A Study of Musical Instrument Classification Using Gaussian Mixture Models and Support Vector Machines

Janet Marques and Pedro J. Moreno

Cambridge Research Laboratory
Technical Report Series

CRL 99/4
June 1999

Cambridge Research Laboratory

The Cambridge Research Laboratory was founded in 1987 to advance the state of the art in both core computing and human-computer interaction, and to use the knowledge so gained to support the Company's corporate objectives. We believe this is best accomplished through interconnected pursuits in technology creation, advanced systems engineering, and business development. We are actively investigating scalable computing; mobile computing; vision-based human and scene sensing; speech interaction; computer-animated synthetic persona; intelligent information appliances; and the capture, coding, storage, indexing, retrieval, decoding, and rendering of multimedia data. We recognize and embrace a technology creation model which is characterized by three major phases:

Freedom: The lifeblood of the Laboratory comes from the observations and imaginations of our research staff. It is here that challenging research problems are uncovered (through discussions with customers, through interactions with others in the Corporation, through other professional interactions, through reading, and the like) or that new ideas are born. For any such problem or idea, this phase culminates in the nucleation of a project team around a well-articulated central research question and the outlining of a research plan.

Focus: Once a team is formed, we aggressively pursue the creation of new technology based on the plan. This may involve direct collaboration with other technical professionals inside and outside the Corporation. This phase culminates in the demonstrable creation of new technology which may take any of a number of forms—a journal article, a technical talk, a working prototype, a patent application, or some combination of these. The research team is typically augmented with other resident professionals—engineering and business development—who work as integral members of the core team to prepare preliminary plans for how best to leverage this new knowledge, either through internal transfer of technology or through other means.

Follow-through: We actively pursue taking the best technologies to the marketplace. For those opportunities which are not immediately transferred internally and where the team has identified a significant opportunity, the business development and engineering staff will lead early-stage commercial development, often in conjunction with members of the research staff. While the value to the Corporation of taking these new ideas to the market is clear, it also has a significant positive impact on our future research work by providing the means to understand intimately the problems and opportunities in the market and to more fully exercise our ideas and concepts in real-world settings.

Throughout this process, communicating our understanding is a critical part of what we do, and participating in the larger technical community—through the publication of refereed journal articles and the presentation of our ideas at conferences—is essential. Our technical report series supports and facilitates broad and early dissemination of our work. We welcome your feedback on its effectiveness.

Robert A. Iannucci, Ph.D.
Vice President, Reearch & Advanced Development

A Study of Musical Instrument Classification Using Gaussian Mixture Models and Support Vector Machines

Janet Marques and Pedro J. Moreno

June 1999

Abstract

In this paper, we present a preliminary study of musical instrument classification for use in an audio file annotation system. Using a sound segment 0.2 seconds in length, the classifier can determine the instrument source with a 30% error rate: bagpipes, clarinet, flute, harpsichord, organ, piano, trombone, or violin. The classifier was built after experimenting with different parameters such as feature type and classification algorithm. The features examined were linear prediction coefficients, FFT based cepstral coefficients, and FFT based mel cepstral coefficients. Gaussian Mixture Models and Support Vector Machines were the two classification algorithms studied.

CRL Technical reports are available on the CRL's web page at
http://www.crl.research.digital.com.

Compaq Computer Corporation
Cambridge Research Laboratory
One Kendall Square, Building 700, Suite 721
Cambridge, Massachusetts 02139
USA

1 Introduction

Over the last decade there has been a great deal of work on speech/speaker recognition research. Progress has been made on the analysis of speech waveforms, in its perception by humans, and in the use of different statistical methods for classification. On the other hand, the field of instrument classification and recognition has been studied less. In this paper, we attempt to apply some of the knowledge gained in speech research to the field of instrument classification.

The interest of building computer systems to classify instruments is evident. For example, many Internet search sites, such as AltaVista and Lycos, are evolving from purely textual indexing to multimedia indexing. It is estimated that there are approximately thirty million multimedia files on the Internet with no effective method available for searching their audio content (Swain, 1998).

Audio files could be easily searched if every sound file had a corresponding text file that accurately described people's perceptions of the file's audio content. For example, in an audio file containing only speech, the text file could include the speakers' names and the spoken text. In a music file, the annotations could include the names of the musical instruments. Generating these transcriptions manually is not a feasible alternative, hence automatic methods able to effectively index multimedia files, many of which contain music, are key.

As we mentioned earlier, there has been a great deal of research concerning the automatic annotation of speech files. Currently, it is possible to annotate a speech file with spoken text and name of speaker using speech recognition and speaker identification technology. Researchers have achieved a word error rate of 17.4% for "found speech", speech not specifically recorded for speech recognition (Ligget and Fisher, 1998). Speaker identification systems have been developed to distinguish among approximately 50 voices with a 3.2% error rate (Reynolds and Rose, 1995).

The automatic annotation of non-speech sounds has received less attention. Wold, Blum, Keislar, and Wheaton (1996) built a system that differentiates between the following sound classes: laughter, animals, bells, crowds, synthesizer, and various musical instruments. Scheirer and Slaney (1997) were able to classify sounds as speech or music with a 1.4% error rate. Han, Par, Jeon, Lee, and Ha (1998) have built a system that differentiates between classical, jazz, and popular music with a 45% error rate.

Most of the work done in music annotation has focused on note identification. Moorer (1977) built a system that could produce a score for music containing one or more harmonic instruments. However, the instruments could not be vibrato or glissando, and there were strong restrictions on notes that occurred simultaneously. Subsequently, better transcription systems have been developed (Katayose and

Inokuchi, 1989), (Kashino, Nakadai, Kinoshita, and Tanaka, 1995), and (Martin, 1996).

There have not been many studies done on musical instrument identification. Kaminskyj and Materka (1995) built a classifier for four instruments: piano, marimba, guitar, and accordion. It had an impressive 1.9% error rate. However, in their experiments the training and test data were recorded using the same instruments in the same laboratory. Therefore, their system accuracy will most likely decrease substantially when tested with music played with different instruments in a different studio.

In another study, researchers built a classifier that could distinguish between saxophone and oboe music. The sound segments classified were between 1.5 and 10 seconds long. In this case, the test set and training set were recorded using different instruments and under different conditions. The average error rate was 7.5% (Brown, 1999).

Martin and Kim (1998) built a system that could identify 15 musical instruments using isolated tones. The test set and training set were recorded using different instruments and under different conditions. It had a 28.4% error rate. Since the classifier used isolated tones, we believe that the system would have limited use in an audio annotation system.

In this study, a musical instrument classifier was built that could distinguish between eight types of solo music: bagpipe, clarinet, flute, harpsichord, organ, piano, trombone, and violin. Since the Internet does not contain many files with solo music, this type of system is not immediately practical. However, it does show "proof of concept". Using the same techniques, this work can potentially be extended to include other types of sound such as musical style (jazz, classical, etc.) and sound effects.

A more immediate use for this work is in audio editing applications. Currently, these applications do not use information such as instrument name for traversing and manipulating audio files. For example, a user must listen to an entire audio file in order to find instances of specific instruments. Audio editing applications would be more effective if annotations were added to the sound files (Wold, Blum, Keislar, and Wheaton, 1996).

The outline of the paper is as follows. In section 2 we describe the sound database, the choice of feature set, and the classification algorithms. In section 3 we present our results. We explore the different feature sets, classification algorithms, and the effect of using test data originating from the same source as the training data. We finish the paper with our conclusions and suggestions for future work.

2 Database and System Description

2.1 Sound Database

The training and test data were recorded from 16 compact disks (CDs). We had two solo CDs for each of the musical instruments studied. One CD was used for training data, and one CD was used for test data. We recorded approximately ten minutes of music from each training CD and approximately two minutes of music from each test CD. The audio was sampled at 16 kHz using 16 bits per sample and was stored in AU file format. The amplitude was linearly scaled to the range -1 to 1.

We divided the recorded audio into segments 0.2 seconds in length. We experimented with segment lengths varying from 0.1 seconds to 0.4 seconds. However, our classification results were quite similar for all lengths. We settled on a 0.2 second segment duration for our experiments. In addition, segments with an average amplitude (after scaling) between -0.01 and 0.01 were not used. This automatically removed any silence from the training and test sets. This threshold value was determined by listening to a random portion of the data. Lastly, each segment's average loudness was normalized to 0.15. We normalized the segments in order to remove any loudness differences that may have existed between the CD recordings.

We then composed the training and test sets by randomly choosing a subset of segments from the recorded audio, 1024 training segments and 100 test segments for each instrument. We emphasize that the training and test sets were disjoint and were recorded from different CDs.

2.2 Audio Segment Representations

Several alternatives are possible when converting a fixed duration sound segment into a vector. For example one can explore information contained in the spectral envelope, the phase, or the time evolution of the signal. We decided to experiment with feature set representations that are popular in the speech recognition and coding fields. We believe that the reasons that make these representations valid for speech processing are also valid, to a first degree of approximation, in music processing. We tried three different feature sets: linear prediction coefficients (LPC), FFT based cepstral coefficients, and FFT based mel cepstral coefficients.

2.2.1 Linear Prediction Features

The LPC feature parameterization assumes the speech production model shown in Figure 1. The source $u(n)$ is a series of periodic pulses produced by air forced through the vocal chords, the filter $H(z)$ represents the vocal tract, and the output $o(n)$ is the speech signal (Rabiner and Juang, 1993).

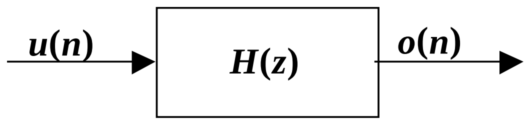

Figure 1 Linear prediction model for speech and music production.

The LPC feature set attempts to approximate the vocal tract system, $H(z)$, with an all-pole model,

$$H(z) = \frac{G}{1 - \sum_{i=1}^{p} a_i z^{-i}}, \tag{1}$$

where G is the model's gain, p is the order of the LPC model, and $\{a_1 \ldots a_p\}$ are the model coefficients. These coefficients compose the feature vector.

As a first approximation, the model shown in Figure 1 is also suitable for musical instrument sound production. The source *u(n)* is a series of periodic pulses produced by air forced though the instrument or by resonating strings, the filter *H(z)* represents the musical instrument, and the output *o(n)* represents the music. Linear prediction analysis attempts to approximate the musical instrument system, *H(z)*. Since there are substantial parallels between speech production and musical instrument sound production, we feel that linear prediction is a reasonable model for music analysis.

In our experiments, we computed linear prediction coefficients using an autoregression model of order 16. Before the autoregression method was applied, each audio segment was multiplied by a Hamming window to smooth out discontinuities at the beginning and end of the segment. The gain was discarded and only the filter coefficients were used as features.

2.2.2 Cepstral Features

Unlike the previous representation that tries to estimate parameters of an assumed production model, cepstral analysis tries to estimate the model *H(z)* directly using homomorphic filtering. First, the audio segment is multiplied by a Hamming window to smooth out discontinuities at the beginning and end of the segment. Then, the Fast Fourier Transform (FFT) of the windowed segment is computed. We then compute the logarithm followed by the inverse FFT. This is shown in equation (2). We used the first 16 coefficients of the output as the cepstral feature set.

$$\text{cepstrum}(o) = \text{FFT}^{-1}(\ln|\text{FFT}(o(n))|). \tag{2}$$

It can easily be demonstrated that the first components of the cepstrum correspond to the production model or general shape of the spectrum. The higher components of the cepstrum correspond to fast changing spectral components that can easily be related to the excitation in a typical speech production model (Oppenheim and Schafer, 1989).

2.2.3 Mel Cepstral Features

A variation of the cepstral representation set is the mel cepstrum. This feature representation is identical to the cepstrum except that the signal undergoes a mel transformation before the cepstral transform is calculated. This transformation modifies the signal so that its frequency content is more closely related to a human's perception of frequency content. The relationship is linear for lower frequencies and logarithmic at higher frequencies (Rabiner and Juang, 1993).

The mel transformation is based on human sound perception experiments. Therefore, it represents how humans perceive sound with more frequency resolution at frequencies below 1 kHz and less frequency resolution above. In as much as music is originally created to be optimally perceived by humans, we hypothesize that a mel frequency analysis might improve classification results.

2.3 Classification Algorithms

We explored two different classification algorithms: Gaussian mixture models (GMM) and Support Vector Machines (SVM). GMM is a popular and easy to implement classification algorithm that has been applied to instrument classification problems before (Brown, 1999), (Martin and Kim, 1998). On the other hand SVMs have not been used in the area of instrument classification, but they have outperformed GMMs in a variety of classification tasks.

2.3.1 Gaussian Mixture Models

Given an ensemble of training corpora feature vectors $X = \{\bar{x}_1, \ldots, \bar{x}_m\}$ where $\bar{x}_i \in R^d$ and assuming that the m vectors are statistically independent and identically distributed, the likelihood that the entire ensemble has been produced by instrument C_1 is,

$$p(X = \{\bar{x}_1, \ldots, \bar{x}_m\} \mid C_1) = \prod_{i=1,m} p(\bar{x}_i \mid C_1). \tag{3}$$

If we assume that the likelihood of a vector can be expressed with a mixture of Gaussian distributions then,

$$p(\bar{x}_i \mid C_1) = \sum_{l=1}^{K} P(l \mid C_1)\ p(\bar{x}_i \mid l, C_1) \text{, where}$$
$$p(\bar{x}_i \mid l, C_1) = \frac{\exp\left(-1/2\,(\bar{x}_i - \mu_{l,1})^t\, \Sigma_{l,1}^{-1} (\bar{x}_i - \mu_{l,1})\right)}{\sqrt{(2\pi)^d \left|\Sigma_{l,1}\right|}} \qquad (4)$$

$P(l \mid C_1)$ is the prior probability of Gaussian l for instrument class C_1, and $p(\bar{x}_i \mid l, C_1)$ is the likelihood of vector $\bar{x}_i$ being produced by Gaussian l within instrument class C_1. The parameters of this Gaussian distribution are the mean vector $\mu_{l,1}$ and the diagonal covariance matrix $\sum_{l,1}$.

During training, we collect all the vectors for a given instrument class and our task is to learn the parameters of the Gaussian mixture, i.e. the mixing weights, the mean vectors and the diagonal covariance matrices. We achieve this goal using the well-known Expectation-Maximization (EM) algorithm. EM is an iterative algorithm that computes maximum likelihood estimates (Dempster, Laird, and Rubin, 1977). The initial Gaussian parameters (means, covariances, and prior probabilities) used by EM are generated via the k-means method (Duda and Hart, 1973).

Once the Gaussian mixture parameters for each instrument class have been found, determining a test vector's class is straightforward. A test vector $\bar{x}$ is assigned to the class that maximizes $p(C_j \mid \bar{x})$, which is equivalent to maximizing $p(\bar{x} \mid C_j)\,p(C_j)$ using Bayes rule. When each class has equal *a priori* probability, the probability measure is simply $p(\bar{x} \mid C_j)$. Therefore, the test vector $\bar{x}$ is classified into the instrument class C_j that maximizes $p(\bar{x} \mid C_j)$.

2.3.2 Support Vector Machines

Support Vector Machines have been used in a variety of classification tasks, such as isolated handwritten digit recognition, speaker identification, object recognition, face detection, and vowel classification. When compared with other algorithms, they show improved performance. This section introduces the theory behind SVMs. Lack of space prohibits a more detailed discussion, but interested readers are referred to (Vapnik, 1995) for an in depth discussion or to (Burges, 1998) for a short tutorial.

The Linearly Separable Case

Suppose we have a set of training samples $\bar{x}_1, \ldots, \bar{x}_m$ where $\bar{x}_i \in R^d$ which are assigned labels $y_1, \ldots, y_m$ (where $y \in \{-1, 1\}$). The labels indicate which of two classes each sample belongs to. Then the hyperplane $(\bar{w} \cdot \bar{x}) + b$ separates the data if and only if

$$(\bar{w} \cdot \bar{x}_i) + b > 0 \quad \text{if} \quad y_i = 1 \tag{5}$$

$$(\bar{w} \cdot \bar{x}_i) + b < 0 \quad \text{if} \quad y_i = -1. \tag{6}$$

We can scale $\bar{w}$ and b so that this is equivalent to

$$(\bar{w} \cdot \bar{x}_i) + b \geq 1 \quad \text{if} \quad y_i = 1 \tag{7}$$

$$(\bar{w} \cdot \bar{x}_i) + b \leq -1 \quad \text{if} \quad y_i = -1 \tag{8}$$

or

$$y_i((\bar{w} \cdot \bar{x}_i) + b) \geq 1 \quad \forall \quad i. \tag{9}$$

To find the optimal separating hyperplane, we need to find the plane that maximizes the distance between the hyperplane and the closest sample. The distance of the closest sample is

$$d(\bar{w}, b) = \min_{\{\bar{x}_i | y_i = 1\}} \frac{\bar{w} \cdot \bar{x}_i + b}{|\bar{w}|} - \max_{\{\bar{x}_i | y_i = -1\}} \frac{\bar{w} \cdot \bar{x}_i + b}{|\bar{w}|}, \tag{10}$$

and from equation (9) we can see that the appropriate minimum and maximum values are ± 1. So we need to maximize

$$d(\bar{w}, b) = \frac{1}{|\bar{w}|} - \frac{-1}{|\bar{w}|} = \frac{2}{|\bar{w}|} \tag{11}$$

Therefore, our problem is equivalent to minimizing $|\bar{w}|^2 / 2$ subject to the constraints expressed in equation (9). By forming the Lagrangian, and solving the dual problem, this can be translated into the following (Burges, 1998): Minimize

$$\sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \bar{x}_i \cdot \bar{x}_j \tag{12}$$

subject to

$$\alpha_i \geq 0 \tag{13}$$

$$\sum_i \alpha_i y_i = 0 \tag{14}$$

The α_i are the Lagrange multipliers; there is one Lagrange multiplier for each training sample. The training samples for which the Lagrange multiplier is non-zero are called *support vectors*, and are such that the equality in equation (9) holds. The samples with Lagrange multipliers of zero could be removed from the training set without affecting the position of the final hyperplane.

This is a well-understood quadratic programming problem, and software packages exist which can find a solution. Such solvers are non-trivial, however, especially in cases where we have large training sets (Osuna, 1998).

The Non-Separable Case

The optimization problem described in the previous section will have no solution if the data is not separable. In order to cope with this scenario, we modify equations (7) and (8) such that the constraints are looser, but a penalty is incurred for misclassification:

$$(\overline{w} \cdot \overline{x}_i) + b \geq 1 - \xi_i \quad \text{if} \quad y_i = 1 \tag{15}$$

$$(\overline{w} \cdot \overline{x}_i) + b \leq \xi_i - 1 \quad \text{if} \quad y_i = -1 \tag{16}$$

$$\xi_i \geq 0 \quad \forall \quad i \tag{17}$$

If $\overline{x}_i$ is to be misclassified, we must have $\xi_i > 1$. This implies that the number of errors is less than $\sum_i \xi_i$. So we may add a penalty for misclassifying training samples by replacing the function to be minimized by $|\overline{w}|^2 / 2 + C(\sum_i \xi_i)$, where C is a parameter which allows us to specify how strictly we want the classifier to fit to the training data. The dual Lagrangian now becomes: Minimize

$$\sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \overline{x}_i \cdot \overline{x}_j \tag{18}$$

subject to

$$0 \leq \alpha_i \leq C \tag{19}$$

$$\sum_i \alpha_i y_i = 0 \tag{20}$$

The Non-Linear Case

The classification framework outlined above is limited to linear separating hyperplanes. However, SVMs can circumvent this problem by mapping the sample points to a higher dimensional space using a non-linear mapping chosen in advance. That is, we choose a map $\Phi : R^d \mapsto H$ where the dimension of H is greater than d. We then seek a separating hyperplane in the higher dimensional space; this is equivalent to a non-linear separating surface in R^d.

When finding a separating hyperplane, the training data always appears in the form of dot products as shown in equation (12). Therefore, in higher dimensional space we are only concerned with the data in the form $\Phi(\bar{x}_i)\cdot\Phi(\bar{x}_j)$. If the dimensionality of H is very large, then this could be difficult, or very computationally expensive to compute. However, if we have a *kernel function* such that $K(\bar{x}_i, \bar{x}_j) = \Phi(\bar{x}_i)\cdot\Phi(\bar{x}_j)$, then we can use this in place of $\bar{x}_i \cdot \bar{x}_j$ everywhere in the optimization problem, and never need to know explicitly what Φ is.

Some examples of kernel functions are the polynomial kernel $K(\bar{x}_i, \bar{x}_j) = (\bar{x}_i \cdot \bar{x}_j + 1)^p$ and the Gaussian radial basis function (RBF) kernel $K(\bar{x}_i, \bar{x}_j) = e^{|\bar{x}_i - \bar{x}_j|^2 / 2\sigma^2}$. The kernel function used in this research was $K(\bar{x}_i, \bar{x}_j) = (\bar{x}_i \cdot \bar{x}_j + 1)^3$. We chose a polynomial of order 3 because it has worked well in a variety of classification experiments. We verified this in our experiments. Other kernels such as the RBF or polynomials of order 2 or 4 also worked reasonably well.

Multi-class classifiers

So far we have only discussed using SVMs to solve two-class problems. However, if we are interested in conducting instrument classification experiments, we will need to choose among multiple classes. The best method of extending the two-class classifiers to multi-class problems is not clear. Previous work has generally constructed a “one vs. all” classifier for each class (Scholköpf, 1995), or constructed a “one vs. one” classifier for each pair of classes.

The “one vs. all” approach works by constructing a classifier for each class which separates that class from the remainder of the data. A given test example $\bar{x}$ is then classified as belonging to the class whose boundary maximizes $(\bar{w}\cdot\bar{x}) + b$. The “one vs. one” approach simply constructs for each pair of classes a classifier which separates those classes. A test example is then classified by all of the classifiers, and is said to belong to the class with the largest number of positive outputs from these sub-classifiers.

In (Weston and Watkins, 1998) a method of extending the quadratic programming problem to multi-class problems is presented. However, the results presented suggest that it performs no better than the more ad-hoc methods of building multi-class classifiers from sets of two-class classifiers.

3 Results and Discussion

We now present results exploring our three feature representations (LPC, cepstrum, and mel cepstrum) and two classification algorithms (SVM and GMM). We also studied the effect of segment length on classification accuracy and examined the implications of using test data originating from the same CDs as the training data.

3.1 Audio Segment Representations

The mel cepstral feature set gave the best results with an overall error rate of 37% classifying segments 0.2 seconds long. We performed this experiment using the Gaussian Mixture Model classification algorithm with 2 mixture components. All of the feature representations were parameterized with 16 dimensional vectors. Figure 2 shows our results.

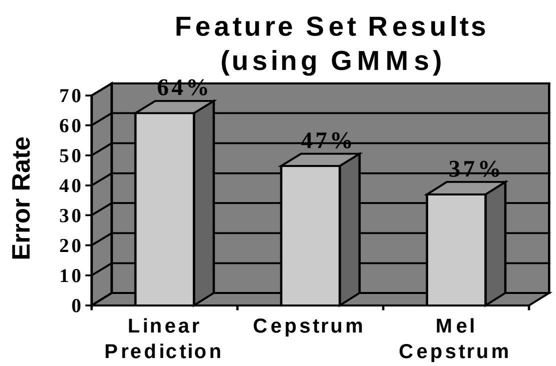

Figure 2 Results for the feature set experiment using a GMM classifier with 2 mixture components. The segments were 0.2 seconds in length.

The cepstral representation performed better than the linear prediction set. This is in agreement with results in speech recognition where LPC coefficients are scarcely used (Rabiner and Juang, 1993). Additionally, the mel scaled cepstral representation gave better performance than the cepstral representation. This is also in agreement with speech recognition results. Therefore, it appears likely that the mel scaling is also beneficial in the music domain.

3.2 Classification Algorithm

The Support Vector Machine classification algorithm gave the best results with an overall error rate of 30% when classifying segments of 0.2 seconds of sound. We used the mel cepstral feature set (16 dimensional vector) and the “one vs. all” algorithm for this experiment. Figure 3 shows the results.

In the SVM experiments, the “one vs. all” algorithm performed slightly better than the “one vs. one” algorithm. In the GMM experiments, we achieved the best results using two Gaussians for each instrument model. Using more than two Gaussians did not improve performance significantly.

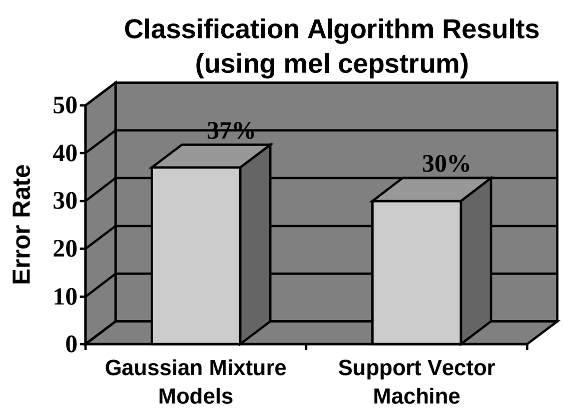

Figure 3 Results for the classification algorithm experiment using a mel cepstral representation and 0.2 second segments. The GMM classifier used two gaussians per class. The SVM was trained with the “one vs. all” multi-class method.

3.3 Classification Based on Sequences of Segments

The previous experiments classify single segments, 0.2 seconds in length. However, it is also interesting to classify longer examples. In this experiment, we classified examples that were two seconds long.

For the SVM classifier, we classified an example using a simple majority rule. First, we divided the sound into 10 segments 0.2 seconds in length. After determining the most likely instrument for each segment, the class with the most votes was chosen as the final instrument.

For the GMM classifier, we divided the sound into 10 segments with the corresponding feature vectors $\{\bar{x}_1, \ldots, \bar{x}_m\}$. Then, we determined the probability that the sequence was played by each of the eight instruments, $C_1 \ldots C_8$, using equation (21). The class with the highest probability was chosen as the final instrument.

$$p(X = \{\bar{x}_1, \ldots, \bar{x}_m\} \mid C_j) = \prod_{i=1,m} p(\bar{x}_i \mid C_j). \tag{21}$$

We ran our experiment using eighty examples of music, two seconds in length, using both the GMM and SVM classifiers. The overall error rate for the 80 sounds was approximately 17%. All of the bagpipe, clarinet, flute, organ, piano, and violin examples were classified correctly. However, 70% of the trombone and harpsichord examples were classified incorrectly. We suspect the trombone error rate was high because the classifier was trained with a tenor trombone, and tested with a bass trombone. We believe that the harpsichord accuracy was low for similar reasons; the system was trained and tested with two harpsichords very different in frequency range.

3.4 Sensitivity to Recording Conditions, Instrument Instance, and Performer

In the experiments described above, the training and test data for each instrument were extracted from different CDs. Thus, the training and test data were recorded in changed conditions using distinct instruments, and different performers. To explore the classifier's sensitivity to recording conditions, instrument instance and performer, we designed an experiment in which the training and test data were recorded in the same acoustic conditions using identical instruments and performers.

We used the mel cepstral feature set and the SVM (one vs. all) classification algorithm. As we expected the error rate decreased by an order of magnitude to 2%. This result is in agreement with Kaminskyj and Materka (1995).

4 Conclusions and Future Work

In this paper, we developed an eight-instrument classifier. Our most successful system had a 30% error rate when classifying 0.2 seconds of audio. It used 16 mel cepstral coefficients as features and employed the Support Vector Machine classification algorithm with the "one vs. all" multi-class algorithm. When the segments used for training and testing the classifiers were recorded in the same acoustic conditions using identical instruments and performers, the classification error rate decreased dramatically to a 2% error rate. We also explored classification based on segment sequences two seconds in length achieving an error rate of 17%.

While the performance of the system is still far from ideal and the size of the corpora is small, we believe this research proves that instrument classification using techniques originating in automatic speech recognition and speech coding is feasible. This work is also one of the first applications of SVM's to music classification.

There are three important areas of future work: (1) Improve the accuracy of the eight-instrument classifier. (2) Add the capability to classify concurrent sounds. (3) Build more practical sound classifiers for use in audio annotation systems.

4.1 Accuracy Improvements

The eight-instrument classifier can be improved by increasing the generality of the training data. In this study, the training data for each instrument was recorded from a single CD. Therefore, each instrument model was trained using just one instrument example. Using more CDs would lead to more general training data.

The accuracy of the eight-instrument classifier can also be improved using temporal information both in the feature representation and in the classifier. For example, the log-lag correlogram representation has been previously used in music classification with some success (Martin and Kim, 1998). A Hidden Markov model classifier could also be used to capture the temporal evolution of the feature set, perhaps improving classification performance (Rabiner and Juang, 1993).

4.2 Classification of Concurrent Sounds

Currently the classifier cannot identify sounds that occur simultaneously. For example, it cannot distinguish between a clarinet and a flute being played concurrently.

There has been a great deal of work in perceptual sound segregation. Researchers believe that humans segregate sound in two stages. First, the acoustic signal is separated into multiple components. This stage is called auditory scene analysis (ASA). Afterwards, components that were produced by the same source are grouped together (Bregman, 1990).

There has not been much progress in automatic sound segregation. Most systems rely on knowing the number of sound sources and types of sounds. However, some researchers have attempted to build systems that do not rely on this data. One group successfully built a system that could segregate multiple sound streams, such as different speakers and multiple background noises (Brown, 1994).

4.3 Additional Sound Classifiers

In order to build an annotation system that will add meaningful labels to any audio file, more sound classifiers will need to be built. Some particularly important classifiers are musical style detectors, music lyric recognizers, and sound effect classifiers.

We believe that it is possible to build an annotation system that can automatically generate descriptive and accurate labels for any sound file. Once this occurs, it will no longer be difficult to search audio files for content.

Acknowledgements

We would like to thank Judith Brown (MIT, Media Lab), Brian Eberman (Compaq, Cambridge Research Lab, now at SpeechWorks), Dave Goddeau (Compaq, Cambridge Research Lab), Keith Martin (MIT, Media Lab), Tomaso Poggio (MIT, Center for Biological and Computational Learning), and Jean-Manuel Van Thong (Compaq, Cambridge Research Lab) for their valuable guidance and advice. We would like to thank Phillip Clarkson (Compaq, Cambridge Research Lab, now at SpeechWorks) for implementing the Support Vector Machine client code used in this research and for providing us with the SVM summary used in this paper. We would also like to thank Edgar Osuna (MIT, Center for Biological and Computational Learning) and Tomaso Poggio for providing us with the Support Vector Machine software.

References

Bregman, A.S. (1990). Auditory Scene Analysis, MIT Press, Cambridge, MA.

Brown, J.C. (1999). “Computer identification of musical instruments using pattern recognition with cepstral coefficients as features,” J. Acoust. Soc. Am., 105, 1933-1941.

Brown, J.G. (1994). “Computational Auditory Scene Analysis,” Computer Speech and Language, 8, 297-336.

Burges, C. (1998). “A Tutorial on Support Vector Machines for Pattern Recognition,” Data Mining and Knowledge Discovery, 2, 2.

Dempster, P., Laird, N.M., and Rubin, D.B. (1977). "Maximum Likelihood from Incomplete Data Using the EM Algorithm," Journal of the Royal Society of Statistics, 39, 1, 1-38.

Duda, R.O. and Hart, P.E. (1973). Pattern Classification and Scene Analysis, John Wiley & Sons, New York.

Han, K., Par, Y., Jeon, S., Lee, G., and Ha, Y. (1998). "Genre Classification System of TV Sound Signals Based on a Spectrogram Analysis," IEEE Transaction on Consumer Electronics, 44, 1, 33-42.

Kaminskyj, I. and Materka, A. (1995). "Automatic Source Identification of Monophonic Musical Instrument Sounds," IEEE International Conference On Neural Networks, 1, 189-194.

Kashino, K., Nakadai, K., Kinoshita, T., and Tanaka, H. (1995). "Application of Bayesian Probability Network to Music Scene Analysis," IJCAI95 Workshop on Computational Auditory Scene Analysis, August, Quebec.

Katayose, H. and Inokuchi, S. (1989). "The Kansei Music System," Computer Music Journal, 13, 4, 72-7.

Ligget, W. and Fisher, W. (1998). "Insights from the Broadcast News Benchmark Tests," DARPA Speech Recognition Workshop, February, Chantilly, VA.

Martin, K. (1996). "Automatic Transcription of Simple Polyphonic Music," MIT Media Lab Perceptual Computing Technical Report #385, July.

Martin, K.D. and Kim, Y.E. (1998). "Musical Instrument Identification: A Pattern-Recognition Approach," presented at the 136th Meeting of the Acoustical Society of America, October, Norfolk, VA.

Moorer, J.A. (1977). "On the Transcription of Musical Sound by Computer," Computer Music Journal, 1, 4, 32-8.

Oppenheim, A. and Schafer, R.W. (1989). Discrete-Time Signal Processing, Prentice Hall, Englewood Cliffs, NJ.

Osuna, E. (1998). "Applying SVMs to face detection," IEEE Intelligent Systems, 23-6, July/August.

Rabiner, L. and Juang, B. (1993). Fundamentals of Speech Recognition, Prentice Hall, Englewood Cliffs, NJ.

Reynolds, D.A. and Rose, R.C. (1995). "Robust Text-Independent Speaker Identification Using Gaussian Mixture Speaker Models," IEEE Transactions on Speech and Audio Processing, 3, 1, 72-83.

Scheirer, E. and Slaney, M. (1997). "Construction and Evaluation of a Robust Multifeature Speech/Music Discriminator," Proceedings of ICASSP, 1331-4.

Scholköpf, B. (1995). "SVMs - A Practical Consequence of Learning Theory," IEEE Intelligent Systems, July/August, 18-21.

Swain, M. (1998). Study completed at Compaq Computer Corporation, Cambridge, MA.

Vapnik, V. (1995). The Nature of Statistical Learning Theory, Springer-Verlag, New York.

Weston, J. and Watkins, C. (1998). "Multi-class Support Vector Machines," Technical Report CSD-TR-98-04, Department of Computer Science, Royal Holloway, University of London, May.

CRL 99/4
June 1999

A Study of Musical Instrument Classification Using Gaussian Mixture Models and Support Vector Machines

Janet Marques and Pedro J. Moreno