A Bayesian Framework for Semantic Classification of Outdoor Vacation Images

Aditya Vailaya, [+]Mário Figueiredo[*], Anil Jain, [#]HongJiang Zhang

Dept. of Comp. Sc. & Eng.
Michigan State University
East Lansing, MI 48824, USA

[+] Instituto de Telecomunicações
Instituto Superior Técnico
1049-001 Lisboa, Portugal

[#] Internet Systems & Applications Lab
Hewlett Packard Labs
Palo Alto, CA 94304, USA

vailayaa@cps.msu.edu, [+]mtf@lx.it.pt, jain@cps.msu.edu, [*]hjzhang@hpl.hp.com

Key words: semantic image classification, Bayesian framework, clustering, salient features, similarity, image database, content-based retrieval.

Abstract

Grouping images into (semantically) meaningful categories using low-level visual features is a challenging and important problem in content-based image retrieval. Based on these groupings, effective indices can be built for an image database. In this paper, we cast the image classification problem in a Bayesian framework. Specifically, we consider city vs. landscape classification and further classification of landscape images into sunset, forest, and mountain classes. We demonstrate how high-level concepts can be understood from specific low-level image features under the constraint that the test images do belong to one of the classes in concern. We further demonstrate that a small codebook (the optimal size of codebook is selected using MDL principle) extracted from a vector quantizer can be used to estimate the class-conditional densities needed for the Bayesian methodology. Classification based on color histograms, color coherence vectors, edge direction histograms, and edge direction coherence vectors as features shows promising results. On a database of $2,716$ city and landscape images, our system achieved an accuracy of 95.3% for city vs. landscape classification. On a subset of 528 landscape images, our system achieves an accuracy of 94.9% for sunset vs. forest & mountain classification and 93.6% for forest vs. mountain classification. Our final goal is to combine multiple 2-class classifiers into a single hierarchical classifier.

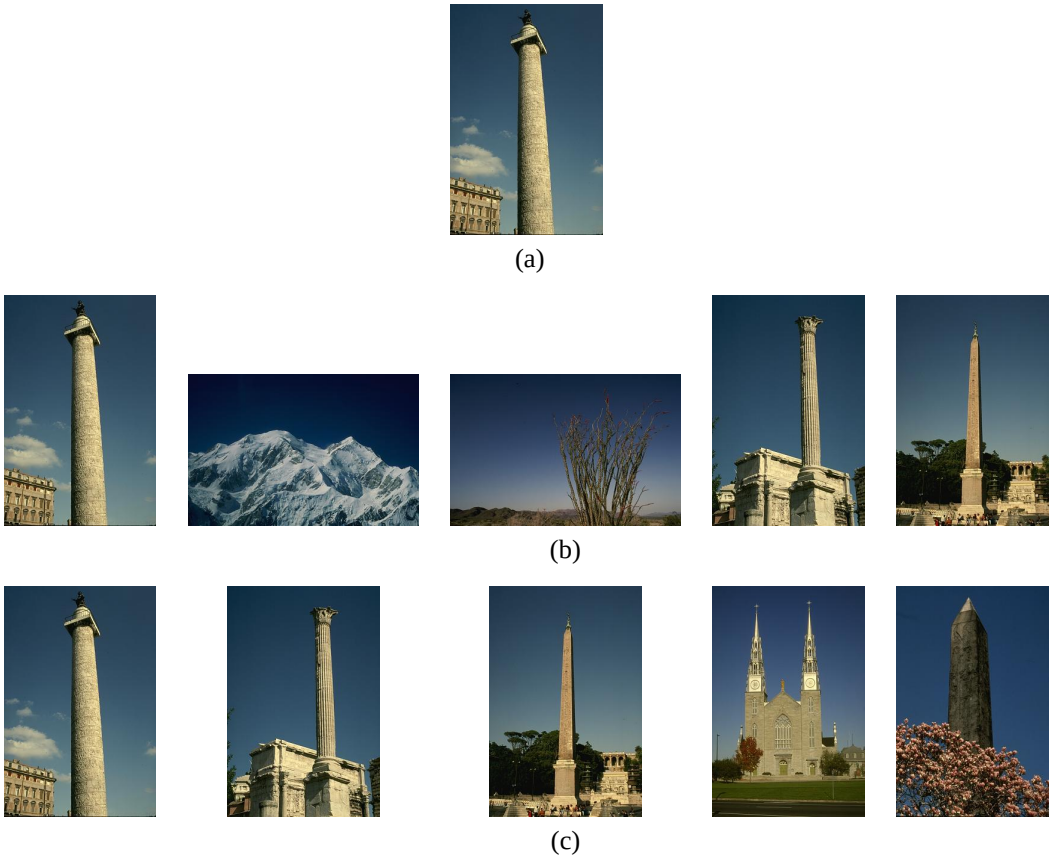
1 Introduction

Content-based image organization and retrieval has emerged as an important area in computer vision and multimedia computing. This is mainly driven by technological breakthroughs which allow us to digitize, store, and transmit images in a very cost effective and efficient manner. A large number of commercial organizations have large image and video collections of programs, news segments, games, paintings, and artifacts that are being digitized. With the development of digital photography, more and more people are able to store their vacation and personal photographs on their computers. Various museums are constructing digital archives of art paintings and pictures of various artifacts [1, 2]. These digital archives can then be stored as virtual museums that are not only more accessible but they also safe-guard and preserve the original artifacts. Travel agencies are interested in digital archives of photographs of holiday resorts. A user could query these databases to plan a vacation. These digital databases are not a dream of the future, but have become a reality. However, in order to make these databases more useful, we need to develop schemes for indexing and categorizing the humungous data. In other words, organizing these image and video libraries into a small number of categories and providing effective indexing is imperative for accessing, browsing, and retrieving useful data in "real-time".

Over the last five years, there has been an intense activity in developing image retrieval methods based on image content. Various systems have been proposed for content-based image retrieval, such as QBIC [3], Photobook [4] and

[*] M. Figueiredo was partially supported by Nato Grant *NATOCRG 960010* & Portuguese PRAXIS XXI program, grant no. 2/2.1/T. I. T./1580/95.

FourEyes [5], SWIM [6], Virage [7], Visualseek [8], Netra [9], and MARS [10]. These systems follow the paradigm of representing images using a set of image attributes, such as color, texture, shape, and layout which are archived along with the images in the database. A retrieval is performed by matching the feature attributes of a query image with those of the database images. User queries are typically based on semantics (e.g., show me a sunset image) and not on low-level image features (e.g., show me a predominantly red and orange image). As a result, most of these image retrieval systems have poor performance for specific queries. For example, Figure 1(b) shows the top-5 retrieved results (based on color histogram features) for the query in Figure 1(a) on a database of $2,145$ images of city and landscape scenes. While the query image has a specific monument (a tower here), some of the retrieved images include landscape scenes (mountains and sky). A successful grouping of these database images into semantically meaningful classes can greatly enhance the performance of a content-based image retrieval system. Figure 1(c) shows the top-5 retrieved results (again based on color histogram features) on a database of 760 *city* images for the same query; clearly, filtering out landscape images from the image database prior to querying improves the retrieval result.



(a)

(b)

(c)

Figure 1: Content-based retrieval results: (a) query image; (b) top-5 retrieved images from a database of $2,145$ city and landscape images; (c) top-5 retrieved images from a database of 760 city images; filtering out landscape images prior to querying clearly improves the retrieval result.

Current image database systems have not adequately addressed this rather difficult problem of effective indexing in large image databases. The main problem is that only low-level features can be reliably extracted from images as opposed to higher level features (objects present in the scene and their inter-relationships). For example, color distributions in terms of histograms can be extracted from any color image, but presence/absence of sky, trees, buildings, furniture, people, etc., cannot be reliably extracted from general images. Thus, the grouping of images cannot be based on high-level concepts as (probably) is done by humans. The main challenge, thereby, lies in grouping images into semantically meaningful categories (or indexing images in a database) based on the available low-level visual features of the images. One attempt to solve this problem is the hierarchical indexing scheme proposed by Zhang and Zhong [11, 12], which uses a Self-Organization Map (SOM) to perform clustering based on color and texture features. This indexing scheme was further applied in [13] to create a texture thesaurus for indexing a database of large aerial photographs. Forsyth et al. [14] used specialized grouping heuristics to classify coherent regions (blobs) in an image

under increasingly stringent conditions to recognize objects in the image. Yu and Wolf [15] used one-dimensional hidden Markov models along horizontal and vertical blocks of images to do scene classification. However, the success of such clustering-based indexing schemes is often limited, largely due to the low-level feature-based representation of image content. Yet, as we shall show in this paper, in constrained environments, *specific* low-level features can be used to discriminate between conceptual image classes. To achieve the goal of automatic categorization and indexing of images in a large database, we need to develop robust schemes to identify salient features of images that capture a certain aspect of semantic content of these images. In other words, we first need to specify/define pattern classes, so that the database images can be organized in a *supervised* fashion.

In this paper, we address the hierarchical classification of outdoor vacation images into *city* and *landscape* classes, and a further classification of a subset of landscape images into *sunset*, *forest*, and *mountain* classes. The city vs. landscape classification problem can be stated as follows: Given an image, classify it as either a city or a landscape image. City scenes can be characterized by the presence of man-made objects and structures such as buildings, cars, roads, etc. Natural scenes, on the other hand, lack these structures. A subset of landscape images can be further classified into one of the sunset, forest, and mountain classes (Figs. 2(a)-(b)). Sunset scenes can be characterized by saturated colors (red, orange, or yellow), forest scenes have predominantly green color distribution due to the presence of dense trees and foliage, and mountain scenes can be characterized by long distance shots of mountains (either snow covered, or barren plateaus). We assume that the input image does belong to one of the classes under consideration. This restriction is imposed because, automatically rejecting images not belonging to the specific classes (city or landscape in the first classification problem, and sunset, mountain, or forest in the second problem) is in itself a very difficult problem.

Our hierarchical organization of outdoor vacation images were motivated by earlier experiments on organization of image databases by human subjects [16]. These experiments with 8 human subjects on a database of 171 vacation images revealed the classification hierarchy as shown in Figure 2(a). Figure 2(b) shows our simplified classification hierarchy. The solid lines show the classification problems addressed in this paper. While the above hierarchy is not in itself complete (a user may be interested in querying the database for images captured in the evening - (day/night classification), or images containing faces (face vs. non-face classification)), it is a reasonable approach to simplify the image retrieval problem.

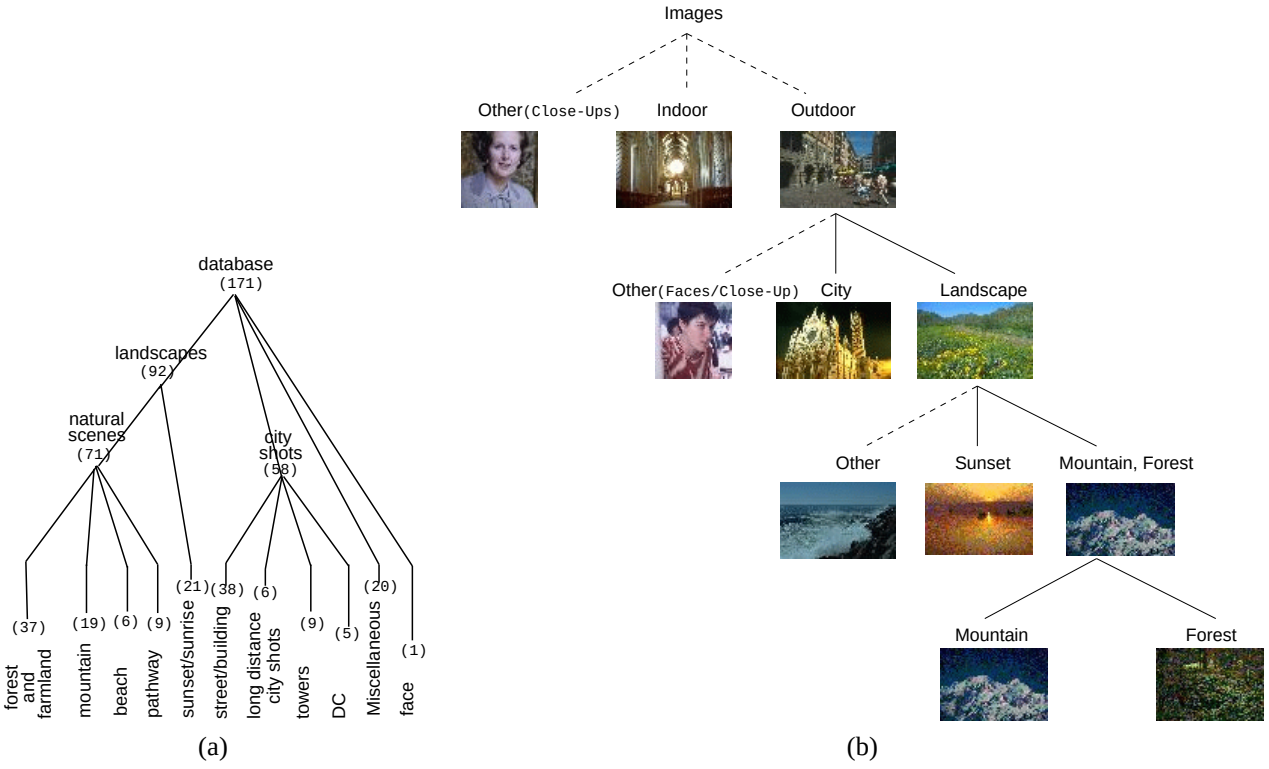


Figure 2: Experiments with human subjects: (a) A hierarchical organization of the 11 categories obtained from groupings provided by human subjects [16]; (b) Simplified semantic classification of images; solid lines show the classification problems addressed in this paper.

We have used a Bayesian approach for the above formulated "semantic" classification problems. We show how *specific* low-level image features can be used for high-level semantic categorization of images under the constraint that images do belong to one of the classes in question. The probabilistic models (class-conditional distributions of the various low-level features) required for the Bayesian approach are efficiently estimated using a Vector Quantization (VQ) framework [17, 18, 19] during a training phase. The MDL principle [20] is used to determine the optimal size of codebook vectors for the various classifiers. The Bayesian approach has the following advantages: (i) a small number of codebook vectors represent a particular class of images, thereby greatly reducing the number of comparisons necessary for each classification; (ii) it naturally allows for the integration of multiple features through the class-conditional densities; and (iii) it not only provides a classification rule, but also assigns a degree of confidence in the classification which may be used to build a reject option into the classifiers.

The paper is organized as follows. In Section 2, we discuss the Bayesian framework for image classification. Section 3 presents an introduction to Vector Quantization (VQ) and density estimation along with a detailed description of the MDL principle for estimating an optimal codebook size. Section 4 discusses the implementation issues, such as the choice of our feature sets and the application of the MDL principle to select an optimal codebook size for the various classifiers. The classifier performances are reported in Section 5. Section 6 finally concludes the paper and presents directions for future research.

2 Bayesian Framework

Bayesian theory provides a formal (probabilistic) framework for image classification problems. It requires that all assumptions be explicitly specified to build models (observation model, prior, loss function) which are then used to generate an "optimal" decision/classification rule. Optimality here means that, under the assumed models, there does not exist any other classification rule which has a lower classification error. The Bayesian paradigm has been successfully adopted in a number of image analysis and computer vision (both low and high level) problems, such as restoration, segmentation, and classification (see [21, 22] and the references therein). However, its use in content-based retrieval from image databases is just being realized [18].

2.1 Basic Elements

The Bayesian framework requires that all the entities involved in decision making be adequately formalized:

- Each observed image $\boldsymbol{x}$ belongs to a set $\mathcal{I}$ of possible images.
- The set $\mathcal{I}$ is assumed to be partitioned into K classes $\Omega = \{\omega_1, \omega_2, ...\omega_K\}$; these classes are exhaustive and mutually exclusive, i.e., any image $\boldsymbol{x}$ from $\mathcal{I}$ belongs to one and only one class.
- Each observed image $\boldsymbol{x}$ is modeled as a sample of a random variable $\mathbf{X}$, whose class-conditional probability density function is written as $f_{\mathbf{X}}(\boldsymbol{x}|\omega_n)$.
- An *a priori* knowledge concerning the classes is expressed via a probability function defined on the set of classes, $\{p(\omega_1), p(\omega_2), ..., p(\omega_K)\}$.
- A loss function, $\mathcal{L}(\omega, \hat{\omega}) : \Omega \times \Omega \rightarrow \mathcal{R}$, specifying the loss incurred when class $\hat{\omega}$ is chosen and the true class is ω. As is common in classification problems, we adopt the "$0/1$" loss function; $\mathcal{L}(\omega, \omega) = 0$, and $\mathcal{L}(\omega, \hat{\omega}) = 1$, if $\omega \neq \hat{\omega}$.
- Finally, the solution of the classification problem is a decision rule $\delta(\boldsymbol{x}) : \mathcal{I} \rightarrow \Omega$ which maps any possible observed image into one of the available classes.

2.2 Image Features

In many image analysis problems, it is typical that the classification is based on, say, m features extracted from the observed image, rather than directly on the raw pixel values. Let $\boldsymbol{y} = \{\boldsymbol{y^{(1)}}, \boldsymbol{y^{(2)}}, \ldots, \boldsymbol{y^{(m)}}\}$ denote the set of m features based on which the classification procedure must operate. For computational simplicity it is typical to assume that the features are conditionally independent. As a result, the class-conditional density functions can be written as

$$f_{\mathbf{X}}(\boldsymbol{x} \mid \omega) \equiv f_{\mathbf{Y}}(\boldsymbol{y} \mid \omega) = \prod_{i=1}^{M} f_{\mathbf{Y}^{(i)}}(\boldsymbol{y^{(i)}} \mid \omega). \tag{1}$$

The classification problem can be stated as: "given a set of observed features, $\boldsymbol{y}$, from an image $\boldsymbol{x}$, classify $\boldsymbol{x}$ into one of the classes in Ω.

2.3 Classification Rule

In the Bayesian framework, all inferences have to be based on the *a posteriori* probability function, which is obtained by combining the class-conditional observation models with the *a priori* class probabilities. This is done via Bayes law

$$p(\omega \mid \boldsymbol{y}) = \frac{f_{\mathbf{Y}}(\boldsymbol{y} \mid \omega)\, p(\omega)}{f_{\mathbf{Y}}(\boldsymbol{y})}, \tag{2}$$

where the denominator, $f_{\mathbf{Y}}(\boldsymbol{y})$, in Eq. (2) is the unconditional (or marginal) probability density function of the observed features, which serves simply as a normalizing constant.

The adopted "$0/1$" loss function leads to what is the most common criterion in Bayesian classification problems: choose the class whose *a posteriori* probability is maximum. This is known as the *maximum a posteriori* (MAP) criterion, and is given by

$$\hat{\omega} = \delta(\boldsymbol{x}) = \arg\max_{\omega \in \Omega} \{p(\omega \mid \boldsymbol{y})\} = \arg\max_{\omega \in \Omega} \{f_{\mathbf{Y}}(\boldsymbol{y} \mid \omega)\, p(\omega)\}. \tag{3}$$

In addition to reporting the MAP classification of a given image, say ω_k, the Bayesian approach also assigns a degree of confidence to that classification, which is proportional to $p(\omega_k \mid \boldsymbol{y})$. We next describe a procedure to estimate the class-conditional density functions.

3 Density Estimation Using Vector Quantization

The performance of the Bayes classifier clearly depends on the ability of the feature set $\boldsymbol{y}$ to discriminate among the various classes. Moreover, since the class-conditional densities have to be estimated from training data, the accuracy of these estimates is also critical. Automatically choosing the right set of features for a given classification task is a difficult problem and we do not discuss the issue here. We concentrate instead on estimating the class-conditional densities for which we adopt a Vector Quantization (VQ) approach [19].

3.1 Introduction to Vector Quantization

Vector Quantization (as its name suggests) is a compression algorithm that is applied to vectors rather than scalars. It takes as input a p-dimensional vector and quantizes it into a p-dimensional *reproduction vector*. In the compression and communication applications, a Vector Quantizer is described as a combination of an encoder and a decoder. A p-dimensional VQ consists of two mappings: an encoder γ which maps the input alphabet ($\mathbf{A}$) to the channel symbol set ($\mathbf{M}$), and a decoder β which maps the channel symbol set ($\mathbf{M}$) to the output alphabet ($\hat{\mathbf{A}}$), i.e., $\gamma(\boldsymbol{y}) : \mathbf{A} \rightarrow \mathbf{M}$ and $\beta(\boldsymbol{v}) : \mathbf{M} \rightarrow \hat{\mathbf{A}}$. A distortion measure $\mathcal{D}(\boldsymbol{y}, \hat{\boldsymbol{y}})$ specifies the cost associated with quantization, where $\hat{\boldsymbol{y}} = \beta(\gamma(\boldsymbol{y}))$. Usually, an optimal quantizer minimizes the average distortion under a size constraint on $\mathbf{M}$. An input vector $\boldsymbol{y} \in \mathbf{A}$ is quantized into one of the K output vectors $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$, also referred to as *codebook vectors*, such that

$$\mathcal{D}(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{i}}) \leq \mathcal{D}(\boldsymbol{y}, \hat{\boldsymbol{y}}_{\boldsymbol{j}}),\ \forall\, 1 \leq j \leq K. \tag{4}$$

These codebook vectors define a partition of the feature space, according to Eq. (4), into the so-called Voronoi cells, $\{S_i,\, i = 1, 2, \ldots, K\}$. Figure 3 shows an example of such a 2-D Voronoi tessellation where the $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ are shown as square dots. As the data points get closer, the cells become more compact. According to Eq. (4), an input vector is assigned the codebook vector of the cell it falls into. A comprehensive study of VQ, choice of distortion measures, and use of VQ in classification and compression, is presented in [23, 19].

3.2 VQ as a Density Estimator

Vector quantization provides an efficient tool for density estimation [19]. Consider n training samples from a class ω. In order to estimate the class-conditional density of the feature vector $\boldsymbol{y}^{(\boldsymbol{i})}$ given the class ω, i.e., $f_{\mathbf{Y}^{(\mathbf{i})}}(\boldsymbol{y}^{(\boldsymbol{i})} \mid \omega)$, a vector quantizer is used to extract q (with $q < n$, hopefully $q \ll n$) codebook vectors, $\mathbf{v}_{\mathbf{j}}^{(\mathbf{i})}$ $(1 \leq j \leq q)$, from the n training samples. It has been shown (see [19]) that in the so-called high-resolution approximation (i.e., for sufficiently

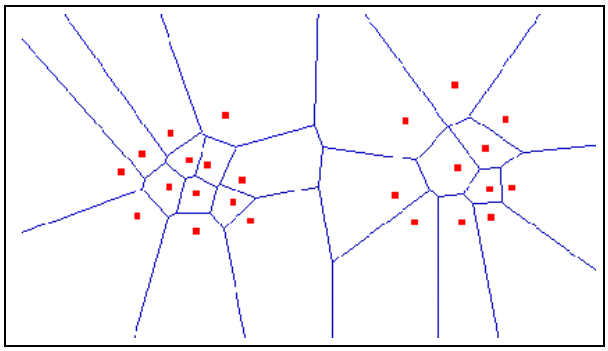


Figure 3: Voronoi Tessellation for 2-D data points in two clusters.

small Voronoi cells), the class-conditional density can be approximated as a piecewise-constant function over each cell $S_j^{(i)}$, with value

$$f_{\mathbf{Y}^{(i)}}(\boldsymbol{y}^{(i)} \mid \omega) \approx \frac{m_j^{(i)}}{Vol(S_j^{(i)})}, \tag{5}$$

where $m_j^{(i)}$ and $Vol(S_j^{(i)})$ are the ratio of training samples falling into cell $S_j^{(i)}$ and the volume of the cell $S_j^{(i)}$, respectively. This approximation fails if the Voronoi cells are not sufficiently small, as is the case when the dimensionality of the feature vector $\boldsymbol{y}^{(i)}$ is large. The class-conditional densities can then be approximated using a kernel-based approach [19, 18], approximating the density by a mixture of Gaussians, each centered at a codebook vector. In most VQ algorithms, the codebook vectors are iteratively selected by minimizing the MSE (mean square error) which is the sum of the Euclidean distances of each training sample from its closest codebook vector. Hence, an identity covariance matrix can be assumed for the Gaussian components used to represent the densities [18], the resulting (approximate) class-conditional densities being

$$f_{\mathbf{Y}^{(i)}}(\boldsymbol{y}^{(i)} \mid \omega) \propto \sum_{j=1}^{q} m_j^{(i)} * \exp(-\|\boldsymbol{y}^{(i)} - \boldsymbol{v}_j^{(i)}\|^2/2). \tag{6}$$

A more comprehensive approach would be to use the Mahalanobis distance [24] in estimating the codebook vectors; but, if feature dimensionality is high and the number of training samples is small, the estimated covariance matrices are likely to be singular.

3.3 Selecting Codebook Size

A key issue in using vector quantization for density representation is the choice of the codebook size. It is clear that, given a training set, the VQ-approximated likelihood (probability) of that training set will keep increasing as the dimension of the codebook grows; in the limit, we would have a code vector for each training sample, with the corresponding probability equal to one. To address this issue, we adopt the *minimum description length* (MDL) principle [20]. MDL is an information-theoretic criterion which has recently been used for several problems in computer vision and image processing (see [25], and references therein). The MDL criterion states that the description code-length to be minimized by the estimate must include both the data (training samples) code-length and also the code-lengths of the parameters ($\boldsymbol{\theta}_{(q)}$, containing the codebook vectors, $\{\mathbf{v}_\mathbf{j}^{(\mathbf{i})} : 1 \leq j \leq q\}$, and the weights $m_j^{(i)}$).

Our first key observation is that looking for an ML estimate of the parameters in the the mixture in Eq. (6) is equivalent to looking for the Shannon code for which the observations have the shortest code-length [20]; this is so because Shannon's optimal code-length[1], for some set of observations, $\mathcal{Y} : \{\boldsymbol{y}^{(i)}(1), \ldots, \boldsymbol{y}^{(i)}(n)\}$, obeying some joint probability density function $f(\mathcal{Y}|\boldsymbol{\theta}_{(q)})$, is simply [26, 25]

$$L(\mathcal{Y}|\boldsymbol{\theta}_{(q)}) = -\log f(\mathcal{Y}|\omega, \boldsymbol{\theta}_{(q)}). \tag{7}$$

Under the assumption of independent samples, $\boldsymbol{y}^{(i)}(j)$ $(1 \leq j \leq n)$, the joint likelihood in Eq. (7) can be written as

$$L(\mathcal{Y}|\boldsymbol{\theta}_{(q)}) = -\sum_{j=1}^{n} \log f(\boldsymbol{y}^{(i)}(j)|\omega, \boldsymbol{\theta}_{(q)}); \tag{8}$$

[1] In bits or *nats*, if base-2 or natural logarithms are used, respectively [26].

in our case, each of the marginals in the above likelihood is the one in Eq. (6).

The second fundamental fact is that the parameters themselves are also part of the code, in the following sense: a code word representing $\mathcal{Y}$ can not be decoded by itself; only a full knowledge of $f(\mathcal{Y}|\boldsymbol{\theta}_{(q)})$ (i.e., of its parameters) allows reconstructing the code and respective decoder. Accordingly, the MDL criterion states that the description code-length to be minimized by the estimate must also include the code-lengths of the parameters. The resulting criterion for the choice of q (codebook size) is then

$$\widehat{q} = \arg\min_{q} \left\{ L(\mathcal{Y}|\boldsymbol{\theta}_{(q)}) + L(\boldsymbol{\theta}_{(q)}) \right\}. \tag{9}$$

Finally, concerning the parameter description length, $L(\boldsymbol{\theta}_{(q)})$, the common choice is $L(\boldsymbol{\theta}_{(q)}) = (\zeta(q)/2)\log n$, where n is the sample size and $\zeta(q) = \{q + q\ \dim(\boldsymbol{y^{(i)}})\}$ is the number of real-valued parameters needed to specify a q^{th}-order model and $\dim()$ represents the dimension of the feature space [20]. This is an asymptotically optimal choice, which is only valid when all the parameters depend on all the data, which is not the case in the present problem. The weights $m_j^{(i)}$ are, in fact, estimated from all the data; however, each $\mathbf{v_j^{(i)}}$ is estimated from the $m_j^{(i)}$ samples that fall in the associated cell. Accordingly, we use the following parameter description length

$$L(\boldsymbol{\theta}_{(q)}) = \frac{q}{2}\log n + \frac{\dim(\boldsymbol{y^{(i)}})}{2}\sum_{j=1}^{q}\log(m_j^{(i)}\ n), \tag{10}$$

where the first term accounts for the weights m_j while the second one corresponds to the codebook vectors themselves.

4 Implementation Issues

We have built the hierarchical classifier on an image database of $2,716$ images of city and landscape scenes collected from various sources. The images are of varying sizes ranging from 150×150 to 750×750 and are represented by 24-bit color per pixel. No pre-processing is done on the images. The ground truth for the $2,716$ images was assigned based on a single subject. The final labeling assigned $1,128$ images to the city class and $1,588$ images to the landscape class.

4.1 Image Features

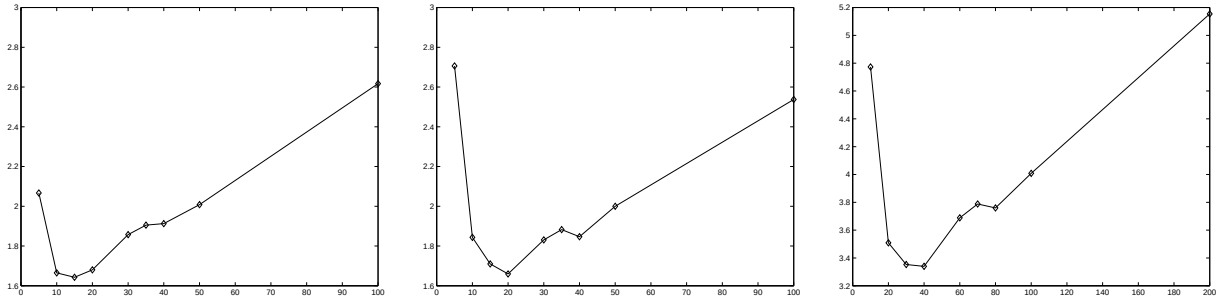
The accuracy of the Bayesian classifiers depends on the underlying low-level representation of the images. The more discriminative the features, better is the classification accuracy and using just any feature will not yield good classification results. As shown in [16], edge direction features (histograms and coherence vectors) have sufficient discriminative capability for city vs. landscape classification; therefore, we use these edge direction features for classification. This is so because, city scenes tend to have man-made structures which have structured horizontal and vertical edges, whereas landscape images have edges randomly distributed. Since, landscape images have characteristic distribution of colors (sky is generally blue, grass is green, etc), we also use color features (histograms and coherence vectors, described in [16]) as another set of features. Table 1 briefly describes the qualitative attributes of the various classes and the features used to represent them.

Classification Problem	Qualitative Attributes	Low-level Features
City vs. Landscape	distribution of edges	edge direction histograms and coherence vectors
Sunset vs. Forest vs. Mountain	global color distributions and saturation values	color histograms and coherence vectors in HSV space

Table 1: Feature saliency: Qualitative attributes of various outdoor vacation photograph classification problems and associated low-level features used for discrimination.

4.2 Vector Quantization

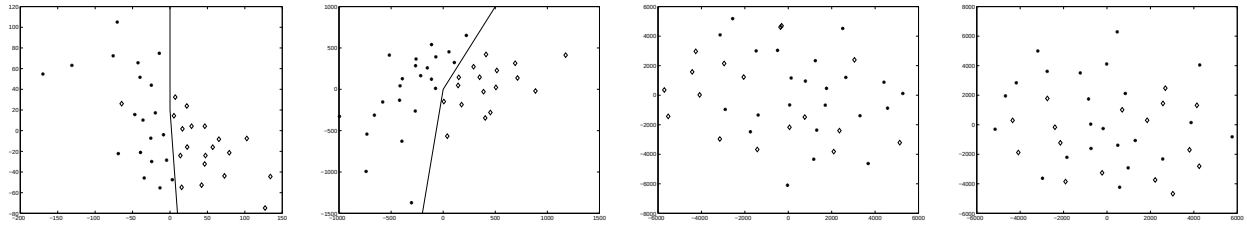
We have used the LVQ_PAK package [27] for vector quantization. For every class, half of the database samples were used to train the VQ for each of the four features (edge direction histograms, edge direction coherence vectors, color histograms, and color coherence vectors). The MDL principle [20] described in Section 3.3 was used to determine the codebook size from the training samples for each of the four features for the city vs. landscape classification problem. Figures 4(a)-(c) show the plots of $L(\mathcal{Y}|\boldsymbol{\theta}_{(q)}) + L(\boldsymbol{\theta}_{(q)})$ (criterion to be minimized in Eq. (9)) vs. the codebook size, q, using the edge direction coherence vector for (a) city, (b) landscape, and (c) both the classes. As can be seen in the figures, $q \sim 15$ minimizes the criterion in Eq. (9) for the city class, whereas $q \sim 20$ is optimal for the landscape class. Combining the two classes, $q \sim 40$ is the optimal size of the codebook based on the training samples. Color coherence vector also yielded an optimal size of 40 codebook vectors for the two classes combined, although the curve was more noisy. Based on the above analysis, 40 codebook vectors were selected for the city vs. landscape classification for each of the four features. For further classification of landscape images, a codebook of 5 vectors was selected for each class.



(a) (b) (c)

Figure 4: Determining codebook size for edge direction coherence vectors; (a) city class; (b) landscape class; (c) city and landscape classes combined; x-axis represents the codebook size and y-axis represents the optimality criterion to be minimized.

In order to verify the claim that edge direction features have more discriminatory power for the given classification problem, we used Sammon's non-linear projection algorithm [24] to plot the codebook vectors for each of the four features in two dimensions (Figures 5 (a)-(d)). The edge direction-based codebook vectors for the city and landscape classes can be more easily distinguished than the color-based codebook vectors, which confirms that the edge direction features have more discriminatory power for the given classification problem.



(a) (b) (c) (d)

Figure 5: $2D$ plots of codebook vectors showing the discrimination ability of (a) edge direction histogram features, (b) edge direction coherence vector features, (c) color histogram features, and (d) color coherence vector features; $*$ represents the landscape patterns and $\diamond$ represents the city patterns; 40 codebook vectors were extracted from the training samples.

Given an input image, the classifier compares its features with the stored codebook vector features of a particular class (say, "city" class) and estimates the class-conditional probabilities for each of the features using Eq. (6). These probabilities are then used to estimate the a posteriori probability (Eq. (2)) of the image having come from the city class, given its observed feature vectors.

5 Experimental Results

We conducted a variety of experiments to measure the classification accuracies of the various classifiers. The experiments included measuring the performance on an independent test set (hold-out error) as well as on the training patterns (re-substitution error). Each of the query images was input to the classifier and the a posteriori class probabilities were computed for each of the features individually, i.e., the feature set $\boldsymbol{y}$ had only one element (edge direction histogram, edge direction coherence vector, color histogram, color coherence vector feature), as well as for the combination of features, i.e., the feature set $\boldsymbol{y}$ had two elements (one edge direction feature and one color feature). The query image was assigned to the class with the highest a posteriori probability.

5.1 City Vs. Landscape Classification

Table 2 shows the classification results for the city vs. landscape classification problem. Edge direction coherence vector provides the best accuracy of 96.7% for the training data and 92.7% for the test data. A total of 126 images were misclassified (a classification accuracy of 95.3%) when the edge direction coherence vector was combined with the color histogram feature. Of these 56 were city images and 70 were landscape images. Figures 6(a) and (b) show a representative subset of the misclassified city and landscape images, respectively. The main reasons for the misclassification of city images can be identified as follows: (i) long distance city shots at night (inability to extract edges), (ii) top view of city scenes (lack of vertical edges), (iii) highly textured buildings, and (iv) trees obstructing the buildings. Most of the misclassified landscape images had strong vertical edges from tree trunks, close-ups of stems, fences, etc., that led to their assignment to the city class. The results show that the edge direction coherence vector features have sufficient discrimination power for the city vs. landscape classification problem. Combining color feature does not drastically improve the classification accuracies.

Test Data	EDH	EDCV	CH	CCV	EDH & CH	EDH & CCV	EDCV & CH	EDCV & CCV
Training Set	94.7	96.7	83.7	83.5	94.8	95.4	96.4	96.9
Test Set	92.0	92.7	75.4	76.0	92.5	92.8	93.4	93.8
Entire Database	93.4	94.7	79.6	79.8	93.7	94.1	94.9	95.3

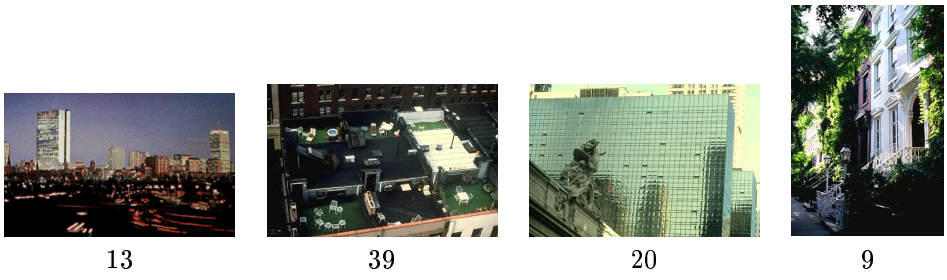
Table 2: Classification accuracies (in %) for city vs. landscape classification problem; the features are abbreviated as follows: edge direction histogram (EDH), edge direction coherence vector (EDCV), color histogram (CH), and color coherence vector (CCV).

5.2 Further Classification of Landscape Images

While our limited experiments on human subjects [16] revealed classes such as sunset/sunrise, forest and farmland, mountain, pathway, water scene, etc., these groups were not consistent among the subjects in terms of the actual labeling of the images. We found it extremely difficult to generate a semantic partitioning of landscape images. We thus restricted classification of landscape images into three classes that could be more unambiguously distinguished, namely, sunset, forest, and mountain classes. Figures 7 (a)-(c) show typical images from the forest, mountain, and sunset classes. A subset of 528 landscape images were classified into the above defined classes. Of the 528 images, a human subject labeled 177, 196, and 155 images as belonging to the forest, mountain, and sunset classes, respectively. A 2-stage classifier was constructed. First, we classify an image into either sunset or the forest & mountain class. We next address the forest vs. mountain classification problem.

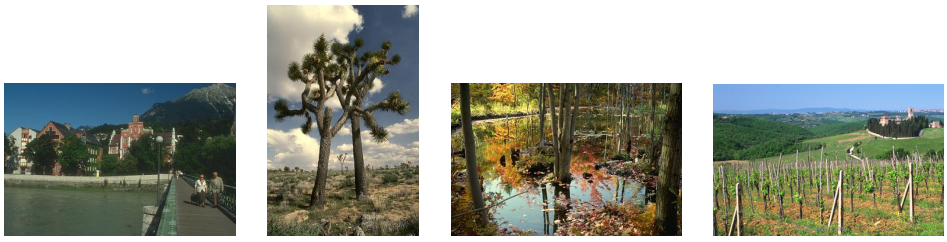
Table 3 shows the classification results for the individual features for the classification of landscape images into sunset vs. forest & mountain classes. The color coherence vector provides the best accuracy of 96.2% for the training data and 93.5% for the test data. Color features do much better than the edge direction features here, since color distributions remain more or less constant for natural images (blue sky, green grass, trees, plants, etc). A total of 26 images were misclassified (a classification accuracy of 95.1%) when the color coherence vector was combined with edge direction coherence vector. We find that there is not much improvement in the classification accuracy with the combination of features. This shows that color coherence vector has sufficient discrimination ability for the given classification problem.

Table 4 shows the classification results for the forest vs. mountain classification problem (373 images in the database). Color coherence vector provides the best accuracy of 94.7% for the training data and 91.4% for the test



13 39 20 9

(a)



42 32 8 1

(b)

Figure 6: A subset of the misclassified (a) city and (b) landscape images and the corresponding confidence values (in %) associated with the true class using a combination of edge direction coherence vector and color histogram features.

Test Data	EDH	EDCV	CH	CCV	EDH & CH	EDH & CCV	EDCV & CH	EDCV & CCV
Training Set	88.3	88.3	96.2	96.2	95.9	95.9	95.5	95.9
Test Set	86.3	89.0	89.7	93.5	90.1	93.9	90.5	94.3
Entire Database	87.4	88.7	93.0	94.9	93.0	94.9	93.0	95.1

Table 3: Classification accuracies (in %) for sunset vs. forest & mountain classification.

data. A total of 24 images were misclassified (a classification accuracy of 93.6%) when the color histogram feature was combined with either of the edge direction features. Again, the combinations of features did not perform better than the individual color features, showing that color features are quite adequate for this classification problem.

Test Data	EDH	EDCV	CH	CCV	EDH & CH	EDH & CCV	EDCV & CH	EDCV & CCV
Training Set	83.4	78.1	92.0	94.7	94.1	92.5	93.6	93.1
Test Set	87.1	77.2	91.4	91.4	93.0	91.9	93.5	93.0
Entire Database	85.3	77.7	91.7	93.1	93.6	92.2	93.6	93.0

Table 4: Classification accuracies (in %) for forest vs. mountain classification.

6 Conclusion and Future Work

Content-based indexing and retrieval has emerged as an important area in computer vision and multimedia computing. User queries are typically based on semantics and not on low-level image features. It is a challenging problem to provide high-level semantic indices into large databases. In this paper, we show that specific high-level semantic categories can be learnt using specific low-level image features under the constraint that the test images do belong to one of the classes in concern. Specifically, we have developed a hierarchical classifier for classifying outdoor vacation images. Outdoor images are classified into city and landscape class and a subset of landscape images are classified

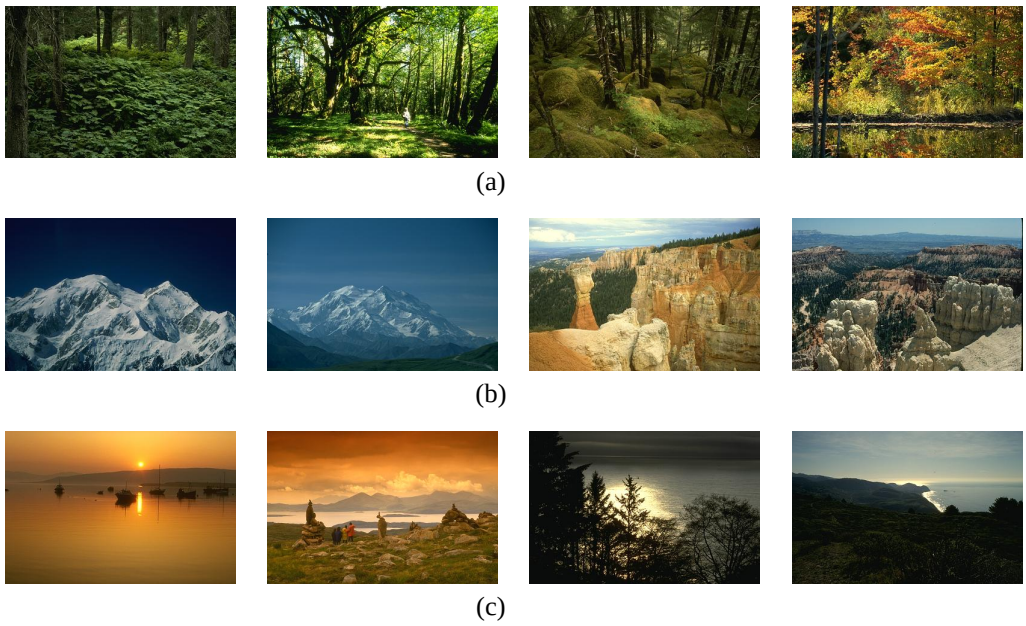


Figure 7: Typical (a) forest, (b) mountain, and (c) sunset/sunrise images.

into the sunset, forest, and mountain class. The classification problems have been formalized using the Bayesian framework wherein Vector Quantization is used to estimate the class-conditional probability densities of the observed features. The MDL principle is applied to extract the optimal codebook size for the various classifiers. Our Bayesian approach has the following advantages: (i) a small number of codebook vectors represent a particular class of images, thereby greatly reducing the number of comparisons necessary for each classification; (ii) it naturally allows for the integration of multiple features through the class-conditional densities; and (iii) it not only provides a classification rule, but also assigns a degree of confidence in the classification which may be used to build a simple reject option into the classifiers. Classifications based on color histograms, color coherence vectors, edge direction histograms, and edge direction coherence vectors as features show promising results.

The accuracy of the above classifiers depends on the feature set used, the training samples, and their ability to learn from the training samples. We are currently working on adding a progressive/incremental learning paradigm to the classifiers, so that they can improve their performance over time as more training data is presented. Another challenging issue is to introduce a reject option. In the simplest form, the a posteriori class probabilities can be used for rejection (rejecting images with say less than 60% probability of belonging to any class). We are looking at other means of adding the reject option into the system. Finally, we intend to add other classifiers into the system, such as indoor vs. outdoor, day vs. night, people vs. non-people, text vs. non-text classification, etc. These classifiers can be added along with the present hierarchy to generate semantic indices into the database.

References

[1] B. Holt and L. Hartwick, "Visual image retrieval for applications in art and art history," in *Proc. SPIE Vol. 2185: Storage and Retrieval for Image and Video Databases II*, pp. 70–81, February 1994.

[2] B. Kroepelien, K. Miller, and A. K. Jain, "SILVERSMITH: An Expert System for Norwegian Silver Tankards," in 14*th International Conference on Pattern Recognition*, (Brisbane, Australia), August 16-20 1998.

[3] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz, "Efficient and effective querying by image content," *Journal of Intelligent Information Systems*, vol. 3, pp. 231–262, 1994.

[4] A. Pentland, R. W. Picard, and S. Sclaroff, "Photobook: Content-based manipulation of image databases," *SPIE Vol 2185: Storage and Retrieval for Image and Video Databases II*, pp. 34–47, February 1994.

[5] R. W. Picard and T. P. Minka, "Vision texture for annotation," *Multimedia Systems: Special Issue on Content-based Retrieval*, vol. 3, pp. 3–14, 1995.

[6] H. J. Zhang, C. Y. Low, S. W. Smoliar, and J. H. Wu, "Video parsing retrieval and browsing: An integrated and content-based solution," in *Proc. ACM Multimedia '95*, (San Francisco, CA), pp. 15–24, November, 5-9 1995.

[7] A. Hampapur, A. Gupta, B. Horowitz, C. F. Shu, C. Fuller, J. Bach, M. Gorkani, and R. Jain, "Virage video engine," in *Proc. SPIE: Storage and Retrieval for Image and Video Databases V*, (San Jose), pp. 188–197, February 1997.

[8] J. R. Smith and S. F. Chang, "Visualseek: A fully automated content-based image query system," in *Proc. ACM Multimedia*, pp. 87–98, Nov 1996.

[9] W. Y. Ma and B. S. Manjunath, "Netra: A toolbox for navigating large image databases," in *Proc. Int. Conf. on Image Proc.*, vol. 1, (Santa Barbara, CA), pp. 568–571, Oct 26-29 1997.

[10] S. Mehrotra, Y. Rui, M. Ortega, and T. S. Huang, "Supporting content-based queries over images in MARS," in *Proc. IEEE Int. Conf. on Multimedia Computing and Systems*, 1997.

[11] H. J. Zhang and D. Zhong, "A scheme for visual feature based image indexing," in *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases*, (San Jose, CA), pp. 36–46, February 1995.

[12] D. Zhong, H. J. Zhang, and S.-F. Chang, "Clustering methods for video browsing and annotation," in *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases*, (San Jose, CA), February 1995.

[13] W. Y. Ma and B. S. Manjunath, "Image indexing using a texture dictionary," in *Proc. SPIE Conference on Image Storage and Archiving System*, vol. 2606, (Philadelphia, PA), pp. 288–298, October 1995.

[14] D. A. Forsyth, J. Malik, M. M. Fleck, H. Greenspan, T. Leung, S. Belongie, C. Carson, and C. Bregler, "Finding pictures of objects in large collections of images," in *International Workshop on Object Recognition for Computer Vision*, (Cambridge, England), April 13-14 1996. http://www.cs.berkeley.edu/projects/vision/publications.html.

[15] H.-H. Yu and W. Wolf, "Scenic classification methods for image and video databases," in *Proc. SPIE, Digital Image Storage and Archiving Systems*, pp. 363–371, 1995.

[16] A. Vailaya, A. K. Jain, and H. J. Zhang, "On Image Classification: City vs. Landscape," in *IEEE Workshop on Content-Based Access of Image and Video Libraries*, (Santa Barbara, CA), pp. 3–8, June 21 1998. To also appear in Pattern Recognition, 1998.

[17] R. M. Gray, "Vector quantization," *IEEE ASSP Magazine*, vol. 1, pp. 4–29, April 1984.

[18] N. Vasconcelos and A. Lippman, "Library-based coding: a representation for efficient video compression and retrieval," in *Data Compression Conference'97*, (Snowbird, Utah), 1997.

[19] R. M. Gray and R. A. Olshen, "Vector quantization and density estimation," in *SEQUENCES97*, 1997. http://www-isl.stanford.edu/ gray/compression.html.

[20] J. Rissanen, *Stochastic Complexity in Stastistical Inquiry*. Singapore: World Scientific, 1989.

[21] R. C. Dubes and A. K. Jain, "Random field models in image analysis," *Journal of Applied Statistics*, vol. 16, no. 2, pp. 131–164, 1989.

[22] S. Z. Li, *Markov Random Field Modeling in Computer Vision*. Tokyo: Springer-Verlag, 1995.

[23] P. C. Cosman, K. L. Oehler, E. A. Riskin, and R. M. Gray, "Using vector quantization for image processing," *Proc. IEEE*, vol. 81, pp. 1326–1341, September 1993.

[24] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, New Jersey: Prentice Hall, 1988.

[25] M. Figueiredo and J. Leitão, "Unsupervised image restoration and edge location using compound Gauss-Markov random fields and the MDL principle," *IEEE Transactions on Image Processing*, vol. IP-6, pp. 1089–1102, August 1997.

[26] T. Cover and J. Thomas, *Elements of Information Theory*. New York: John Wiley & Sons, 1991.

[27] T. Kohonen, J. Kangas, J. Laaksonen, and K. Torkkola, "LVQ_PAK: A program package for the correct application of Learning Vector Quantization algorithms," in *Proc. Intl' Joint Conf. on Neural Networks*, (Baltimore), pp. I 725–730, June 1992.