Project Report for 15781

Classification of Music Genre

Muralidhar Talupur, Suman Nath, Hong Yan

As the demand for multimedia grows, the development of information retrieval systems including information about music is of increasing concern. Radio stations and music TV channels hold archives of millions of music tapes. Gigabytes of music files are also spread over the web. To automate searching and organizing the music files based on their genre is a challenging task. In this report, we present an approach to identifying genre of music file based on their content. We present experimental result that shows that our approach is effective in identifying the genre of the music file with acceptable level of confidence.

1. Introduction

Vast musical databases are currently accessible over computer networks (e.g., the Web), creating a need for sophisticated methods to search and organize these databases. Because music is a multifaceted, multi-dimensional medium, it demands specialized representations, abstractions and processing techniques for effective search that are fundamentally different from those used for other retrieval tasks. The project "Exploiting Style as Retrieval and Classification Mechanism" undertaken by Roger Dannenberg and his group aims to develop hierarchical, stochastic music representations and concomitant storage and retrieval mechanisms that are well suited to music's unique characteristics by exploiting reductionist theories of musical structure and performance (i.e., musical style).

As a part of this whole effort, we decided to explore if it is possible to make prediction about music genre/type based on the content of the audio file. Work done previously shows that the spectro-temporal features are sufficiently rich to allow for coarse classification of music from brief sample of the music. We use two machine learning algorithms: Neural Networks and Learning Vector Quantization. We restrict ourselves to four genres: Classical, Rock, Jazz and Country, although the approach we use can possibly be extended to much larger classification set. This intuitively seems to be easier than recognizing styles of individual composers.

The rest of the report is organized as follows. Section 2 describes our overall approach. Section 3 shows our experimental result. In section 4 we analyze our result and this is followed by a brief conclusion in section 5.

2. The Approach

In this section we describe our overall approach to preprocessing the music file, extracting features and using them to train the networks.

2.1 Preprocessing Data

Given an audio clip, we first preprocess it to get our input data, that will be used both in training and testing phase. The preprocessing steps are as follows.

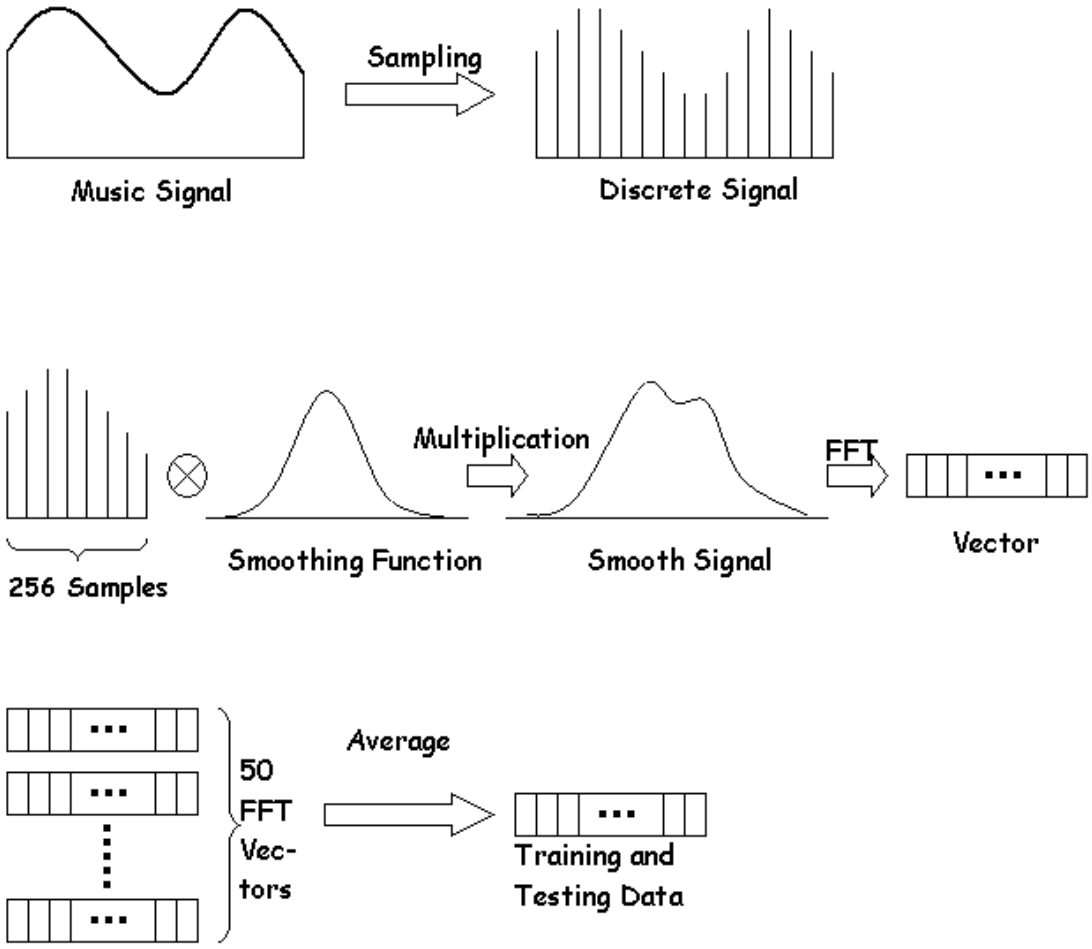


Figure 1: Preprocessing of audio clip to get the input data

1. Sample the audio signal at the rate of 28 KHz. At this sampling rate, signals over 14 KHz are ignored. As humans cannot hear signals over 20 KHz, those frequencies need not considered. And at sampling rate as low as 8 KHz, as in telephone facility, different genres of music can still be differentiated. So we chose some sampling rate that lies between these two limits.
2. Apply Fourier Transform to 256 samples, after multiplying them with Laplace smoothing function. The number of samples (256) was chosen such that it was large enough to capture the essential features of the music.
3. Get the amplitude spectrum from the result of the Fourier Transform at the first 128 frequencies. Thus we have a vector of length 128.

4. Then we slide the window of samples, take the next 256 samples, repeat the same process and so on. We do this till the end of the audio clip.
5. Now we make groups of 50 consecutive vectors and take average of each group. Such vectors (each of length 128) are used as inputs to our learning algorithms.

Each of these input vectors is used independently of other vectors, though they may have come from the same audio clip. In our experiments, we have taken around 400 vectors from each audio clip. Given the sampling rate, 400 vectors correspond to about 3 minutes of music and each vector corresponds to 0.42 second of music. While testing a sample of music, we test each of the 400 vectors and then take the plurality classification, that is the largest group, as the genre of the music. By this method, we have been able to classify all the audio clips we used for our testing phase correctly. Although plurality classification is always correct, individual vectors do not get always classified correctly. From now on, we will use the term "likelihood of genre X" to mean the percentage of vectors classified as genre X, and the term "confidence" to express the likelihood of the majority genre.

Apart from averaging 50 vectors to obtain an input vectors, we have tried out three other features.

a. Take the difference of consecutive vectors and the average these difference vectors to obtain the input vectors. This method did not yield any good result though.
b. Take the average of 50 vectors and then take the difference of the vectors so obtained to get the input vectors. This method gave results that were better than the previous approach, but not as good as those of averaging method.
c. We also tried out using envelope of the audio clip. The highest frequency component in the envelope we used was 440Hz. We used a Lisp program to find the envelope of the audio clip and then got input vectors using the same approach we used in the averaging method. This method too does not perform well. This method was preliminary tried out to see if it is able to increase the confidence with which Rock and Country were classified. Initially 'average of FFT approach' was not giving good confidence in classification between Rock and Country using Neural Network, so we tried this envelope method.

The results for all the approaches will be given later. To obtain the Fourier transform of audio sample, we used the program named "Nyquist".

2.2 Building the Networks

2.2.1 The neural network

The structure of the neural network that we used is as follows:

1. Four output units, one for each genre.
2. One hidden layer, having 30 hidden units
3. Input layer, having 128 units.

The number of hidden units was chosen experimentally. We found that, if the number of units in the hidden layer is increased above 30, the learning phase becomes slow, although the testing/training accuracy does not improve much. The number of input units was chosen arbitrarily, we have done some experiments (described later in this report) that, sort of, justifies the number.

2.2.2 The Linear Vector Quatization (LVQ) Networks

Learning Vector Quantization is a neural network based method to find a good set of reference vectors to be stored as a nearest neighbor classifier's reference set. It is often used for speech recognition and it has high performance. So, we decided to try genre classification using LVQ towards the end of the project.

An LVQ network contains a Kohonen layer which learns and performs the classification. LVQ assigns equal number of Processing Elements (PEs) for each class of the Kohonen.

The basic LVQ trains and then uses Kohonen layer as follows:

1. In the training mode, the distance of a training vector to each PE is computed and the nearest PE is declared to be the winner.
2. If the winner PE is in the class of the training vector, it is moved toward the training vector.
3. If the wining PE is not in the class of the training vector, it is moved away from the training vector. This is referred to as repulsion.
4. During the training process, the PEs assigned to a class migrate to the region associated with their class.
5. In the classification mode, the distance of an input vector to each PE is computed and again the nearest PE is declared to be the winner. The input vector is then assigned to the class of that PE.

We here consider a variant of LVQ, which is called LVQ2, since it performs better in the situation where, when the winning PE gives wrong result, the second best PE is in the right class. While experimenting with basic LVQ, we found that whenever we make a wrong prediction, the 2^{nd} best prediction is in fact the correct classification. So we decided to improve our prediction by using LVQ2. The parameters of the LVQ2 network we used in the experiment are as follows:

- Number of inputs: 128
- Number of outputs: 4
- Number of PEs in Kohonen layer: 100
- Number of iteration: for LVQ1: 500,000, for LVQ2: 50,000

The first two parameters (input and output) are based on our problem specification. The rest are based on experiment. We tried for several possible alternatives and selected the best value.

2.3 Training Phase

As discussed above, from each audio clip, we obtain about 400 vectors and each of them is used to train the network independently. We used 24 audio clips of each genre, that 96 in all. The number of training vectors was approximately 38,000.

2.4 Testing Phase

We used 40 audio clips, 10 per each genre, to test the network. All of the clips were classified correctly with varying confidence. The result for various features used is given below and a comparison is made.

3 Experimental Results

In this section we present the result of various experiments we have done. Almost all the experiments described here were done on Neural Networks. We considered LVQ only during the last stages of the project. To train LVQ network, we used the features that performed the best in the case of Neural Networks. LVQ better than Neural networks in most of the cases.

3.1 Selecting the best feature

As we discussed before, we tried four different features to use in the whole experiment: average of the FFT vectors, average of differences of successive FFT vectors, difference of averages of successive FFT vectors, average of envelope of the spectrum. The result of trying these different features is shown in following graph.

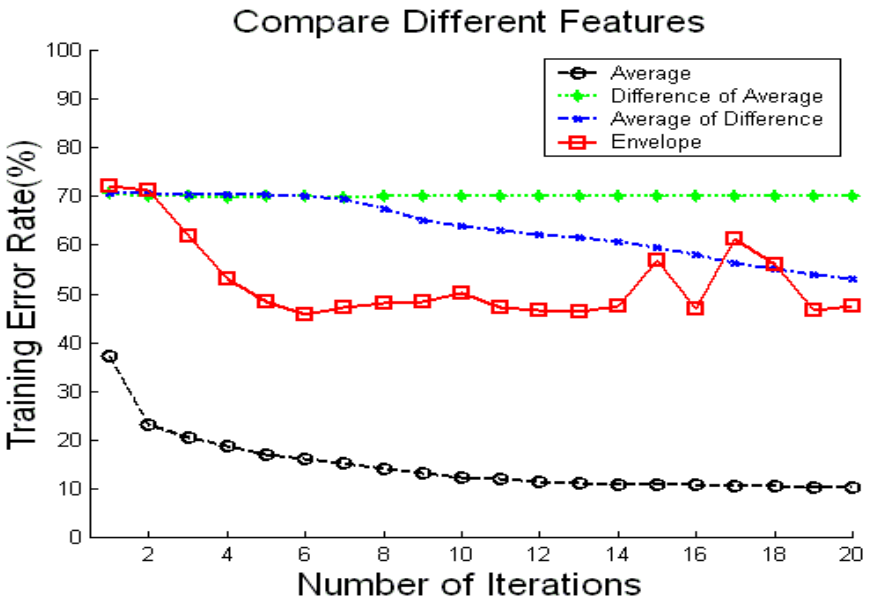


Figure 2: Comparing effectiveness of different features.

As can be seen from the above statistics, the simple averaging approach outperforms the other approaches. In fact, in those cases the training error is more than 50%, which is bad.

So, we used simple averaging approach throughout all the experiments. From now on, we will talk only about the "average of FFTs" approach only, since the other methods are way behind this method.

3.2 Fixing number of iteration

We then concentrated on the question of how many iterations of each learning algorithm needs to go through to get acceptable performance. The result is shown in figure 3.

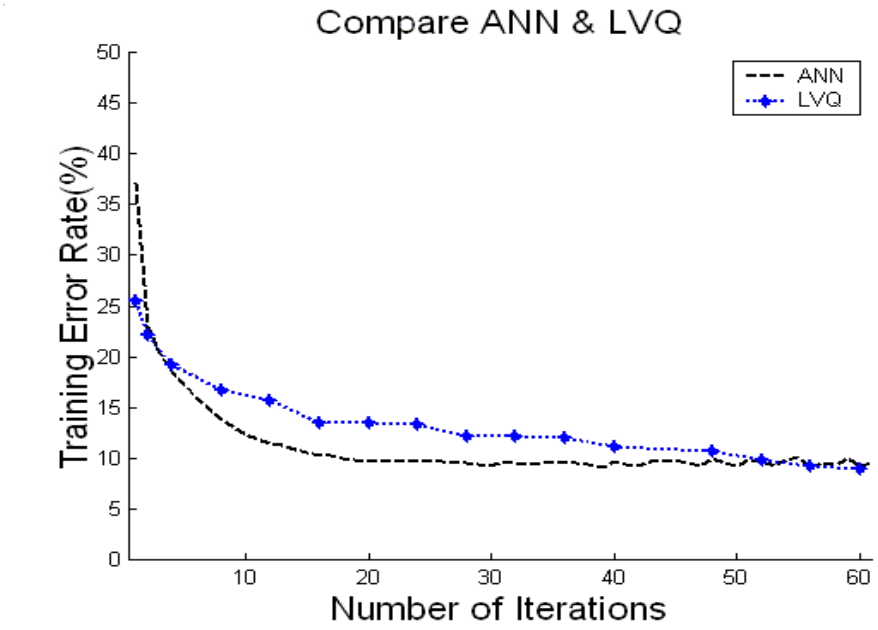


Figure 3: Number of iterations (×10,000) and training error

The graph shows, to have less than 10% training error, Neural Network needs about 200,000 iterations and LVQ needs around 550,000 iterations. We used this value for number of iterations for rest of the experiments.

3.3 Testing Error

We trained Neural network and LVQ network, using the averaging of FFT feature and reasonable number of iterations. We used approximately 38,000 input data to train the networks, and then tested their performance on separate set of data. Both the networks were able to classify each of the 3-minute audio clips, with different degree of confidence. Recall that *confidence* is the fraction of FFTs classified correctly. A number of FFT data are generated from an audio clip, and majority of the classifications of FFTs is taken as the classification of the audio clip. So, in spite of correct classification, the confidence may be less than 100%, when some of the FFTs are classified incorrectly. Figure 4 shows the result. Here, by the term testing error rate, we mean the percentage of FFTs misclassified for each genre. So, the quantity (100 – testing error rate) expresses classification confidence for that genre.

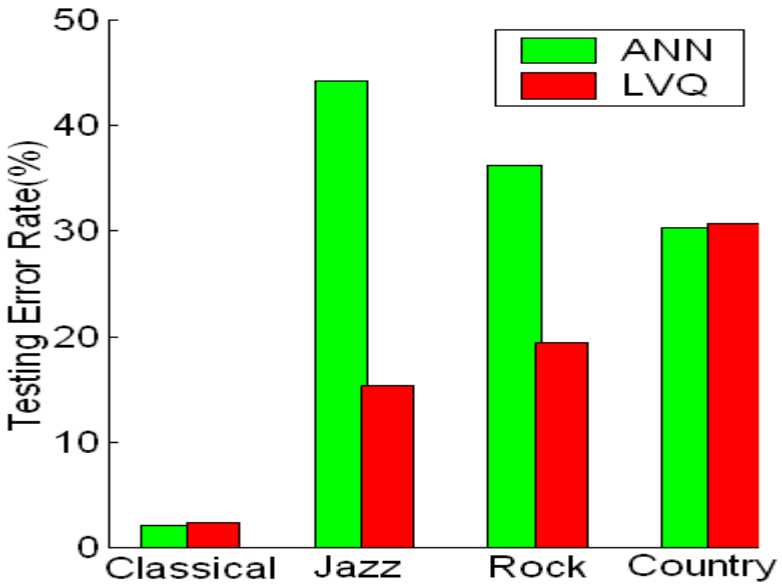


Figure 4: Classification error in different genre

The graph shows that, in most of the case, LVQ network performs better than neural network and its confidence is always more than 70%. It is also found that, classical music is the easiest to classify. LVQ network has the most difficulty in classifying country music, while neural network has the most difficulty in classifying Jazz music.

LVQ algorithm finds an optimal set of reference vectors for classification purposes. During learning, reference vectors are shifted into an optimal position, whereas during recall a nearest neighbor classification technique is used. Since LVQ performs so well in classifying music into different genres from the 128-element vectors described previously, it can be inferred that for the four genres, the vectors obtained form fairly well defined clusters.

3.4 Training set size

Following experiment shows how number of training FFTs affects the confidence of the prediction in the case of Neural Networks. We varied the number of FFTs used for training and found the testing error. The graph below shows the result.

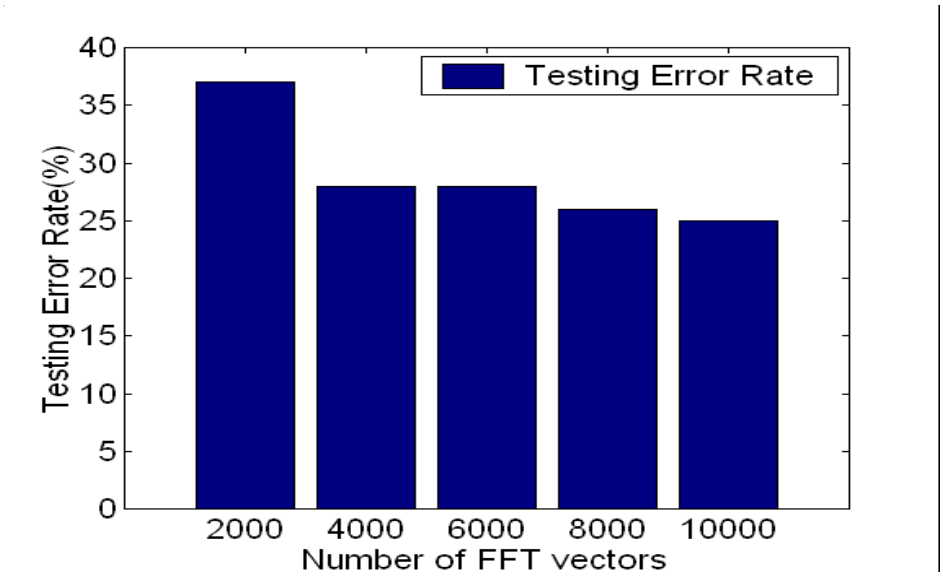


Figure 5: Effect of training size on Testing error

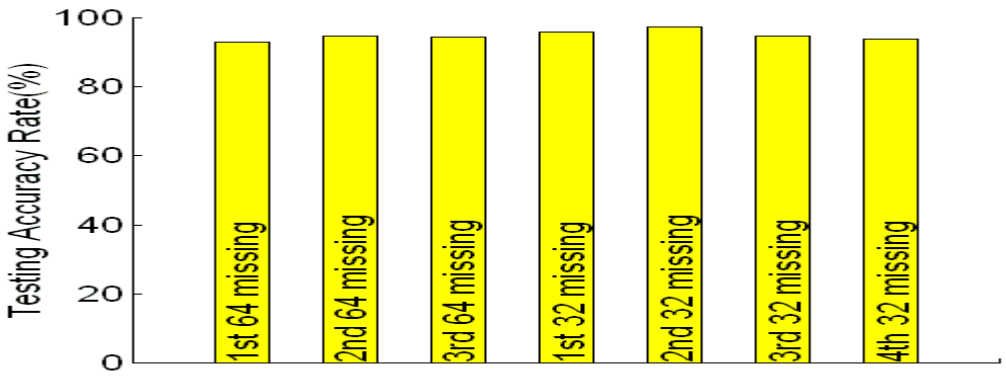
When only 2000 FFTs are used (which is equivalent to 40 input vectors) the testing error (i.e. 100 - confidence of prediction) is around 35%, for 10000 FFTs it is around 25% (this testing is over all genres).

4 Sensitivity Analysis

Sensitivity Analysis was carried out to see which part of 128-long input vector contributes maximum in prediction (note that, entries in the FFT vectors are sorted in ascending order of the frequency). To see which frequency FFT data contributes most in classification, we did the following experiments:

1. We removed the first 32 elements of each of the base (128-long) input vector and used the new 96-long vector as input in the experiments. We did the same experiment by removing next 32 elements, then next 32 elements, so on.
2. We repeated the above step by removing 64 elements at a time from the base input vector. We removed the front 64, then the middle 64 and finally the last 64 elements.

It can be seen from the data that different genre have different regions of “importance” in the 128 element input vector. And also note from the above data that some subset of 128 elements lead to better classification for some genres. For example, for Jazz, dropping any 64 elements leads to much higher classification accuracy. In the following section we try to identify best subset of 128 elements for each genre based on the experimental results.



.

Figure 6: Comparing confidence of recognizing classical music with different parts of features missing.

Classical is the easiest to recognize among all the genres we considered. The confidence always remains over 90% for all the different subsets we have considered. It can be seen from that any 64 elements of the base input vectors can be used to make good prediction and the classification is best(~98%) when all the elements of the vector are used. But the other three genres have widely varying confidences.

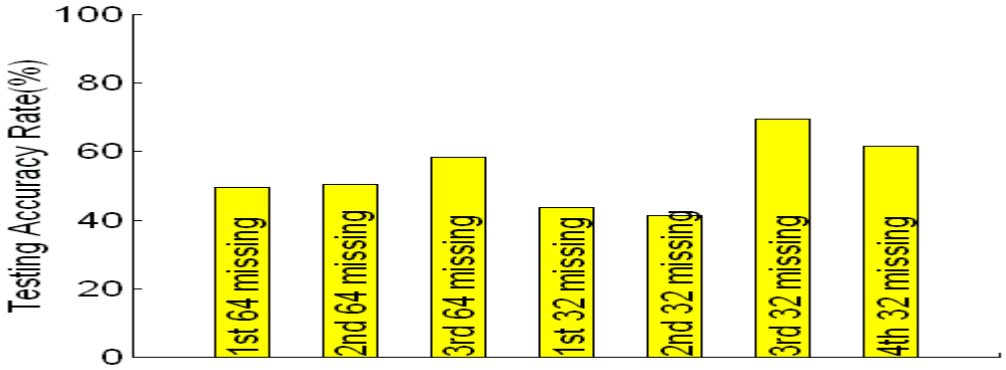


Figure 7: Comparing confidence of recognizing country music with different parts of features missing.

For the Country music when all the 128 elements of the input vector are used the classification confidence is 63% and this does not improve much when different subsets of element are considered. When any 64 elements are left put, the confidence of prediction falls down to below 60%. But when the elements indexed from 65 to 96 are left out, the confidence increases to 70%, an increase of 7%, which is not much considering that this increase over only 10 songs. This is the only case when the confidence increases, so the best subset for Country would be elements indexed 1 through 64 and those indexed 97 through 128. Among these elements, the order of importance is as follows: elements indexed 33 through 64 being the most important (when these are dropped the confidence drops to 41%) followed by elements indexed 1 through 32(when these are dropped the confidence drops to 43%) followed by elements indexed 97 through 128.

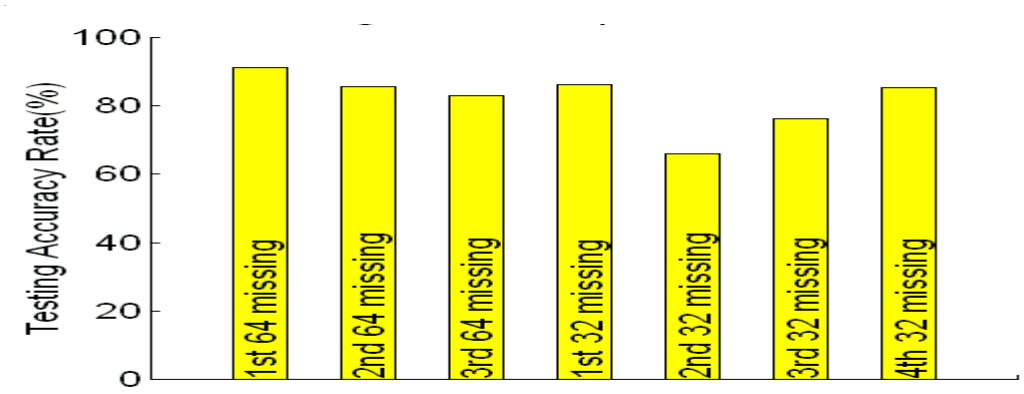


Figure 8: Comparing confidence of recognizing jazz music with different parts of features missing.

For Jazz, the learning is much better when a subset of 128-elements is used. The confidence of prediction when all the 128 elements are used is 56% and when any subset of these 128 elements is used the confidence increases by at least 10%, the maximum increase being 35%, which significant even though this increase is over 10 songs. When the first 64 elements are dropped the confidence of prediction increases to 91%, an increase of 35%. In the cases when the middle 64 and the last 64 are dropped the confidence increases to around 85%. So the last 64 elements will give the best prediction for Jazz. Interesting thing here is when additional elements are added to this subset the confidence drops (when elements indexed 1 through 32 are added the confidence drops to 66%, a drop of 25%). In fact subsets having 64 elements perform better than subsets having 96 elements. When additional elements are added to any of the three 64 element subset the performance drops as can be seen from the above data (the only exception being the case when elements 65 through 97 (confidence increases to 85%) are added to the subset of elements 1 through 64 (the confidence for this subset is 83%).

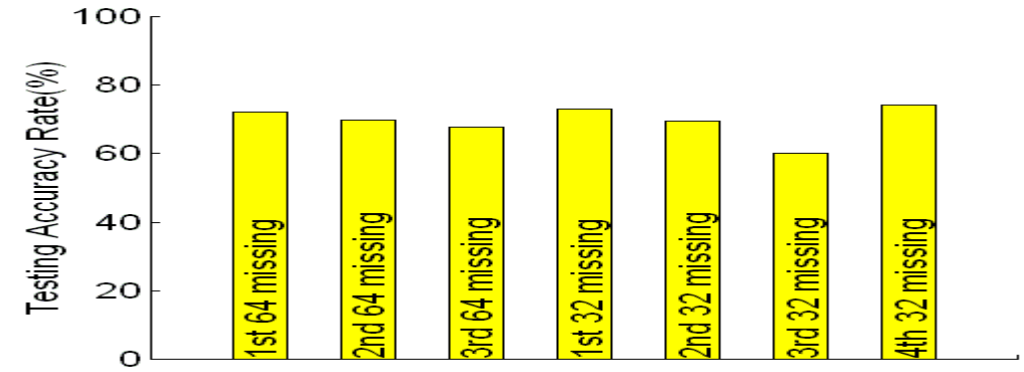


Figure 9: Comparing confidence of recognizing rock music with different parts of features missing.

For Rock, the performance does not vary much with subset of features used (it hovers around 70% always). The elements from 65-96 seem to be of more importance as the confidence drops to 60% from 70% when these elements are dropped (when other elements are dropped the confidence either increases or stays the same).

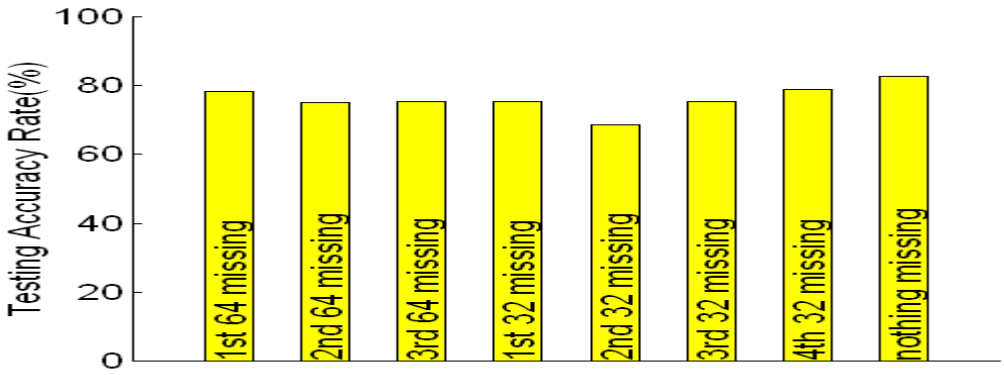


Figure 10: Comparing confidence of recognizing all genres with different parts of features missing.

Across all the genres, the subset of elements indexed 1 through 96 gives the best confidence followed by subset of elements indexed 1 through 64. In fact in al most all the case the subsets give better average confidence than 128 elements set. The only case where the confidence drops is when elements indexed 33 through 64 are dropped. So these subset of elements seem to be the most important in deciding the genre of the music. Dropping of elements indexed 97 through 128 gives an increase of 7.5% per genre, which is the highest among all subsets of features considered. So this subset of features is probably the misleading or difficult to learn.

5. Conclusion

From results described above it can be concluded that a coarse classification of music into four genres can be done easily by machines. And the confidence with which it is done is also significant. The work described in this report can be extended to include further genres like Techno etc. Another direction would be to classify music by artist/composer. This we believe will be more difficult than classifying into different genres; an analysis of the kind described in the report may not be enough to differentiate between different artists/composers.

To build on the work in this report, sensitivity analysis can be carried out at a finer level. The sensitivity analysis that we have done gives us only a coarse picture of the importance of each of the 128 element vectors. It would also be interesting to see how the classification accuracy changes with the sampling frequency (which in our experiments was 28kHz and also with the window size (256 samples in our experiments).

References

[1] Hörnel D., and Frank Olbrich, Comprative Style Analysis with Neural Networks. *ICMC Proceedings* 1999, pp. 433-436.

[2] Perrott, D., and R. Gjerdingen, Scanning the dial: An exploration of factors in the identification of musical style, 1999: oral presentation at The 1999 Conferences of the Society for Music Perception and Cognition.

[3] Poechmueller, W.; Glesner, M.; Juergs, H, Is LVQ Really Good for Classification?-An Interesting Alternative, *Neural Networks, 1993., IEEE International Conference on*, 1993 Page(s): 1207 -1212 vol.3