Fast Recognition of Musical Genres Using RBF Networks

Douglas Turnbull and Charles Elkan

Abstract—This paper explores the automatic classification of audio tracks into musical genres. Our goal is to achieve human-level accuracy with fast training and classification. This goal is achieved with radial basis function (RBF) networks by using a combination of unsupervised and supervised initialization methods. These initialization methods yield classifiers that are as accurate as RBF networks trained with gradient descent (which is hundreds of times slower). In addition, feature subset selection further reduces training and classification time while preserving classification accuracy. Combined, our methods succeed in creating an RBF network that matches the musical classification accuracy of humans. The general algorithmic contribution of this paper is to show experimentally that RBF networks initialized with a combination of methods can yield good classification performance without relying on gradient descent. The simplicity and computational efficiency of our initialization methods produce classifiers that are fast to train as well as fast to apply to novel data. We also present an improved method for initializing the k-means clustering algorithm which is useful for both unsupervised and supervised initialization methods.

Index Terms—Radial basis function network, musical genre, initialization method, feature subset selection.

1 Introduction

THE classification of music by genre is difficult to automate. Each listener has an experience that is different from that of other listeners when listening to the same piece of music. Our complex perception of sound is influenced by auditory memory, emotions, and social context. Creating a deep representation of emotion or social context is beyond the reach of current artificial intelligence methods, but we show here that an automated musical classification system can extract information from previously heard audio tracks in order to recognize the genre of new tracks.

Our system uses radial basis function (RBF) networks for music classification [2], [11]. RBF networks have two advantages over other classifiers. First, in addition to supervised learning methods, we are able to use unsupervised learning methods to find clusters of audio sounds without presupposed class labels. For example, much of Elvis Presley's music is thought of as rock and roll, even though it is closely derived from the blues music tradition. Unsupervised learning will allow the King's music to be clustered with blues or rock based strictly on audio content. The RBF network distinguishes between these two genres using weights that are learned after the class labels for the samples are included.

The second advantage is that, given good initialization methods, RBF networks do not require much training time when compared with other classifiers. Traditionally, the training of an RBF network requires a large amount of training time because it involves finding good basis function parameters using gradient descent. We show that training time can be reduced by combining multiple initialization methods to provide parameters for the basis functions. Training time is further reduced by using feature subset selection to eliminate unneeded features.

• *The authors are with the Department of Computer Science and Engineering, AP&M Building, Room 4856, University of California San Diego, La Jolla, California 92093-0114.*
E-mail: {dturnbul, elkan}@cs.ucsd.edu.

Manuscript received 20 Jan. 2004; revised 6 July 2004; accepted 2 Oct. 2004; published online 17 Feb. 2005.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-0023-0104.

After a brief introduction to RBF networks and audio feature extraction in Section 2, we develop multiple methods for initializing radial basis functions, as well as describe gradient descent and feature subset selection in Section 3. Section 4 compares classification results using various combinations of initialization methods, using different feature subset sizes, and incorporating gradient descent. The final section contains a discussion of the results and outlines ideas for future research.

2 RBF Networks and Musical Feature Extraction

Radial basis function networks generally have a basis function layer and a linear discriminant layer, as shown in Fig. 1. The input to our RBF network is a vector $\mathbf{x}$ of extracted features, $x_1, \dots, x_d$ from an audio signal. We choose M basis functions for our network, where each function computes the distance from $\mathbf{x}$ to a prototype vector μ. We use unnormalized Gaussians for our basis functions: $\phi_j(\mathbf{x}) = \exp\left\{-\frac{||\mathbf{x}-\mu_j||^2}{2\sigma_j^2}\right\}$. The parameters μ_j and σ_j for each function are determined using methods discussed in Section 3.

2.1 Original Image Code

The top layer is a linear discriminant that outputs a weighted sum of the basis functions. The equation for a single output y_k is $y_k(\mathbf{x}) = \sum_{j=1}^{M} w_{kj}\phi_j(\mathbf{x}) + w_{k0}$, where w_{k0} is the weight of the bias. We find the optimal weights by minimizing the sum of squares error function

$$E = \frac{1}{2}\sum_n \sum_k (y_k(\mathbf{x^{(n)}}) - t_k^{(n)})^2, \tag{1}$$

where $y_k(x^{(n)})$ is the value of the kth output node for the nth data point and $t_k^{(n)}$ is the target value of the kth output node for nth data point. The target value for $t_k^{(n)}$ is 1 if the nth data point is labeled as class k. Otherwise, $t_k^{(n)}$ is 0. To solve (1), we use a method presented in Section 3.4.3 of [1] that computes the pseudoinverse of the matrix Φ, where $(\Phi)_{nj} = \phi_j(\mathbf{x^{(n)}})$.

Each of the output nodes, $y_1, \dots, y_C$, represent a musical genre. When the network is presented with an input vector of the extracted features from an audio sample, the genre that matches the output node with the highest value is picked to be the genre for that audio sample. The percentage of correctly classified novel input vectors determines the classification performance of a network.

For our music classification system, the input feature vector $\mathbf{x}$ is made up of features from three categories: timbral texture, rhythmic content, and pitch content features [15]. Timbral texture features are standard features used for music-speech discrimination and speech recognition. These include Mel-frequency cepstral coefficients (MFCC) and features based on the short time Fourier transform (STFT) of the audio signal. The STFT features include spectral centroid, spectral roll-off, spectral flux, and zero-crossings over the texture window. The rhythmic content features involve beat strength, amplitude, and tempo analysis. The pitch content features contain information about the pitches, such as the frequency of the dominant chord and the pitch intervals between secondary pitches.

3 Constructing RBF Networks

Constructing a good RBF network for classification involves a number of decisions. First, the dimension of the input vector can be manipulated using feature selection. (This is discussed in Section 3.5.) We also need to determine the number of radial basis functions present in the middle layer of the RBF network. If we choose too few, the network will not be able to separate the data. If

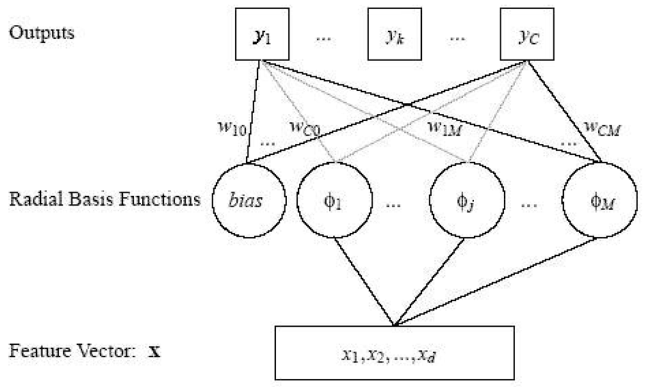

Fig. 1. The structure of a generic radial basis function (RBF) network with a d-dimensional input vector, M radial basis function nodes, and C output nodes.

we choose too many, the network will overfit the data. In practice, the number of basis functions is related to the initialization methods we decide to use. Finally, once the parameters are initialized, they can be refined using gradient descent.

3.1 Unsupervised RBF Initialization

Unsupervised initialization is based on finding clusters within the training set. A major problem with clustering algorithms is that they converge to a poor local optimum due to a bad initialization [10].

We propose a new initialization technique called *subset furthest first* (SFF). The standard furthest first algorithm starts with a randomly chosen center and iteratively adds the next center by finding the point that has the largest minimum distance to all previously selected centers [7]. One drawback of this technique is that it tends to find the outliers in the data set; these are typically not representative of the true clusters. Our improvement is to apply the furthest first algorithm to a subset of the data points. Using a smaller subset, the total number of outliers that furthest-first can find is reduced and, thus, the proportion of nonoutlier points obtained as centers are increased. The size of the subset is $\lceil ck \ln k \rceil$ where k is the number of clusters and c is a constant greater than one. For our application, $c = 2$ works well. This subset size is the number of data points we must sample in order to obtain with high probability at least one sample point from each of k clusters, assuming that the clusters are equal in size. See the Appendix for a proof of this result.

Once the initial locations of the centers have been determined, the standard k-means algorithm finds their final locations [4], [9]. We take each cluster center to represent a radial basis function. The d-dimensional vector μ for the basis function is the location of the cluster center and the scalar σ is the standard deviation of the distance from the cluster center to each of the points that are assigned to that center. Alternatively, a $d \times d$ covariance matrix Σ could be used in place of σ for each of the basis functions, but doing so would increase excessively the number of free parameters in the classifier.

3.2 Supervised RBF Initialization

Supervised initialization uses known class information about the training data to suggest parameters for the basis functions. Our first supervised initialization method uses maximum likelihood estimation to find the parameters of a Gaussian model (MLG) for each class. Using MLG, we construct one radial basis function for each class by averaging all the points within that class. Let $Class_k$ be the set of all the data points belonging class k. Then, $\mu_{Class_k} = \frac{1}{|Class_k|} \sum_{\mathbf{x^{(n)}} \in Class_k} \mathbf{x}^{(n)}$ and $\sigma^2_{Class_k} = \frac{1}{|Class_k|} \sum_{\mathbf{x^{(n)}} \in Class_k} ||\mathbf{x}^{(n)} - \mu_k||^2$, where $|Class_k|$ is the number of data points in class k.

The second supervised method, in-class k-means (ICKM), divides the training set into subsets based on class. The k-means algorithm, using subset furthest first initialization, is run on each of the subsets to obtain cluster centers. Again, each cluster center represents a radial basis function in the network. Note that ICKM with one cluster per class produces the same prototypes as MLG.

ICKM is less complex than other supervised clustering methods, such as Learning Vector Quantization (LVQ) [6], in that prototypes for each class only depend on data points within that class. This means that we may end up with multiple prototypes from different classes that are located close to one another. This may be a problem if we are concerned with directly classifying novel points with these prototypes. However, we learn the upper layer weights of the RBF network to adjust for the fact that prototypes may be close to one another.

3.3 Multiple Initialization Methods

While other researchers have explored various initialization methods for RBF networks [1], [8], none, to our knowledge, have suggested using a combination of initialization methods to build an RBF network. For example, if we have 10 classes, we could create a network with 46 radial basis functions where 10 are determined from MLG, 6 are found using KM, and 30 are found using ICKM.

Note that all three of our initialization methods are implemented by KM. Once we have a fast and effective implementation of KM [4], we gain the benefit of using all three initialization methods without much additional implementation. In addition, alternative clustering algorithms can be substituted for KM [5].

3.4 Improving Parameters with Gradient Descent

It is possible to improve the performance of an RBF network by iteratively updating the means and standard deviations of the radial basis functions using gradient descent [1]. This is done by calculating the derivative of the error function (1) with respect to σ_j and μ_{ji} for each basis function j and feature i. The formula for $\frac{\partial E}{\partial \sigma_j}(x^{(n)})$ is

$$\sum_k \left\{ y_k(\mathbf{x}^{(n)}) - t_k^{(n)} \right\} w_{kj} \exp\left(-\frac{||\mathbf{x}^{(n)} - \mu_j||^2}{2\sigma_j^2} \right) \frac{||\mathbf{x}^{(n)} - \mu_j||^2}{\sigma_j^3}$$

and the formula for $\frac{\partial E}{\partial \mu_{ji}}(x^{(n)})$ is

$$\sum_k \left\{ y_k(\mathbf{x}^{(n)}) - t_k^{(n)} \right\} w_{kj} \exp\left(-\frac{||\mathbf{x}^{(n)} - \mu_j||^2}{2\sigma_j^2} \right) \frac{(x_i^{(n)} - \mu_{ji})}{\sigma_j^2}.$$

We update the means and standard deviations by moving them against the gradient: $\sigma_j \leftarrow \sigma_j - \eta_1 \frac{\partial E}{\partial \sigma_j}$ and $\mu_{ji} \leftarrow \mu_{ji} - \eta_2 \frac{\partial E}{\partial \mu_{ji}}$, where η_1 and η_2 are small, decreasing values called the *learning rates*. For each epoch, we update σ_j and μ_{ji} for each data point using *online learning*. An epoch is defined as using all of the data points once during training. For online learning, we randomly shuffle all the data points at the beginning of an epoch and then update the parameters σ_j and μ_{ji} by using one data point at a time. We repeat this process for a fixed number of epochs.

Overfitting can occur when network parameters (RBF parameters and upper level weights) are trained to reflect the specific training data set rather than general phenomena. This is corrected by reserving a section of the training data, called the *hold-out* set, that is not used for the training of the parameters. Instead, the error on the hold-out set is calculated using (1). The network parameters for the epoch in which the hold-out set error is the smallest are saved and restored after gradient descent stops running. If there is a large number of consecutive epochs in which the hold-out error increases, we can stop gradient descent before it reaches the predefined fixed number of epochs.

A second method to prevent overfitting is regularization. Overfitting tends to occur when the weights in the upper layer of the RBF network begin to take on large positive and negative values to reduce the error function. These large values create a large variance for the classification accuracy. One way to avoid this is to shrink the values of the weights using ridge regression [6]. However, our experiments with ridge regression did not produce better classification results.

3.5 Feature Subset Selection

We can sometimes improve accuracy, as well as reduce the amount of computation, by using a subset of the features. This can be particularly useful when there are redundant and/or noisy features. The key task is to determine which features are useful and which features are unneeded. The brute force method is to try all possible combination of features. Instead, we use *Forward Stepwise Selection* [6] as our heuristic for finding good subsets of features. First, we train d networks, each with one of the features. The feature used for training the network with the best performance is selected. The next feature is selected by training $d-1$ networks, each with the previously selected "best" feature and one of the remaining features. The second feature is selected based on the network with the best performance. Each feature is added one at a time by constructing networks and testing their performance. The total number of networks constructed using Forward Stepwise Selection is $d^2/2$.

In addition, we may be able to further reduce computation using other dimensionality reduction techniques, such as principal component analysis (PCA) [6]. However, this was not explored since we were interested in identifying informative features in the original set of features. PCA involves creating new features that are linear combinations of our original features. It is often hard to understand the meaning of each of these features and the number of original features required is not actually reduced.

4 Experimental Results

Using a data set created by Tzanetakis and Cook [15], we start with 1,000 30-second audio samples, where each of the 10 musical genres has 100 examples. Using the feature extraction techniques implemented in MARSYAS [14], we extract a vector of 30 values where 19 values are timbral texture features (10 MFCC and 9 STFT features), six are rhythmic content features, and five are pitch content features.

Each trial is done with 10-fold cross-validation. In each of the 10 trials, we break the data set into three sets: training, hold-out, and test. The training set uses 800 samples to find the parameter for the RBF network. The hold-out set of 100 samples is used during gradient descent to prevent over-fitting. The test set of 100 samples is used after the network parameters have been found. Classification accuracy for an experiment is the average fraction of novel samples from the test set that are correctly matched to their known genre.

Note that in 11 of the following 12 trials, we construct networks with 90 basis functions. (The first trial, Trial A, has 10 basis functions since it must have exactly one basis function per class.) Our goal is to fix the number of basis functions and examine the performance of networks for different initialization methods, various feature subset sizes, and the effect of gradient descent. Our goal is not optimal model selection [8]. The number 90 is used because it shows decent performance and does not cause overfitting.

4.1 Supervised versus Unsupervised Initialization Methods

Table 1 shows the results for networks initialized with different combinations of the three initialization methods. We use a fixed subset of 15 features for all trials, chosen using Forward Stepwise Selection as explained in the next section.

TABLE 1
Accuracy Using Multiple Initialization Methods and the Same Subset of 15 Features

	# of RBFs				Subset	Performance
EX	MLG	KM	ICKM	Total	# Features	$\hat{\mu}$
A	10	-	-	10	15	0.575
B	-	90	-	90	15	0.646
C	-	-	90	90	15	0.696
D	10	80	-	90	15	0.676
E	10	-	80	90	15	0.703
F	-	40	50	90	15	0.694
G	10	30	50	90	15	0.710

Trials A, B, and C show performance using only one initialization method. MLG gives the poorest performance, but it is able to produce 57.5 percent classification accuracy with only 10 basis functions. ICKM produces better results than KM with the same number of basis functions. We expect this behavior since ICKM uses class information during clustering.

Trials D, E, and F use two of the three methods to initialize the basis functions. Trial G uses all three methods and produces the best results. However, the improvement is not statistically significant compared with the results from Trials C through F. Statistical significance is defined by viewing each trial as $n = 1,000$ Bernoulli events. We compute the symmetric two-tailed 95 percent confidence interval for each trial using the formula that the expected standard deviation of the number of successes observed is $\sqrt{p(1-p)/n}$, where $p \approx 0.7$ is the average classification accuracy.

4.2 Feature Subset Selection

For feature subset selection, we fix the RBF network structure and iteratively add new features to a growing set of previously selected features using forward subset selection. (See Section 3.5.) In Table 2, we show the classification performance for networks that are constructed using the same basis function initialization methods but varying in feature subset size. Trial G from Table 1 is included for comparison.

Trial I shows that good classification can be achieved with a subset of 10 features. Using more than 10 features does not significantly improve classification accuracy. This is not surprising

TABLE 2
Subsets of Features: All Trials Use the Same RBF Network Structure, but Vary in Subset Size

	# of RBFs				Subset	Performance
EX	MLG	KM	ICKM	Total	# Features	$\hat{\mu}$
H	10	30	50	90	5	0.603
I	10	30	50	90	10	0.691
G	10	30	50	90	15	0.710
J	10	30	50	90	20	0.708
K	10	30	50	90	25	0.715
L	10	30	50	90	30	0.704

TABLE 3
Accuracy Before and After Basis Function Parameters σ_j and μ_{ji} are Modified Using Gradient Descent

	# of RBFs				Before GD	After GD
EX	MLG	KM	ICKM	Total	$\hat{\mu}$	$\hat{\mu}_{GD}$
A	10	-	-	10	0.575	0.684
B	-	90	-	90	0.646	0.714
G	10	30	50	90	0.710	0.710

in that we expect some features to be redundant due to the close coupling between various features. Other features may be noisy and may degrade performance.

It is interesting to note which features are chosen first by the subset selection method. The first six features are timbral texture features (five STFT and one MFCC), but there is both a rhythmic content and a pitch content feature selected in the first 10 features.

4.3 Multiple Initialization Methods versus Gradient Descent

One of our central goals is to test the hypothesis that networks using multiple initialization methods can perform as well as networks that are trained using gradient descent. If so, we can quickly train an effective classifier because computing the weights of the upper layer of the RBF network is a closed form operation and the initialization methods are fast compared to gradient descent.

In Table 3, we repeat three trials from Table 1 and run gradient descent to improve network performance. In each trial, creating a network without gradient descent takes seconds, whereas applying gradient descent takes hours to compute given the same workstation. In two of the three trials, we see an improvement using gradient descent as expected. However, the classification accuracy of a network initialized using all three methods (Trial G) is approximately the same as the best performance found using networks that have been improved with gradient descent.

One interesting result is that good classification occurs with a small network (Trial A) that is improved using gradient descent. By observing the hold out set, we see that the parameters of the smaller network migrate over the course of many (~ 100) epochs, whereas overfitting in larger networks (Trials B and Trial G) occurs after just a few epochs (< 10).

The small network will be preferred in situations where the amount of training time is not an issue, there is a high premium on the evaluation time of novel data, or if storage is an issue. The larger classifier will be preferred when we want to limit training time.

5 Discussion

In an equivalent test using the same data set, Tzanetakis and Cook found that humans achieved 70 percent music classification accuracy [15]. Using RBF networks, we are able to achieve 71.5 percent classification accuracy. This performance is comparable to that found by Tzanetakis and Li (71 percent) using support vector machines and linear discriminant analysis with the same data and extracted features [16]. It is not our opinion that the classification of music into genres is limited to 71 percent accuracy, but rather that the method of collecting and labeling the audio samples provides an upper bound.

One suggestion for improvement is to work with expert musicologists and cognitive scientists to first develop a better system for labeling music and then build a data set that captures this system. RBF networks are a good choice of classifier because they are well-suited to work with flexible labeling systems. For example, they can allow for multiple class labels per music sample and real-valued targets depending on the strength of association between a sound sample to a class. For example, Elvis' *Heartbreak Hotel* can be given strength of 0.6 in Blues and a strength of 0.7 in Rock. With the current data set, every song is given a strength of 1.0 for one genre and 0.0 for all other genres. A more flexible classification system is cognitively plausible in that we as humans often classify and subclassify music into a number of genres, streams, movements, and generations that are neither mutually exclusive nor always agreed upon.

We have shown that feature subset selection can be used to improve classification. The next step is to use feature extraction technology to automatically extract many more features ($\geq 10,000$) and then use feature selection to obtain a set of features that is better than the human-selected set used in this paper. This has been a successful technique used by computer vision researchers to classify images (see [13], [17]). Features from images are automatically extracted by taking subimages of different sizes and locations, altering resolution and scaling factors, and applying image filters. In the case of an audio file, we can extract similar features by taking sections of different lengths and starting locations, playing with pitches and tempos in the frequency domain, and applying any number of digital filters such as comb filters.

Appendix
Proof of SFF Subset Size

In Section 3.1, we introduced a new k-means initialization method called Subset Furthest First (SFF). In this Appendix, we show why we choose a subset of size $\lceil ck \ln k \rceil$. The problem, as stated below, is a variant of the *Coupon Collector Problem*, which is addressed in [12].

Consider a data set of n points that consists of k clusters where each cluster has the same number of data points. Our initialization method starts by randomly selecting a subset of m points. Our goal is to determine how large m must be so that we know with high probability that at least one point from each of the k clusters is contained within the subset. Before we can prove this statement, we start with the following:

Lemma 1. *For all $k > 1$, $(1 - 1/k)^k \leq e^{-1}$.*

Proof. Proposition B.3 of [12]. □

Let E_i^m be the event that we have not picked a point from cluster i in our subset of m points. Let E^m be the event that there is at least one of the k clusters for which we have not picked a point. That is, E^m is the union of E_i^m for $1 \leq i \leq k$.

Lemma 2. *The probability of event E^m is at most $ke^{-\frac{m}{k}}$.*

Proof. The proof makes use of Lemma 1, the union bound,[1] and assumes that each cluster is equally represented in the data set. □

$$\Pr[E^m] = \Pr\left[\bigcup_{i=1}^{k} E_i^m\right] \leq \sum_{i=1}^{k} Pr[E_i^m] = k \cdot \left(1 - \frac{1}{k}\right)^m \leq k \cdot e^{-\frac{m}{k}}.$$

When we set $m = \lceil ck \ln k \rceil$ for some constant $c > 1$, we find that the probability of E^m is at most k^{1-c}. In this paper, we use $c = 2$. This assures us that the probability of E^m falls off with the number of clusters. For example, with $k = 20$, $c = 2$, and, thus, $m = 120$, the probability that we have not selected at least one point from each cluster is less than 0.05.

1. The union bound is the fact that the probability of a union of events is at most the sum of the probabilities of the individual events. This fact is true even if the events are not independent.

Acknowledgments

The initial motivation for this work stemmed from research by D. Turnbull with P. Cook and G. Tzanetakis at Princeton University. Some RBF network code was developed with N. Alldrin and A. Smith at UC San Diego. The authors would also like to thank G. Cottrell, G. Balzano, E. Wiewiora, and three anonymous reviewers for their helpful feedback during the writing of this paper.

References

[1] C. Bishop, *Neural Networks for Pattern Recognition.* Oxford Univ. Press, 1995.
[2] D. Broomhead and D. Lowe, "Multivariable Functional Interpolation and Adaptive Networks," *Complex Systems,* vol. 2, pp. 321-355, 1988.
[3] R. Duda, P. Hart, and D. Stork, *Pattern Classification,* second ed. John Wiley & Sons, 2001.
[4] C. Elkan, "Using the Triangle Inequality to Accelerate k-Means," *Proc. 20th Int'l Conf. Machine Learning,* pp. 147-153, 2003.
[5] G. Hamerly and C. Elkan, "Alternatives to the k-Means Algorithm that Find Better Clusterings," *Proc. 11th Int'l Conf. Information and Knowledge Management,* pp. 600-607, 2002.
[6] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning.* Springer, 2001.
[7] D. Hochbaum and D. Shmoys, "A Best Possible Heuristic for the k-Center Problem," *Math. of Operations Research,* vol. 10, no. 2, pp. 180-184, 1985.
[8] N. Karayiannis and G.M. Mi, "Growing Radial Basis Neural Networks: Merging Supervised and Unsupervised Learning with Network Growth Techniques," *IEEE Trans. Neural Networks,* vol. 8, no. 6, pp. 1492-1506, 1997.
[9] J. MacQueen, "On Convergence of k-means and Partitions with Minimum Average Variance," *Annals of Math. Statistics,* vol. 36, p. 1084, 1965.
[10] M. Meila and D. Heckerman, "An Experimental Comparison of Several Clustering and Initialization Methods," *Proc. 14th Ann. Conf. Uncertainty in Artificial Intelligence,* pp. 386-395, 1998.
[11] J. Moody and C. Darken, "Fast Learning in Networks of Locally-Tuned Processing Units," *Neural Computation,* vol. 1, no. 2, pp. 281-294, 1989.
[12] R. Motwani and P. Raghavan, *Randomized Algorithms.* Cambridge Univ. Press, 1995.
[13] K. Tieu and P. Viola, "Boosting Image Retrieval," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 56, nos. 1-2, pp. 17-36, 2000.
[14] G. Tzanetakis and P. Cook, "Marsyas: A Framework for Audio Analysis," *Organised Sound,* vol. 4, no. 30, pp. 169-175, 2000.
[15] G. Tzanetakis and P. Cook, "Musical Classification of Audio Signals," *IEEE Trans. Speech and Audio Processing,* vol. 10, no. 5, pp. 293-302, 2002.
[16] G. Tzanetakis and T. Li, "Factors in Automatic Musical Classification of Audio Signals," *Proc. IEEE Workshop Applications of Signal Processing to Audio and Acoustics,* 2003.
[17] S. Ullman, M. Vidal-Naquet, and E. Sali, "Visual Features of Intermediate Complexity and Their Use in Classification," *Nature Neuroscience,* vol. 5, no. 7, pp. 682-687, 2002.