Music Genre Classification of Audio Signals

Mrugank Alat, Pradeep Kumar Govindaraju, and Sachin Mohla

***Abstract*—Music genre classification is an interesting but challenging task given its inherent subjective nature. Through this project we treat genre classification of raw audio signals as a machine learning problem. We extensively discuss relevant feature extraction necessary for classification and various machine learning techniques applied to it. Based on the dataset selected we discuss the accuracy of classification of each algorithm applied and also give relevant insights. We obtained the best classification accuracy of 66.73% for a 10 genre classification problem using adaboost with Support vector machines as the weak learner**

***Keywords*—Genres, Classification, Music.**

I. Introduction

Music genres are categorical labels created by humans to characterize pieces of music. These characteristics typically are related to the instrumentation, rhythmic structure, and harmonic content of the music[1]. The classification of songs is subjective but it serves the purpose of classifying songs into various categories so that users can retrieve music based on genres they prefer. Study has shown that humans are quite capable of classifying genres themselves but automating this process has numerous applications. Automation will allow easier organization of music online. Music recommendation applications on smartphones and over the internet will learn users taste in music based on genres and will act as dedicated music recommendation systems. Applications like Pandora, Drinkify are popular examples. What powers these applications are machine learning algorithms. Through our project we explore various machine learning techniques that make music genre classification possible and reliable.

The feature extraction for audio signal is not straightforward as in the case of speech signal. Many of the features are still based on extraction models used for speech. Although these do provide a good classification accuracy, they do not essentially capture all the characteristics of an audio signal. Another issue is related to the subjective nature of genres. Genres evolve over time and the variations of songs within a genre are quite random which makes it tough to classify them. Complexity of classification increases with more genres and sub genera involved.

II. Feature Extraction

A. Timbral Texture Features

The timbral features based to represent timbral texture are based on the features mentioned in [1] which make use of speech classification and music-speech discrimination features. The following features are used to represent timbral texture in our experiments.

1.Spectral Centroid: The spectral centroid is defined as the center of gravity of the magnitude spectrum of the STFT

$$C_t = \frac{\sum_{n=1}^{N} M_t[n] * n}{\sum_{n=1}^{N} M_t[n]} \tag{1}$$

where $M_t[n]$ is the magnitude of the Fourier transform at frame t and frequency bin n . The centroid is a measure of spectral shape and higher centroid values correspond to brighter textures with more high frequencies.

2.Spectral Rolloff: The spectral rolloff is defined as the frequency R_t below which 85% of the magnitude distribution is concentrated

$$\sum_{n=1}^{R_t} M_t[n] = 0.85 * \sum_{n=1}^{N} M_t[n] \tag{2}$$

The rolloff is another measure of spectral shape.

3.Spectral Flux: The spectral flux is defined as the squared difference between the normalized magnitudes of successive spectral distributions

$$F_t = \sum_{n=1}^{N} \left(N_t[n] - N_{t-1}[n]\right)^2 \tag{3}$$

where $N_t[n]$ and $N_{t-1}[n]$are the normalized magnitude of the Fourier transform at the current frame t , and the previous frame $t-1$, respectively. The spectral flux is a measure of the amount of local spectral change.

4.Time Domain Zero Crossings:

$$Z_t = \frac{1}{2} \sum_{n=1}^{N} sign(x[n]) - sign(x[n-1]) \tag{4}$$

where the $sign$ function is 1 for positive arguments and 0 for negative arguments and $x[n]$ is the time domain signal for frame t. Time domain zero crossings provide a measure of the noisiness of the signal.

5. Mel-Frequency Cepstral Coefficients: Automatic classification of audio has a long history of originating from the speech recognition[1]. Mel-frequency cepstral coefficients (MFCC) are a set of perceptually motivated features that form an important part of the feature vector description for speech recognition problems. Given the nature of these coefficients it makes good sense to apply these to music genre classification problem as well.

6.Low-Energy Feature: Low energy is the only feature that is based on the texture window rather than the analysis window.

Mrugank Alat, Pradeep Kumar Govindaraju and Sachin Mohala are with the Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA, 93106 USA e-mail: [mrugankvalat, pkg, sachinmohla]@umail.ucsb.edu, .

This project is done under the guidance of Professor Upamanyu Madhow,Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA, 93106 USA e-mail: madhow@ece.ucsb.edu

It is defined as the percentage of analysis windows that have less RMS energy than the average RMS energy across the texture window. As an example, vocal music with silences will have large low-energy value while continuous strings will have small low-energy value.

B. Rythmic Feature

1.Beat Histogram: Most automatic beat detection systems provide a running estimate of the main beat and an estimate of its strength.In order to characterize musical genres more information about the rhythmic content of a piece can be utilized. The beat histogram is used to represent the rhythmic content of a song through feature vectors. The feature set for representing rhythm structure is based on detecting the most salient periodicities of the signal.

Fig.1 shows the flow diagram of beat analysis algorithm. The signal is first decomposed into various sub-bands using filter banks. For our project, the signal was decomposed to 6 (200, 400, 800, 1600, 3200, 6400) subbands. Following this decomposition, the time domain amplitude envelope of each band is extracted separately. The autocorrelation of the resulting sum of the envelopes is computed and the dominant peaks of the autocorrelation function are used to construct the beat histogram.

C. Feature Vector

Keeping these Spectral Centroid,Spectral Rolloff, Spectral Flux, Time Domain Zero Crossings and five MFCC coefficients for each frame in the song we still end up with a huge matrix of coefficients.To counter this, we modeled the entire matrix of coefficients as a multivariate gaussian distribution. Giving equal weightage to each of these features, we obtain a 9 dimensional multivariate gaussian distribution.

So, to represent any song we obtain the mean vector and the standard deviation from the covariance matrix of the multivariate gaussian distribution that we obtained from the previous step. So, the final feature vector consists the following.

- 4 spectral features' mean and covariance
- 5 MFCC coefficients' mean and covariance
- 1 Time domain zero crossings value
- 4 Beat Histogram features

These makes the final feature vector size as 23.

III. Detailed Algorithms

A. Mel Frequency Cepstral Coefficients

Here, we present the steps involved in MFCC feature extraction and explain how they form a relevant part of our feature vector descriptor.

Step 1: Frame the signal into short overlapping frames The first step involves dividing the signal into frames by applying a windowing function at fixed intervals. The signal is usually framed in the range of 20ms - 40ms per frame. The motivation behind this is to model small sections of the signal that are statistically stationary. To obtain accurate statistical estimates we also maintain an overlapping section of about 10-15ms between the frames. Hamming window is typically the choice for windowing function as it removes edge effects.

Step 2: For each frame, calculate the periodogram estimate of the power spectrum This step is motivated by the human cochlea (an organ in the ear) which vibrates at different spots depending on the frequency of the incoming sounds. Depending on the location in the cochlea that vibrates (which wobbles small hairs), different nerves fire informing the brain that certain frequencies are present. The periodogram estimate obtained using FFT of the signal over a given frame performs a similar job for us. It identifies all the frequencies present in the frame.

Step 3: Apply the Mel filter bank to the power spectra, sum the energy in each filter

The cochlea can not discern the difference between two closely spaced frequencies. This effect becomes more pronounced as the frequencies increase. For this reason, we take clumps of periodogram bins and sum them up to get an idea of how much energy exists in various frequency regions. This is performed by the Mel filter bank: the first filter is very narrow and gives an indication of how much energy exists near 0 Hertz. As the frequencies get higher our filters get wider as we become less concerned about variations. We are only interested in roughly how much energy occurs at each spot. The Mel scale tells us exactly how to space the filter banks and how wide to make them.

Step 4: Take the logarithm of all filter bank energies. Once we have the filter bank energies, we take the logarithm of them. This is also motivated by human hearing: we don't hear loudness on a linear scale. Generally to double the perceived volume of a sound we need to put 8 times as much energy into it. This means that large variations in energy may not sound all that different if the sound is loud to begin with.

Step 5: Take the DCT of the log filter bank energies There are two main reasons why this is performed. Because the filter banks are all overlapping, the filter bank energies are quite correlated with each other. The DCT de-correlates the energies and we obtain roughly 26 DCT coefficients. But notice that only 12 of these 26 DCT coefficients are kept. This is because the higher DCT coefficients represent fast changes in the filter bank energies and it turns out that these fast changes actually degrade performance, so we get a small improvement by dropping them.

We found that out of the extracted 12 MFCC coefficients the $2^{nd} - 6^{th}$ MFCC coefficients are the most relevant ones. Our classification results did not vary much by including more MFCC coefficients. So to keep the representation of the song compact we decided to retain only the 2nd - 6th MFCC coefficients. The first one which represents the DC component was also ignored as it does not really capture any genre specific information.

B. Beat Histogram

The following building blocks are used for the beat analysis feature extraction.

1) Envelope Extraction:

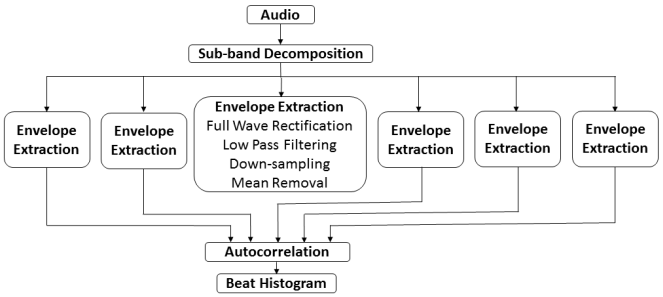

Fig. 1. Beat Histogram Flow

a) Full Wave Rectification: It is applied in order to extract the temporal envelope of the signal rather than the time domain signal itself.

$$y[n] = |x[n]| \tag{5}$$

b) Low-Pass Filtering: A one-pole value with an alpha value of 0.99 is used to smooth the envelope.

$$y[n] = (1 - \alpha)x[n] + \alpha y[n - 1] \tag{6}$$

c) Downsampling: Downsampling the sequence makes the algorithm computationally cheaper without affecting performance because of the large periodicities for beat analysis. In this project, the signal is downsampled by a factor of 16.

d) Mean Removal: It is used to make the signal zero centered.

2) Autocorrelation: The individual envelopes after extraction are added together and the autocorrelation of the resultant is computed. The peaks of the autocorrelation function correspond to the time lags where the signal is most similar to itself.

3) Beat Histogram: The peaks of the autocorrelation function that are in the appropriate range for beat detection are selected and added to a beat histogram (BH).The bins of the histogram correspond to beats-per-minute (bpm) from 40 to 200 bpm. For each peak of the autocorrelation function the peak amplitude is added to the histogram so as to give more weight to stronger peaks in histogram calculation. Based on experimentation, a set of features based on the BH are calculated in order to represent rhythmic content and are shown to be useful for automatic musical genre classification.

- A0, A1: relative amplitude (divided by the sum of amplitudes) of the first, and second histogram peak.
- RA: ratio of the amplitude of the second peak divided by the amplitude of the first peak.
- SUM: overall sum of the histogram (indication of beat strength).

IV. Classification & Results

We used Marsys (Music Analysis Retrieval and Synthesis for Audio Signals) dataset [5]. It consists of 1000 audio tracks each 30 seconds long. It contains 10 genres, each represented by 100 tracks. The tracks are all 22050Hz Mono 16-bit audio files in .au format. We used Libsvm [7] for support vector machines and Weka [6] for neural network and boosting.

TABLE I. Confusion Matrix for 10 genre classification using KNN

	me	cl	hi	po	bl	co	di	ja	re	ro
me	**20**	0	0	0	3	0	3	0	0	4
cl	0	**28**	0	0	0	0	0	1	0	1
hi	0	0	**10**	6	1	0	4	0	7	2
po	0	0	1	**26**	0	1	2	0	0	0
bl	4	0	0	0	**9**	5	1	0	3	8
co	1	2	1	0	5	**14**	1	1	1	4
di	1	0	4	1	2	1	**17**	0	1	3
ja	0	1	1	7	2	5	2	**8**	1	3
re	0	0	4	8	1	4	0	1	**11**	1
ro	6	1	6	0	5	1	2	1	0	**8**

TABLE II. Confusion Matrix for 4 genre classification using KNN

	me	cl	hi	po
me	**28**	0	1	1
cl	1	**29**	0	0
hi	3	0	**21**	6
po	0	0	2	**28**

A. K-Nearest Neighbor (k-NN) Classification

k-NN is perhaps the most straightforward supervised classification technique in machine learning. In simple words, the k-NN achieves the classification task by identifying the nearest neighbors to a query example and using those neighbors to determine the class of the query.k-NN is simple in terms of implementation and the training is fast. Since each song is a 23 dimensional vector we do the exhaustive search with distance metric being the ”euclidian distance”. The only parameter to tweak was the number of neighbors used for identifying the class of the queried test data. The parameters after several tests was set to 10. The training set consists of 70% songs picked from each genre and the test data-set consists of 30% songs from each genre set.

The classification for 10 genres has a total accuracy of 50.33%. Given the 23 dimensional vector representation of each song the classification results are quite satisfactory. Of course with higher dimensional data we can expect some improvement in classification, but this gives us a fair idea of the genres that are being classified correctly and genres that are poorly classified due to inherent vector representation. The k-NN as expected behaves like a weak classifier since our dataset is not very huge and we are trying to classify 10 different genres. k-NN generally needs large training samples to develop a very robust classification model. k-NN is biased to the k value (in our case 10) we choose. So the classification may not be very accurate and will not perform equally on different datasets.

To show better classification results with k-NN we reduced the complexity of problem by restricting the classification

problem to 4 genres - Metal,Classical,Hiphop and Pop. As expected we achieved an accuracy of 88.33%. So clearly k-NN is suitable for smaller number of classes and is not reliable when problem is blown to a 10 class classification problem.

B. K-Means

K-means using distance norm as the cost metric would more or less perform similar to the already described k-NN classification. To build a more suitable unsupervised classification model we reduced our study to the MFCC vector in order to validate this model. Since the MFCC vector (10 dimensional vector) is a representation of multivariate gaussian distribution of each song, we need a better distance metric that can basically compute the divergence of these two distributions. Kullback-Lieber (KL) divergence provides a good estimate of the variation of any two probability distributions. This can be extended to our multivariate gaussian modeling of songs.

$$D(\widehat{f}||\widehat{g})) = \frac{1}{2}[log\frac{|\sum_{\widehat{g}}|}{|\sum_{\widehat{f}}|} + Tr[\sum_{\widehat{g}}^{-1}\sum_{\widehat{f}}^{-1} - d + (\mu_{\widehat{f}} - \mu_{\widehat{g}})] \quad (7)$$

For a univariate gaussian distribution the calculation of KL divergence is straightforward as in eq.(7) but it is not so simple in the case of the multivariate distribution.For the multivariate distribution case eq.(8) provides a good closed form expression.

$$D_{variational}(f||g) = \sum_a \pi_a log\frac{\sum_{a'} \pi_{a'} e^{-D(f_a||f_{a'})}}{\sum_b w_b e^{-D(f_a||g_b)}} \quad (8)$$

The D-variational provides the best closed form solution to our problem. For more details on the exact implementation of the variation approach of estimating the KL divergence of two multivariate GMMs please refer to [3]. Since KL divergence is not symmetric we need to compute the divergence for both D(f||g) and D(g||f). Our final distance metric is defined as in eq. (9).

$$KL = 0.5 * (D(f||g) + D(g||f) \quad (9)$$

We first applied the technique to a smaller data set of 4 genres (metal,classical,hiphop,pop) and reduced the dataset to just the MFCC vectors in order to validate our method and the use of KL divergence as the distance metric. We had an interesting observation - The convergence is quite poor in the case where centroids are initialized randomly. To counter this issue we decided to reduce the data set of 4 genre classes - 389 total songs (11 songs from the given genre set result in invalid MFCC coefficient calculation and hence removed from data set) to 269 songs. The remaining 120 songs were used as a separate dataset for random centroid initialization. Since we deal with a 4 genre clustering problem here we initialize our 4 centroids with 4 songs chosen randomly from the remaining 120 songs of the 4 genres. We see that the algorithm converges well and the resulting centroids give us a good classification.

TABLE III. CONFUSION MATRIX FOR 10 GENRE CLASSIFICATION USING SVM

	me	cl	hi	po	bl	co	di	ja	re	ro
me	**83**	0	1	0	5	0	4	1	0	6
cl	0	**86**	3	0	0	3	0	6	0	1
hi	2	1	**40**	10	1	1	6	0	22	7
po	0	0	1	**75**	0	3	8	7	4	2
bl	9	1	3	0	**64**	6	1	5	2	9
co	1	2	0	7	7	**60**	6	9	2	6
di	3	0	4	13	2	1	**56**	0	6	15
ja	3	8	0	0	5	3	0	**68**	1	12
re	1	0	6	13	3	7	2	2	**61**	5
ro	13	2	4	4	7	11	11	8	3	**37**

The algorithm iteratively executes 30 times given 120 songs of dataset for random initialization and we report the best result based on the combination that gives the least distance metric. This concept can be extended to a larger dataset and larger clustering problem.

The centroid initialization is a critical step for clustering in k means. A random initialization results in poor classification with unreliable results over iterations. By using a systematic initialization where we choose our initial centroid locations as songs chosen randomly from a separate dataset that closely represent the genres we wish to classify, we see that this gives us a better clustering model for the actual dataset that we want to classify.One of the major drawback is that the k means is susceptible to the centroid initialization. Given this inherent problem of k means clustering we restricted our classification results to 4 genres only with just the MFCC vector as the feature vector. Classification accuracy achieved using K means on the 4 genres - 85.275%. This result is quite similar to the classification accuracy achieved using entire 23 vector representation of songs in k-NN.

C. Support Vector Machines & Boosting

We used Sequential minimal Optimization algorithm [4] for training a support vector machine. This implementation normalizes all the features and solves the Quadratic programming problem that arises during the training stage. For handling multi-class classification using support vector machines, we used pair wise classification or also called as one -vs-one classification for each of the class combinations. Each of the one-vs-one classifier will vote for one class. The class which gets the maximum votes will b e considered as the result of the multi class classification.

We used a polynomial kernel with the SVM as shown in eq.10. Polynomial Kernels represent the similarity of vectors in the feature space over polynomials of the original variables allowing learning of non-linear models.

TABLE IV. CONFUSION MATRIX FOR 4 GENRE CLASSIFICATION USING SVM

	me	cl	hi	po
me	**96**	0	3	1
cl	2	**94**	2	1
hi	5	0	**78**	7
po	0	1	4	**94**

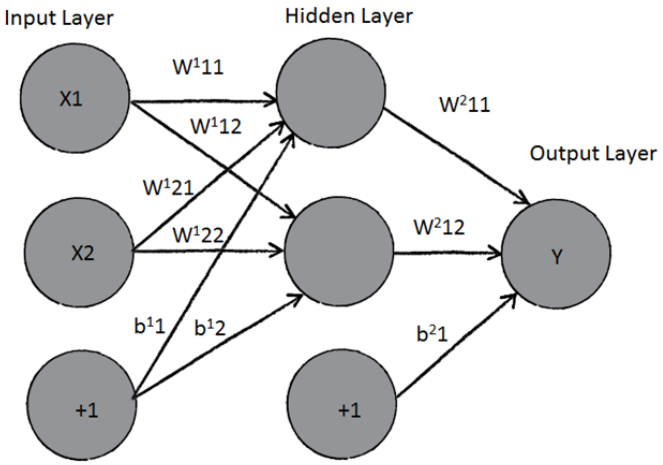

Fig. 2. Basic blocks of Neural Network

$$K(x,y) = (\sum_{i=1}^{n} x_i y_i + c)^2$$
$$= \sum_{i=1}^{n}(x_i^2)(y_i^2) + \sum_{i=2}^{n}\sum_{j=1}^{i-1}(\sqrt{2}x_i x_j)(\sqrt{2}y_i y_j \qquad (10)$$
$$+ \sum_{i=1}^{n}(\sqrt{2c}x_i)(\sqrt{2c}y_i) + c^2$$

The accuracy achieved is 63.7% with a second order polynomial kernel mentioned in 10 and C parameter for SVM as 4.0. The confusion matrix for the same is given in Table III.The same configuration resulted in 93.32% for the 4 classes metal, classical, hip-hop and pop.The corresponding confusion matrix is gien in TableIV.

Since we got around 60% accuracy with SVM, which is greater than 50%, it qualifies for being used as a weak learner for boosting. We experimented adaptive boosting by considering the support vector machine model explained earlier and achieved a classification accuracy of 66.73% for 1000 iterations.

D. Neural Network

The neural network is a very commonly used classifier because of its ability to approximate any polynomial functions. A simple neural network architecture is shown in Fig.2. It has an input layer consisting of 2 nodes and output layer has only one node i.e. it is used for binary classification. It also has one hidden layer with 2 nodes. Xi denotes the input units and Y is the output. The nodes labeled as +1 are the bias unit. The weight $W^L mn$ to denote the parameter (or weight) associated with the connection between unit m in layer L, and unit n in layer L+1. These weights are computed using generic back propagation algorithm which aims to error function.

TABLE V. CONFUSION MATRIX FOR 10 GENRE CLASSIFICATION USING NEURAL NETS

	me	cl	hi	po	bl	co	di	ja	re	ro
me	**85**	1	3	0	4	0	3	2	0	2
cl	1	**75**	0	0	1	2	1	13	2	4
hi	4	0	**45**	6	5	0	6	0	21	3
po	0	2	3	**69**	0	4	9	4	7	2
bl	4	1	4	0	**58**	12	3	6	4	8
co	1	7	1	5	7	**58**	6	4	3	8
di	2	2	11	15	0	4	**55**	0	2	9
ja	1	11	0	4	6	8	0	**63**	1	6
re	1	0	12	7	6	6	7	1	**57**	3
ro	11	3	5	2	10	12	18	5	4	**30**

TABLE VI. CONFUSION MATRIX FOR 4 GENRE CLASSIFICATION USING NEURAL NETS

	me	cl	hi	po
me	**95**	0	4	1
cl	3	**95**	0	1
hi	5	0	**78**	7
po	1	1	5	**93**

Neural network Architecture For this project, the three layered neural network was used. The input layer had 23 nodes and output layer has 10 nodes (one for each genre) as shown in Fig.3. In this figure, the green nodes represents the input nodes; red nodes represents the hidden layer nodes and yellow nodes represent the output nodes. Unipolar sigmoid was used as activation function. For theabove configuration, experiments performed with keeping lamda and alpha constant revealed that for 10 nodes the trade-off between computational time and accuracy was very good. Experimenting with lamda and alpha value with same neural network architecture, we found that for learning rate=0.03 and alpha=0.2 optimum accuracy was obtained.

The overall accuracy obtained was 61.27% for 10 genres. The confusion matrix obtained for the same parameters is shown in Table V

TABLE VII. COMBINATIONS OF FEATURES & CLASSIFICATION RESULTS

	Neural Nets	**SVM**
Beat Histogram (BH) Features (4)	24.37	19.82
MFCC (10)	58.44	54.61
STFT (9)	47.93	47.79
BH and MFCC (14)	58.14	55.40
BH and STFT (13)	51.16	50.05
MFCC and STFT (19)	**60.06**	**62.31**
BH, STFT and MFCC (23)	**61.47**	**63.73**

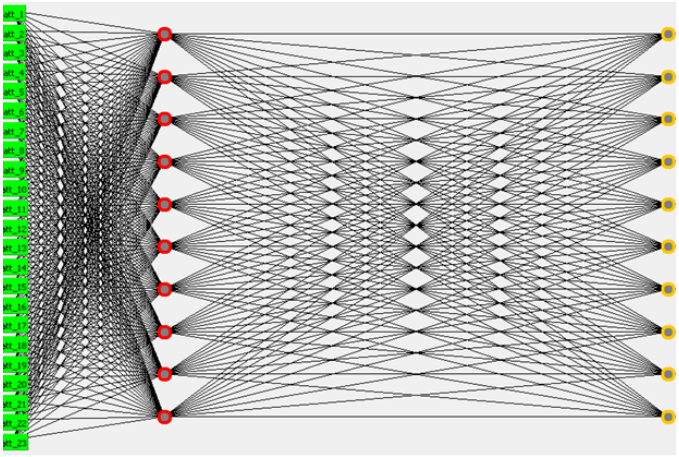

Fig. 3. Neural Network Simulation for 23 dimensional feature

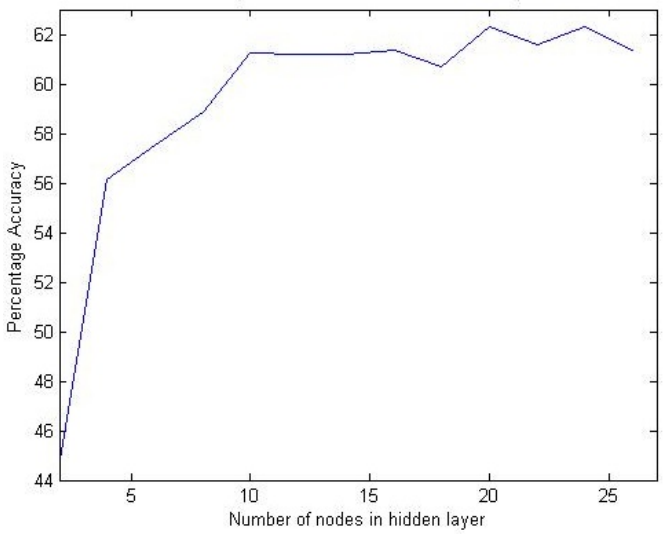

Fig. 4. Plot of accuracy vs. number of hidden layer nodes

V. Conclusion

Based on our results it is quite evident that speech inspired feature extraction provides the best classification accuracy. Features specific to audio signals such as rhythmic variation and beats provide little improvement in the overall accuracy of the classification. Overall we obtained best classification result using adaboost with SVM as weak learner with an accuracy of 66.73% for a 10 genre classification problem. With a smaller genre set of 4 genres we achieve accuracy of up to 93.32%. This is a promising result which points to the fact that despite having a compact representation of feature vectors we can still achieve high classification accuracy.

VI. Future Work

To test the robustness of our feature vectors and classification accuracy a larger dataset is required. Moving forward we expect more work done on the extraction of features better suited to music signals rather than speech. The classification using machine learning can be extended to Convolutional neural networks which depending on the dataset might give more accurate classification accuracy.

References

[1] Tzanetakis, G.; Cook, P., "Musical genre classification of audio signals," Speech and Audio Processing, IEEE Transactions on , vol.10, no.5, pp.293,302, Jul 2002

[2] B. Logan, Mel frequency cepstral coefficients for music modeling, in Proc. Int. Symp. Music Information Retrieval (ISMIR), 2000.

[3] Hershey, J.R.; Olsen, P.A., "Approximating the Kullback Leibler Divergence Between Gaussian Mixture Models," Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on , vol.4, no., pp.IV-317,IV-320, 15-20 April 2007

[4] John C. Platt. 1999. Fast training of support vector machines using sequential minimal optimization. In Advances in kernel methods, Bernhard Schlkopf, Christopher J. C. Burges, and Alexander J. Smola (Eds.). MIT Press, Cambridge, MA, USA 185-208.

[5] MARSYAS Dataset. http://marsyasweb.appspot.com/download/data_sets/

[6] WEKA Data mining software. http://www.cs.waikato.ac.nz/ml/weka/

[7] Libsvm- A Library for Support Vector Machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm/