, November 1991. SPIE- The International Society for Optical Enginering.