EURASIP Journal on Applied Signal Processing 2003:2, 151–159

Automatic Hierarchical Color Image Classification

Jing Huang
Department of Computer Science, Cornell University, Ithaca, NY 14853, USA
Email: huang@cs.cornell.edu

S. Ravi Kumar
Department of Computer Science, Cornell University, Ithaca, NY 14853, USA
Email: ravi@cs.cornell.edu

Ramin Zabih
Department of Computer Science, Cornell University, Ithaca, NY 14853, USA
Email: rdz@cs.cornell.edu

Received 20 March 2002 and in revised form 6 November 2002

Organizing images into semantic categories can be extremely useful for content-based image retrieval and image annotation. Grouping images into semantic classes is a difficult problem, however. Image classification attempts to solve this hard problem by using low-level image features. In this paper, we propose a method for hierarchical classification of images via supervised learning. This scheme relies on using a good low-level feature and subsequently performing feature-space reconfiguration using singular value decomposition to reduce noise and dimensionality. We use the training data to obtain a hierarchical classification tree that can be used to categorize new images. Our experimental results suggest that this scheme not only performs better than standard nearest-neighbor techniques, but also has both storage and computational advantages.

Keywords and phrases: image classification, color correlogram, classification tree.

1. INTRODUCTION

The proliferation of the worldwide web has given easy access to an explosively growing volume of visual data. Unfortunately, this data on the web is both scattered and unstructured, making search and retrieval of information difficult. Such requirements have created great demands for effective and flexible systems to manage digital images/videos (e.g., [1, 2, 3, 4, 5, 6]). Large digital libraries, which are built by collecting resources from different sites [5, 7, 8], can make searching relatively easier.

Most of the above systems generate low-level image features such as color, texture, shape, motion, and so forth, for image indexing and retrieval. This is partly because low-level features (e.g., color histograms, texture patterns) can be computed automatically and efficiently. However, the semantics of images, with which users prefer most of their interaction, are seldom captured by low-level features. Currently, there is no effective method to automatically generate good semantic features of an image. One common compromise is to obtain some semantic information through manual annotations. Since visual data contains rich information, the manual annotation process may be subjective and inconsistent. In addition, it is difficult to capture the content of an image using words, not to mention the tedious manual labor involved in such a process. Another recent innovative approach, taken by the IMKA system [6], utilizes a medianet framework which combines the low-level features and semantic concepts in the same network and supports perceptual and semantic relationships among concepts, as the wordnet does.

Image classification

Image classification attempts to classify images into semantic categories by using low-level image features, and therefore, bridges the gap between high-level semantics and low-level features. The categorization of images into classes can be helpful both for semantic organizations of digital libraries and for obtaining automatic annotations of images.

The classification of natural imagery is quite hard in general since real images from the same semantic class may have large variations (see Figure 1) and images from different semantic classes might share a common background (such as images from "clouds" and "aviation," and images from "waves" and "dolphins and whales" in Figure 1). These issues limit the applicability of object-based and knowledge-based approaches.

Figure 1: Sample images from various classes.

A common approach to image classification involves addressing the following three issues: (i) how to represent an image, (ii) how to organize the data, and (iii) how to classify an image. Acquiring "nice" features and carefully modeling, the feature data are vital steps in this approach. Common features include color, texture, and shape information of an image. Some also integrate visual information and text accompanying an image [3, 5, 9].

As noted before, image classification can lead to a semantic organization of a digital database. Though this type of organization can be done in several ways, a hierarchical approach has multifold advantages, (i) easy browsing and

navigation through the database, (ii) efficient retrieval, and (iii) ergonomically friendly presentation of the database. For instance, webseek, a web image search engine [7], uses hierarchical semantic structure for collecting and searching images from the web. The image categories and hierarchies are preset by human design. Such an approach were also taken by [10, 11] with very limited categories.

Our approach

In this paper, we propose a new scheme for automatic hierarchical image classification. We assume that a training set of images with known class labels is available. We use an easy-to-compute low-level feature, banded color correlograms, which has been shown to be effective and efficient for content-based image retrieval [12]. Using banded color correlograms for the training images, we model the feature data using *singular value decomposition* (SVD) [13] and constructing a *classification tree*. Once the classification tree is obtained, any new image can be easily classified. Our recursive method for constructing the classification tree is summarized below.

At each level of the classification tree, we aim to choose the best modeling of the training data. We first eliminate the *noise* (or irrelevant variations) from the feature vectors using SVD (or two-mode factor analysis). This step not only reduces the dimensionality of the feature vectors but also rearranges the feature space to reflect the major correlation patterns in the data and ignores the smaller, less important variations.[1] Using this noise-tolerant SVD representation, we next classify each image in the training data using the nearest-neighbor algorithm with the first neighbor (which is the image itself) dropped (this is similar to leave-one-out cross-validation scheme). Based on the performance of this classification, we then partition the set of classes into two subclasses such that the intra-subclass association is maximized while simultaneously the inter-subclass disassociation is minimized. This is accomplished using *normalized cuts* [15]. Finally, the subclasses and those training images that were correctly classified with respect to the subclasses are worked upon recursively to obtain a hierarchical classification tree, with the hope of improving the classification performance.

Notice that a different SVD representation is used at each level of the tree. This flexibility in our method gives us the freedom to choose the size of the SVD representation as demanded by each level, which in turn is dictated by the characteristics of classes involved.

We test our method on 11 fairly representative classes of Corel images. These 11 image classes are aviation photography, British motor car collection, Canadian Rockies, cats and kittens, clouds, dolphins and whales, flowers, night scenes, spectacular waterfalls, sunsets around the world, and waves. These images contain a wide range of content (scenery, animals, objects, etc.) and colors.

[1] SVD has been successfully used in latent semantic indexing for document retrieval [14].

We test our scheme using banded color correlograms and color histograms as features and compare our method to the nearest-neighbors algorithm directly applied to both color features. Our results suggest that this hierarchical scheme is able to perform consistently better than the already effective nearest-neighbor algorithm (see [16]). The classification tree we obtain also conforms with the semantic content of the 11 classes. Our results also suggest that correlograms have more *latent semantic* structures (than histograms) that can be extracted by SVD procedure.

Organization

The rest of the paper is organized as follows. Section 2 briefly describes the previous work in automatic image classification. Section 3 contains a brief description of the banded color correlogram we use in our experiments; Section 4 outlines how to use SVD to model feature vectors; and Section 5 describes our hierarchical classification method. Section 6 contains our experimental results and Section 7 concludes our discussions.

2. RELATED WORK

Since classification itself is a long-studied research area, different classifiers can be tried on image classification (e.g., k-nearest neighbor, decision trees, Bayesian nets, maximum likelihood (ML) analysis, maximum a posterior (MAP) analysis, linear discriminant analysis, neural networks, etc.). Not much work has been done on how to organize or select features. In the following, we review some previous work in image classification.

Vailaya et al. [11] use block image features and binary MAP classifier. An Image is first divided into blocks, and features are extracted from individual blocks. A few codebook vectors are used to estimate the class-dependent Gaussian mixture densities of the observed features. The image classes are organized by the following predefined hierarchical categories: the first level is indoor/outdoor; the second level for outdoor images is city/landscape; the third level for landscape images is sunset/mountain-forest; and the last one is to classify mountain/forest. These hierarchical five classes are fairly distinguishable from one another in terms of color and texture compared to the eleven classes in Figure 1, which are used for our test data. The binary classification at each level is over 90%. If the error propagation is included, the average classification accuracy of the five classes is degraded to 84%.

The *configural recognition* scheme proposed by Lipson et al. [17] is a knowledge-based scene classification method. A model template, which encodes the common global scene configuration structure using qualitative measurements, is hand crafted for each category. An image is then classified to the category whose model template best matches the image by deformable template matching. The average percentage of correct classification on four classes of scenery (snowy mountains, snowy mountains with lakes, fields, and waterfalls) is about 64%. Torralba and Oliva [18] also use templates, which are trained from linear discriminant filters

that take account of spatial information. Degrees of naturalness, verticalness, and openness are used to classify city centers, skyscape, mountain, and beach scenes.

Carson et al. [16] propose a new representation for images. Each image is thought to consist of several *blobs*; each blob is coherent in color and texture space.[2] All the blobs in the training data of 14 image categories are clustered into about 180 "canonical" blobs using Gaussian models with diagonal covariance. Each image is then assigned a score vector which measures the nearest distance from each canonical blob to the image. These score vectors are used to train a decision-tree classifier. The results of this method are compared to color histograms with the decision-tree classifier. Interestingly, the color histograms seem to perform better than blobs.[3]

All the above works have only focused on visual features. Gevers et al. [10] try to integrate visual and textual features for web image classification. The textual information extracted from HTML tags is not always helpful for classification. For example, experiments of classifying images into portraits/nonportraits show that the textual information does not help much. This is due to the inconsistent textual descriptions. In the case of classifying photographic/synthetic images, the visual and textual features contribute equally to the classification. Hence, the composite features achieve better accuracy for this task. Paek et al. [9] also integrate visual and textual features for photograph classification. Text is extracted from accompanying text of images contained in news articles. The standard TF*IDF vectors are generated from text information, and a parallel OF*IIF vectors are produced from visual information. The OF*IIF vectors are supposed to be based on objects (which are parallel to words) in images although the real implementation in [9] used cluster centroids of 8×8 image blocks. The integrated vectors improves over the individual ones by about 3% in performance of an indoor/outdoor classification.

3. BANDED COLOR CORRELOGRAMS

In this section, we briefly review the banded color correlograms that we use in our experiments.

If we treat the color histogram as a probability distribution of colors in an image, we ask the following question: pick any pixel p_1 of the image $\mathcal{I}$ at distance k away from p_1, pick another pixel p_2, what is the probability that p_2 has the same color as p_1? The answer gives us the conditional probability distribution that depicts the spatial correlation between the same color pixels. The color correlogram describes how this spatial correlation of colors changes with distances. We give the formal definitions below.

Let I be an $n_1 \times n_2$ image. The colors in I are quantized into m colors $c_1, \ldots, c_m$. For a pixel $p = (x, y) \in I$, let $I(p)$ denote its color. Let $I_c \triangleq \{p \mid I(p) = c\}$. Thus, the notation $p \in I_c$ is synonymous with $p \in I, I(p) = c$. For convenience, we use the L_∞-norm to measure the distance between pixels, that is, for pixels $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$, we define $|p_1 - p_2| \triangleq \max\{|x_1 - x_2|, |y_1 - y_2|\}$. We denote the set $\{1, 2, \ldots, n\}$ by $[n]$. The size of I is denoted by $|I| = n_1 n_2$.

Histogram

The *color histogram* (henceforth histogram) h of I is defined for $i \in [m]$ by

$$h_{c_i}(I) \triangleq \Pr_{p \in I}\left[p \in I_{c_i}\right]. \tag{1}$$

Thus, $h_{c_i}(I)$ gives for any pixel in I, the probability that the color of the pixel is c_i. Given the count $H_{c_i}(I) \triangleq |\{p \mid p \in I_{c_i}\}|$, it follows that $h_{c_i}(I) = H_{c_i}(I)/(n_1 n_2)$.

Autocorrelogram

Let a *distance set* D be fixed a priori (e.g., $D \subseteq [\min\{n_1, n_2\}]$). Let $d = |D|$. Then, the *autocorrelogram* of I is defined, for $i \in [m]$ and $k \in D$, as

$$\begin{aligned}\alpha_{c_i}^{(k)}(I) &\triangleq \Pr_{p_1 \in I_{c_i}, p_2 \in I}\left[p_2 \in I_{c_i} \mid |p_1 - p_2| = k\right] \\ &= \frac{\left|\{p_1, p_2 \in I_{c_i} \mid |p_1 - p_2| = k\}\right|}{H_{c_i}(I) \cdot 8k}.\end{aligned} \tag{2}$$

Given any pixel p of color c_i in the image, $\alpha_{c_i}^{(k)}$ gives the probability that a pixel at a distance k from the given pixel has the same color of p. (The factor $8k$ is due to the properties of L_∞-norm used to compute the distance between pixels.) Note that the size of the autocorrelogram is md. Since local correlations between colors are more significant than global correlations in an image, a small value of d is sufficient to capture the spatial correlation.

We now define the *banded autocorrelogram* as

$$\beta_{c_i}(I) \triangleq \frac{1}{k}\sum_{k'=1}^{k} \alpha_{c_i}^{(k')}(I). \tag{3}$$

This measure computes the local *density* of color c_i's correlation with itself, thus suggesting one kind of a local structure of colors. Note that the size of banded autocorrelogram is m, that is, the same as that of histogram. We use $\beta(I)$ to denote the banded autocorrelogram of I, treated as vectors in an m-dimensional space.

We use the L_1 (or the city-block) distance measure for comparing histograms and banded autocorrelograms because it is simple and robust. For simplicity, we will address banded autocorrelograms merely as correlograms for the remainder of the paper.

4. SINGULAR VALUE DECOMPOSITION

In this section, we briefly review the SVD that we use for organizing image feature vectors.

Without loss of generality, let $m \geq n$. For an $m \times n$ matrix A, the *SVD* of A is given by $A = U\Sigma V^T$ (see [13]), where

[2] This is one kind of color- and texture-based image segmentation method.

[3] Several explanations were given for this performance degradation.

(i) U is an $m \times n$ matrix, and Σ, V are $n \times n$ matrices;
(ii) U and V are column orthonormal, that is, $U^T U = V^T V = I_n$;
(iii) $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0)$, where $r = \text{rank}(A)$ and the *singular values* are $\sigma_1 \geq \sigma_2 \cdots \geq \sigma_r > 0$.

The first r columns of U and V together with the nonzero singular values actually are the eigenvectors and the r nonzero eigenvalues of AA^T and $A^T A$, respectively. Several efficient algorithms exist to compute the SVD of a matrix, especially if the matrix is known to be sparse.

The SVD of a matrix can be used to obtain lower-rank approximations of the matrix. If we take the first k columns of U and V (denoted by $U_{[k]}$ and $V_{[k]}$) and the leading $k \times k$ submatrix of Σ (denoted by $\Sigma_{[k]}$), and define

$$A_k \triangleq U_{[k]} \Sigma_{[k]} V_{[k]}^T = \sum_{i=1}^{k} U_i \Sigma_{i,i} V_i^T, \tag{4}$$

then A_k is the best rank k approximation of A, that is,

$$\min_{\text{rank}(B)=k} |A - B|_2 = |A - A_k|_2 = \sigma_{k+1}. \tag{5}$$

This property of the SVD helps to obtain a good trade-off between the quality of approximation and the size of the approximation (i.e., k). (To compute A_k, we use the MATLAB built-in function SVD.)

The advantages of SVD are nicely exploited in *latent semantic indexing* (LSI) for document retrieval [14]. The SVD, in some sense, derives the underlying structure that is hidden in A. The approximation A_k can be thought of as dampening the noise and that is present in the original matrix A. When SVD is applied to feature vectors, it not only eliminates the noise in the feature vectors but also reduces the dimension of the feature when $k < m$.

We outline our approach of using SVD with correlograms. Let $\mathcal{I} = \{I_1, \ldots, I_n\}$ denote the set of training images and let m be the number of color quantizations. We define the matrix $A_{i,j}(\mathcal{I}) \triangleq \beta_{c_i}(I_j)$. We compute the SVD of $A(\mathcal{I})$ to be $A(\mathcal{I}) = UDV^T$. Let $A_k = U_{[k]} \Sigma_{[k]} V_{[k]}^T$ be an approximation to A. We can choose $U_{[k]}$ as the basis for the new k-dimensional feature space. Then, $V_{[k]}$ is the new representation for the correlograms in this reduced feature space. When we have a new image that needs to be classified, we first compute its correlogram q, then project q onto the reduced feature space by computing

$$q' = q \cdot U_{[k]} \cdot \Sigma_{[k]}^{-1}. \tag{6}$$

Now, the question is how to choose k for the approximation. We use the following heuristic to pick the k. Note that we want to find the best approximation A_k such that the SVD representation of correlograms gives the best classification results using nearest-neighbor rule. Instead, we compute the classification for each k between the number of classes (i.e., c) to an upper limit k^* and choose the best k in this range. Now, we show how to choose k^*. Notice that the singular values of A correspond to the eigenvalues of AA^T, which is the correlation matrix of local color density for the training images. We set k^* to be the k^*th biggest eigenvalue within 2% of the maximum eigenvalue, that is, we ignore those correlations whose values are less than 2% of the maximum correlation.[4]

Note that, in the above SVD method, histograms can be used instead of correlograms. We will see (Section 6) that the performance with correlograms is much better than with histograms.

5. THE HIERARCHICAL CLASSIFICATION SCHEME

Image classification is the problem of classifying images into known semantic classes. Let $\mathcal{C} = \{C_1, \ldots, C_c\}$ be the image classes known as a priori. We assume that we have a set $\mathcal{S}$ of training images whose class membership is known and we set $\mathcal{T}$ of images that need to be classified. We want to build a classification tree from training images. At each level of the classification tree, we aim to choose the best modeling of the training data. We first use SVD to eliminate the noise from the training data as described in Section 4. We then classify each image in the training data using the nearest-neighbor algorithm with the first neighbor dropped (similar to the leave-one-out cross-validation scheme). Based on the performance of this classification, we then split the classes into two subclasses such that the intra-subclass association is maximized while simultaneously the inter-subclass disassociation is minimized. This is accomplished using normalized cuts [15]. Finally, the subclasses and those training images that were correctly classified with respect to the subclasses are worked upon recursively to obtain the hierarchy in the classification tree, with the hope of improving the classification performance.

5.1. Confusion matrix

We construct the matrix $A(\mathcal{S})$ as indicated in Section 4 and compute its SVD: $A(\mathcal{S}) = U\Sigma V^T$. Then, we choose the best approximation A_k that gives the best classification of $\mathcal{S}$ on itself. The details are the following.

For an image $I \in \mathcal{S}$, and $C(I)$ is the class of I, let $\beta_k(I)$ denote the k-dimensional reduced SVD representation of I. We consider each $I \in \mathcal{S}$ as a query and obtain the class $C'(I)$, where

$$C'(I) \triangleq C\Big(\arg \min_{J \in \mathcal{S} \setminus \{I\}} \{|\beta_k(I) - \beta_k(J)|\}\Big). \tag{7}$$

In other words, $C'(I)$ is the class assigned by the nearest-neighbor classification when all images other than I itself are considered. Intuitively, this procedure helps to find the best association patterns between the classes of SVD.

Now, the $c \times c$ *confusion matrix* M is then defined by

$$M_{i,j} = \text{size of } \{I \mid C(I) = C_i,\ C'(I) = C_j\}. \tag{8}$$

[4] There is no good heuristic for choosing k. The rule of thumb is finding the k that gives the best performance [19].

The diagonal entries of M are the number of images that are correctly classified, while the off-diagonal entries are the misclassifications. The average percentage of correct classification is just the sum of the diagonal entries ($\text{trace}(M)$) divided by the size of $\mathcal{S}$.

5.2. Normalized cuts

We now show how to partition the confusion matrix M on the basis of maximizing the interclass association and minimizing the intraclass disassociation simultaneously. First, we review some basic definitions from graph theory.

Given a weighted graph $G = \langle V, E\rangle$ with $w(u, v)$, being the weight of an edge (u, v), the *mincut* is defined to be a partition of $V = V_1 \cup V_2$ such that

$$\text{cut}_w(V_1, V_2) \triangleq \sum_{(u,v)\in V_1\times V_2} w(u, v) \tag{9}$$

is minimized. Mincuts can be computed in polynomial time using network flow techniques.

The confusion matrix M defines a natural directed graph. The mincut in this graph corresponds to a partition of the classes into M_1 and M_2 such that the number of misclassifications among these classes is minimized. A partition of M according to the mincut, however, sometime favors cutting small sets [20], that is, one of V_1 or V_2 is very small. This problem is considered in [15], where normalized cuts are introduced.

Formally, the normalized cut is given by the best partition of $V = V_1 \cup V_2$ that minimizes

$$\text{ncut}_w(V_1, V_2) \triangleq \text{cut}_w(V_1, V_2)\left(\frac{1}{\text{cut}_w(V_1, V)} + \frac{1}{\text{cut}_w(V_2, V)}\right). \tag{10}$$

The partition based on normalized cut is shown to have the property that minimizes the disassociation between the groups and maximizes the association within the group.

Define the diagonal matrix $M'_{i,i} = \sum_j M_{i,j}$. Normalized cuts can be computed reasonably well and efficiently by computing the second smallest eigenvalue of the system defined by $(M - M')x = \lambda M'x$ and using some additional heuristics. The details can be looked up in [15].

We use normalized cuts to partition M into M_1 and M_2, accordingly, we obtain a partition of the classes $\mathcal{C}$ into $\mathcal{C}_1$ and $\mathcal{C}_2$.

5.3. Classification tree

Using the normalized cuts, we can build the classification tree recursively. Given the original set of classes $\mathcal{C}$, we compute the partition $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$ based on normalized cuts. We define $\mathcal{S}_1 = \{I \in \mathcal{S} \mid C'(I) \in \mathcal{C}_1,\ C(I) \in \mathcal{C}_1\}$ and $\mathcal{S}_2 = \{I \in \mathcal{S} \mid C'(I) \in \mathcal{C}_2,\ C(I) \in \mathcal{C}_2\}$. In this way, the images that are misclassified across $\mathcal{S}_1$ and $\mathcal{S}_2$ are not considered from now on. We then recursively work on classifying $\mathcal{S}_1$ (resp., $\mathcal{S}_2$) with $\mathcal{C}_1$ (resp., $\mathcal{C}_2$) as the set of classes (see [21] for detailed algorithm).

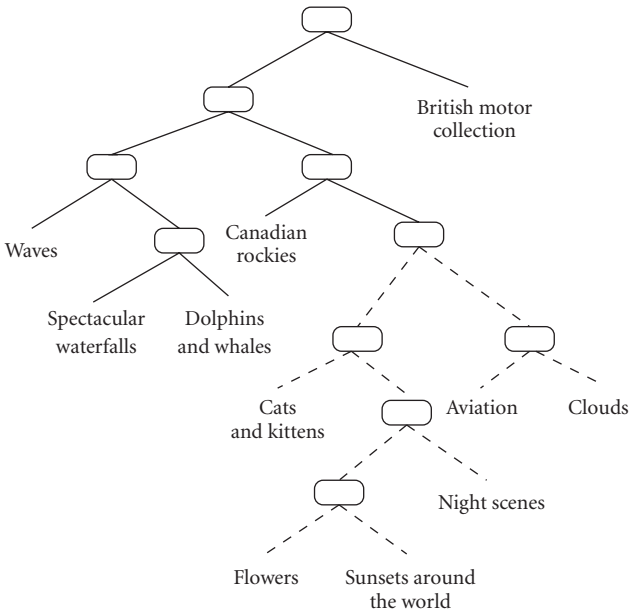

Figure 2: The classification tree obtained from the first training set using correlograms. The dotted lines indicate the trimmed portions.

5.4. Trimming

Sometimes, the performance of the classification on the training data does not always improve level by level using reduced SVD representations. This is because some variations that are important to a set of classes may be removed by the SVD reduction. In this case, it does not pay off to recursively split such classes. Notice that this scenario can be detected automatically by comparing the performance of the tree before and after trimming on the training set. More precisely, we perform a trimming procedure on the tree we obtain from the algorithm in the following manner: if the classification correctness of a node in the tree is higher than that of its two children, then we trim both the children; otherwise we keep the child with the higher correctness than the node itself and trim the other child. For instance, Figure 2 shows a classification tree (corresponding to our sample set) with the trimmed portions marked.

6. EXPERIMENTS AND RESULTS

6.1. Experiments

We choose 11 image classes from Corel collections: aviation photography, British motor car collection, Canadian Rockies, cats and kittens, clouds, dolphins and whales, flowers, night scenes, spectacular waterfalls, sunsets around the world, and waves (for some samples, see Figure 1). These images contain a wide range of content (scenery, animals, objects, etc.), colors, and lighting conditions. We delete some images in each class which are inconsistent with the rest of the class (as in [16]) and leave 90 images in each class. Since we use the nearest-neighbor rule with the classification tree, we want to make sure that the color distributions of training

images and test images are more or less the same. Therefore, we randomly shuffle the images in each class and take 70 images as the training set and the rest 20 images as the test set. By doing so three times (to ensure fairness), we obtain three sets of training data and test data.

To compute color histograms and color correlograms, we quantize the RGB color space into $8 \times 8 \times 8 = 512$ colors (3 bits for each color channel).[5] This level of quantization is good enough for the SVD to extract the underlying structure, while not being too big (unlike 6912 colors used in [16]) so as to affect efficiency.

6.2. Results

We test both color correlograms and color histograms on the hierarchical classification approach and compare the hierarchical approach with the nearest-neighbor classification. The three classification trees from three training sets are more or less the same and are consistent with the color content of the 11 classes. We only present the tree from the first data set (Figure 2). For the sake of simplicity, we abbreviate the names for the 11 classes: aviation (A1), British motors (B1), Canadian Rockies (C1), cats and kittens (C2), clouds (C3), dolphins and whales (D1), flowers (F1), night scenes (N1), spectacular waterfalls (S1), sunsets around the world (S2), and waves (W1). From the classification tree, we see that A1 and C3 share the same parent because of the same sky background; similarly, W1, S1, and D1 are grouped together because of the same water background.

The confusion matrices for different methods are shown in Table 1, Table 2, and Table 3. The classification behavior of the classification tree is quite different from the nearest-neighbor. The classification tree is better than the nearest-neighbor in that (i) the overall number of misclassifications between classes is smaller and (ii) the overall number of correct classifications is larger.

The average percentage of correctness of the three test sets is summarized in Table 4. With correlograms and the classification-tree scheme, the average accuracy of classifying 11 image classes is about 82%, comparable to the 84% accuracy in [11] for a hierarchy of 5 image classes.[6] The results show that the hierarchical method is consistently better than the nearest-neighbor classification, and the color correlogram is consistently better than the color histogram.

Using the simple nearest-neighbor (NN) classification, the correlogram performs 3% better than the histogram; using the classification tree (CT), the correlogram performs 21% better than the histogram. Using the classification tree, the correlogram improves 3% over the nearest-neighbor; it improves 7% over the nearest-neighbor on the histogram. Note that the average data size of the SVD representations is about fifteen, 3% of the original size. The average number of nonleaf nodes in the classification trees is five after trimming. Therefore, the computation and storage of the data for the classification saves about 85%, which is significant.

Table 1: Class-confusion matrix for trimmed classification tree (correlogram).

	A1	C3	C1	D1	W1	B1	C2	S1	F1	N1	S2
A1	17	0	0	3	0	0	0	0	0	0	0
C3	2	15	0	0	0	0	2	0	0	0	1
C1	0	2	16	2	0	0	0	0	0	0	0
D1	0	1	0	17	0	0	1	1	0	0	0
W1	0	1	1	4	13	0	0	1	0	0	0
B1	0	0	0	0	0	20	0	0	0	0	0
C2	0	0	0	0	0	0	20	0	0	0	0
S1	0	1	0	2	0	0	0	17	0	0	0
F1	1	0	0	0	0	1	1	0	17	0	0
N1	0	0	0	0	0	0	1	0	0	18	1
S2	0	0	0	0	0	1	0	0	2	0	17

Table 2: Class-confusion matrix for the nearest-neighbor classification (correlogram).

	A1	C3	C1	D1	W1	B1	C2	S1	F1	N1	S2
A1	16	1	0	2	0	0	1	0	0	0	0
C3	1	14	0	0	0	0	1	2	0	0	1
C1	0	0	15	5	0	0	0	0	0	0	0
D1	0	1	0	19	0	0	0	0	0	0	0
W1	0	0	0	2	17	0	0	1	0	0	0
B1	0	0	0	0	0	14	4	2	0	0	0
C2	0	0	0	0	0	0	20	0	0	0	0
S1	0	1	0	2	1	0	0	16	0	0	0
F1	1	0	0	0	0	0	2	0	14	1	2
N1	0	0	0	0	0	0	1	0	0	19	1
S2	0	2	0	0	0	0	0	0	0	2	16

Table 3: Class-confusion matrix for the nearest-neighbor classification (histogram).

	A1	C3	C1	D1	W1	B1	C2	S1	F1	N1	S2
A1	14	0	2	0	1	0	2	1	0	0	0
C3	0	13	1	2	0	0	3	1	0	0	1
C1	0	1	16	0	2	0	1	0	0	0	0
D1	0	0	0	17	2	0	1	0	0	0	0
W1	0	0	0	3	16	0	0	1	0	0	0
B1	0	0	0	0	0	17	1	1	0	1	0
C2	0	0	0	0	0	0	20	0	0	0	0
S1	0	1	0	0	2	1	0	16	0	0	0
F1	0	0	0	0	0	1	1	0	18	0	0
N1	0	0	0	0	0	0	1	0	0	13	6
S2	0	2	0	0	0	0	0	3	0	2	13

[5] We also tried the HSV color space. The results do not change much.

[6] It is not meant to compare numbers here because the data set are different.

Table 4: Correctness classification on three data sets.

	NN			CT(Trim)		
	1	2	3	1	2	3
Hist.	0.786	0.746	0.786	0.696	0.677	0.668
Corr.	0.818	0.800	0.786	0.850	0.805	0.823

Remark 6.1. We notice from the results that the color histogram performs consistently worse with the classification tree than with the nearest-neighbor, while the color correlogram performs consistently better with the classification tree than with the nearest-neighbor. This suggests that correlograms have an underlying latent semantic structure (local color density). Color histograms do not seem to have such a property.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a hierarchical image classification method based on an automatic constructed classification tree. We use banded color correlograms as visual features and the SVD on the correlograms to extract a latent semantic structure of images for classification into semantic categories. SVD not only reduces the dimensionality of features but also removes the noise in the data. At each level of the classification tree, SVD is used to best model the data in terms of lowest classification errors. The data of a nonleaf node is then divided by the normalized cuts, which maximizes the intra-subclass variation while simultaneously minimizes the inter-subclass variation, to obtain the best classification results.

Our tests on 11 classes of Corel natural scene images show that our method using this scheme and a classification tree not only performs better than the nearest-neighbor classification but also saves much computation and data storage. In addition, the results also suggest that the correlogram is more suitable for the image classification task than the color histogram. It will be interesting to use *feature-weighting* techniques [22] and textual information to further assist SVD to get latent semantic structures from training data. The integration of visual and textual features in our framework needs to be studied.

ACKNOWLEDGMENT

This work was done when the first two authors were at Cornell University.

REFERENCES

[1] M. Flickner, H. Sawhney, W. Niblack, et al., "Query by image and video content," *The QBIC System, IEEE Computer*, vol. 28, no. 9, pp. 23–32, 1995.

[2] A. P. Pentland, R. Picard, and S. Sclaroff, "Photobook: content-based manipulation of image databases," *International Journal of Computer Vision*, vol. 18, no. 3, pp. 233–254, 1996.

[3] J. R. Smith and S.-F. Chang, "Visually searching the web for content," *IEEE MultiMedia*, vol. 4, no. 3, pp. 12–20, 1997.

[4] S. Mehrotra, Y. Rui, M. Ortega, and T. S. Huang, "Supporting content-based queries over images in MARS," in *Proc. IEEE International Conference on Multimedia Computing and Systems*, pp. 632–633, Ottawa, Ontario, Canada, June 1997.

[5] T. Gevers and A. Smeulders, "The PicToSeek WWW image search systems," in *IEEE International Conference on Multimedia Computing and Systems*, vol. 1, pp. 264–269, Florence, Italy, June 1999.

[6] A. B. Benitez, S.-F. Chang, and J. R. Smith, "IMKA: a multimedia organization system combining perceptual and semantic knowledge," in *Proc. ACM Multimedia*, pp. 630–631, Ottawa, Ontario, Canada, 2001.

[7] S-F. Chang, J. R. Smith, M. Beigi, and A. Benitez, "Visual information retrieval from large distributed online repositories," *Communications of the ACM*, vol. 40, no. 12, pp. 63–67, 1997.

[8] S. Sclaroff, L. Taycher, and M. La Cascia, "ImageRover: a content-based image browser for the world wide web," in *Proc. of IEEE Workshop on Content-Based Access of Image and Video Libraries*, pp. 2–9, San Juan, PR, USA, June 1997.

[9] S. Paek, C. Sable, V. Hatzivassiloglou, et al., "Integration of visual and text-based approaches for the content labelling and classification of photographs," in *ACM SIGIR '99 Workshop on Multimedia Indexing and Retrieval*, Berkeley, Calif, USA, August 1999.

[10] T. Gevers, F. Aldershoff, and A. Smeulders, "Classification of images on internet by visual and textual information," in *Internet Imaging, SPIE*, San Jose, Calif, USA, January 2000.

[11] A. Vailaya, M. Figueiredo, A. Jain, and H. J. Zhang, "Bayesian framework for hierarchical semantic classification of vacation images," *IEEE Trans. on Image Processing*, vol. 1, no. 1, pp. 117–130, 2001.

[12] J. Huang, S. R. Kumar, M. Mitra, and W. J. Zhu, "Spatial color indexing and applications," in *Proc. of 8th International Conf. on Computer Vision*, 1998.

[13] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, Md, USA, 1989.

[14] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society of Information science*, vol. 41, no. 6, pp. 391–407, 1990.

[15] J. Shi and J. Malik, "Normalized cuts and image segmentation," in *Proc. 16th IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 731–737, San Juan, PR, USA, June 1997.

[16] C. Carson, S. Belongie, H. Greenspan, and J. Malik, "Color- and texture-based image segmentation using EM and its application to image querying and classification," submitted to *IEEE Trans. on Pattern Analysis and Machine Intelligence*.

[17] P. Lipson, E. Grimson, and P. Sinha, "Configuration based scene classification and image indexing," in *Proc. 16th IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1007–1013, San Juan, PR, USA, June 1997.

[18] A. Torralba and A. Oliva, "Semantic organization of scenes using discriminant structural templates," in *Proc. International Conf. on Computer Vision*, pp. 1253–1258, Corfu, Greece, 1999.

[19] H. Borko and M. Bernick, "Automatic document classification," *Journal of the ACM*, vol. 9, pp. 512–521, 1962.

[20] Z. Wu and R. Leahy, "An optimal graph theoretic approach to data clustering: theory and its application to image segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1101–1113, 1993.

[21] J. Huang, *Color-spatial image indexing and applications*, Ph.D. thesis, Dept. of Computer Science, Cornell University, Ithaca, NY, USA, 1998.

[22] D. Wettschereck, D. W. Aha, and T. Mohri, "A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms," *Artificial Intelligence Review*, vol. 11, no. 1-5, pp. 273–314, 1997, Special issue on lazy learning algorithms.

Jing Huang is a research staff member at IBM T. J. Watson Research Center. She received the B.S. and the M.S. degrees in applied mathematics from Tsinghua University, Beijing, China, and the Ph.D. in computer science from Cornell University. Her Ph.D. work focused on computer vision and content-based image retrieval. After joining the IBM T. J. Watson Research Center, she switched to work on automatic speech recognition. Her research interest also includes machine learning and information extraction.

S. Ravi Kumar is a research staff member at IBM Almaden Research Center. He received the B.S. degree in computer engineering from Anna University, Madias, India, and the M.S. degree from Indian Institute of Science, Bangalore, India. He finished his Ph.D. study in computer science from Cornell University in 1998. His Research includes theory of computation, especially in randomization, complexity theory, and web algorithms.

Ramin Zabih is an Associate Professor of computer science at Cornell University. He received his B.S. and M.S. degrees from MIT, and the Ph.D. degree from Stanford University. Since 2001 he has also held a joint appointment as an Associate Professor of radiology at Cornell Medical School. His research interests lie in early vision and its applications, especially in medicine. He has served on numerous program committees, including the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) in 1997, 2000, 2001, and 2003, and the IEEE International Conference on Computer Vision (ICCV) in 1999 and 2001. He is best known for his work on the use of graph algorithms for vision. He organized the IEEE Workshop on Graph Algorithms in Computer Vision, held in conjunction with ECCV in 1999, and served as a Guest Editor for a special issue of IEEE Transactions on Pattern Analysis and Machine Intelligence on the same topic. Two of his papers received the Best Paper Award at the European Conference on Computer Vision in 2002.