# Combining Image Compression and Classification Using Vector Quantization

Karen L. Oehler, *Member, IEEE*, and Robert M. Gray, *Fellow, IEEE*

***Abstract*—Statistical clustering methods have long been used for a variety of signal processing applications, including both classification and vector quantization for signal compression. We describe a method of combining classification and compression into a single vector quantizer by incorporating a Bayes risk term into the distortion measure used in the quantizer design algorithm. Once trained, the quantizer can operate to minimize the Bayes risk weighted distortion measure if there is a model providing the required posterior probabilities, or it can operate in a suboptimal fashion by minimizing only squared error. Comparisons are made with other vector quantizer based classifiers, including the independent design of quantization and minimum Bayes risk classification and Kohonen's LVQ. A variety of examples demonstrate that the proposed method can provide classification ability close to or superior to LVQ while simultaneously providing superior compression performance.**

***Index Terms*—Image compression, image classification, vector quantization, image coding, statistical clustering.**

## I. INTRODUCTION

IMAGE compression and classification play important roles in making digital images useful. Consider a typical computerized tomography (CT) image consisting of a matrix of gray-scale pixel values: 512 pixels × 512 pixels × 12 bits resolution. This format requires 393 kilobytes of memory per image. Given the steadily increasing number of images produced by hospitals, compression is increasingly important for prolonging local storage and speeding transmission. Because lossless or invertible compression can typically provide only a compression ratio of less than 2:1–4:1, lossy compression techniques become necessary to achieve significant compression. When storage or communication capacity dictates the use of lossy compression, the overall goal is to preserve the highest fidelity possible, ideally producing a compressed image which either has no perceptual difference from the original or which has only statistically insignificant differences from the original in a specific application, e.g., medical diagnosis or interpretation [1]. Improved quality can always be achieved at the expense of higher bit rates, but good compression systems must provide a trade-off between fidelity and bit rate that is good if not optimal. In attempting to obtain an efficient and accurate representation of an original image for a reduced number of bits, a fundamental goal of compression is to extract the information in an image important for a particular application.

In many scientific and medical applications, images are used by human experts to make specific decisions or inferences. The actions of human experts can sometimes be mimicked by sophisticated computer algorithms based on ideas from image analysis and expert systems design. These algorithms can either assist a human observer in identifying features or provide an automated second opinion as a check. For both human and computer decisions, one can measure the quality of an image processing algorithm at least in part by how accurate the decisions based on the processed image are in comparison with those based on the original. Image analysis is a broad and complicated area, and in general sophisticated techniques involve large amounts of computer processing on entire images in order to reach decisions. Simple low-level local classification involving only small regions of an image, however, can assist human observers by highlighting special areas of interest and can simplify further classification/analysis algorithms by incorporating preprocessing into the digital representation. For some applications, such local classifiers may provide performance competitive to that using the sophisticated full-frame methods, especially in light of their relative simplicity.

Classification of small groups of pixels can be viewed as a form of compression since it extracts the information of primary interest in a specific application, or it can be viewed as a form of image enhancement, since it provides a segmentation of an image into disjoint regions labeled by class membership. For example, classification of microcalcifications and masses in digital mammograms permits a radiologist to add false color to regions classified as suspicious, providing an automated second opinion. Conversely, compression can be viewed as a form of classification since it assigns a template or codeword in a small set to input pixel groups drawn from a large set in such a way as to provide a good approximation. All inputs yielding a common binary representation can be considered to belong to a common class, and compression is achieved by replacing the original input by a class label. Not surprisingly, these similarities of goals between compression and classification have resulted in a long history of similar methods for algorithm design and occasional direct applications from one field to the other.

A principal difference between compression and classification is the choice of cost measures used to quantify performance. In compression one measures image quality (or rather the lack of quality) by an average distortion, typically mean squared

Manuscript received Oct. 13, 1993; revised Aug. 15, 1994. Recommended for acceptance by R. Duin.

K.L. Oehler is with the Integrated Systems Laboratory MS 446, Texas Instruments, Inc., Dallas, TX 75265-5474.

R.M. Gray is with the Information Systems Laboratory, Department of Electrical Engineering, Stanford University, Stanford, CA 94305-4055.

IEEECS Log Number P95043.

error similar to quantitative distortion measure. Average distortion is traded off against bit rate as given by the average number of bits per pixel required by the compressed image. In classification one measures quality by probability of classification error or, more generally, by average Bayes risk, which can be viewed as an average distortion. Average Bayes risk is typically traded off against some measure of the complexity of the algorithm used to do the classification, typically the number of nodes in a decision tree, but possibly also the average number of nodes traversed when making a decision using a tree-structured algorithm. In both cases an average distortion is traded off against something like bit rate. Another difference between compression and classification is that in the latter the classes are usually known beforehand, whereas in the former the classes can be chosen in order to maximize the overall quality.

One method which has proved important to both compression and classification is statistical clustering. In general the problem is to assign a label to every input in such a way as to minimize an average distortion (such as squared error for compression or Bayes risk for classification) subject to a constraint on the complexity of the algorithm (such as average bit rate for compression or average computation for classification).

The combination of compression with classification is not new, but previous work has concentrated on systems that either focused entirely on using a compression algorithm to perform classification without regard to optimality of either compression or classification, or that simply cascaded a compression algorithm with a classification algorithm designed for the outputs of the compressor. The former systems include isolated utterance speech recognition where several compression algorithms, one for each utterance, are applied to the input. The one that works best (yields the lowest distortion) identifies the class [2], [3], [4]. The latter systems include image compression systems consisting first of compression to yield small average squared error followed by a classifier designed to minimize probability of error given the compressor output [5], [6]. We here describe a means of explicitly combining the two operations of compression and classification into a single code by combining the two quality measures into the distortion measure used to design the code. A Lagrange weighting permits one to weight the compression and classification importance. Perhaps surprisingly, in some problems the inclusion of some compression into a classifier can actually result in classification accuracy nearly equal to that achievable on the original data, while simultaneously obtaining significant compression and reduction in classifier complexity. On the other hand, the inclusion of classification into a compression system can enhance the usefulness of the compressed image by providing a segmentation into interesting classes, while costing virtually nothing in terms of average distortion for a given bit rate. The combination of compression and classification can be useful even in problems where only compression or only classification constitutes the design goal.

The combination of compression and classification into a single mapping can be done in the context of either viewpoint. We adopt the compression viewpoint because it is more commonly described in terms of general distortion measures and it readily extends to the combined system. In this context, a minimum distortion data compression system or source coder can be modeled as a vector quantizer (VQ), a mapping of input vectors into a finite collection of templates or reproduction codewords called a codebook. VQ is a method of lossy compression that applies statistical techniques explicitly to optimize distortion/bit rate trade-offs. (See, for example, [7], [8].) A distortion measure quantifying the cost or loss of reproducing a specific original image by a decompressed reproduction is assumed for both design and implementation of a compression code. VQ operates on individual image subblocks or vectors (e.g., blocks of 4 pixels × 4 pixels). For each $k$-dimensional input vector $X$, the VQ encoder $\alpha$ determines the best match from a collection of $N$ reproduction vectors or codewords, $C = \{\hat{X}_1, \hat{X}_2, \ldots, \hat{X}_N\}$ and puts out the binary representation of the chosen codeword's index; $\alpha(X) = i$ if $\hat{X}_i$ is selected. The sequence of indices so generated can then be either stored or transmitted, depending on the application. The VQ decoder $\beta$ (or "inverse quantizer") reverses this process as it has the index $i$ as input and puts out the reproduction $\hat{X}_i = \beta(i)$. The decoder is a simple table lookup. This low complexity decompression is an advantage for VQ in comparison with other compression techniques requiring computation at the decoder.

The overall action of a VQ $q$ with input $X$ is to produce a reconstruction $q(X) = \beta(\alpha(X))$. The encoder partitions the input vector space into $\mathcal{P} = \{R_1, R_2, \ldots, R_N\}$, where the disjoint cells are defined by $R_i = \{X : \alpha(X) = i\}$. Thus if $X \in R_i$, the encoder is $\alpha(X) = i$ and the decoder is $\beta(i) = \hat{X}_i$. The VQ is completely described by the encoder-decoder combination $(\alpha, \beta)$ or, equivalently, by the partition-codebook combination $(\mathcal{P}, C)$.

The performance of a VQ (or any other compression scheme) is usually measured by two competing attributes: the average bit rate required to perfectly describe the indices $i$ ($k^{-1}\log_2 N$ bits per pixel [bpp] if all of the indices have identical length as binary vectors) and the average of an objective distortion measure. A distortion measure can be used for code optimization by explicitly designing codes to minimize average distortion subject to a bit rate constraint. We here begin with the simple squared error because of its simplicity:

$$d(X, \hat{X}) = \|X - \hat{X}\|^2 = \sum_{j=1}^{k} |X(j) - \hat{X}(j)|^2 ,$$

where $X = (X(1), \ldots, X(k))$ is a $k$-dimensional vector. The method extends to any distortion measure for which Lloyd centroids exist.

Given a distortion measure, a common measure of the performance of a compression scheme is the average distortion

$$D = E[d(X, q(X))] = \int d(x, q(x)) dF_X(x),$$

where $F_X(x)$ is the distribution of the vectors. In the case where $F_X$ is an empirical distribution based on a sample sequence of

vectors $\{x_1, \ldots, x_L\}$, this becomes a sample average

$$D = \frac{1}{L}\sum_{i=1}^{L} d(x_i, q(x)).$$

When the squared error distortion is used, the average distortion is called the mean squared error (MSE), and the result is often reported by normalizing by a constant $D_0$ and giving the result in a negative dB scale to produce a signal-to-noise ratio, SNR = $10 \log_{10} (D_0 / D)$. A common choice for $D_0$ is $E[\| X - E(X) \|^2]$, the minimum average distortion achievable with a 0 rate code. For some image types $D_0$ is set to $k$ times the square of the maximum allowable pixel value, in which case the SNR is called a peak SNR (PSNR).

A common approach to VQ design is to use clustering techniques to minimize the average distortion subject to constraints on bit rate and code structure. The Lloyd clustering algorithm [9], [8] for full search unconstrained VQ involves an iterative application of two optimality conditions:

- For a given decoder $\beta$ (or, equivalently, codebook $C$), the optimal encoder is a minimum distortion or nearest neighbor selection of a reproduction codeword from the codebook. For the case of a squared error distortion, this is the minimum mean squared error (MMSE) or Euclidean nearest neighbor rule.
- For a given encoder $\alpha$ (or, equivalently, partition $\mathcal{P}$), the optimal decoder assigns to each index $i$ the conditional centroid of all input vectors $X$ for which $\alpha(X) = i$. In the case of squared error, the optimal decoder output for an encoder output index $i$ is the conditional expectation $E[X \mid \alpha(X) = i]$.

The Lloyd algorithm begins with an initial code and performs this iteration, successively optimizing the encoder and decoder for each other, until the decrease in average distortion falls below some predetermined threshold. Variations on this algorithm have found extensive application during the past thirty-five years to quantization and to the related area of statistical clustering and classification under other names, including Forgey's algorithm [10] and $k$-means [11].

We next turn to classification and assume a joint probability distribution on a pair $(X, Y)$, where $X$ is an observed random vector of pixels and $Y$ is a "class label" which takes values in a finite set $\mathcal{H} = \{1, \ldots, M\}$; we wish to accurately guess the class $Y$ when only the observable $X$ is known. Analogous to a VQ encoder this is accomplished by a mapping of the input vector into a finite set, which we here call a classifier and denote $\gamma(X) \in \mathcal{H}$. We again can define a distortion measure and performance, but here the usual form is a Bayes risk $B(\gamma)$ defined by

$$\begin{aligned} B(\gamma) &= \sum_{k=1}^{M}\sum_{j=1}^{M} C_{jk} \Pr(\gamma(X) = k \text{ and } Y = j) \\ &= \int dF_X(x) \left[\sum_{k=1}^{M} 1(\gamma(x) = k) \sum_{j=1}^{M} C_{jk} \Pr(Y = j | X = x)\right], \end{aligned} \tag{1}$$

where $C_{jk}$ represents the cost of classifying $X$ as class $k$ when the true class (i.e., $Y$) is $j$ and where $1(\textit{expression})$ is 1 if the *expression* is true and 0 otherwise. The costs $C_{jk}$ can be chosen to reflect the fact that classification errors can have different consequences. We assume that $C_{kk} = 0$ for all $k$. In the case of equal costs for incorrect decisions, i.e., $C_{jk} = 1$ for $j \neq k$ and 0 for $j = k$, the Bayes risk is simply the probability of error, $\Pr(\gamma(x) \neq Y)$ and the minimum Bayes risk classifier becomes a maximum a posteriori (MAP) classifier. In general the optimal classifier in the sense of minimizing the Bayes risk is immediately seen from (1) to be

$$\gamma_{\text{Bayes}}(x) = \operatorname*{argmin}_k \sum_{j=1}^{M} C_{jk} \Pr(Y = j | X = x).$$

Our goal here is to combine classification with the compression process. In particular, instead of viewing the original random vector $X$ of the pair $(X, Y)$ and classifying it using a Bayes classifier, the vector $X$ is first quantized and then classified. Suppose that $q$ is a quantizer with encoder $\alpha$, partition $\mathcal{P} = \{R_1, \ldots, R_N\}$, and decoder $\beta$, then a classifier $\delta$ associates a class label $\delta(i) \in \mathcal{H}$ with each cell $R_i$ for every index (encoder output) $i = 1, \ldots, N$.

The introduction of quantization before classification raises a natural question: Since quantizing will lose information and hence likely damage the classifier performance, would it not be better to always classify based on the original, unquantized, data? There are two distinct responses. First, one can consider a quantizer as simply part of the classifier and one of the free components for design. Even if the overall goal is purely classification, this can yield several benefits. As the bit rate of the quantizer and the size of the training set increase, the performance of an optimal classifier for the quantized data should converge to that of an optimal classifier on the original data [12], [13], [14]. If performance can be made close to the optimum by using a modest bit rate quantizer, the complexity of the overall system can be drastically reduced since the classifier operating on the quantizer output can be implemented as a simple table lookup. The addition of the compression component to a classification problem yields as a by-product a compressed version of the image without seriously reducing the quality of the classifier. This in turn may make the image more useful in future, yet unspecified, tasks requiring a reconstruction of the original image. Finally, the inclusion of the extra constraint naturally provides an algorithm not suggested by the classification goal alone, which nonetheless allows one to weight the classification performance as being the more important criterion and which may eventually yield better classification algorithms. The second response to the rhetorical question is that in some cases a quantizer step is forced on the designer by a storage or transmission capacity constraint. For example, if the original image is analog as are the vast majority of x-rays, then quantization is necessary before a digital classification algorithm can be applied. In this case our goal can be viewed as finding an algorithm for jointly designing a quantizer and classifier in order to yield performance close to that achievable by an optimal classifier operating on the original data.

The cascade of VQ encoder $\alpha$ and decision rule $\delta$ induces a

classification rule on the original vector, $\gamma(X) = \delta(\alpha(X))$. Because the decision rule is constrained by the quantizer partition, it cannot perform better than the Bayes classifier operating on $x$ in the sense of minimizing Bayes risk. Furthermore, if $\gamma$ is an optimal classifier given $X$, it need not be true that $\gamma(\beta(\alpha(X)))$ is also a good classifier except in the limit of large bit rate and asymptotically accurate decompression. The goal here is to jointly design $\alpha$, $\beta$, and $\delta$ so that SNR and Bayes risk are both good.

A simple approach to the problem of combined compression and classification is to separately and independently design the VQ and classifier. For example, a VQ could be designed to minimize mean squared error (which we will refer to as an MMSE VQ) and then a classifier could be designed for the VQ output $\alpha(X)$ so as to minimize average Bayes risk. Hilbert [16] applied Lloyd-like clustering techniques to a training set to produce a code whose outputs were labeled using a maximum likelihood classifier. This provided an early form of independent VQ and classifier design. More recently McLean [5] cascaded a transform VQ with a MAP classifier to perform combined compression and texture classification. We shall refer to this design approach as *independent design*.

The independent design approach has the obvious flaw that the first step uses one optimality criterion, minimizing mean squared error, while the second uses another, minimizing average Bayes risk. This mismatch has been observed in the statistical detection literature and several ad hoc algorithms have been developed for matching scalar quantizer design to the detection of signals in Gaussian noise [17], [18] based on asymptotic approximations of low signal power or high quantizer bit rate.

An alternative approach to the problem is to design a VQ with an MMSE encoder explicitly to operate as a classifier and to ignore the compression aspect. This approach has a rich history and falls in the general class of nonparametric classification methods which use a learning set to estimate the probability densities [19]. The earliest example of using a method formally equivalent to VQ encoding for classification is the classical nearest neighbor (NN) algorithm which makes classification decisions based directly from a labeled learning set [19], [20]. The NN classifier can be viewed as a VQ which has an entire labeled learning set as a codebook. Viewed as a VQ, however, the nearest neighbor algorithm has the obvious fault of having an impractically large codebook as it contains the entire learning set. This implies both large computational complexity for classification and the lack of effective compression, although methods exist for pruning the codebook [21]. More recently several methods have emerged for modifying the nearest neighbor approach by clustering the codebook [22], [23], [24]. Perhaps the best known clustered VQ for classification, however, is the approach of Kohonen et al. [25], [26], called "learning vector quantization" (LVQ). The encoder operates as an ordinary MMSE selection of a representative from the codebook, but the codebook is designed to attempt to reduce classification error implicitly rather than reducing mean squared error. Kohonen argued that for the case of Gaussian data, the partition induced by a VQ can approximate that required for a Bayes estimator; his algorithm is based on this intuition. Kohonen's algorithm has aspects in common with MacQueen's original $k$-means algorithm [11] and Stone's generalized nearest neighbor algorithm [27], and it can be considered a clustered simplification of the nearest neighbor approach. Kohonen's general goal was to imitate a Bayes classifier with less complexity than other neural network approaches, but there is no explicit minimization of classification error in the code design. The ability to compress is not explicitly considered in LVQ.

As an alternative to a full search VQ which performs an MMSE search of the entire codebook, one can use a tree structured VQ (TSVQ). This can provide a significant reduction in complexity and a variable rate code with a progressive structure. For the usual compression application, the codes can be designed by using extensions of ideas from classification and regression tree design [15] combined with clustering algorithms. (See, e.g., [8], [28], [29], [30].) A TSVQ tree is grown by successively splitting nodes and then optimally pruned until the desired rate is reached. The technique also provides a natural modification for performing classification: the node to split next can be chosen as the one which contributes most to the classification error, which forces the tree to grow more in those areas that it had difficulty classifying [31], [32]. Another splitting method that considered classification rather than squared error was introduced by Kramer [33]. He used heuristics to guide the sigmoid least mean square (LMS) training of a linear classifier for each node of a tree-structured classifier.

Our approach strongly resembles other VQ approaches to classification in implementation, but not in design. All VQ approaches use a VQ codebook and encoder which finds the "best" match within the codebook for an input vector. LVQ and other VQ classifiers select the best match using a Euclidean nearest neighbor rule, and the match is used only to classify. The proposed technique can use a Euclidean nearest neighbor rule to do the matching once the code is designed, but a nearest neighbor rule with respect to a modified distortion measure explicitly incorporating both squared error and Bayes risk in a Lagrangian form is used during codebook design. The modified distortion measure can also be used once the codebook design is completed if the posterior class probabilities are known, e.g., if one has a parametric model for the source.

Several examples, including conditionally uniform and Gaussian random vectors, CT lung scans, and aerial images, are considered to demonstrate that the proposed algorithm provides superior or comparable classification performance to other VQ-based methods while significantly improving the compression performance. When the posterior class probabilities are known, the proposed algorithm improves the classifier performance over all other methods considered for the given examples, including LVQ.

## II. Vector Quantization and Classification

Let $q$ be a $k$-dimensional vector quantizer with codebook $C$, partition $\mathcal{P}$, encoder $\alpha$, and decoder $\beta$. Let $\delta$ be a classifier

assigning a class label $\delta(i) \in \mathcal{H}$ to each possible encoder output $i = 1, \ldots, N$, producing an overall classification rule of $\gamma(x) = \delta(\alpha(x))$.

The compression performance measured by mean squared error is

$$D(\alpha,\beta) = \sum_{i=1}^{N} E\left[d(X,\beta(i))|\alpha(X)=i\right]\Pr(\alpha(X)=i). \tag{2}$$

The classification performance measured by Bayes risk is

$$\begin{aligned} B(\alpha,\delta) &= \sum_{k=1}^{M}\sum_{j=1}^{M} C_{jk}\Pr\left(\delta(\alpha(X))=k \text{ and } Y=j\right) \\ &= \sum_{i=1}^{N}\Pr(\alpha(X)=i)\sum_{k=1}^{M} 1(\delta(i)=k) \\ &\quad \cdot \sum_{j=1}^{M} C_{jk}\Pr\left(Y=j\,|\,\alpha(X)=i\right). \end{aligned} \tag{3}$$

The MSE does not depend on the classifier $\delta$ and the Bayes risk does not depend on the decoder $\beta$. This immediately yields two important optimality properties of a combined VQ and classifier. From (2) we have that

$$D(\alpha,\delta) \geq \sum_{i=1}^{N} \min_{y} E\left[d(X,y)|\alpha(X)=i\right]\Pr(\alpha(X)=i), \tag{4}$$

an unbeatable lower bound which is achievable if for the given encoder $\alpha$ the decoder is chosen as the Lloyd centroid, i.e., as

$$\beta_{\text{Lloyd}}(i) = \arg\min_{y} E\left[d(X,y)|\,\alpha(X)=i\right],$$

regardless of the classifier. Similarly, from (3) we have that

$$B(\alpha,\delta) \geq \sum_{i=1}^{N}\Pr(\alpha(X)=i)\min_{k}\left\{\sum_{j=1}^{M} C_{jk}\Pr\left(Y=j\,|\,\alpha(X)=i\right)\right\},$$

an unbeatable lower bound which is achievable if the classifier is chosen to be the Bayes classifier, which minimizes Bayes risk given a particular encoder $\alpha$, i.e.,

$$\delta_{\text{Bayes}}(i) = \arg\min_{k}\left\{\sum_{j=1}^{M} C_{jk}\Pr\left(Y=j\,|\,\alpha(X)=i\right)\right\}.$$

This classifier depends only on the encoder $\alpha$ and not on the decoder $\beta$.

We can summarize these optimality properties as follows:

PROPERTY 1: *Given an encoder $\alpha$, then the Lloyd decoder $\beta_{\text{Lloyd}}$ minimizes the MSE, regardless of the classifier.*

PROPERTY 2: *Given an encoder $\alpha$, then the Bayes classifier $\delta_{\text{Bayes}}$ minimizes the overall Bayes risk.*

Properties 1 and 2 imply that for a given encoder, the design of the decoder and the classifier are independent. The key issue is how to design the encoder, a problem for which several approaches will be considered. Properties 1 and 2 are only slight modifications of well known properties for the separate compression (minimum average distortion) and classification (minimum Bayes risk) problems.

In order to simultaneously consider the compression and classification abilities of the encoder, we use a Lagrangian modified distortion expression which includes both ordinary distortion and classification error:

$$J_\lambda(\alpha, \beta, \delta) = D(\alpha, \beta) + \lambda B(\alpha, \delta). \tag{5}$$

The Bayes risk is incorporated into the distortion measure with a Lagrangian multiplier in the same way that average codeword length is incorporated into a modified distortion measure in the design of entropy constrained VQ [28], and our approach should be viewed as a variation on entropy constrained VQ. This measure allows flexible trade-offs between compression and classification priorities: when $\lambda \to 0$, we obtain an ordinary MMSE VQ and an independent design of VQ and classifier; when $\lambda \to \infty$, we obtain a minimum Bayes risk classifier based on a VQ structure.

From (5) the modified distortion $J_\lambda$ is

$$\begin{aligned} &J_\lambda(\alpha,\beta,\delta) \\ &= \int dF_X(x)\left[d\left(x,\beta(\alpha(x))\right) + \lambda\sum_{k=1}^{M}\sum_{j=1}^{M} C_{jk}\Pr\left(\delta(\alpha(X))=k, Y=j|X=x\right)\right] \\ &\geq \int dF_X(x)\min_{i}\left\{d\left(x,\beta(i)\right) + \lambda\sum_{k=1}^{M}\sum_{j=1}^{M} C_{jk}1(\delta(i)=k)\Pr\left(Y=j|X=x\right)\right\}. \end{aligned}$$

For a given decoder $\beta$ and classifier $\delta$, this expression is minimized by minimizing the integrand for each value of $x$. This provides a third optimality property:

PROPERTY 3: *For a given decoder $\beta$ and classifier $\delta$, the optimal encoder $\alpha$ is given by*

$$\alpha(x) = \arg\min_{i}\left\{d\left(x,\beta(i)\right) + \lambda\sum_{k=1}^{M}\sum_{j=1}^{M} C_{jk}1(\delta(i)=k)\Pr\left(Y=j|X=x\right)\right\}$$

This is a nearest neighbor encoder with respect to a modified (non-Euclidean) distortion measure. The optimal encoder can make better classification decision regions for cases when the Bayes regions partitioned by the optimal Bayes rule are not well approximated by Euclidean Voronoi cells. Unlike an ordinary MMSE encoder, the optimal encoder allows the curving of quantizer regions in order to improve classification.

The three optimality properties taken together provide a descent algorithm for the design of a VQ with a modified distortion measure comprising both squared error and Bayes risk. We shall refer to such a quantizer as a Bayes risk weighted VQ or, more simply, a Bayes VQ.

## III. Code Design

The VQ-based methods used in the simulations are described next.

### A. Independent Design

Here the encoder $\alpha$ is matched to the decoder $\beta$ by using an MMSE rule. The encoder and decoder are designed by the iterative Lloyd algorithm to produce an ordinary MMSE VQ. Following convergence the Bayes classifier is applied to the VQ outputs. In the equal cost binary class case, this means simply classifying each codeword index according to the majority of class labels quantized into that index in the training set. For this case the encoder and codebook design are based entirely on minimizing squared error alone without regard to classification.

### B. Learning Vector Quantization

The VQ design techniques of Kohonen and his colleagues [25], [26] use clustering techniques to design a VQ with an MMSE encoder specifically for classification. MacQueen's original $k$-means algorithm [11] encodes each training vector in succession, each time mapping it to the index of the nearest codeword in the codebook, and then modifying the codeword to be a centroid of all input vectors that have mapped into that index since training began. All other codewords are left unchanged. Let the size $N$ codebook at time $n-1$ be described as $\{y_1(n-1), y_2(n-1), \ldots, y_N(n-1)\}$. Suppose that the next training vector $x(n)$ selects $y_i(n-1)$, the $i$th reproduction codeword, as the nearest neighbor and hence selects index $i$. Then the new centroid for $i$ is formed as

$$y_i(n) = \frac{n-1}{n} y_i(n-1) + \frac{1}{n} x(n) = y_i(n-1) + \frac{1}{n}\big(x(n) - y_i(n-1)\big).$$

Defining $a(n) = 1/n$, the $k$-means update rule becomes

$$\begin{aligned} &y_i(n) = y_i(n-1) + a(n)\big(x(n) - y_i(n-1)\big) \\ &\text{if } d\big(x(n), y_i(n-1)\big) \le d\big(x(n), y_l(n-1)\big), \text{ all } l. \end{aligned} \tag{6}$$

Kohonen's LVQ1 replaces the "update the centroid of the selected index" of (6) with a rule that depends on whether or not the class label of the selected index matches that of the training vector: if yes, the update rule of (6) is used to incorporate the new input vector into codeword; if no, then the update rule is instead $y_i(n) = y_i(n-1) - a(n)\,(x(n) - y_i(n-1))$, effectively trying to separate the new input vector from the mistaken class. All other codewords are left unaffected. Here $0 < a(n) < 1$, and $a(n)$ should either be constant or decrease monotonically with time. A variation of this basic method, optimized learning rate LVQ1 (OLVQ1), is used here [34].

### C. Bayes Risk Weighted VQ (Bayes VQ)

A Bayes risk weighted VQ uses the Bayes risk weighted distortion of (5) as a generalized distortion measure and uses Properties 1–3 to define a descent algorithm for quantizer design, that is, an iterative algorithm to modify decoder $\beta$, classifier $\delta$, and encoder $\alpha$ in such a way that the average Bayes risk weighted distortion is nondecreasing.

We start with some initial coder $(\alpha^{(0)}, \beta^{(0)}, \delta^{(0)})$ and iteratively apply an improvement transformation $(\alpha^{(t+1)}, \beta^{(t+1)}, \delta^{(t+1)}) = T(\alpha^{(t)}, \beta^{(t)}, \delta^{(t)})$ so that $J_\lambda(\alpha^{(t)}, \beta^{(t)}, \delta^{(t)})$ is nonincreasing in $t$. Since $J_\lambda$ is bounded below by 0, we know that $J_\lambda^{(t)}$ must converge as $t \to \infty$. The transformation $T$ is implemented by successive application of Properties 2, 1, and 3 to the previous coder.

Both design and implementation of the algorithm require knowledge of the posterior class probabilities, Pr $(Y|X)$. We here consider two cases: parametric and nonparametric. In the parametric examples it is assumed that these probabilities are known and hence the design method is exactly as described above and the encoder, decoder, and classifier are exactly as produced in the final iteration of the design algorithm. This is the case, for example, if the random objects are generated from a known probabilistic model such as a Gaussian distribution with different classes corresponding to different variances or means. In the nonparametric case these probabilities must be estimated. Since the VQ is designed based on a learning set of empirical data, this same set can be used to estimate the posteriors during design by relative frequencies. For example, $P(Y = j | X = x)$ is estimated by $t_j/t$ where $t_j$ is the number of times $x$ occurred with class label $j$ and $t$ is the number of occurrences of $x$.

Unfortunately, knowledge of $P(Y|X)$ within the learning set does not immediately provide a useful estimate of the conditional probabilities outside the training set. While one could use some form of multidimensional interpolation or kernel estimation technique, the computational complexity involved can be extreme. For this reason we have chosen here to use a simple MMSE encoder (Euclidean nearest neighbor) for the nonparametric examples outside the training set, as do other VQ classification techniques including LVQ. Unlike those techniques, however, we do use the estimated posteriors in the design of the codebook. The parametric (optimal weighted Bayes encoder with true posterior probabilities) and suboptimal (MMSE) encoders are both considered for our parametric examples to demonstrate the achievable gains when the posteriors are known and can be used. Methods of estimating the class posteriors with low complexity and using these estimates to approximate optimal encoding outside the training data are developed in [35], [36].

Two important implementational issues are the choice of $\lambda$ and the stopping rule used to halt the iterations. There is no general rule for optimal selection of $\lambda$. We chose values yielding good performance on the training data. There are several stopping rules that could be used to determine when to halt the iteration process. We could select a particular $\lambda$ and iterate until the decrease in average distortion $J_\lambda$ falls below a certain threshold. Increasing $\lambda$ generally reduces the Bayes risk while increasing the average ordinary distortion $D$. The stopping rule used here is to set a maximum allowable increase in the ordinary distortion (over the distortion obtained with regular VQ with $\lambda = 0$) and then iterate over increasing values of $\lambda$ until the maximum is reached. This can also be used as a

method to select an appropriate $\lambda$. If the suboptimum MMSE encoder is to be used once the codebook is designed, then there is no guarantee that iterations based on the optimal encoder will always improve Bayes risk with the MMSE encoder. Hence, we cease increasing $\lambda$ if the Bayes risk with the MMSE encoder decreases.

## IV. Examples

### A. Simulated Data

We consider two simulated examples where the distributions are known. In both cases the input vector is a mixture of two equally likely distributions of two-dimensional data. Equal costs are assumed. The compression and classification results are compared with two other methods of VQ design: independent design and LVQ.

#### *A.1. Diamond Example*

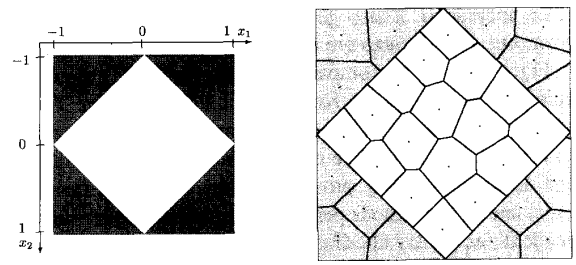
Suppose that the vectors $X=(X_1, X_2)$ belonging to Class 1 are uniformly distributed over the diamond $|x_1| + |x_2| \leq 1$ and the vectors belonging to Class 2 are uniformly distributed over $|x_1| + |x_2| > 1$ where $|x_1| \leq 1$ and $|x_2| \leq 1$. The ideal classification is a diamond shape as shown in Fig. 1(a); the white center region is Class 1, and the gray corner regions are Class 2. The simplicity of this separable problem provides a good means of visualizing the effects of the algorithm.

The vector quantizers are designed using a training sequence consisting of 10,000 randomly chosen vectors and evaluated using a test set of 10,000 (different) vectors. Various codebook sizes ($N$ = 8, 16, 32, 64) are considered. The simulations are repeated a total of five times with different random seeds, and the averages are reported.

For this separable example, the true probabilities and the probabilities estimated from the training set are equivalent. The Voronoi diagram for a sample codebook designed using the optimal encoder for both training and test (parametric Bayes VQ) is shown in Fig. 1(b). Note that this encoder attains perfect classification.

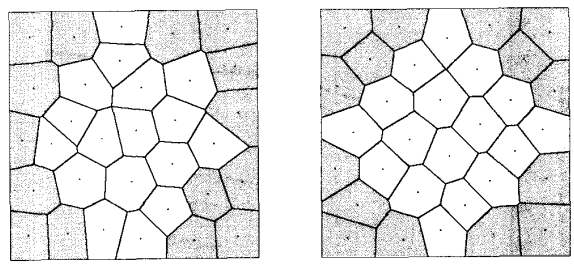
For comparison we consider nonparametric encoders outside the learning set, i.e., an MMSE encoding. The following three codebook design methods are compared: *Independent Design* using an MMSE VQ followed by a Bayes classifier designed using the relative class frequencies of the quantized training set. *Bayes VQ* where the value of $\lambda$ is initialized to 0.01 and allowed to increase by a multiplicative factor of 1.1 until mismatch is detected. During training, optimal encoder probabilities are based on frequency of occurrence in the training set. During testing, ordinary MMSE encoding is used. *LVQ* designed using *OLVQ1* using 50,000 iterations. The codebook is initialized using the LVQ_PAK *eveninit* algorithm.

The partition and classification for a codebook with 32 codewords using independent design are shown in Fig. 2(a). Again, the white regions are labeled as Class 1 and the gray regions are Class 2. No classification information is used when constructing this codebook. In contrast, the partition and classification for the same size codebook designed using Bayes




(a) (b)

Fig. 1. Diamond example: (a) Original Classification, (b) Parametric Bayes VQ Voronoi diagram.



(a) (b)

Fig. 2. Diamond example Voronoi diagram:
(a) Independent Design,
(b) Bayes VQ.

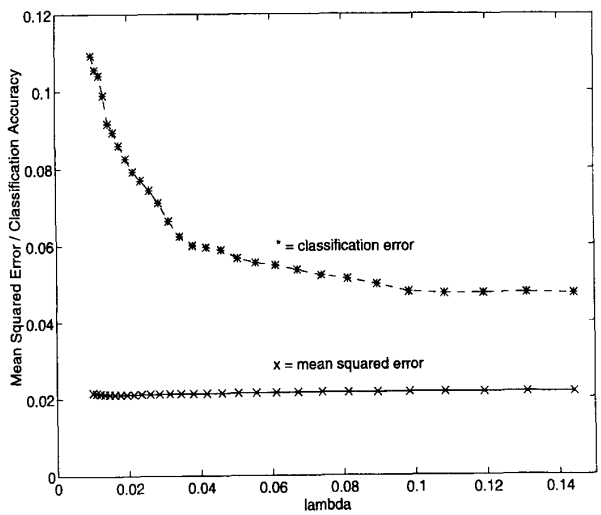



Fig. 3. Diamond example: effect of $\lambda$ on classification error and mean squared error for codebook size 32.

VQ are shown in Fig. 2(b). The codeword boundaries are much more closely aligned with the ideal classification boundaries, improving the classification error. A plot showing the effect of $\lambda$ on the classification error and on mean squared error for a Bayes VQ codebook of this size is given in Fig. 3.

As expected, increasing $\lambda$ generally reduces the classification error. Increasing $\lambda$ also generally increases the mean squared error, but the increases are extremely small (less than 2%).

Fig. 4 provides a comparison of the Bayes VQ algorithm to independent design and Kohonen's LVQ algorithm. The classification performance generally improves as the number of codewords increases. We observe from Fig. 4 that Bayes VQ and Kohonen's LVQ produce comparable classification results, and both methods perform substantially better than independent design. By altering the usual VQ codebook design, we can obtain improved classification. The ability to compress, measured by mean squared error, is examined in Fig. 5. As expected, squared error decreases as the number of codewords

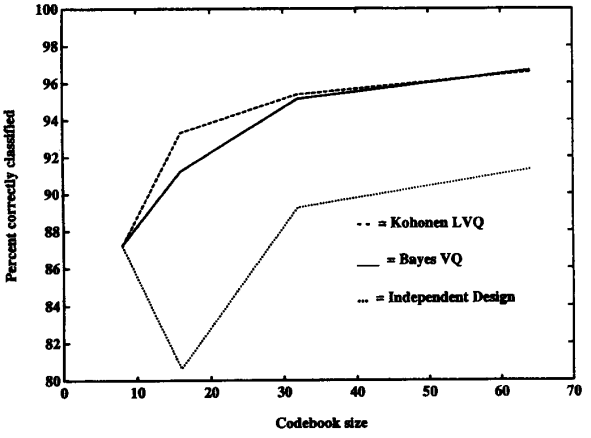



Fig. 4. Diamond example: classification comparison of design methods.

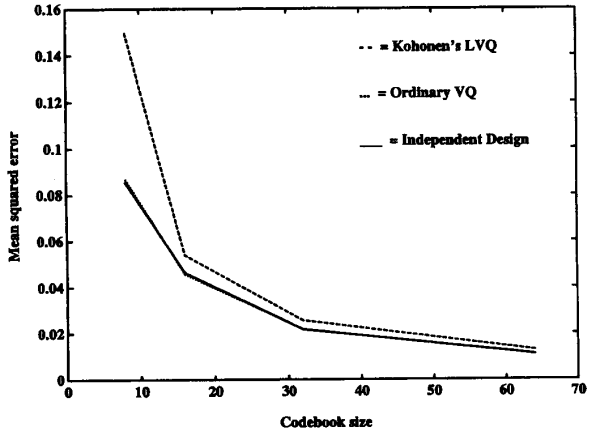



Fig. 5. Diamond example: MSE comparison of design methods.

increases. Bayes VQ and independent design are comparable, with Kohonen's LVQ performing worse.

In this example, Bayes VQ does very well. It classifies virtually as well as Kohonen's LVQ (a method intended strictly for classification not compression) *and* it compresses as well as independent design.

### *A.2. Concentric Gaussian Example*

The next example considered is a nonseparable problem taken from Kohonen [26]. The mixture in this case consists of two concentric two-dimensional (bivariate) Gaussian distributions, each with zero mean. The coordinates are independent for both classes with variance = 1 for Class 1 and variance = 4 for Class 2. These distributions overlap, so no decision rule provides perfect classification. The Bayes decision rule, which minimizes the Bayes risk (or classification error since we have equal costs), is easily determined to be a circle about the origin of radius 1.923. This optimal decision rule will classify 73.624% of vectors correctly on average.

As before, we design and evaluate the codebooks using sets of 10,000 random vectors, averaging the results over five simulations. We first use the true probabilities in the parametric encoder to design various sized codebooks using the parametric Bayes VQ algorithm. In this case, the true probabilities are substantially different from those estimated from the training data. Once the codebooks have been designed, two types of encoding outside the training sequence are presented: optimal encoding using the true probabilities and ordinary MMSE encoding. Compression and classification results are given in Tables I and II. Results are based on data outside the training sequence averaged over five simulations.

TABLE I
CONCENTRIC GAUSSIAN EXAMPLE:
PERCENT VECTORS CORRECTLY CLASSIFIED

| Codebook Design Method | Codebook size 8 | 16 | 32 | 64 |
|---|---|---|---|---|
| Parametric Bayes VQ MMSE encoder | 73.03 | 73.11 | 73.50 | 73.65 |
| Parametric Bayes VQ Optimal encoder | 73.63 | 73.61 | 73.63 | 73.60 |
| Bayes VQ | 71.13 | 69.56 | 72.26 | 73.07 |
| Kohonen LVQ | 72.14 | 72.47 | 72.78 | 72.24 |
| Independent Design | 70.11 | 70.41 | 71.94 | 73.01 |

TABLE II
CONCENTRIC GAUSSIAN EXAMPLE:
MEAN SQUARED ERROR

| Codebook Design Method | Codebook size 8 | 16 | 32 | 64 |
|---|---|---|---|---|
| Parametric Bayes VQ MMSE encoder | 1.20 | 0.67 | 0.34 | 0.18 |
| Parametric Bayes VQ Optimal encoder | 1.24 | 0.69 | 0.35 | 0.19 |
| Bayes VQ | 1.18 | 0.63 | 0.34 | 0.18 |
| Kohonen LVQ | 1.46 | 0.81 | 0.47 | 0.26 |
| Independent Design | 1.21 | 0.65 | 0.35 | 0.19 |

Like the diamond example, three methods of design are considered for the nonparametric algorithm. Classification results are given in Table I. In this example, the classification abilities of the three design methods are much more similar than in the diamond example. Kohonen's LVQ method does best for the three smaller codebook sizes, but the Bayes VQ

method does best for the largest codebook size. The strong performance of the independent design method suggests that Gaussian distributions are well suited to classification by VQ in general, and the design methods that stress classification (like Kohonen's LVQ and Bayes VQ) can provide little gain over independent design. The mean squared error for the three design methods is given in Table II. As in the diamond example, the Bayes VQ and independent design perform noticeably better than Kohonen's LVQ method.

The classification performance of codebooks designed using the known distributions to determine $\Pr(Y = j \mid X = x)$ (parametric Bayes VQ) is significantly better than codebooks designed using probabilities determined by the class labels of the training set. The parametric Bayes VQ methods correctly classify at least 73.0% of the vectors. Although subsequent encoding of the test data using the true probabilities performs better than the ordinary MMSE encoding, both perform quite well. For example, the size 8 parametric Bayes VQ codebook with MMSE encoder classifies 73.0% correctly. This compares with the 71.1% classification from the Bayes VQ which used estimates from the training set rather than the true probabilities during codebook design. The same size parametric Bayes VQ codebook with optimal encoder classifies 73.6% correctly, equivalent to the theoretically optimal Bayes classifier. Further comparisons of Bayes VQ with an MSE encoder and LVQ on Gaussian mixtures are treated by Wesel et al. [37].

### B. Image Data

The Bayes VQ algorithm is next used to identify tumors in CT lung images and to classify man-made and natural regions in aerial images. The compression and classification results for the aerial application are compared with independent design VQ and the LVQ design. Because the CT application uses unbalanced costs, comparison with LVQ is omitted as LVQ has no provision for such costs.

#### *B.1. CT Images*

First consider CT images where we wish both to compress the images and to identify pulmonary tumor nodules. The 12 bit grayscale images are of size $512 \times 512$ pixels. Class 1 corresponds to healthy tissue, and Class 2 corresponds to tumor. The locations of the tumors are determined by trained radiologists. The training set consists of 10 images plus the tumor vectors from five additional images; the additional tumor training vectors are added because of the low average percentage of tumor vectors in the data. We test on similar images outside the training set. Here classification costs of $C_{12} = 1$, $C_{21} = 100$ are chosen to reflect the relative importance of the two error types constructed using various values of $\lambda$. Each codebook contains 256 codewords with dimension $2 \times 2$, producing a fixed rate 2 bpp code (additional entropy coding would reduce this to about 1.56 bpp).

A representative test image yielded the following statistics: SNR = 26.69 dB, sensitivity 83.3%, specificity = 96.4%, predictive value positive = 4.4%, and overall Bayes risk = 0.070. Sensitivity is the percentage of tumor vectors which are correctly classified as tumor. Specificity is the percentage of nontumor vectors which are correctly classified as nontumor. Predictive value positive (PVP) is the percentage of vectors classified as tumor that are actually tumor vectors. The algorithm was able to identify most of the tumor vectors in the test images, with up to 83.3% sensitivity. It should be recalled that the classifier operates only on $2 \times 2$ pixel intensity blocks with no context. The original test image is shown in Fig. 6(a). The image is shown after windowing the intensity values to improve the contrast in the pixel value region of interest (the same adjustment performed by radiologists). This image contains three circular tumors in the left lung. The true positions of the three tumors are shown in Fig. 6(b), where bright regions represent tumor and dark regions represent nontumor. Compression and classification results for the test image are shown in Fig. 6(c) and (d). The algorithm correctly identifies substantial parts of each of the three tumors. Even though the algorithm does identify as tumor a fair amount of the image that is not tumor, such a phenomenon would not greatly mar the diagnostic aid.

TABLE III
AVERAGE PSNR AND CLASSIFICATION ABILITY FOR AERIAL IMAGE SIMULATION

| Codebook Design Method | Percent Correctly Classified | PSNR (dB) |
|---|---|---|
| Bayes VQ | 74.0 | 24.5 |
| Kohonen LVQ | 74.9 | 23.8 |
| Independent Design | 71.0 | 24.4 |

#### *B.2. Aerial Images*

Here the goal is to highlight subblocks in aerial images that are classified as being man-made as opposed to natural in order to attract the attention of human observers. The classification of the training set (learning set) of aerial photographs is done by hand-labeling those features (man-made and natural regions) that are to be recognized in subsequent images. The training set consisted of aerial images of the San Francisco Bay Area provided by ESL, Inc. The images were $512 \times 512$ pixels of 8 bit grayscale. Each $8 \times 8$ 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 $8 \times 8$ vectors to simplify the task of the human classifier, the codebook construction and image encoding were carried out using $4 \times 4$ pixel vectors.

Because of the limited labeled training data available, simulations were performed using a form of 6-fold cross-validation [15] to improve the quality of the results. The training set consisted of six images containing both urban and rural regions. Codebooks were trained using data from all but one of the images, and the resulting codebooks were tested on the image not used for training. This process was repeated until all the images were used for testing. Using equal classification costs, size 256 full search codebooks are constructed using $4 \times 4$ vectors (providing a fixed bit rate of 0.5 bpp). The value of $\lambda$ is increased from 0.01 until it exceeds 1,000 or mismatch is detected

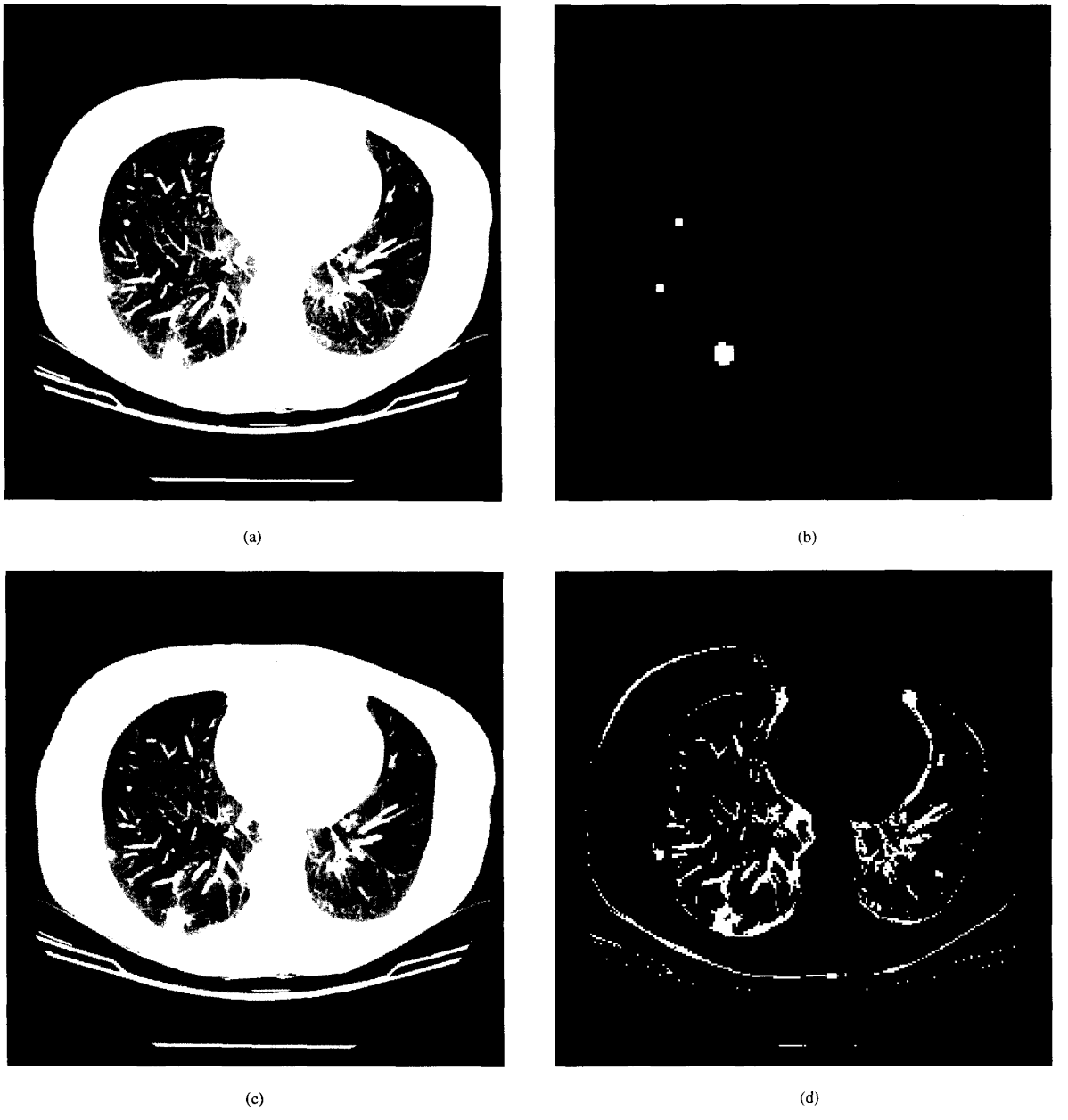


Fig. 6. CT image: (a) original 12 bpp image, (b) true tumor locations, (c) compressed at 2 bpp, (d) classification using Bayes VQ.

Fig. 7(a) and (b) shows a typical test image and the hand-labeled classification for this image. Fig. 7(c) and (d) shows the image after compression at 0.5 bpp and demonstrates the classification produced by Bayes VQ.

The classification is denoted by showing subblocks that the encoder classified as natural as black and showing subblocks classified as man-made as white. For this image, the codebook correctly classifies 74% of the vectors. The ability to classify is computed with respect to the hand-labeling of the training and test images, which is an imperfect "gold standard" affected by the lower resolution (larger block size) of the hand-labeling and the human observer's perception and inevitable inconsistencies. Results averaged over all six images are given in Table III. Table III also contains results from independent design VQ and Kohonen's LVQ algorithm with the same size codebooks (for Kohonen, the training sequence was cycled through five times).

The Bayes VQ algorithm does quite well on the aerial image task. The method classified an average of 74% of the test image vectors correctly, performing slightly better than the

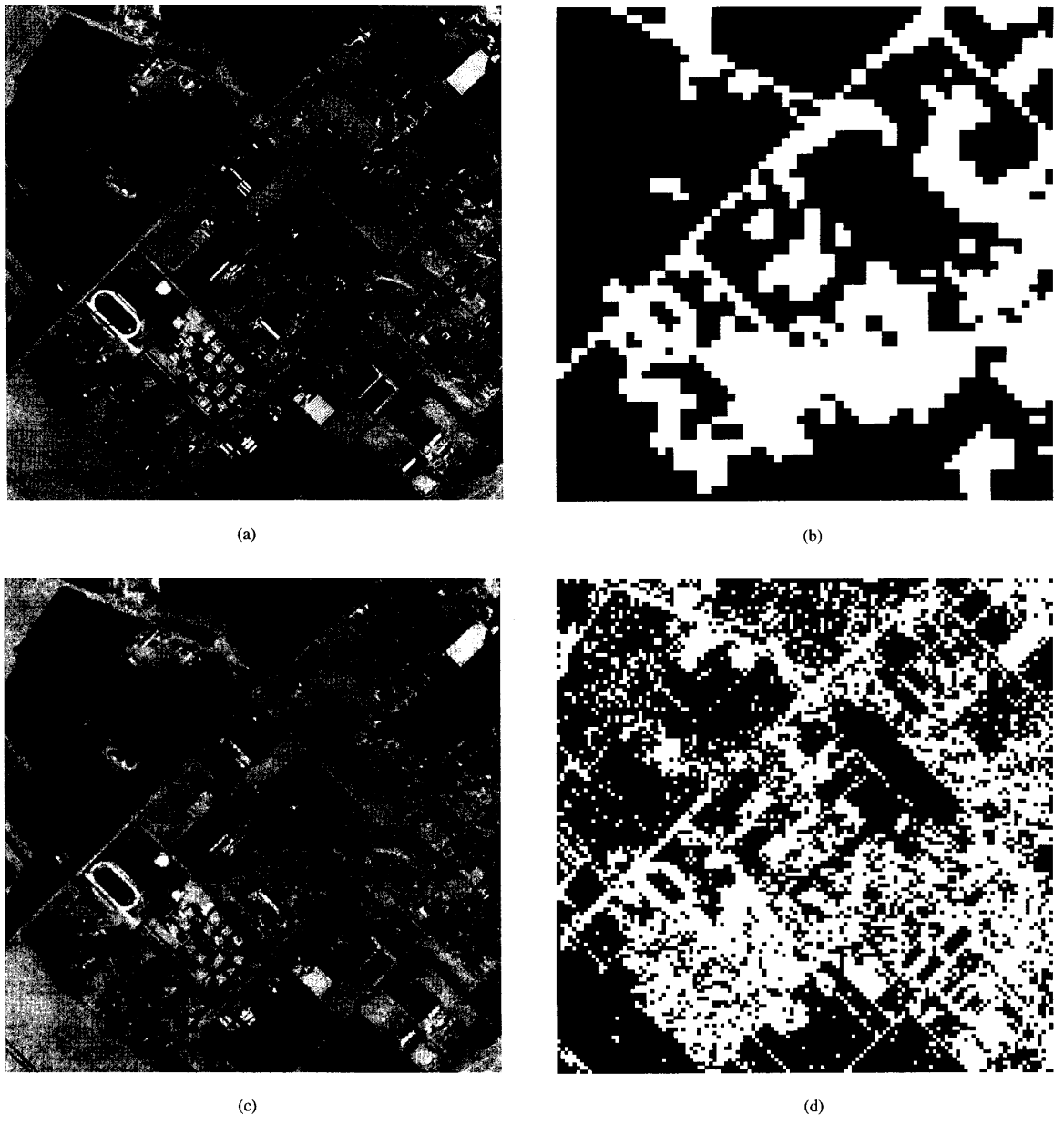


Fig. 7. Aerial image: (a) original 8 bpp image, (b) hand-labeled classification, (c) compressed at 0.5 bpp, PSNR = 26.2 dB, (d) classification using Bayes VQ. (Man-made subblocks shown white, natural subblocks shown black.)

tree-structured method described in [31] (the PSNR is also higher). However, independent design applied to these same tasks also does reasonably well. On the aerial images, independent design correctly classified 71% of the vectors, 3% less than Bayes VQ. One possible explanation is that these real image tasks have class distributions similar to the concentric Gaussian task, so that the simple clustering provided by the independent design algorithm is sufficient to generate a good classifier. The Kohonen LVQ algorithm classified 1% more of the vectors correctly with a loss of 0.7 dB in PSNR.

## V. Extension to Tree-Structured VQ

The basic idea of Bayes risk weighted VQ extends to TSVQ in a natural way. Tree-structured code growing and pruning techniques [29], [28], [30], [8] are applied to the modified distortion measure $J_\lambda$ rather than to the ordinary distortion $D$. Such tree growing techniques can be considered as a new variation of traditional classification tree design algorithms such as CART [15].

This technique was used to construct tree-structured codebooks for the diamond example discussed previously. With an

average bit rate of 2.5 bits per pixel (the same rate as a full search codebook with 32 codewords), the tree-structured code provided an average classification of 94.0% and an average mean squared error of 0.026. In comparison, the Bayes VQ size 32 full search codebook provided 95.1% classification and 0.022 mean squared error. The slight loss in classification and compression performance is offset by a substantial reduction in the time required for design and encoding. The tree-structured encoder requires only an average of 10 distance computations per vector compared to the 32 distance computations needed for full search encoding. This complexity advantage is even more significant with higher dimensions or larger codebooks. The modified distortion measure might be combined with other structured VQ codebook design techniques to develop low-complexity efficient quantizers for compression and classification. It is not as obvious how to adapt Kohonen's method to structured VQ formulations.

## VI. Conclusions

An algorithm has been introduced for the design of vector quantizers combining the goals of compression and classification. This was accomplished by combining an ordinary compression distortion measure such as squared error with a Bayes risk term to measure classification accuracy. A clustering algorithm such as the Lloyd algorithm then produces a code minimizing a weighted combination of squared error and Bayes risk, where the weighting can be adjusted according to relative importance. We here considered the parametric case, where the necessary conditional class probabilities are known, and the specific nonparametric case where these probabilities are estimated by relative frequencies within the training set and a suboptimal MMSE encoder is used outside the training set. Extensions to simple class probability estimators inside and outside the training set are pursued in [35], [36]. It was found that the suboptimal MMSE encoder works quite well in comparison to the optimal encoder in some problems, while performing quite poorly in others. We suspect that the MMSE encoder is best for problems resembling composite Gaussian densities such as Kohonen's Gaussian example.

## Acknowledgments

The authors acknowledge the helpful comments of Richard Olshen, Keren Perlmutter, and Andrew Nobel. This research was supported by the National Science Foundation under grants NSF MIP–9016974 and NSF MIP–9311190.

## References

[1] P.C. Cosman, H.C. Davidson, C.J. Bergin, C. Tseng, L.E. Moses, R.A. Olshen, and R.M. Gray, "The effect of lossy compression on diagnostic accuracy of thoracic CT images," *Radiology*, vol. 190, no. 2, pp. 517–524, 1994.

[2] R. Hamabe, Y. Yamada, M. Murata, and T. Namekawa, "A speech recognition system using inverse filter matching technique," *Proc. Ann. Conf. Inst. of Television Engineers*, Kyushu University, June 1981 (in Japanese).

[3] J.E. Shore and D.K. Burton, "Discrete utterance speech recognition without time alignment," *Proc. ICASSP*, May 1982, p. 907.

[4] S.-S. Huang and R.M. Gray, "Spellmode recognition based on vector quantization," *Speech Communication*, vol. 7, pp. 41–53, 1988.

[5] G.F. McLean, "Vector quantization for texture classification," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 637–649, May/June 1993.

[6] K.L. Oehler and R.M. Gray, "Combining image classification and image compression using vector quantization," *Proc. 1993 IEEE Data Compression Conference*, J.A. Storer and M. Cohn, Eds. Snowbird, Utah: IEEE Computer Society Press, Mar. 1993, pp. 2–11.

[7] H. Abut, Ed., *Vector Quantization*, IEEE Reprint Collection. Piscataway, N.J.: IEEE Press, May 1990.

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

[9] S.P. Lloyd, "Least squares quantization in PCM," Unpublished Bell Laboratories Technical Note. Portions presented at the Institute of Mathematical Statistics Meeting, Atlantic City, N.J., Sept. 1957. Published in the March 1982 special issue on quantization of the *IEEE Trans. Information Theory*, 1957.

[10] E. Forgey, "Cluster analysis of multivariate data: Efficiency vs. interpretability of classification," *Biometrics*, vol. 21, p. 768, 1965 (Abstract).

[11] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proc. 5th Berkeley Symp. on Math. Stat. and Prob.*, vol. 1, pp. 281–296, 1967.

[12] A.B. Nobel, "Histogram regression estimation using data-dependent partitions," in preparation.

[13] G. Lugosi and A.B. Nobel, "Consistency of data-driven histogram methods for density estimation and classification," Beckman Institute technical report UIUC-BI-93-01, University of Illinois, Urbana-Champaign, 1993, submitted for publication.

[14] A.B. Nobel and R.A. Olshen, "Termination and continuity of greedy growing for tree structured vector quantizers," submitted for publication.

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

[16] E.E. Hilbert, "Cluster compression algorithm: A joint clustering/data compression concept," Publication 77-43, Pasadena, Calif.: Jet Propulsion Laboratory, Dec. 1977.

[17] H.V. Poor and J.B. Thomas, "Applications of Ali-Silvey distance measures in the design of generalized quantizers for binary decision systems," *IEEE Trans. Comm.*, vol. 25, pp. 893–900, Sept. 1977.

[18] G.R. Benitz and J.A. Bucklew, "Asymptotically optimal quantizers for detection of i.i.d. data," *IEEE Trans. Inform. Theory*, vol. 35, pp. 316–325, 1989.

[19] E. Fix and J.L. Hodges, Jr., "Discriminatory analysis, nonparametric discrimination, consistency properties," Project 21-49-004, Report No. 4, Randolph Field, Texas: USAF School of Aviation Medicine, Feb. 1951.

[20] T.M. Cover and P.E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inform. Theory*, vol. 13, pp. 21–27, 1967.

[21] P.A. Devijver and J. Kittler, "On the edited nearest neighbor rule," *Proc. 5th Int'l Conf. on Pattern Recognition*, 1980, pp. 72–80.

[22] Q. Xie, C.A. Laszlo, and R.K. Ward, "Vector quantization technique for nonparametric classifier design," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 12, pp. 1,326–1,330, Dec. 1993.

[23] K. Popat and R.W. Picard, "Novel cluster-based probability model for texture synthesis, classification, and compression," *Proc. SPIE Visual Comm. and Image Processing*, Boston, Nov. 1993.

[24] K. Popat and R.W. Picard, "Cluster-based probability model applied to image restoration and compression," *Proc. ICASSP*, Adelaide, Australia, 1994.

[25] T. Kohonen, *Self-organization and Associative Memory*, Berlin: Springer-Verlag, 3rd ed., 1989.

[26] T. Kohonen, G. Barna, and R. Chrisley, "Statistical pattern recognition with neural networks: Benchmarking studies," *IEEE Int'l Conf. Neural Networks*, July 1988, pp. 61–68.

[27] C.J. Stone, "Consistent nonparametric regression," *Annals of Statistics*, vol. 5, pp. 595–645, 1977.

[28] P.A. Chou, T. Lookabaugh, and R.M. Gray, "Entropy-constrained vector quantization," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 37, pp. 31–42, Jan. 1989.

[29] P.A. Chou, T. Lookabaugh, and R.M. Gray, "Optimal pruning with applications to tree-structured source coding and modeling," *IEEE Trans. Inform. Theory*, vol. 35, no. 2, pp. 299–315, Mar. 1989.
[30] E.A. Riskin and R.M. Gray, "A greedy tree growing algorithm for the design of variable rate vector quantizers," *IEEE Trans. Signal Process.*, vol. 39, pp. 2,500–2,507, Nov. 1991.
[31] K.L. Oehler, P.C. Cosman, R.M. Gray, and J. May, "Classification using vector quantization," *Conf. Record 25th Asilomar Conf. on Signals, Systems, and Computers*, Pacific Grove, Calif., pp. 439–445, Nov. 1991.
[32] K.L. Oehler, *Image Compression and Classification Using Vector Quantization*, PhD Dissertation, Stanford University, 1993.
[33] J. Kramer, "Tree structured neural net classifier," Tech. Rep. #19900805, Stanford University Center for Design Research, 1990.
[34] T. Kohonen, J. Kangas, J. Laaksonen, and K. Torkkola, "LVQ_PAK: The learning vector quantization program package, version 2.1," Tech. Rep., Helsinki University of Technology, Laboratory of Computer and Information Science, Finland, Oct. 1992.
[35] K. Perlmutter, R.M. Gray, K.L. Oehler, and R.A. Olshen, "Bayes risk weighted tree structured vector quantization with estimated class posteriors," *Proc. IEEE Data Compression Conf.*, Snowbird, Utah, April 1993, pp. 274–283.
[36] K.O. Perlmutter, R.M. Gray, K.L. Oehler, and R.A. Olshen, "Bayes risk weighted vector quantization with estimated class posteriors," submitted for publication, 1994.
[37] R.D. Wesel and R.M. Gray, "Bayes risk weighted VQ and learning VQ," *Proc. IEEE Data Compression Conf.*, Snowbird, Utah, April 1994.

**Karen L. Oehler** received the BS degree from Rice University in 1987 and the MS and PhD degrees from Stanford University in 1989 and 1993, all in electrical engineering. She was a Winston Churchill Scholar and received a National Science Foundation Graduate Fellowship. Since 1993, she has been a Member of Technical Staff at the Integrated Systems Laboratory of Texas Instruments, Dallas, Texas. A member of the IEEE, her interests include image and video compression and image classification.

**Robert M. Gray** received the BS and MS degrees from the Massachusetts Institute of Technology in 1966 and the PhD degree from the University of Southern California in 1969, all in electrical engineering. Since 1969 he has been with Stanford University, where he is currently a Professor and Vice Chair of the Department of Electrical Engineering. He is a Fellow of both the Institute of Mathematical Statistics and the IEEE and was awarded the 1993 Society Award of the IEEE Signal Processing Society.