Automatic Image Orientation Detection

Aditya Vailaya, *Member, IEEE*, HongJiang Zhang, *Senior Member, IEEE*, Changjiang Yang, Feng-I Liu, and Anil K. Jain, *Fellow, IEEE*

***Abstract*—We present an algorithm for automatic image orientation estimation using a Bayesian learning framework. We demonstrate that a small codebook (the optimal size of codebook is selected using a modified MDL criterion) extracted from a learning vector quantizer (LVQ) can be used to estimate the class-conditional densities of the observed features needed for the Bayesian methodology. We further show how principal component analysis (PCA) and linear discriminant analysis (LDA) can be used as a feature extraction mechanism to remove redundancies in the high-dimensional feature vectors used for classification. The proposed method is compared with four different commonly used classifiers, namely k-nearest neighbor, support vector machine (SVM), a mixture of Gaussians, and hierarchical discriminating regression (HDR) tree. Experiments on a database of 16 344 images have shown that our proposed algorithm achieves an accuracy of approximately 98% on the training set and over 97% on an independent test set. A slight improvement in classification accuracy is achieved by employing classifier combination techniques.**

***Index Terms*—Bayesian learning, classifier combination, expectation maximization, feature extraction, hierarchical discriminant regression, image database, image orientation, learning vector quantization, support vector machine.**

I. Introduction

CONTENT-BASED image organization and retrieval has emerged as an important area in computer vision and multimedia computing, due to the technological advances in digital imaging, storage, and networking. With the development of digital photography as well as inexpensive scanners, it is possible for us to store vacation and family photographs on our personal computers. This has created a need for developing image management systems that assist the user in storing, indexing, browsing, and retrieving images from a database. All image management systems require information about the true image *orientation*. When a user scans a picture, she expects the resulting image to be displayed in its correct orientation, regardless of the orientation in which the photograph was placed on the scanner. Thus, an image management system is expected to correctly orient the input images. Currently, this operation of orientation detection is performed manually.

Manuscript received January 9, 2001; revised February 13, 2002. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Josiane B. Zerubia.

A. Vailaya is with Agilent Technologies, Palo Alto, CA 94303-0867 USA (e-mail: aditya_vailaya@agilent.com).

H. Zhang is with Microsoft Research China, Beijing 100080, China (e-mail: hjzhang@microsoft.com).

C. Yang, F.-I. Liu, and A. K. Jain are with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: yangcha1@cse.msu.edu; liufeng@cse.msu.edu; jain@cse.msu.edu).

Publisher Item Identifier 10.1109/TIP.2002.801590.

Fig. 1. An image whose orientation was incorrectly estimated by our algorithm.

Automatic image orientation detection is a very difficult problem. Humans use object recognition and contextual information to identify the correct orientation of an image. Unfortunately, the state-of-the-art computer vision techniques still cannot infer the high-level knowledge abstraction of the objects in the real world [33]. The alternative method is to exploit the low-level features (e.g., spatial color distributions, texture, etc.) from the images for orientation detection [15]. Fig. 1 illustrates the difficulty in image orientation estimation, where the true orientation cannot be detected unless you first recognize the object present in the image. Close-up images, low-contrast images, or images of uniform or homogeneous texture (e.g., sunset/sunrise and indoor images) pose additional problems for robust orientation estimation.

We assume that the input image is restricted to only four possible rotations that are multiples of 90°, i.e., the photograph which is scanned can differ from its correct orientation by 0° (no rotation), 90°, 180°, or 270°. This is reasonable since photographs placed on a scanner are usually aligned with horizontal or vertical boundaries of the scanner plate. The orientation detection problem is then defined as the following four-class classification problem: given an image (a scanned photograph), determine its correct orientation from among the four possible orientations of (0°—no rotation, 90°, 180°, and 270°). Note that an image in an arbitrary orientation can easily be rotated into one of the previous four orientations by aligning the image boundaries horizontally and vertically. Fig. 2(a) shows an image in four possible orientations and Fig. 2(b) shows the true orientation detected by our algorithms for the four images in Fig. 2(a).

The literature on image orientation detection is rather sparse. Most of the literature has emphasized related topics such as page orientation detection [4], [5], [23], [28], [43]. In this paper, we present a Bayesian framework for image orientation detection. We use spatial color moments as the features for classification.

1057-7149/02$17.00 © 2002 IEEE

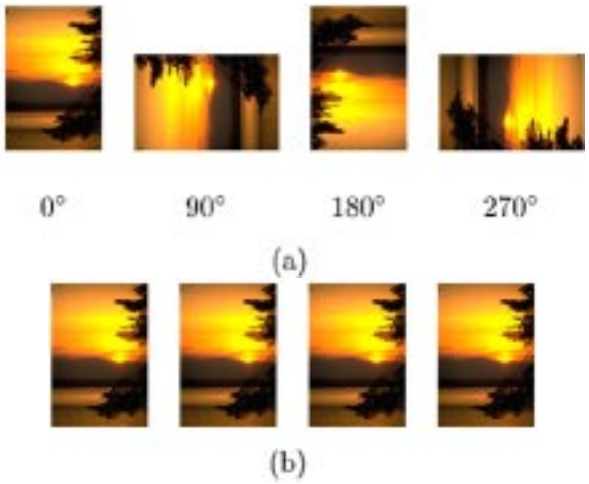

Fig. 2. Orientation detection: (a) four possible orientations of an image and (b) correct orientations detected by our algorithm.

Each image is represented by a multidimensional feature vector and the classification problem uses a Bayesian learning framework. The class-conditional probability density functions of the observed feature vector are estimated by the learning vector quantization (LVQ) technique [36].

The paper is organized as follows. We present our Bayesian learning framework with classification results in Section II. Section III discusses methods for feature extraction and selection to improve the robustness of the classifier. We compare the proposed Bayesian classifier with other classifiers commonly reported in the literature in Section VI. Section V presents results with classifier combination and we finally conclude in Section VI.

II. Proposed Method

Bayesian methods have been successfully adopted in many image analysis and computer vision problems. However, its use in content-based retrieval from image databases is just being realized [36], [41], [42]. The orientation detection problem formulated above is addressed here using the Bayes decision theory. Each image is represented by a feature vector extracted from the image. The probabilistic models required for the Bayesian approach are estimated during a training phase; in particular, the class-conditional probability density functions of the observed feature vector are estimated under a vector quantization (VQ) framework [14], [16], [17], [36]. Consider n training samples from a class ω. A vector quantizer is used to extract q ($q < n$) codebook vectors, $\boldsymbol{v_j}$ ($1 \leq j \leq q$), from the n training samples. A modified MDL principle, described in Section II-C, is used to select the optimal codebook size, q. The class-conditional density of a feature vector $\boldsymbol{y}$ given class ω, i.e., $f_{\mathbf{Y}}(\boldsymbol{y}|\omega)$ is then approximated by a mixture of Gaussians (with identity covariance matrices), each centered at a codebook vector, resulting in

$$f_{\mathbf{Y}}(\boldsymbol{y}|\omega) \propto \sum_{j=1}^{q} m_j * \exp(-||\boldsymbol{y} - \boldsymbol{v_j}||^2/2) \qquad (1)$$

where m_j is the proportion of training samples assigned to $\boldsymbol{v_j}$. The Bayesian classifier is then defined using the *maximum a posteriori* (MAP) criterion as follows:

$$\hat{\omega} = \arg\max_{\omega \in \Omega} \{p(\omega|\boldsymbol{y})\} = \arg\max_{\omega \in \Omega} \{f_{\mathbf{Y}}(\boldsymbol{y}|\omega),\, p(\omega)\} \qquad (2)$$

where $\Omega = \{\omega_1, \omega_2, \omega_3, \omega_4\}$ is the set of pattern classes and $p(\omega)$ represents the *a priori* class probability.

A number of VQ techniques have been developed and proposed in the literature. These include entropy-constrained vector quantization (ECVQ) [6], Bayes vector quantization [31], and learning vector quantization (LVQ) [25], [26]. Since we have a classification problem at hand, we concentrate only on supervised algorithms for VQ. Therefore, we did not consider ECVQ for vector quantization. While Bayes VQ is a parametric vector quantization technique (needing a measure of posteriori density), LVQ is a nonparametric vector quantization technique. Therefore, it does not attempt to obtain the posteriori estimates of the underlying probability density models that generate the data. This is of high significance in most image classification problems where the training data is limited and the dimensionality of the feature set is very high. Under these circumstances, parametric approaches generally perform worse than nonparametric techniques. We present experimental results comparing LVQ-based classifier to a parametric approach, a mixture of Gaussians using EM algorithm to estimate parameters of the mixture (henceforth, this classifier will be referred to as a mixture of Gaussian classifier) in Section II-B and Section IV, to demonstrate our point. The mixture of Gaussian (using EM) classifier tends to perform significantly worse under high-dimensional data, but its performance improves as the dimensionality of the data is reduced considerably, i.e., ratio of number of training samples to feature dimensionality increases. Due to this ability of nonparametric approaches to scale well to high-dimensional data, we have selected LVQ as the VQ algorithm in our approach.

A. Implementation Issues

There are many potential features which can be used to represent an image. Different features have different abilities to detect the four possible orientations of the scanned image. Since global image features are not invariant to image rotation, we use local regional features for the classification [38]. An image is represented in terms of $N \times N$ blocks ($N = 10$) and the features are extracted from these local regions. A number of features (e.g., color moments in the LUV color space [35]; color histograms in the HSV color space [37]; edge direction histograms [37]; and MSAR texture features [30]) were evaluated for their ability to discriminate between the four possible orientations of an image. Feature saliency was evaluated using a k-nearest neighbor classifier. Preliminary results showed that spatial color moment features (especially, after normalization—see Section IV-E for results with k-NN classifier on normalized spatial color moment data) yielded a much higher accuracy for the four-class classification problem than the color histogram, edge direction histogram, and MSAR features. Certain image classes such as outdoor images, tend

to have uniformity in spatial color distributions (sky is on top and typically blue in color; grass is typically green and toward the bottom of an image, etc.) and direction of illumination (illumination is usually from the top), while texture features have much lower variations with changes in image orientation. We therefore, trained our Bayesian classifier using 600 (100 blocks $*$ {3 mean and 3 variance values of L, U, and V components}) spatial color moment features. The features were normalized to the same scale as follows:

$$y_i' = \frac{y_i - \min_i}{\max_i - \min_i} \tag{3}$$

where y_i represents the ith feature component of a feature vector y, $\min_i$ and $\max_i$ represent the range of values for the ith feature component over the training samples, and y_i' is the scaled feature component.

B. Selecting Classification Scheme

We are posed with a problem of estimating the underlying posterior probability distribution given a high-dimensional feature vector (600 dimensions) and a relatively small number of training samples. In order to simplify the problem, it is assumed that the underlying class-conditional probability distribution can be represented as a mixture of Gaussians and the Bayes rule can be applied for classification. However, estimating the parameters of the class-conditional probability distribution is still a difficult problem. The expectation maximization (EM) technique [8], [12] and vector quantization (VQ) are two main techniques employed to model the data. While the EM technique attempts to estimate the parameters of a mixture of Gaussians, VQ attempts to quantize the data both for compression and classification. Due to the high-dimensionality of the feature vector, nonparametric techniques are expected to outperform parametric techniques (nonparametric techniques have a better scaling ability) for estimating the class-conditional probability distribution (see Section II-D for an empirical demonstration of this property). Kohonen's LVQ [25], [26] is an ideal choice for such a classifier, since it is both supervised (using positive and negative examples for selecting the codebook vectors) and nonparametric in nature. Therefore, we have used LVQ for estimating codebook vectors and then fitting Gaussian kernels over the codebook vectors for classification. Ideally, the Gaussian kernels fitted over the codebook vectors can be estimated using the EM algorithm. However, due to the high-dimensionality of the feature vector and relatively small number of training samples [20], we empirically estimate the parameters of the Gaussian (in our case, the variance, since the weight and mean of the Gaussian components are represented by the (normalized) number of training samples closest to a codebook vector, and the codebook vectors, respectively).

C. Selecting Optimal Codebook Size

A key issue in using vector quantization for density estimation is the choice of the codebook size. It is clear that, given a training set, the VQ-approximated likelihood (probability) of the training set will monotonically increase 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 to select the "optimal" codebook size [32]. The MDL criterion states that the description code-length to be minimized by the estimate must include not only the data code-length but also the code-lengths of the parameters. The resulting criterion for the choice of q (codebook size) is then

$$\hat{q} = \arg\min_q \left\{ L\left(\mathcal{Y}|\boldsymbol{\theta}_{(q)}\right) + L\left(\boldsymbol{\theta}_{(q)}\right) \right\} \tag{4}$$

where the first term represents the data code-length and the second term represents the code-length of the parameters.

The first key observation behind MDL is that finding an ML estimate is equivalent to finding the Shannon code for which the observations have the shortest code-length [32]; this is so because Shannon's optimal code-length,[1] for a set of observations, $\mathcal{Y}$: $\boldsymbol{y^{(i)}}(1), \ldots, \boldsymbol{y^{(i)}}(n)\}$, obeying a joint probability density function $f(\mathcal{Y}|\boldsymbol{\theta}_{(q)})$, is simply [7], [11]

$$L\left(\mathcal{Y}|\boldsymbol{\theta}_{(q)}\right) = -\log f\left(\mathcal{Y}|\omega, \boldsymbol{\theta}_{(q)}\right). \tag{5}$$

Under the assumption of independent samples, $\boldsymbol{y^{(i)}}(j)$ $(1 \leq j \leq n)$, the joint likelihood in (5) can be written as

$$L\left(\mathcal{Y}|\boldsymbol{\theta}_{(q)}\right) = -\sum_{j=1}^{n} \log f\left(\boldsymbol{y^{(i)}}(j)|\omega, \boldsymbol{\theta}_{(q)}\right). \tag{6}$$

In our case, each of the marginals in the above likelihood is given by (1) and $\boldsymbol{\theta}_{(q)}$ contains the codebook vectors, $\{\mathbf{v_j^{(i)}}: 1 \leq j \leq q\}$, and the weights $m_j^{(i)}$.

Concerning the parameter description length, $L(\boldsymbol{\theta}_{(q)})$, the most commonly used 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 qth-order model and $\dim(.)$ represents the dimensionality of the feature space [32]. This is an asymptotically optimal choice, which is only valid when all the parameters depend on all the data, which is not applicable here. 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 obtain the following *modified* MDL (MMDL) criterion

$$\hat{q} = \arg\min_q \cdot \left\{ L\left(\mathcal{Y}|\boldsymbol{\theta}_{(q)}\right) + \frac{q}{2}\log n + \frac{\dim\left(\boldsymbol{y^{(i)}}\right)}{2} \sum_{j=1}^{q} \log\left(m_j^{(i)} n\right) \right\} \tag{7}$$

where the first term is the negative log-likelihood of the observations, the second one accounts for the weights $m_j^{(i)}$, while the third one corresponds to the codebook vectors themselves.

D. Experimental Results

We have tested our orientation detection system on a database of 16 344 images, which consists of 7980 training samples (1995 samples per class) and 8364 test samples (2091 samples per class). The database consists of professional quality images from Corel stock photo library. The LVQ package described in

[1] In bits or *nats*, if base-2 or natural logarithms are used, respectively [7].

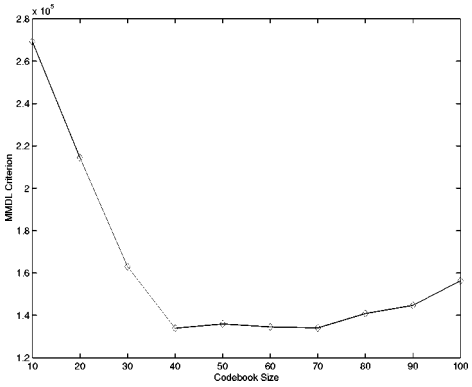

Fig. 3. Determining the optimal codebook size.

[27] was used to determine codebook vectors. The MMDL criterion, described in Section II-C, was used to determine the codebook size. Fig. 3 shows the plot of $L(\mathcal{Y}|\boldsymbol{\theta}_{(q)})+L(\boldsymbol{\theta}_{(q)})$ [criterion in (4)] versus the codebook size, q, for the spatial color moment features for all the four classes. We found that $q \sim 40$ (ten codebook vectors per class) minimizes the criterion in (7) for all the four classes. The classifier yielded an accuracy of 98.6% on the training set and 96.8% on the independent test set.

We compared the previous LVQ-based classifier to a mixture of Gaussian classifier using the EM algorithm for parameter estimation. We have employed the method described in [10], wherein the number of mixture components is automatically estimated along with the mixture parameters by incorporating a MDL technique within the EM steps. The EM method was employed in a supervised fashion, learning a mixture for each class separately and then classifying based on the MAP classifier (a diagonal covariance was assumed for the distribution). The mixture of Gaussian method yielded a single Gaussian component for each of the four classes. We feel this is mainly an artifact of the high-dimensional nature of the feature set (600 dimensions) and availability of a small number of training samples (1995 samples). The mixture of Gaussian classifier yielded an accuracy of 79.3% on the training data and 88.4% on the test data. Section IV demonstrates how the performance of the mixture of Gaussian classifier (using the EM approach) improves significantly as the dimensionality of the feature set is reduced.

III. Feature Extraction

The performance of a classifier trained on a finite number of samples starts deteriorating after a point as more and more features are added (*curse of dimensionality* [20]). A limited yet salient feature set simplifies both the pattern representation and the resulting classifiers that are built using the chosen representation. Consequently, the classifier will be faster and use less memory, and be able to alleviate the curse of dimensionality.

To reduce the dimensionality of the feature vector, we utilized the principal component analysis (PCA) and linear discriminant analysis (LDA) [9], [22]. PCA, which is one of the most well-known feature extractor, selects the m largest eigenvectors of the $d \times d$ covariance matrix of the d-dimensional patterns as the new features, $m \ll d$. Whereas PCA is an unsupervised linear feature extraction method, LDA [21] uses the category information associated with each training pattern. Let S_w denote the within-class scatter matrix, and S_b the between-class scatter matrix. The linear discriminant analysis finds a subspace, which maximizes the value of $\mathrm{trace}(S_w^{-1}S_b)$. Given c classes, LDA extracts $c-1$ features. We applied both of the previous methods to extract the features from the original 600-dimensional data. We visualized the feature extraction results in two-dimensional (2-D) and three-dimensional (3-D) projections of the original data (Fig. 4). Using PCA, we were able to effectively reduce the feature dimensionality from 600 to 100, while maintaining greater than 95% of the variance of the original data

$$\left(\frac{\sum_{i=1}^{m} \lambda_i}{\sum_{i=1}^{d} \lambda_i} > 0.95 \right).$$

However, the resulting features did not help in improving the classification accuracy. In fact, the performance of the classifier trained on the reduced (100-dimensional) feature vector, extracted using PCA, dropped significantly (as compared to the classifier trained on the original 600-dimensional feature vector) both on the training and the independent test data (see Fig. 6). Since PCA is an unsupervised feature extraction method, the m uncorrelated features (extracted using PCA), which are used to represent the original d-dimensional patterns, are not necessarily the best for pattern classification. The classifier trained on features extracted using LDA yielded higher accuracies (see Fig. 6). The Chernoff faces are also used to represent the mean vectors (in the 3-D LDA space) of the four categories [Fig. 4(d)]. We can clearly see that the images belonging to the four different orientations are well separated in a 3-D feature space obtained using LDA.

Another method to define the features is to derive them automatically from the input image, using the HDR tree method [19]. Instead of relying on a human expert to define the features, this method automatically and implicitly derives the features from training images, while building the classifier. However, due to the complexity of extracting features (both in terms of time complexity and robustness of extracted features) automatically from a small set of training images, we have employed the HDR tree method to derive features from the original feature vectors (color moment features), rather than directly from the image.

IV. Classifier Comparison

While there are a large number of classification schemes reported in the literature, we compare the proposed LVQ-based classifier with the following four well-known classifiers: the k-nearest neighbor rule (k-NN), the support vector machine (SVM), a mixture of Gaussians, and the hierarchical discriminating regression (HDR) tree. The choice of these four classifiers is based on the following observations:

1) k-NN is a very well-known and one of the simplest (to implement) nonparametric classification scheme;
2) SVMs are one of the most comprehensive nonparametric classification scheme, attempting to apply both feature

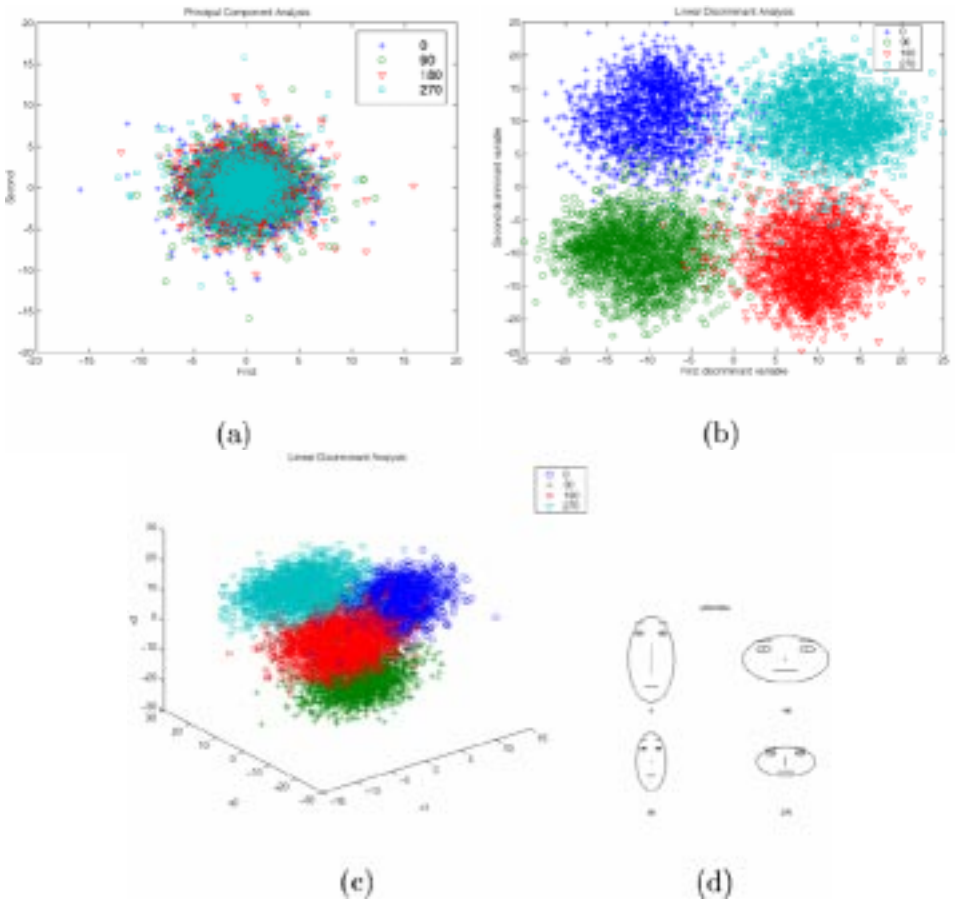

Fig. 4. Two-dimensional representation of the data using (a) PCA and (b) LDA. The original feature space is projected into 2-D space, along the largest two eigenvectors, and the first two discriminant variables, respectively. (c) Three-dimensional representation of the LDA space. (d) Chernoff faces corresponding to the mean vectors of the four categories in the 3-D LDA space.

normalization and Occam's Razor principle in classifier design;

3) HDR tree is a powerful technique devised for high-dimensional data, which also applies implicit feature extraction during classifier design;
4) both SVM and HDR tree have been shown to specifically work for high-dimensional feature space and hence, our choice to compare classification performance with these techniques;
5) mixture of Gaussian classifier is suitable in situations where the class-conditional densities are not unimodal (especially true for high-dimensional feature vectors).

A. k-Nearest Neighbor Rule

The k-NN rule [9] assigns a test pattern to the majority class among its k nearest neighbors using a performance optimized value for k. There is no separate training procedure for k-NN rule. It is a robust classifier, which gives a good classification accuracy in practice. The asymptotic error rate of k-NN rule is bounded by twice the Bayes error rate. The drawback of k-NN rule is its large computational requirement. Also, when the data are not properly scaled, the k-NN rule employing the Euclidean distance does not perform well. So, data normalization is inevitable in most cases.

B. Support Vector Machine

Support vector machine [3], which was introduced by Vapnik [39], [40], utilizes the structural risk minimization principle. It is primarily a dichotomy classifier. The optimization criterion is the width of the margin between the classes, i.e., the empty area around the decision boundary defined by the distance to the nearest training samples. These patterns, called the *support vectors*, finally define the classification function. The SVMs use optimization methods to maximize the gap between the classes. An SVM with a large margin separating two classes has a small VC dimension, which yields a good generalization performance. The computational complexity of the training procedure (a quadratic minimization problem) is one of the drawbacks of SVM. A number of classifiers were trained using different kernels (linear, polynomial, radial basis function, and sigmoid) for SVM. The best classification accuracy was achieved when a polynomial kernel function of degree 3 was used. Therefore, we report results for this SVM classifier only.

C. Hierarchical Discriminating Regression Tree

The HDR algorithm [19] casts classification and regression problems into a unified regression framework. This unified view enables classification problems to use numeric information in the output space–distance metric among clustered class labels for coarse and fine classifications. Clustering is

performed in both output space and input space at each internal node of the regression tree. Clustering in the output space provides virtual labels for computing clusters in the input space. Features in the input space are automatically derived from the clusters in the output space. These discriminating features span the subspace at each internal node of the tree. A hierarchical probability distribution model is applied to the resulting discriminating subspace at each internal node. To relax the per-class training sample requirement of traditional discriminant analysis techniques, a sample-size dependent negative-log-likelihood (NLL) is employed.

D. Mixture of Gaussian

A mixture of Gaussians can be used to model a data set comprising of several distinct populations. A Gaussian mixture model, $G(\mathbf{y})$, with k components can be written as

$$G(\mathbf{y}) = \sum_{i=1}^{k} w_i * p(\mathbf{y}|\boldsymbol{\theta}_\mathbf{i}) \tag{8}$$

where w_is are the mixing probabilities and $\boldsymbol{\theta}_\mathbf{i}$ is the set of parameters (the mean and the covariance matrix) defining the ith component of the Gaussian mixture. The expectation maximization (EM) algorithm [8], [12] is a commonly employed parametric technique for estimating parameters of a mixture of Gaussians, here w_i and $\boldsymbol{\theta}_\mathbf{i}$. We have employed the method described in [10], wherein the number of mixture components is automatically estimated along with the mixture parameters by incorporating a MDL technique within the EM steps. Although the EM method is unsupervised, we employed it in a supervised fashion, learning a mixture for each class separately. The final classifier is designed using the estimated mixture parameters and then classifying based on the MAP classifier. The main limitations of this approach are that it doesn't scale well to large dimensional feature vectors and does not use the complete training data (each class is separately treated independent of the others) while estimating the mixture parameters.

E. Experimental Results

We implemented the k-NN algorithm in the following four ways: voting, distance weighted [1], voting with normalized data and distance weighted with normalized data. Hold out method was used to compute the classification accuracy with 1995 samples per class as the training set and 2091 samples per class as the test set. We notice that the data normalization improves the classification performance. We also studied the performance of k-NN after feature extracting using PCA and LDA. These classification results are plotted in Fig. 5, which shows that k-NN with LDA has the best performance.

The LVQ-based classifier was applied to the original normalized features, features extracted with PCA, and features extracted with LDA. The accuracies, as a function of the codebook size for features extracted using PCA and LDA, are shown in Fig. 6. LVQ with LDA achieves much better performance in these experiments. The MMDL principle extracted a total of ten codebook vectors for the four-class problem for the LVQ with LDA classifier. Due to space constraints, we do not show the graph of the MMDL criterion versus codebook size in this case. While the classification accuracy seems to be maximized for 100 codebook vectors (although, only insignificantly), the MMDL criterion (a function of accuracy and code length) is minimized for ten codebook vectors. In order to maintain a consistent approach, we still use a diagonal covariance for each Gaussian kernel placed on a codebook vector. However, this assumption is not necessary, In fact, EM algorithm can be applied to estimate the parameters of the Gaussian kernel representing each codebook vector (from the training samples falling under this codebook vector). We remove this assumption (and use a full covariance matrix) when we employ the EM algorithm with LDA-derived features.

We also applied SVM, HDR tree, and a mixture of Gaussian (assumed a diagonal covariance matrix for each component) methods to the original training data. The classification accuracy of SVM on the training set is 100% and on the test set is 94.95%. At the same time, HDR tree obtained 100% accuracy on the training data set, and 93.8% on the test data set. Both these methods do implicit data normalization. The mixture of Gaussian classifier yielded much lower results with accuracy of

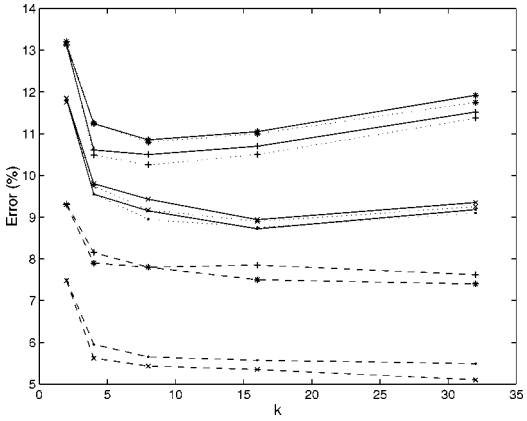

Fig. 5. Error rates of k-NN classifier w.r.t. k; scheme for classification are represented as follows: voting ($*$), distance weighted ($+$), voting with normalized data (x), and distance weighted with normalized data ($.$); results are shown before feature extraction (solid lines), feature extraction using PCA (dotted lines), and feature extraction using LDA (dashed lines).

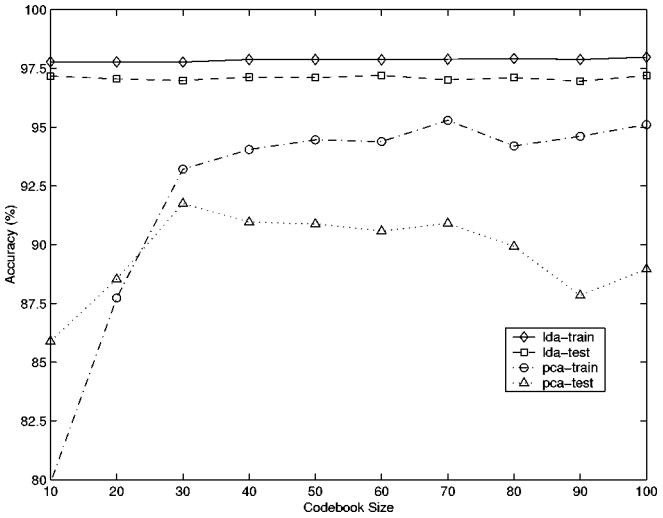

Fig. 6. Accuracy of LVQ-based classifier on features extracted using PCA and LDA.

TABLE I
PERFORMANCE COMPARISON OF FIVE CLASSIFIERS

Method	Accuracy		Time (sec)	
	Training	Testing	Training	Testing (per image)
LVQ	98.6%	96.8%	410.00	0.005
LVQ with LDA	97.8%	97.2%	70.00	0.000025
Mixture of Gaussian	79.3%	88.4%	3,600.00	0.001
Mixture of Gaussian with LDA	97.7%	97.3%	50.00	0.0005
k-NN	100%	91.25%	0.0	0.18
k-NN with LDA	100%	94.75%	0.0	0.01
SVM	100%	94.95%	7,020.00	0.06
SVM with LDA	97.54%	96.60%	1,850.00	0.02
HDR	100%	93.80%	3,901.00	0.10

79.3% on the training data and 88.4% on the test data. We also applied SVM and mixture of Gaussian classifiers to the reduced feature space generated by LDA. In the reduced feature space, the mixture component entries can be assumed to have a full covariance matrix. Feature extraction not only leads to a speed up in classification, but also improves the classification accuracy. The accuracy of the mixture of Gaussian classifier improved to 97.7% on the training set and 97.3% on the test set. Table I shows the performance comparison of the five classifiers on a SUN UltraSparc 10 machine with 256 MB RAM. We would like to emphasize that the performance of all the five classifiers is impressive. The best results (in terms of accuracy and speed) were obtained by the LVQ-based classifier and the mixture of Gaussian classifier using LDA for feature extraction. Coincidently, both the LVQ-based and mixture of Gaussian classifiers estimated the same number of components (ten codebook vectors in total). The use of a full covariance matrix in the mixture of Gaussian classifier increases its computation time during classification. Note that while the LVQ-based classifier maintains good accuracy at higher dimensionality, the mixture of Gaussian classifier does not scale appropriately. We feel this is mostly due to the lack of sufficient number of training samples in the higher dimension, affecting the parameter estimation provided by the EM algorithm. However, a major limitation of both the LVQ-based and mixture of Gaussian classifiers is that they do not attempt any feature selection or normalization during classifier design. The onus is left on the human designer to come up with a "good" set of features. SVM and HDR tree attempt feature normalization and extraction during classifier design.

We next present detailed results of the LVQ-based classifier using LDA for feature extraction. Fig. 7 shows a subset of the images whose orientations were correctly detected. We found the classifier performance to be near perfect for long-distance outdoor images and images with sky present in them. The classifier performance drops on difficult images, such as images with uniform texture and background, close-up images, symmetric images, and images of low contrast. Fig. 8 shows a subset of the misclassified images. Since the classifier performance is near perfect for outdoor images and tends to decrease for images with a constant background or symmetric images, it seems that the classifier bases classification on gradation of illumination present in an input image.

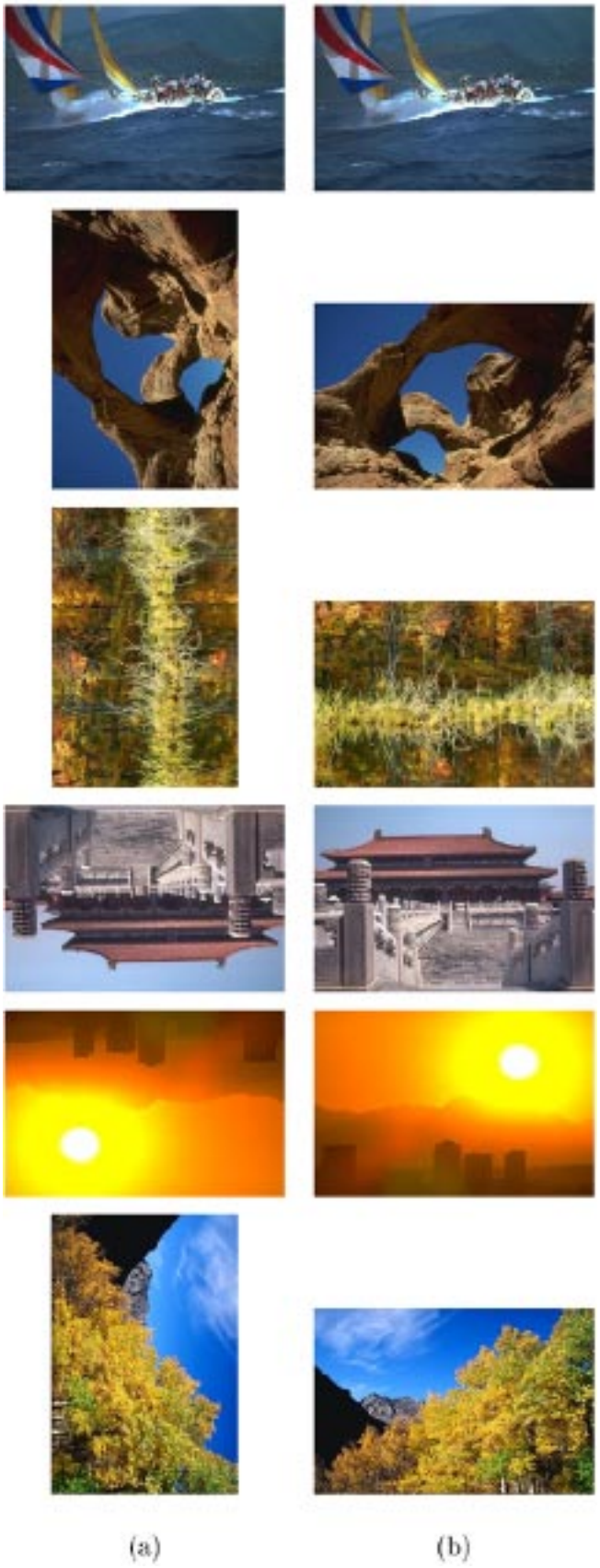

Fig. 7. Subset of images in the database whose orientations were correctly detected. (a) Input images and (b) detected orientations.

Fig. 8. Subset of images in the database that were misclassified. (a) Input images, (b) detected orientations, and (c) true orientations.

V. Classifier Combination

A large number of experimental studies have shown that classifier combination can exploit the discrimination ability of individual feature sets and classifiers [22], [24]. Bagging [2], [34] and boosting [13], [18] are two commonly used methods of combining classifiers based on statistical re-sampling techniques. Bagging uses bootstrap techniques (randomly draw n patterns with replacement from the original training data of size n) to generate a number of "new" training sets. Different classifiers are then designed using these training sets. The final classification is based on the combined output (linear combination) of the various classifiers. Boosting is another statistical re-sampling technique where individual classifiers are trained hierarchically to learn more subtle regions of a classification problem. Each classifier in the hierarchy is trained on a training set that overemphasizes (assigns more weights to) the patterns that were misclassified in the earlier stages.

A number of studies [2], [13], [18], [29], [34] have shown that bagging and boosting can improve the classification accuracy of "weak" classifiers (classifiers with near chance accuracies). However, if the classifier performance is good, bagging and boosting do not guarantee any improvement (the bagged or boosted classifier may actually perform worse than the original classifier). Specifically, Mao [29] showed that boosting can in fact reduce the classification accuracy for robust and efficient classifiers under a reject option. Mao argued that for robust classifiers, the misclassifications are mostly due to the lack of sufficient discrimination ability of the underlying features used for classification. Under these circumstances, boosting does not improve classification accuracies and using additional features with higher discrimination ability for these misclassified patterns is a more practical approach. Since our individual classifier performances are very good, we present results of combining classifiers using the bagging ensemble only. For bagging, we first used each of the bootstrapped data sets, which were created by randomly drawing 798 samples (10% of the training set) with replacement from the original training set, to train the component classifiers. The final decision/classification was made on the output of each component classifier by voting. Table II shows the classification accuracy using bagging. As expected (since our individual classifier performances are very good), bagging only slightly improves classifier performance. We also combined the four classifiers (k-NN, LVQ-based, HDR, and SVM) together. The final decision was again based on the voting of the output of the four classifiers. The classification accuracy using the combined classifiers on the training dataset was 99.4%, and the accuracy on the test dataset was 97.3%. This result is only a slight improvement in the overall classification accuracy over individual classifiers.

TABLE II
Performance of Bagging

Component Classifier	Accuracy		Time (sec)	
	Training	Testing	Training	Testing (per image)
LVQ	99.2%	97.4%	460.00	0.11
LVQ with LDA	97.8%	97.4%	95.00	0.005
SVM	99.80%	95.08%	7,850.00	0.61
SVM with LDA	97.58%	96.65%	2,150.00	0.18

VI. Conclusions

Automatic image orientation detection is an important problem, which is extremely difficult in practice, since no contextual information is available. Using a LVQ-based classifier, we have presented a *novel* solution to the previous problem. The proposed approach gives high classification accuracy ($\sim$97.4%) and is extremely fast (40 images/ms given the features have been extracted). We have shown that PCA and LDA can be applied to extract a relatively small number of features from the high-dimensional data. While PCA reduces feature dimensionality, its unsupervised nature does not improve classification accuracy, and can actually lower classifier performance. However, the LDA technique greatly reduced the dimensionality of the data (from 600 to three) while maintaining excellent classification accuracy. We have compared our proposed LVQ-based classifier to four commonly used classifiers, namely, HDR tree, SVM, mixture of Gaussians (parameters estimated using EM), and k-NN classifier. Only SVM classifier achieved accuracies comparable to the LVQ-based classifier. A mixture of Gaussian classifier also achieved excellent performance on features extracted using LDA. However, its performance does

not scale to a high-dimensional feature vector. A limitation of the LVQ-based classifier is that it does not perform any implicit feature selection or normalization during classifier design and the onus is placed on the designer. On the other hand, both SVM- and the HDR tree-based classifiers have a built-in feature extraction and normalization. Finally, we utilized the classifier combination techniques to improve the accuracy and robustness of the classifiers.

REFERENCES

[1] T. Bailey and A. K. Jain, "A note on distance-weighted k-nearest neighbor rules," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-8, pp. 311–313, 1978.

[2] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[3] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[4] R. S. Caprari, "Algorithm for text page up/down orientation determination," *Pattern Recognit. Lett.*, vol. 22, no. 4, pp. 311–317, 2000.

[5] B. B. Chaudhuri and U. Pal, "Skew angle detection of digitized Indian script documents," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 182–186, Feb. 1997.

[6] P. A. Chou, T. Lookabaugh, and R. M. Gray, "Entropy-constrained vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, no. 1, pp. 31–42, 1989.

[7] T. Cover and J. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

[8] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. R. Statist. Soc. B*, vol. 39, no. 1, pp. 1–38, 1977.

[9] R. O. Duda, P. E. Hart, and D. G. Stork, Eds., *Pattern Classification*, 2nd ed. New York: Wiley, 2000.

[10] M. Figueiredo and A. K. Jain, "Unsupervised selection and estimation of finite mixture models," in *Proc. 15th Int. Conf. Pattern Recognition*, Barcelona, Spain, Sept. 2000, pp. 87–90.

[11] M. Figueiredo and J. Leitão, "Unsupervised image restoration and edge location using compound Gauss–Markov random fields and the MDL principle," *IEEE Trans. Image Processing*, vol. 6, pp. 1089–1102, Aug. 1997.

[12] M. Figueiredo, J. Leitão, and A. K. Jain, "On fitting mixture models," in *Energy Minimization Methods in Computer Vision and Pattern Recognition*, E. Hancock and M. Pellilo, Eds. Berlin, Germany: Springer-Verlag, 1999, pp. 54–69.

[13] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[14] A. Gersho and R. Gray, *Vector Quantization and Signal Compression*. Norwell, MA: Kluwer, 1992.

[15] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Reading, MA: Addison-Wesley, 1992.

[16] R. M. Gray, "Vector quantization," *IEEE ASSP Mag.*, vol. 1, pp. 4–29, Apr. 1984.

[17] R. M. Gray and R. A. Olshen, "Vector Quantization and Density Estimation," in *SEQUENCES97*, 1997, [Online]. Available: http://www-isl.stanford.edu/~gray/compression.html.

[18] J. Huang and R. Kumar, "Boosting algorithms for image classification," in *Proc. ACCV'2000*, Taipei, Taiwan, R.O.C., Jan. 2000.

[19] W.-S. Hwang and J. Weng, "Hierarchical discriminant regression," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, pp. 1277–1293, Nov. 2000.

[20] A. K. Jain and B. Chandrasekaran, "Dimensionality and sample size considerations in pattern recognition practice," in *Handbook of Statistics*. Amsterdam, The Netherlands: North-Holland, 1982, pp. 835–855.

[21] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice-Hall, 1988.

[22] A. K. Jain, R. Duin, and J. Mao, "Statistical pattern recognition: A review," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, pp. 4–37, Jan. 2000.

[23] E. Kavallieratou, N. Fakotakis, and G. Kokkinakis, "Skew angle estimation in document processing using Cohen's class distributions," *Pattern Recognit. Lett.*, vol. 20, no. 11–13, pp. 1305–1311, 1999.

[24] J. Kittler, M. Hatef, and R. P. W. Duin, "On combining classifiers," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 226–239, Mar. 1998.

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

[26] ——, "Improved versions of learning vector quantization," in *Proc. Int. Joint Conf. Neural Networks*, San Diego, CA, June 1990, pp. 545–550.

[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. Int. Joint Conf. Neural Networks*, Baltimore, MD, June 1992, pp. I725–I730.

[28] D. Le, G. Thoma, and H. Wechsler, "Automated page orientation and skew angle detection for binary document images," *Pattern Recognit.*, vol. 27, no. 10, pp. 1325–1338, 1994.

[29] J. Mao, "A case study on bagging, boosting, and basic ensembles of neural networks for OCR," in *Proc. IEEE Int. Joint Conf. Neural Networks*, Anchorage, AK, May 1998.

[30] J. Mao and A. K. Jain, "Texture classification and segmentation using multiresolution simultaneous autoregressive models," *Pattern Recognit.*, vol. 25, no. 2, pp. 173–188, 1992.

[31] K. O. Perlmutter, S. M. Perlmutter, R. M. Gray, R. A. Olshen, and K. L. Oehler, "Bayes risk weighted vector quantization with posterior estimation for image compression and classification," *IEEE Trans. Image Processing*, vol. 5, pp. 347–360, Feb. 1996.

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

[33] L. Shapiro and G. Stockman, *Computer Vision*. Englewood Cliffs, NJ: Prentice-Hall, 2000.

[34] M. Skurichina and R. P. W. Duin, "Bagging for linear classifiers," *Pattern Recognit.*, vol. 31, no. 7, pp. 909–930, 1998.

[35] A. Vailaya, M. Figueiredo, A. Jain, and H.-J. Zhang, "Content-based hierarchical classification of vacation images," in *Proc. IEEE Multimedia Systems'99*, vol. 1, Florence, Italy, June 7–11, 1999, pp. 518–523.

[36] A. Vailaya, M. Figueiredo, A. K. Jain, and H.-J. Zhang, "Image classification for content-based indexing," *Trans. Image Processing*, vol. 10, pp. 117–130, Jan. 2001.

[37] A. Vailaya, A. K. Jain, and H. J. Zhang, "On image classification: City images vs. landscapes," *Pattern Recognit.*, vol. 31, no. 12, pp. 1921–1936, 1998.

[38] A. Vailaya, H.-J. Zhang, and A. Jain, "Automatic image orientation detection," in *Proc. IEEE ICIP'99*, Kobe, Japan, Oct. 25–28, 1999.

[39] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[40] ——, *Statistical Learning Theory*. New York: Wiley, 1998.

[41] N. Vasconcelos and A. Lippman, "Library-based coding: A representation for efficient video compression and retrieval," in *Data Compression Conf. '97*, Snowbird, UT, 1997.

[42] ——, "A Bayesian framework for semantic content characterization," in *Proc. IEEE Conf. Computer Vision Pattern Recognition*, Santa Barbara, CA, June 1998, pp. 566–571.

[43] B. Yu and A. K. Jain, "A robust and fast skew detection algorithm for generic documents," *Pattern Recognit.*, vol. 29, no. 10, pp. 1599–1630, 1996.

Aditya Vailaya (M'01) received the B.Tech. degree from the Indian Institute of Technology, Delhi, in 1994 and the M.S. and Ph.D. degrees from Michigan State University, East Lansing, in 1996 and 2000, respectively.

He joined Agilent Laboratories, Palo Alto, CA, in May 2000, where he is currently applying pattern recognition techniques for decision support in bioscience research. His research interests include bioinformatics, pattern recognition and classification, machine learning, image and video databases, and image understanding.

Dr. Vailaya received the Best Student Paper Award at IEEE International Conference on Image Processing, 1999.

HongJiang Zhang (S'90–M'91–SM'97) received the Ph.D. degree from the Technical University of Denmark and the B.S. degree from Zhengzhou University, China, both in electrical engineering, in 1982 and 1991, respectively.

From 1992 to 1995, he was with the Institute of Systems Science, National University of Singapore, where he led several projects in video and image content analysis and retrieval and computer vision. He also worked at the Massachusetts Institute of Technology (MIT) Media Lab, Cambridge, in 1994 as a Visiting Researcher. From 1995 to 1999, he was a Research Manager at Hewlett–Packard Labs, where he was responsible for research and technology transfers in the areas of multimedia management, intelligent image processing and Internet media. In 1999, he joined Microsoft Research Asia, where he is currently a Senior Researcher and Assistant Managing Director, mainly in charge of media computing and information processing research. He has authored three books, over 150 refereed papers and book chapters, seven special issues of international journals in multimedia processing, content-based media retrieval, and Internet media, as well as numerous patents or pending applications. He currently serves on the editorial boards of five professional journals and a dozen committees of international conferences.

Dr. Zhang is a member of ACM.

Changjiang Yang received the B.E. degree in automation from the University of Science and Technology, China, in 1996 and the M.S. degree from the Institute of Automation, Chinese Academy of Sciences, in 1999. Currently he is pursuing the Ph.D. degree at the Computer Vision Laboratory, University of Maryland, College Park.

His research interests include computer vision, image processing, and pattern recognition.

Feng-I Liu received the B.S. degree in information and computer education from the National Taiwan Normal University, Taiwan, R.O.C., in 1998 and the the M.S. degree in computer science from Michigan State University, East Lansing, in 2001.

Her research interest includes medical images analysis and pattern recognition.

Anil K. Jain (S'70–M'72–SM'86–F'91) is a University Distinguished Professor with the Department of Computer Science and Engineering at Michigan State University, East Lansing. He served as the Department Chair from 1995 to 1999. His research interests include statistical pattern recognition, computer vision, and biometric authentication. He is the co-author of *Algorithms for Clustering Data* (Englewood Cliffs, NJ: Prentice-Hall, 1988), edited the book *Real-Time Object Measurement and Classification* (Berlin, Germany: Springer-Verlag, 1988), and co-edited the books, *Analysis and Interpretation of Range Images*, (Berlin, Germany: Springer-Verlag, 1989), *Markov Random Fields*, (New York: Academic, 1992), *Artificial Neural Networks and Pattern Recognition* (Amsterdam, The Netherlands: Elsevier, 1993), *3D Object Recognition* (Amsterdam, The Netherlands: Elsevier, 1993), and *BIOMETRICS: Personal Identification in Networked Society* (Norwell, MA: Kluwer, 1999).

He received the Best Paper Awards in 1987 and 1991 and certificates for outstanding contributions in 1976, 1979, 1992, and 1997 from the IEEE Pattern Recognition Society. He also received the 1996 IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award. He was the Editor-in-Chief of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (1990–1994). He is a Fellow of the IAPR. He received a Fulbright Research Award in 1998 and was named a Fellow of the John Simon Guggenheim Memorial Foundation in 2001.