A Timbre Recognition System Using Hidden Markov Models

Doug Van Nort
Signal Processing and Control Laboratory
Faculty of Music - McGill University
555 Sherbrooke St. West
Montreal, QC Canada H3A 1E3
doug@music.mcgill.ca

1. ABSTRACT

This paper describes a system for recognition of the timbre of isolated instruments. This system implements a discrete Hidden Markov Model for classification as well as a k-means algorithm for clustering purposes. The system is tested and results are reported.

2. INTRODUCTION

A *Hidden Markov Model* (HMM) is a stochastic model consisting of a set of N states $\{S_1, \ldots, S_N\}$, along with probability distributions that govern transition between states, initial state, and observation probability from a defined grammar at each given state. It is useful in the description of time-series events. For more information the reader is directed to [Rabiner 1989]. Rather than describe the mathematics of the technique, this paper describes an implementation of HMMs for the purpose of automatic recognition of musical instruments. The approach is to classify the timbre of an instrument, where timbre is described by an appropriate *feature vector*. Specifically, we look here at isolated instruments. Many similar examples exist in the case of speech recognition, inlcuding those reported in [Rabiner 1989],[Ostendorf et al. 1996], [Kwong and Chau 1997] and [Selouani and Shaughnessy 2001]. Not as much work has dealt with the use of HMMs for instrument recognition, but some examples are discussed in [Herrera-Boyer et al. 2003] as well as [Lee and Chun 2002]. However, more work has focused on other methods of instrument recognition as well as on HMMs for other audio tasks. For the former see [Herrera-Boyer et al. 2003] and [Fraser and Fujinaga 1999] for different approaches, and for examples of the latter see [Depalle et al. 1993] and [Raphael 1999].

This work presented in [Lee and Chun 2002] is the most similar to that which is presented here. However, there are several major differences. The aforementioned uses continuous observation densities via Gaussian Mixture Models, harmonic partials as features and seemingly uses the entire audio signal. In contrast, this approach uses a discrete observation density, the features come from both attack time and cepstral coefficients and only the first .5 seconds of audio is used.

3. THE SYSTEM

As stated in the introduction, this implementation uses discrete HMMs, and the features used for classification are attack time and cepstral coefficients as well as their first order difference (delta cepstrum). The system was coded in Matlab, and used the McGill University Master Samples as test data. The steps in the process are as follows, and will be discussed individually: 1. Digitize audio 2. Extract features 3. Vector quantization/Clustering 4. Map features of training data to codebook 5. Train HMMs 6. Classify test data

3.1 Digitize Audio

The audio samples used were from the McGill University Master Samples (MUMS). The samples used were of violin, flute and trumpet. Each instrument was modeled by a separate HMM, and for training purposes 64, 54 and 48 samples were used for each respective instrument. This fairly large number or samples was achieved by using the left and right channels of the stereo files separately, a decision that is justified by the fact that the instruments are recorded with different mic placements for each channel. The belief that this would cause a timbral difference is supported by the fact that stereo pairs were classified differently by the trained models in several instances.

3.2 Extract Features

Twenty-one features were used for describing timbre. They were attack time, the first 10 cepstral coefficients and their derivatives. Low order cepstral coefficients and their first order difference are used to describe the slowly changing qualities of a spectrum, and thus are frequency independent. As we here use a range of pitches to train our models, these are a logical choice for this system. See [Dubnov and Rodet 1998] and [Eronen and Klapuri 2000] for more discussion of the benefits of these particular features. Attack time has been linked to a human's perception of timbre, and so it stands to reason that this will provide useful as a descriptor. The algorithm that defines attack time looks for both the point at which the signal rises above some minimal threshold (signal beginning) and the maximum value over the first quarter length of the audio signal (attack end). The distance in samples between these two points is our defined attack time.

After finding attack time, 100 analysis frames of 256 samples with an overlap of 50% are used to extract cepstrum coefficients. Thus, roughly the first .5 seconds of audio samples are used for extraction purposes.

3.3 Vector Quantization/Clustering

A discrete HMM system is utilized, and so the infinite number of possible feature vectors in $\mathbf{R}^{21}$ need to be quantized to some suitable discrete "codebook." In [Rabiner 1989] it is mentioned that a larger code book reduces quatization error, but that the benefit is less after a certain point. Thus, a tradeoff exists between computation of codebook and the decrease in error that it gives you. Here we find a near optimal codebook for our particular set of training observations, by use of a k-means clustering algorithm. As the total number of observations exceeds 16,000 (100 observations per sample), the relatively small number of 73 codebook entries was used. Even with this number of entries, the algorithm takes over 10 minutes to compute on a 1.25 Ghz G4 laptop. The k-means algorithm returns a centroid for 73 different clusters, and this collection becomes our discrete codebook of possible observations.

3.4 Map Features to Codebook

In order to train the HMMs or to classify the test data, the observed features vectors need to be mapped to "legal" entries within the observation codebook. This is achieved by first assigning numbers to the 73 entries of the codebook, comparing the Euclidean distance of a given vector to all of the entries and then mapping the input feature vector to the number representing the nearest neighbor in the codebook. After this point, each observation sequence consists of 100 (one for each analysis frame) integers ranging from 1-73. These are the values that are used to actually train the HMM.

3.5 Model Creation and Training

As said previously, a different HMM is used for violin, flute and trumpet. A left-to-right model is used, and so the initial state distribution is trivially a probability of 1 for state 1 and zero for the other states. The initial transition and observation probabilites are chosen at random. We model each analysis frame as a state, and so the observation probability matrix is of size 100x73 while the transition probability matrix is 100x100. The training set observations are used to update model parameters (transition and observation probability matrices), using the Baum-Welch EM algorithm.

3.6 Classification

After the three models have been trained, observation sequences (mapped to the codebook) which were not used for training are tested on the models. The probabilty of test observation O given model H, or $P(O|H)$, is calculated for each model by the "forward algorithm." The model that returns a maximum is then chosen as the correct instrument/class. To test the system, 20 samples were used each for violin, flute and trumpet for a total of 60 test samples. The probability (log-likelihood) was calculated for each HMM, and the model that gave the highest likelihood value was returned as the proper class.

As a variant on this approach, another test was conducted in which each model possessed its own observation codebook. Thus, the sequences were classified based on the relative nature of the transitions, as opposed to the absolute classification of sequence values and transitions, as in the main system. It was expected that this secondary experiment would yield less successful results, which was the case. The results of both experiments are reported in table 1.

	Violin	Flute	Trumpet
Single Codebook	95%	85%	65%
Multiple Codebook	50%	75%	60%

Table 1: Success Rate of Timbre Classification System

4. CONCLUSION

The system presented here has given some promising results, including a correct classification rate of 95% in the case of violin and 85% in the case of flute. If the system can prove to be this successful on larger databases with more instruments, then it may be a viable option as it requires only a discrete observation set, a simple feature set and only a small portion of audio. Further, aside from the computation of the k-means clustering of the large observation database, all of the computations are relatively fast to compute.

5. REFERENCES

Depalle, P., G. Garcia, and X. Rodet (1993). Tracking of partials for additive sound synthesis using hidden markov models. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Dubnov, S. and X. Rodet (1998). Timbre recognition with combined stationary and temporal features. In *International Computer Music Conference*, pp. 102–108.

Eronen, A. and A. Klapuri (2000). Musical instrument recognition using cepstral coefficients and temporal features. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Fraser, A. and I. Fujinaga (1999). Towards real-time recognition of acoustic musical instruments. In *International Computer Music Conference*, pp. 175–177.

Herrera-Boyer, P., G. Peeters, and S. Dubnov (2003). Automatic classification of musical instrument sounds. *Journal of New Music Research 32*(1), 3–21.

Kwong, S. and C. Chau (1997). Analysis of parallel genetic algorithms on hmm based speech recognition system. *IEEE Transactions on Consumer Electronics 43*(4), 1229–1233.

Lee, J. and J. Chun (2002). Musical instrument recognition using hidden markov model. In *Asilomar Conference on Signal, System and Computer*.

Ostendorf, M., V. V. Digalakis, and O. A. Kimball (1996). From hmm's to segment models: A unified of stochastic modeling for speech recognition. *IEEE Transactions on Speech and Audio Processing 4*(5), 360–378.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pp. 257–286.

Raphael, C. (1999). Automatic segmentation of acoustic musical signals using hidden markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence 21*(4).

Selouani, S. and D. O. Shaughnessy (2001). Hybrid architecture for complex phonetic features classification: A unified approach. In *International Symposium on Signal Processing and It's Applications (ISSPA)*, pp. 719–722.