Producing Self Tuned Music Using Machine Learning Tools

A THESIS SUBMITTED TO BRAC UNIVERSITY IN

PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND

ENGINEERING

Supervised by

Moin Mostakim

Co-supervised by

Md. Shamsul Kaonain

Submitted by

Akifa Tasneem, Student ID: 13101192

Mostofa Saif Reza, Student ID: 12201106

Navid Anindya, Student ID: 13101233

Department of Computer Science and Engineering

BRAC University

Dhaka, Bangladesh

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

DECLARATION

We hereby solemnly declare that the research work embodying the results reported in this thesis entitled **"Producing Self Tuned Music Using Machine Learning Tools"** submitted by Akifa Tasneem, Mostofa Saif Reza and Navid Anindya has been carried out under supervision of Moin Mostakim, Lecturer of Department of Computer Science and Engineering and co-supervision of Md. Shamsul Kaonain, Lecturer of Computer Science and Engineering department at BRAC University, Dhaka. It is further declared that the research work presented here is original and has not been submitted to any other institution for any degree or diploma.

Certified by

Moin Mostakim	**Md. Shamsul Kaonain**
Lecturer	Lecturer
Department of Computer	Department of Computer
Science and Engineering,	Science and Engineering,
BRAC University, Dhaka	BRAC University, Dhaka

Signature of Authors

Akifa Tasneem (13101192)	**Mostofa Saif Reza (12201106)**	**Navid Anindya(13101233)**

ABSTRACT

Automatic music genre classification is one of the important tasks for the Music Information Retrieval (MIR). With the development of the knowledge on Machine Learning, researchers have implemented different methods to implement automatic music genre classification. In this paper, we have done secondary research on various papers that have used different mechanisms to achieve results for genre detection. We will check for the performance of genre detection using kNN (k-Nearest Neighbor Classifier) Classifier and the SVM (Support Vector Machine) classifier. We used MATLAB and specific toolboxes made by MIRLab such as Machine Learning Toolbox and Speech and Audio Processing Toolbox and used different kinds of features for classifiers such as MFCC (Mel-frequency cepstral coefficients). We will use the GTZAN dataset to do the classification using the classifiers. This goal is to see which classifier algorithm on MFCC features performs optimally on the GTZAN dataset. We have used various papers as references on this topic.

ACKNOWLEDGEMENT

We would like to show our gratitude to the Almighty for giving us the strength and patience to complete the work.

This work would not have been possible without the enormous help and support of our Supervisor Moin Mostakim, who believed and was there always for us. We are fortunate to work under his supervision.

We are thankful to our co advisor Md. Shamsul Kaonain, for his continuous support and encouragement for our work. We are fortunate to get his remarks for our paper.

We are thankful to our companions, Sayed Erfan Arefin providing testing equipment and helpful cooperation for which we can successfully complete this work.

Last but not the least, we would like to thank our family members for their blessing, inspiration, encouragement and moral support during this time period. Their patience motivation and inspiration has helped us a lot to achieve our goal.

Chapter 1

INTRODUCTION

1.1 INTRODUCTION

Genres can be defined as a category or style to distinguish from music form and musical style. Instrumentation, rhythmic structure and pitch content are the keys which differ each genres from other. Genres such as classic, jazz, pop, metal etc. There are multiple methodologies to detect the music genre. However, music genre classification has been a challenging task in the field of music information retrieval (MIR). Music genres are hard to systematically and consistently describe due to their inherent subjective nature.

In existing methods, the process of genre classification is composed of two steps: in the first step, relevant features (that represent the instrumentation, rhythmic structure, pitch content, etc.) are extracted from the signal, and a vector of features (feature-vector) is built. In the second step, a classification algorithm is applied on the feature-vector, such as k-nearest neighbor (kNN) or Gaussian mixture model (GMM), support vector machines (SVM) or neural networks, boosting techniques etc.

In this paper, we will classify ten genres: rock, Reggae, pop, metal, jazz, hip-hop, disco, country, classical and blues. To classify that we will be investigating various machine learning algorithms including multi-class SVM and kNN classifier. To characterize our data we relied on Mel-Frequency Cepstral Coefficients (MFCC) and other features using the MIRToolbox in

MATLAB. It is a library that has functions built in for various audio feature extraction that can be used in classifiers for classification. Then machine learning algorithms were applied on the extracted features. [1]

Later, we will suggest to use the extracted data from genre detection to classify various drum beats that will be generated based on the audio signal. It is the plan for our future work.

1.2 MOTIVATION

Audio feature extraction and calculating datasets based on audio files is a relatively unsaturated topic in the world of computing. Based on limited resources and libraries available, researchers have tried to face this challenge of processing audio features in computing. Voice, audio and music recognition are slowly coming into light because of these researches. But not much have still being done on this topic.

In a world where people are using voice search and commanding computers and devices using voice input, where millions of songs and music is available on the internet where people try to acquire them and listen to them actively, audio based researches are not being done in that speed. People want to know which music belongs to who and so that music services such as Spotify or iTunes can give those suggestions based on what they usually listen to. Automatic genre detection is a field that is being researched but still a lot is not ventured. This can help people detect genre of a music and people can have music suggestions in front of them automatically. Also this feature can be also used in one of our future work, producing music based on audio input.

This process of automatic genre detection has similar structures to extracting audio features that can be also used to help in researches based on speech recognition and music production.

Machine learning is not yet being used properly in audio based researches and we wanted to help the field to gather some data on music genre detection based on a particular dataset and also open up ideas on more uses of the genre detection process.

1.3 OBJECTIVES

- Extract various audio features using MIRToolbox to determine which features are useful in the GTZAN dataset to classify using SVM and kNN Classifiers
- Use kNN (k-Nearest Neighbor) classifier to take a comparison dataset to check if it performs better than SVM (Support Vector Machine) Classifier.
- Ten genres (rock, reggae, pop, metal, jazz, hip-hop, disco, country, classical, blues) will be used to check for the optimal performance of the classifiers compared with the GTZAN dataset
- Usage of automatic genre detection in future work of producing music

1.4 THESIS OUTLINE

The rest of our paper contains as follows:

Chapter 2 is the background study that covers literature review.

Chapter 3 is where we have proposed a structure for automatic genre detection using MATLAB.

Chapter 4 is about the selected classifiers and how they are implemented here.

Chapter 5 is where the results are compared and shown.

Chapter 6 contains the conclusion and the discussion about the future challenges

Chapter 2

LITERATURE REVIEW

In our paper, we are using various machine learning algorithms, including multi-class SVM, and kNN to classify 10 genres and check with the dataset from the calculated data. We relied on Mel-frequency Cepstral Coefficients (MFCC) and other features to characterize our data as recommended by previous work in this field [3]. We then applied the classifiers using the features extracted to find our result. We used MIRToolbox [4] in MATLAB to figure out and used MATLAB libraries for the classifiers to detect genres.

Currently, there are many different music genre datasets on the Internet, such as Million Song Dataset [5], Latin Music Database, Last.fm Dataset [6], GTZAN Genre Collection [7] and etc. Each datasets may have different information and format. For example, Million Song Dataset include audio analysis features, track, artist, release and so on. Other datasets may contain other music related information.

Just a quick note about the dataset, which we will refer to later from time to time. Since we wanted to have some comparable results, the selection of dataset was based on the aforementioned paper. The authors use the well-known in the MIR community GTZAN dataset. Even though the dataset is considered 'standard' in music genre classification problem, it has a few flaws, which quite significantly impact the interpretation of the results of any model trained on the GTZAN. In fact, there is even a paper describing in details all the cons of the GTZAN dataset [2]. Some of the problems described there are misclassifications, distortions and duplicates in the data.

Another problem with GTZAN is related not to its construction itself, but the fact that it was made in 2000. Though some may argue, it is quite commonly accepted truth, that during the last 16 years, there actually was some contribution to the musical world overall, particularly to the electronic music. GTZAN doesn"t reflect this contribution - in fact it doesn"t recognize electronic music as a genre at all, which may seem pretty odd nowadays [3].

In the Music Information Retrieval field, MIRtoolbox [4], which developed by Olivier Lartillot, Petri Toiviainen and Tuomas Eerola is a powerful MATLAB toolbox which integrate a set of functions to extract musical features from the audio waveform efficiently. This toolbox contains functions to extract music features using MFCC, produce graphs of the audio files and dataset. This also contains libraries to classify the gathered MFCC dataset and use various classifiers on it and check against the original dataset. This has made our work easier to compare and calculate which performs better to be used in our future work.

For classifiers, we will check by normalizing the dataset as it creates a binary values of data to be classified easily. We are going to use two different classifiers: the k-Nearest Neighbor (kNN) classifier and the Support Vector Machine (SVM) classifier. These classifiers will classify on the dataset that will be given after the MFCC feature extraction. The better performing classifier will be used in our future work of producing the music.

Chapter 3

SYSTEM STRUCTURE

We are going to go through a particular flow of processes that will lead up to the detected genres from the audio input. The processes will start with using the dataset to extract feature and create a classification of the dataset. Then we will give an input to check the genre for that particular input from the classification. We are using MATLAB.

3.1 PREPROCESSING

Before we start, we will add the necessary toolboxes to the search path of MATLAB. Note that this preprocessing is simply a way of starting the process of using the necessary library used, rather than a preprocessing algorithm that is used in a dataset. This is simply a part of using machine learning algorithms in our paper.

```
addpath path/users/jang/matlab/toolbox/utility
addpath path/users/jang/matlab/toolbox/sap
%For speech and audio processing
addpath path/users/jang/matlab/toolbox/machineLearning
addpath
path/users/jang/matlab/toolbox/machineLearning/externalTool/lib
svm-3.21/matlab     %For using SVM classifier
```

Figure 1: Code to add path for MIRToolbox in MATLAB

These codes are part of the library of MIRToolbox which comes with various support libraries like “sap” or Speech and Audio Processing that will help process the audio. We will also use SVM classifier and kNN classifier directly from the given MIRToolbox Library. This will help check us data for comparison with the GTZAN dataset.

3.2 DATASET COLLECTION

First of all, we shall collect all the audio files from the corpus directory. Note that:

- The audio files have extensions of "au".
- These files have been organized for easy parsing, with a subfolder for each class.

Our dataset here is GTZAN dataset from which we will collect 1000 files as audio samples. The samples are sorted in 10 different genres.

3.3 FEATURE EXTRACTION

For each audio, we need to extract the corresponding feature vector for classification. A function called mgcFeaExtract() on one file can be used to extract all features dimensions and the graph depicting the features of the dataset. Features that we will need will be chosen accordingly. For example, we will need to have MFCC as part of the feature. We also need to put all the dataset into a single variable "ds" which is easier for further processing, including classifier construction and evaluation.

Note that if feature extraction is lengthy, we can simply load ds.mat which we will save using our code. We will take MFCC as a feature, where they are based on MFCC's statistics, including mean, standard deviation (std), min and max along each dimension. Since MFCC has 39 dimensions, the extracted file-based features has 156 (39*4) dimensions. The MFCC statistics has 4 values: mean, std, min and max so it will be multiplied with the dimension of MFCC to get the total dimension value of MFCC. Using MATLAB, a graph can be plotted to see the result of dataset.

3.4 TESTING DIMENSION REDUCTION

The feature vector file that is generated (ds.mat) has a huge dimension for MFCC (156 as stated above). We need to reduce this for the classifier to perform faster to figure out genre of a song. A method called PCA (Principal Component Analysis) [8] will be used to reduce the dimension of the generated vector file. But we experimented and checked that PCA is not optimized here if we use MFCC as the feature extraction as using PCA will reduce the accuracy of the classification result greatly. So we will **skip the dimension reduction** using PCA and **use MFCC only** as part of the feature extraction using 156 dimensions.

3.5 CLASSIFICATION

We will be checking the dataset after feature extraction using MFCC with two classifiers. One is known as the kNN classifier and another is the SVM classifier.

The kNN classifier checks the nearest data to a constant „k"and checks the dataset to see which data matches k the most. Then it classifies based on the nearest most frequent data and does the classification. These classified data are called classes. Each classified data has a different k value and is compared and checked which match or nearly matches k. A good thing about this classifier is it is very fast and does not really need to train that much to get a result. Fully trained kNN and simply checked kNN yields the same result. We will check if it yields and optimum result. kNN classification is *lazy*, which means it does not check more intricate values inside the dataset that correlates the dataset internally and really does not check the inner parts of the dataset, rather it only matches values with k.

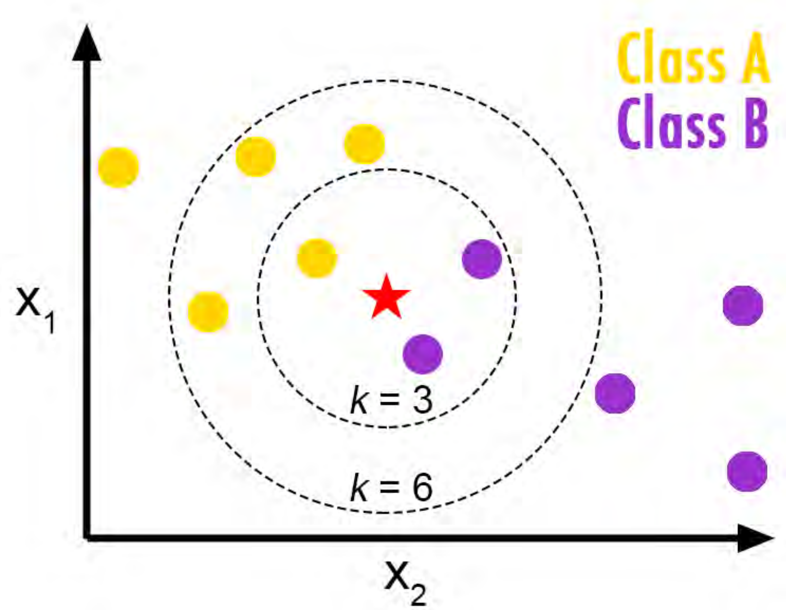

Figure 2: Sample kNN graph

A simple kNN classifier, as stated in above figure, will classify with a constant k with a value and check the nearest data to that value and classify. The k valued classified data will be known as classes.

After kNN we will check the dataset with the SVM classifier. This classifier is more complex and divides dataset using a hyper plane and gathers similar data and classifies them. Each classified data using SVM is known as a vector. These vectors are divided using a line that is known as the hyper plane. This hyper plane divides that divides the data into vectors with a maximum distance from each other. These vectors, comparing to the kNN classifier classes, are the classified data. SVM thoroughly checks the entire dataset and even the internal intricate values.

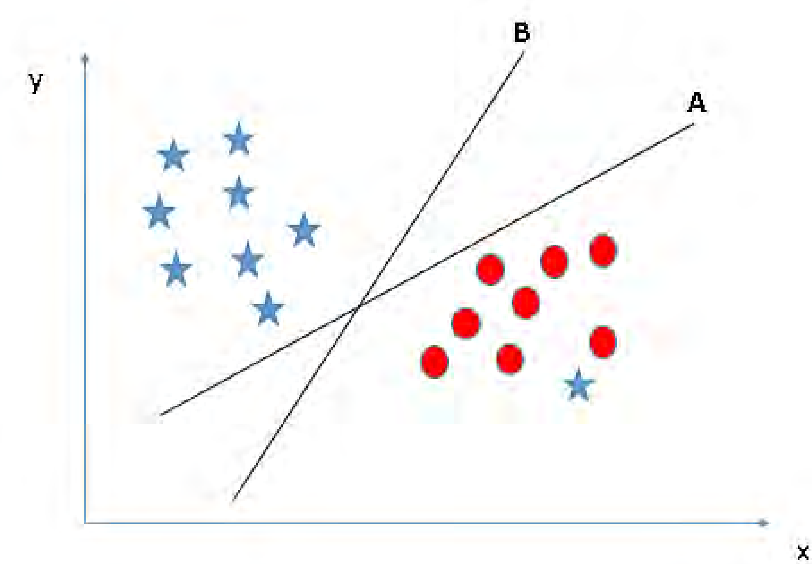

Figure 3: Sample SVM graph

The code will use a MATLAB file called mgcOptSet.m that will put all genre classifier options in a single file. This will make it easier to change options. In the code, the input will be normalized, then classified. We can plot a result chart from it.

The result will be shown as a confusion matrix that will compare the classified dataset with the original dataset and will check how many songs were correctly recognized as that particular genre.

After using kNN, we have tried using SVM to find out if genre detection is better and if is, we will use it as data as part of the later work that will generate drum beat.

Chapter 4

CLASSIFIERS AND IMPLEMENTATION

We will familiarize ourselves with the classifiers a bit before going to the implementation.

4.1 CLASSIFIERS

We are using two classifiers to classify the dataset. One of them is called k-Nearest Neighbor and another one is called SVM or support vector machine.

The k-nearest neighbor algorithm (k-NN) is a non-parametric method used for classification and regression. In *k-NN classification*, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If $k = 1$, then the object is simply assigned to the class of that single nearest neighbor.

In Support Vector Machine, given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

We are going to use both of them and check if they can properly classify and recognize genres from the given dataset. We are using MFCC as the feature extraction.

4.2 IMPLEMENTATION

We will use following SVM code in MATLAB to generate data and plot it:

```
mgcOpt=mgcOptSet;

if mgcOpt.useInputNormalize, ds.input=inputNormalize(ds.input);

end       % Input normalization

cvPrm=crossValidate('defaultOpt');

cvPrm.foldNum=mgcOpt.foldNum;

cvPrm.classifier=mgcOpt.classifier;
```

Figure 4: SVM Code

Here the SVM will run on the feature vector file and before running it will normalize the data first using the function inputNormalize() function. Then use the classified data to plot it and gather result. Below is the sample code for kNN.

```
[rr, ~]=knncLoo(ds);

fprintf('rr=%g%% for original ds\n', rr*100);

ds2=ds; ds2.input=inputNormalize(ds2.input);

[rr2, computed]=knncLoo(ds2);
```

Figure 5: kNN Code

Chapter 5

RESULT

Classification accuracy varied between the different machine learning techniques and genres. We have seen the classifiers work on 10 different genres. A confusion matrix on the kNN classifier and the SVM classifier is shown below to give an idea of our result.

Below in the confusion matrixes, the X-axis represents the original dataset and the Y-axis represents the classified dataset. The input here is the dataset itself, checking against the original one. The original GTZAN dataset provides us with the information that which song resides in what genre. The classified dataset will check against original dataset and then the confusion matrix is created. We will explain the result of both classifiers.

```
desired=ds.output;

confMat = confMatGet(desired, computed);

cmOpt=confMatPlot('defaultOpt');

cmOpt.className=ds.outputName;

confMatPlot(confMat, cmOpt); figEnlarge;
```

Figure 6: Plot confusion matrix

In the code above, we plotted the confusion matrix. The variable “desired” in our dataset input and the variable “computed” is the cross-validated classified result.

	blues	classical	country	disco	hiphop	jazz	metal	pop	reggae	rock
blues	49.00% (49)	0	12.00% (12)	13.00% (13)	2.00% (2)	0	12.00% (12)	0	5.00% (5)	7.00% (7)
classical	0	90.00% (90)	3.00% (3)	0	0	6.00% (6)	0	0	0	1.00% (1)
country	0	0	62.00% (62)	5.00% (5)	2.00% (2)	6.00% (6)	1.00% (1)	4.00% (4)	1.00% (1)	19.00% (19)
disco	5.00% (5)	0	8.00% (8)	55.00% (55)	3.00% (3)	0	2.00% (2)	2.00% (2)	5.00% (5)	20.00% (20)
hiphop	1.00% (1)	0	3.00% (3)	9.00% (9)	57.00% (57)	0	2.00% (2)	7.00% (7)	14.00% (14)	7.00% (7)
jazz	0	7.00% (7)	7.00% (7)	1.00% (1)	0	73.00% (73)	2.00% (2)	1.00% (1)	0	9.00% (9)
metal	1.00% (1)	0	3.00% (3)	5.00% (5)	4.00% (4)	0	76.00% (76)	0	0	11.00% (11)
pop	0	1.00% (1)	8.00% (8)	7.00% (7)	3.00% (3)	0	0	71.00% (71)	5.00% (5)	5.00% (5)
reggae	0	0	4.00% (4)	17.00% (17)	11.00% (11)	0	1.00% (1)	9.00% (9)	54.00% (54)	4.00% (4)
rock	0	0	15.00% (15)	13.00% (13)	1.00% (1)	4.00% (4)	4.00% (4)	0	1.00% (1)	62.00% (62)

Figure 4: Confusion matrix of kNN Classifier

In the figure above, we have the confusion matrix from the kNN classifier. Here, the kNN classifier has correctly figured blues only 49% of the songs, classical 90%, country 62%, disco 55%, hip-hop 57%, jazz 73%, metal 76%, pop 71%, reggae 54% and rock 62%. We can see blues genre is the lowest performing here and less than half the dataset it got it wrong. In an average recognition rate, it has only done recognition of 64.9%.

	Blues	Classical	Country	Disco	Hiphop	Jazz	Metal	Pop	Reggae	Rock
Blues	83.00% (83)	0	2.00% (2)	2.00% (2)	1.00% (1)	2.00% (2)	5.00% (5)	0	0	5.00% (5)
Classical	0	94.00% (94)	0	0	2.00% (2)	2.00% (2)	0	0	0	2.00% (2)
Country	4.00% (4)	0	70.00% (70)	5.00% (5)	0	1.00% (1)	0	6.00% (6)	2.00% (2)	12.00% (12)
Disco	2.00% (2)	0	3.00% (3)	66.00% (66)	7.00% (7)	1.00% (1)	2.00% (2)	4.00% (4)	6.00% (6)	9.00% (9)
Hiphop	3.00% (3)	0	0	4.00% (4)	74.00% (74)	0	2.00% (2)	1.00% (1)	14.00% (14)	2.00% (2)
Jazz	0	5.00% (5)	1.00% (1)	1.00% (1)	0	90.00% (90)	0	0	0	3.00% (3)
Metal	6.00% (6)	0	2.00% (2)	3.00% (3)	1.00% (1)	0	83.00% (83)	0	0	5.00% (5)
Pop	0	0	9.00% (9)	4.00% (4)	2.00% (2)	0	0	80.00% (80)	2.00% (2)	3.00% (3)
Reggae	5.00% (5)	0	4.00% (4)	6.00% (6)	8.00% (8)	0	0	4.00% (4)	71.00% (71)	2.00% (2)
Rock	4.00% (4)	0	12.00% (12)	11.00% (11)	0	3.00% (3)	7.00% (7)	1.00% (1)	3.00% (3)	59.00% (59)

Figure 5: Confusion matrix of SVM Classifier

After the kNN, we plotted the SVM confusion matrix. Here, the SVM classifier has correctly figured blues 83% of the songs, classical 94%, country 70%, disco 66%, hip-hop 74%, jazz 90%, metal 83%, pop 80%, reggae 71% and rock 59%. Here, except rock, all other genres had significant increase in performance. We checked the total recognition rate is now at 77%, which is 12.1% better than the kNN classifier.

CLASSIFIER	AVERAGE RECOGNITION RATE
k-Nearest Neighbor Classifier	64.9%
Support Vector Machine Classifier	77%

Table 1: Comparison of the total classifier output on the dataset

GENRES	kNN Classifier (%)	SVM Classifier (%)
Blues	49	83
Classical	90	94
Country	62	70
Disco	55	66
Hip-hop	57	74
Jazz	73	90
Metal	76	83
Pop	71	80
Reggae	54	71
Rock	62	59

Table 2: Comparison of the genre output based on the classifiers

We can see through the results and conclude that the SVM classifier is significantly better than kNN classifier and we will use it in our future work where we will produce music from the gained data from automatic genre detection.

Chapter 6

CONCLUSION

We can come to a conclusion from our research that for automated genre detection, the SVM classifier is performing better than the kNN classifier. SVM classifier has a lacking here which is it performed worse than the kNN while detecting rock genre. We suggest using a boosting technique such as AdaBoost multiclass boosting algorithm or bagging on the SVM classifier to check if it returns a better result in the provided GTZAN dataset. Binary classifying boosting techniques such as LogitBoost would not work properly as the dataset is classified into multi classes using MFCC. If only binary classification using other feature extraction method such as ZCR (Zero crossing rate) is used, then LogitBoost will perform better [9] in all genres. But for our future work plan, for now the SVM classification is well enough to generate a result.

6.1 FUTURE WORK

Future work may include classification of more genres like EDM using different methods and machine learning algorithms. Moreover, in future we will try to make a computer generated system which will be a complete musical accompanist to the user. It will generate real time beat to keep track with what the user is playing by categorizing the genres. There are two ways that we can see right now to move forward with this one is machine learning tools, another one is neural networks (CNN or RNN). If we are going to use neural networks then there‟s another decision that need to be taken which is what to use between CNN(Convolutional Neural Network) and RNN (Recurrent Neural Network). As we know CNN works with fixed number of inputs and outputs. So fitting any dataset other than GTZAN would be very difficult which doesn‟t include electronic music. On the other hand, RNN use arbitrary inputs, so fitting large data set is not a problem there. More data means more percentage of success.

REFERENCES

1. M. Haggblade, Y. Hong and K. Kao, "Music Genre Classification", *http://cs229.stanford.edu*, 2017. [Online]. Available: http://cs229.stanford.edu/proj2011/HaggbladeHongKao-MusicGenreClassification.pdf. [Accessed: 16- Aug- 2017].
2. B. L. Sturm, "An Analysis of the GTZAN Music Genre Dataset", *stevetjoa.com*, 2017. [Online]. Available: https://stevetjoa.com/static/p7.pdf. [Accessed: 16- Aug- 2017].
3. Fu, A., Lu, G., Ting, K.M., Zhang, D.. "A Survey of Audio-Based Music Classification and Annotation" IEEE Transactions on Multimedia. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5664796&tag=1.
4. R. Jang, "MATLAB Toolboxes Developed by MIR Lab", *http://mirlab.org*, 2017. [Online]. Available: http://mirlab.org/jang/matlab/toolbox/. [Accessed: 16- Aug- 2017].
5. "Million Song Dataset", *labrosa.ee.columbia.edu*, 2017. [Online]. Available: https://labrosa.ee.columbia.edu/millionsong/. [Accessed: 16- Aug- 2017].
6. "Million Song Dataset", *labrosa.ee.columbia.edu*,2017 .[Online]. Available: https://labrosa.ee.columbia.edu/millionsong/lastfm. [Accessed: 16- Aug- 2017].
7. "MARSYAS", *http://marsyasweb.appspot.com*, 2017. [Online]. Available: http://marsyasweb.appspot.com/download/data_sets/. [Accessed: 16- Aug- 2017].
8. Cast, C. Schulze and A. Fauci, "Music Genre Classification", *http://cs229.stanford.edu*, 2013. [Online]. Available: http://cs229.stanford.edu/proj2013/FauciCastSchulze-MusicGenreClassification.pdf. [Accessed: 16- Aug- 2017].

9. C. Tripp, H. Hung and M. Pontikakis, "Waveform-Based Musical Genre Classification", *http://cs229.stanford.edu*. [Online]. Available: http://cs229.stanford.edu/proj2006/TrippHungPontikakis-WaveformBasedMusicalGenreClassification.pdf. [Accessed: 16- Aug- 2017].