Hidden Markov Classification for Musical Genres

Igor Karpov
Rice University
ikarpov@rice.edu

COMP 540: Adaptive Systems
Term Project
Dr. Devika Subramanian
Fall 2002

1. Abstract

The need for an efficient technique for digital music classification based on content arises in many modern settings, including digital media libraries, online search engines, and personal music collections. One of the best examples of such classification is classification among diverse musical genres, where categories are assigned to musical pieces based on their "similarity" with other pieces in those categories. This paper describes a system for classification of digitally sampled music using a combination of digital signal processing techniques and hidden Markov models. Different configurations of this system are tested on a moderately sized collection of four and three music genres, and the results are analyzed.

2. Introduction

With increasing popularity of digitally sampled music formats such as compact discs and MPEG-3 encoded audio files, exciting possibilities for new applications appear. Imagine being able to whistle a tune, say a genre and an approximate time of origin and be quickly presented with a list of relevant musical pieces at your local library. Or, have the same indexing technology work on your personal computer, with a personal MP3 collection. Imagine being able to customize a search engine to look for music similar to songs you have previously enjoyed, or to send your friend a software agent that encodes your musical tastes to some degree.

Many methods have been proposed to tackle the problem of music content search and classification (see next section). Hidden Markov models are known to work very well for certain problems with sequential inputs, such as speech recognition (5). Intuitively, the method models a sequential process with visible outputs and invisible internal state. This structure lends itself well to applications in music, which is a sequential process.

The purpose of this project is to design, test and assess the feasibility of a music classification system using hidden Markov models. Digitally sampled music formats were chosen as the input to the system. A large body of previous work deals with symbolic music formats and pure melodic information (see next section). While this kind of data is easier to work with in some cases, it can loose some of the information contained in digitally sampled music and is not as readily available in most real world applications.

3. Previous Work

Wei Chai and Barry Varcoe at MIT Media Laboratory have used hidden Markov models to classify four different symbolic representations of folk music from three Western European countries (1). Significantly, they have shown that the number of hidden states did not have a dramatic impact on the effectiveness of the system. They have also shown that simple left-right and strict left-right models outperformed HMM's with more complex connection structure between hidden states. Two-way classification accuracies of 75%, 77% and 66% and three-way accuracy of 63% were achieved among very similar classes of music.

In a term project at Stanford, Paul Scott has designed a system for musical genre classification based on digitally sampled content using an artificial neural network (3). While their primary strength is the recognition of static patterns, ANNs have been used with some success in sequential problems such as speech recognition (7). Scott's system used a combination of digital signal feature extraction, preprocessing, and a three-layer feed-forward neural network. His system achieved average successful classification rates of 95% on unseen data after coming to a validation stop on the training data. His work has shown that combining different feature extraction techniques is a good strategy for music classification.

Chris Burges and others at Microsoft Research have proposed Distortion Discriminant Analysis, a new algorithm for digital sound fingerprinting that uses linear neural networks for efficient oriented principal component analysis (OPCA) (4). This work may prove useful in future music classifiers.

Researchers at Microsoft Corp. and Cambridge University have developed a software toolkit called HTK for using hidden Markov models in speech recognition applications (6). Tools from this library are used extensively by the system described in this report.

4. System Setup

A system overview is presented in Figure 1. Preprocessed sampled music is converted to a sequence of feature vectors by the feature extractor. Part of the dataset is used to train a continuous multivariate hidden Markov model for each genre. The trained models are then used to select the genre for unseen data by selecting the model with the highest input probability for that piece.

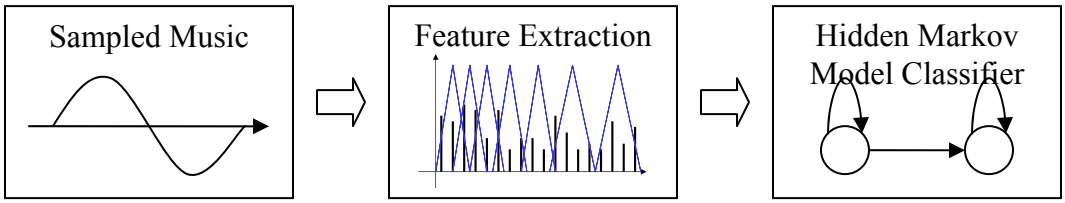

Figure 1: Overview of the system.

4.1. Input Data

A corpus of 252 songs in MP3 format (extracted and encoded from original audio CDs) was collected, 47 Celtic, 69 Western classical, 70 techno and trance, and 66 rock songs in all. The goals guiding the selection of the songs were to get a large variety of artists and styles within each genre. To simplify processing, the songs were converted to PCM-encoded WAV files at sample frequency of 11025 Hz, 2 bytes per sample, mono.

Ten 10-second segments were extracted from each piece. The scheme used in extracting the samples is demonstrated in Figure 2.

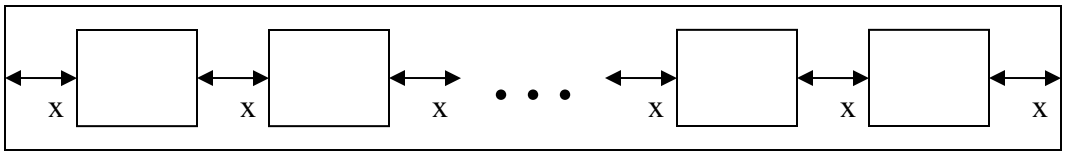

Figure 2: Splitting of the song into 10 10-second samples.
x = ((total samples) – 10*110250)/11.

The data is intrinsically high-dimensional, as can be seen from principal component analysis of the Fourier transform (a basis for much of the feature extraction performed in this project) in Figure 3. Because of this, effective feature extraction techniques are important in dealing with the problem.

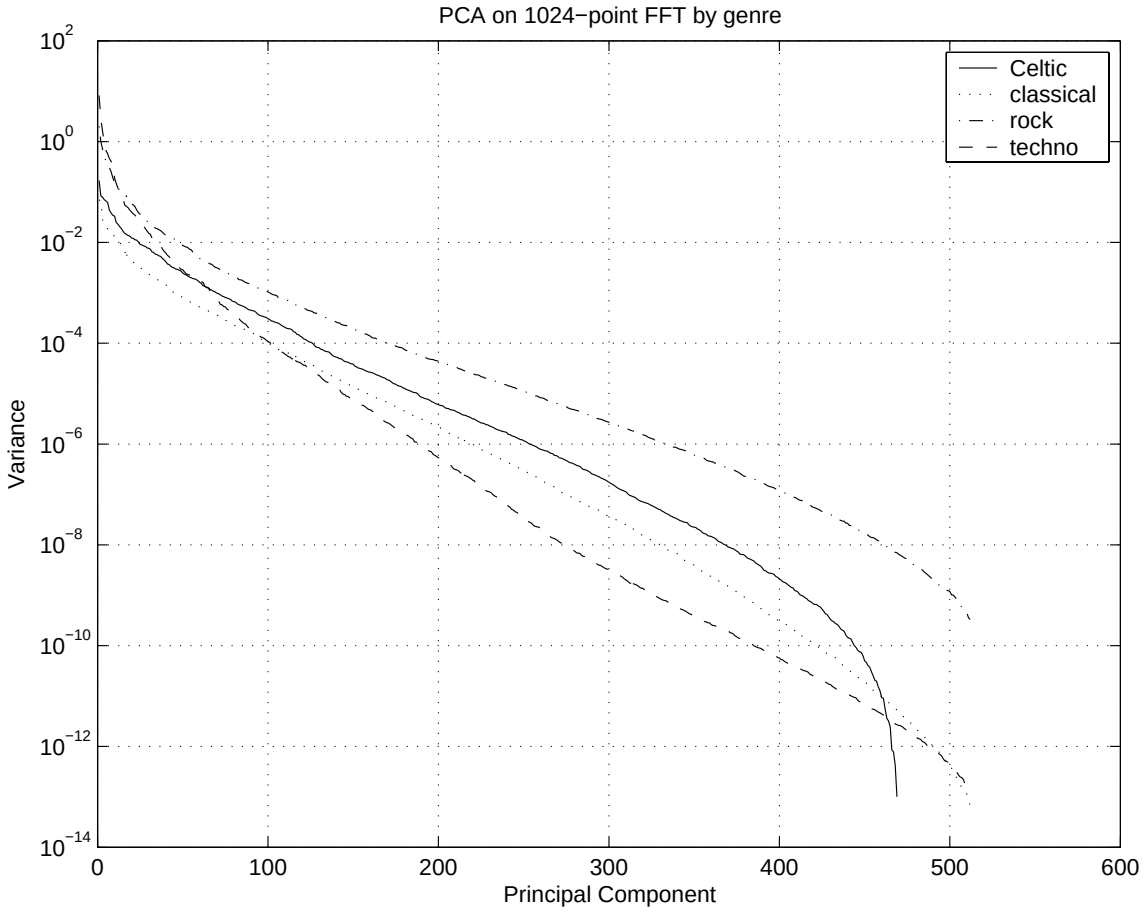

Figure 3: Principal component analysis of 1024-point fast Fourier transform by musical genre.

4.2. Feature extraction

Three distinct types of features were used in separate experiments: Fourier transform-based Mel cepstral coefficients, Mel cepstra with delta and acceleration information, and linear predictive coding.

4.2.1. Mel-Frequency Cepstral Coefficients (MFCC)

The Mel frequency scale was originally developed in phonetics to help model the non-linear nature of human auditory system. To find the cepstral coefficients of a window in an audio signal, the discrete Fourier transform of the Hamming window of the signal is filtered by a bank of triangle filters equally distributed on the Mel frequency scale (the scale is part linear, part logarithmic in frequency). The filter outputs are then fed to a logarithmic function and a cosine transform. 12 cepstral coefficients were used in my experiments. 25 ms windows are taken every 10 ms, resulting in 1000 feature vectors per 10 second audio sample.

The HCopy utility from the HTK toolset was used to convert WAV files to sequences of feature vectors stored in HTK format.

4.2.2. Delta and Acceleration

Delta values $[\Delta(\mathbf{v}_i)]$ and acceleration values $[\mathtt{acc}(\mathbf{v}_i)]$ can be appended to any feature vector $\mathbf{v}_i$. They are computed as $\Delta(\mathbf{v}_i) = \mathbf{v}_i - \mathbf{v}_{i-1}$ and $\mathtt{acc}(\mathbf{v}_i) = \Delta(\mathbf{v}_i) - \Delta(\mathbf{v}_{i-1})$. In my experiments, 12 delta values and 12 acceleration values were appended to the MFCC feature vector bringing the total number of features to 36.

Delta and acceleration values are very important improvements in feature extraction for hidden Markov models because they effectively increase the state definition to include first and second order memory of past states.

4.2.3. Linear Predictive Coding (LPC)

Linear predictive cepstra were used as an alternative to MFCC features. Linear predictive analysis is based on a model of the vocal tract as an all-pole filter with the transfer function:

$$H(z) = \frac{1}{\sum_{i=0}^{p} a_i z^{-i}}$$

Where p is the order of the filter and the coefficients a_i are chosen to minimize the mean square filter prediction error summed over the analysis window and a_0 is defined to be 1. The cepstra can be computed simply as:

$$c_n = -a_n + \frac{1}{n}\sum_{i=1}^{n-1}(n-i)a_i c_{n-i}$$

12 linear prediction (order 14) cepstra were used in the experiments with the same windowing settings (25 ms windows every 10 ms).

4.3. Hidden Markov Model Classifier and Experimental Setup

Hidden Markov models are discrete-time stochastic signal models (5). They can be completely defined by the number of hidden states, a static state transition probability distribution, the observation symbol probability distribution and the initial state distribution. They also observe the Markov property, which states that the system can be completely described at any time by the current and the predecessor state, truncating the history of state transitions that led to the moment. These models have been applied with great success in speech recognition and other domains (6).

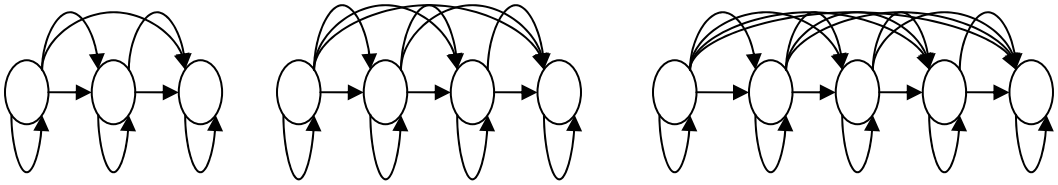

Figure 4: 3, 4, and 5-state HMM's used in the experiments.

A continuous-input hidden Markov model template was created for each genre with random initial parameters. Each state's observation distribution was modeled by a single Gaussian with 12-dimensional mean and 12 by 12 diagonal variance for MFCC and LPC features and 36-dimensional mean and 36 by 36 diagonal variance for MFCC supplemented by delta and acceleration values. Hidden state number was varied between 3, 4 and 5 states. A left-to-right architecture was used (Figure 4).

Equal proportions of all genres were used in all datasets (47 from each genre in 4-way classification and 66 from each genre in 3-way classification). For each experiment, the data was split into a training set (70%) training and a test set (30%). Two types of splitting were used – equal random splitting and constrained random splitting. In equal random splitting, 70% of each genre was selected randomly and assigned to the training set, with the rest assigned to the test set. In constrained random splitting, an additional condition that all samples from a song should be assigned to a single set when possible was imposed. This was done to verify that the generalization of the system was valid between songs and not just due to the similarity within each song.

The training set was used in Baum-Welch reestimation of the parameters of each model: the state transition probabilities and the means and the variances of each output distribution. The estimation was stopped after 20 iterations or after the change in log likelihood fell below a threshold value of 0.0001, whichever came first. HTK HRest and HParse utilities were used for this purpose (6).

The trained HMMs were then combined to build a classifier system. The Viterbi algorithm was used to obtain the log likelihood probability for each test input, and a classification corresponding to the highest value was assigned. HTK HVite tool was used for this purpose (6).

5. Results

5.1. 4-way classification

470 10-second clips from each genre were selected and used for all the experiments in this section. The genres were Techno, Classical, Rock and Celtic. This resulted in 329 per genre (1316 total) clips being used for training and 141 per genre (564 total) clips being used as a test set. Each table represents the average results of 15 cross-validation trials with the dataset split anew for every trial. The row titles represent actual genre, while the column titles represent the classification assigned by the system. Standard deviation is reported in parentheses.

5.1.1. Varying feature extraction

Table 1 shows the classification percentages for 4-way classification by a 4-state HMM system that uses 12 Mel-Frequency Cepstral Coefficients as the feature vector.

Table 1: 12 MFCC

	Tech.	*Class.*	*Rock*	*Celt.*
techno	**88.2** (3.0)	7.5 (2.2)	2.7 (1.6)	1.5 (1.2)
classical	9.1 (1.8)	**74.3** (3.0)	1.7 (0.96)	14.8 (2.6)
rock	4.1 (1.2)	1.6 (1.2)	**82.4** (3.1)	11.9 (2.9)
Celtic	3.6 (2.1)	13.0 (3.4)	12.2 (2.7)	**71.1** (2.6)

Table 2 shows the results of otherwise similar experiment with the 12 MFCC features complemented by delta and acceleration coefficients.

Table 2: 12 MFCC, 12 delta, 12 acceleration

	Tech.	*Class.*	*Rock*	*Celt.*
techno	**92.4** (1.8)	5.4 (2.0)	1.9 (0.96)	0.3 (0.35)
classical	3.0 (1.4)	**88.1** (2.6)	2.4 (1.5)	6.4 (2.1)
rock	2.9 (1.7)	2.1 (0.66)	**84.7** (3.8)	10.3 (3.3)
Celtic	1.5 (0.98)	12.1 (3.1)	14.0 (3.5)	**72.4** (5.1)

Table 3 shows the same experiment performed with 12 linear prediction cepstra.

Table 3: 12 LP Cepstra

	Tech.	*Class.*	*Rock*	*Celt.*
techno	**85.1** (2.8)	8.9 (2.2)	3.5 (1.5)	2.5 (1.5)
classical	10.4 (2.2)	**74.6** (3.6)	1.8 (1.1)	13.1 (3.2)
rock	2.2 (1.4)	2.1 (1.3)	**85.3** (2.7)	10.4 (2.0)
Celtic	2.4 (1.3)	13.1 (3.4)	12.5 (3.2)	**71.9** (4.1)

5.2. 3-way classification

660 10-second clips per genre (1980 total) were used as the dataset for these experiments. The genres were Techno, Classical, and Rock. 462 clips per genre (1386 total) were used for training sets and 198 clips per genre (594 total) were used for testing. Again, 15 cross-validation trials were performed for each result reported. The row titles represent actual genre, while the column titles represent the classification assigned by the system. Standard deviation is reported in parentheses.

5.2.1. Varying number of hidden states

All experiments in this section were conducted using 12 MFCC supplemented by 12 delta and 12 acceleration values. Equal random splitting was used to generate the training and test sets. Table 4 lists the results with a 3-state HMM. Table 5 lists the results of a 4-state HMM, and table 6 lists the results of a 5-state HMM.

Table 4: 3-state HMM

	Tech.	*Class.*	*Rock*
techno	**91.3** (2.2)	7.1 (1.9)	1.5 (1.3)
classical	3.9 (0.87)	**93.7** (1.3)	2.4 (1.0)
rock	2.7 (1.5)	4.0 (1.3)	**93.3** (1.4)

Table 5: 4-state HMM

	Tech.	*Class.*	*Rock*
techno	**93.4** (1.2)	4.4 (1.3)	2.2 (0.7)
classical	3.9 (1.7)	**94.2** (1.9)	1.9 (1.0)
rock	2.4 (1.3)	3.3 (0.90)	**94.3** (2.0)

Table 6:5-state HMM

	Tech.	*Class.*	*Rock*
techno	**93.0** (1.5)	5.0 (1.3)	2.0 (0.94)
classical	5.0 (2.2)	**93.6** (2.3)	1.4 (0.93)
rock	2.4 (0.68)	3.2 (0.94)	**94.4** (1.2)

5.2.2. Verifying cross-song generalization

Table 7 shows the results of an experiment similar to that of table 6, except that constrained random splitting was used, where clips were grouped by song and these groupings were preserved as much as possible in assigning the clips to the training and test sets.

Table 7: constrained random splitting

	Tech.	*Class.*	*Rock*
techno	**93.2** (2.9)	5.1 (2.3)	1.7 (1.2)
classical	7.0 (4.0)	**91.0** (4.0)	2.0 (1.3)
rock	2.6 (2.2)	3.4 (1.7)	**94.0** (3.1)

6. Conclusion

Hidden Markov models have proven to be another viable method for music classification by genre categories. The training process on a sizeable dataset takes about 15 minutes on a Sun Ultra 10 workstation. The classification process is almost instantaneous for a single song. Because this is a prototype system, this performance is expected to improve, making such systems practically applicable in real-world applications.

Some types of music have proven to be more difficult to classify than others. In particular, the Celtic dataset has proven to be difficult to distinguish from both classical and rock categories. In the particular music chosen, classical instrumentation is often used, and fast paced, drum- and base-rich Celtic dance music is often similar to rock. Techno has proven to be easiest to classify. This makes intuitive sense because techno and trance are most different from the other genres.

Varying feature extraction techniques has shown that including velocity and acceleration values improves the performance significantly, while both MFCC and LPC performed about the same.

Three-way classification performed better than 4-way classification. In general, HMM methods seem to work better with few classes that are strongly different.

Varying the number of hidden sates did not lead to dramatic changes in the performance of the system, which confirms the findings of Chai and Varcoe in the domain of digitally sampled music.

I have also demonstrated that the HMM method generalizes across song and artist boundaries, an important property for practically useful systems: the difference between equal and constrained random splitting was insignificant.

7. Future Work

The system presented here can be improved in a number of ways.

First, better feature extraction algorithms can improve the performance dramatically. Optimizing feature extraction techniques such as Distortion Discriminant Analysis proposed by Burges et al may provide a significant improvement for this and other systems (4).

Better packaging of the system as a single executable will make it more useful for practical uses. In order to do this, integration with the HTK library as opposed to the use of separately compiled tools would be beneficial. Some aspects of the library can then be optimized away since the original purpose of HTK was not music classification but speech recognition. Input and conversion utilities would have to be integrated to take MP3 and other popular music formats as input.

Another possible area for investigation is varying the length of the sample considered by the system and the size and frequency of windowing that is performed for feature extraction. It would also be very interesting to correlate music properties such as beat, tempo, timbre, and emphasis with the properties of the classification system.

An exciting possibility is combining several methods of classification, such as neural networks and hidden Markov models to use the strengths of both. While hidden Markov models are in general quicker and simpler, they perform poorly for large numbers of classes with small degrees of differences. HMMs could be employed to initially classify into broad, strongly different categories, and ANNs could then classify finely distributed narrow subcategories. This would simplify the problem complexity for both systems and could lead to better overall results.

8. References

1. Chai, Wei and Barry Vercoe. *Folk Music Classification Using Hidden Markov Models*. Proceedings of International Conference on Artificial Intelligence, June 2001.
2. Chai, Wei. *Melody Retrieval on the Web.* Master thesis. MIT 2001
3. Scott, Paul. *Music Classification using Neural Networks*. EE 373B Project, Spring 2001. Available at http://www.stanford.edu/class/ee373a/musicclassification.pdf.
4. Burges, C.J.C., J.C. Platt, and S. Jana. *Distortion Discriminant Analysis for Audio Fingerprinting*. Microsoft Research Technical Report MSR-TR-2001-116. December, 2001.

5. Rabiner, Lawrence. *A tutorial on hidden Markov models and selected applications in speech recognition*. Proceedings of the IEEE, 77:2(257-286), 1989.
6. Young, Steve, Gunnar Evermann, Dan Kershaw, Gareth Moore, Julian Odell, Dave Ollason, Valtcho Valtchev, Phil Woodland. *The HTK Book* (for HTK Version 3.1). http://htk.eng.cam.edu/. Cambridge University Engineering Department, December 2001.
7. Bengio, Yoshua. *Neural Networks for Speech and sequence Recognition*. International Computer Press, London, 1996.