A Machine Learning Approach to Classify Sinhala Songs Based On User Ratings

H.M.T.Paranagama[1#], M.K.A.Ariyaratne[1], S.C.M.D.S. Sirisuriya[1]

[1] *Department of Computer Science, Faculty of Computing, General Sir John Kotelawala Defence University, Ratmalana.*

[#] tharindrapara@gmail.com

Abstract— *In the music industry there is a need to analyse the significant features that distinguish highly rated songs from lower rated ones. Then an artist can test their music tracks to check whether it will gain potential popularity, before mass production and if the rating is lower they can focus on the significant features present in popular tracks. Our study address this by developing a machine learning approach to classify music tracks based on user ratings. There were many research performed in the area of music genre classification, music recommendation using vanilla neural networks, recurrent neural networks and convolutional neural networks. The research mainly focuses on the classification of Sinhala songs. Our dataset is consisting of 11,000 Sinhala music tracks each having several attributes. From each track we extract 3 meaningful features. For the feature extraction process we used a python library. The output has three distinct classes that specify the user rating. A Multi-layer neural network was implemented. 500 training epochs with 60 neurons in each hidden layer were used. Initially, with 3031 training tracks and 1299 testing tracks we achieved an accuracy of 86%. With this, we conclude that the development of a multilayer neural network to automate the process of determining the rating for a song is in a successful stage compared with the existing approaches.*

Keywords— **Artificial Neural Networks, Classification, Clustering, Feature Extraction**

I. INTRODUCTION

Music has a great influence on humankind from reducing stress to enhancing cognitive power of human beings, While conveying strong messages as well as feelings to the audience in a unique manner. According to forecasts done by PriceWaterhouse Coopers, the global music industry is supposed to generate 47.7 billion U.S. dollars in revenue by 2020.(Statista,1 May 2017) In this type of promising market musicians can easily benefit given that they can satisfy audience expectations. In the context of Sinhala music it has the influence of many cultures such as the British, Indian and Buddhist.(wikipedia,1 May 2017) With these influences the music developed in Sri Lanka has greatly evolved and as a result of this evolution today we experience a more unique production of music. In the past the music produced was more of classical type while modern music incorporates a highly sophisticated level of beat patterns as well as a variety of instruments which generate powerful and complex tones. With the improvement in technology the music industry has taken a huge step forward with more convenient ways to produce, manipulate as well as generate music for the ease of musicians. With these developments in this industry it has made it extremely difficult to analyze and interpret user preferences thus leading to many musicians to fail in the industry while a few highly dominate the industry through their gifted talent. Through research done by a collaboration of the following universities and institutions, McGill University, the University of Cambridge, Rutgers University, City University of New York, and the Stanford Graduate School of Business, They have found out that any music produced can be effectively categorized with the use of three significant features which are Arousal (intensity and energy in music), Valence(Spectrum of emotions in music) and Depth(Intellect and sophistication in music).(Digital Music News,1 May 2017) There are many research which mention different types of characteristics of music that determine user preferences. But there are no proper tools that have been implemented to successfully provide a solution to this problem of determining the potential user rating for a given song based on the underline characteristics. As a rating determines the success of the music track inability to effectively determine the rating will be a barrier to determine the success of a music track. Without any awareness of the success of the product a musician(especially for amateurs) may not be willing to enter the industry or may leave the industry, because of this the objective of our study was to d*evelop a multilayer artificial neural network to successfully determine the rating of a given Sinhala song. In this process of determining the rating we also optimize the algorithm(s) and source code to provide a more efficient and accurate result.*

II. METHODOLOGY

There are extensive research done in the area of music genre classification and sound classification. In a research done by Aaqib Saeed, used machine learning to classify urban sounds to its relevant classes. He used a dataset which consist of various 8732 urban sounds

each consisting of duration of 4s separated in to 10 folders from www.freesound.org from which first three folders were used for training and testing purposes. The dataset has 10 classes which are air conditioner, car horn, children playing, dog bark, drilling, engine idling, gun shots, jack hammer, siren and street music The features that were extracted with the use of a python library known as Librosa and these features were melspectrogram, mel-frequency cepstral coefficients, chroma-stft, spectral contrast and tonnetz .

The extracted features and labels were fed to a multi-layer neural network on the training data. And the testing data were later fed after completion of the training. And here only the features of the testing set were provided to the neural network and the network was supposed to predict/determine the relevant label pertaining to the given features based on prior learning experience. The accuracy is calculated based on the number of correct predictions over the total testing set.

They have used Gradient Descent Optimizer which is an optimisation algorithm used to reduce the cost by a fraction of the learning rate during the training process. Thus optimizing for higher accuracy. Tensorflow has been used in the development of the neural network and the cost function they use is the cross entropy function.

This system does not provide an acceptable level of accuracy. It only has an accuracy of approximately 12% and further the computational cost of this implementation is significantly high(Saeed,1 May 2017).

In a research done by Carlos N. Silla Jr., Alessandro L. Koerich, and Celso A. A. Kaestner which was to automatically classify music into genres.In this system they use multiple feature vectors and pattern recognition techniques. A set of binary classifiers which produces the final result/genre based on a merge of multiple attributes were used to obtain result which were afterward merged to determine the genre. Most popular machine learning algorithms such as k-nearest neighbors, support vector machines, Naive-Bayes are some of them which they used. An important feature identified through their research was the most significant features used for the classification task depended on the origin of the music signal(Carlos N. ,Koerich and Silla Jr,2008).

In a music genre classification research done by Michael Haggblade, Yang Hong and Kenny Kao. In their research they have discussed about three classification algorithms which are k-means,multi-class SVM, and K-Nearest Neighbor(KNN).further they have chosen four classes(jazz, classical, metal, pop) of which one will be assigned with the output. They have used Mel Frequency Cepstral Coefficients(MFCCs) as their feature. They have also extended their system to cluster music into genres based on image features. They have performed the extension using k-means algorithm for clustering and Fourier-Mellin 2D transform to extract features. They have noted an important fact which is that the accuracy decreases if the number of classes exceeds 4. They stated that K-means algorithm faced a huge difficulty in distinguishing between classical and jazz. thus leading to 36% misclassification. They have stated that when considering overall accuracies the KNN and K-means displayed similar accuracies of 80% while the SVM gave them a 87% accuracy and finally the neural network giving them a 96% accuracy(Kenny Kao , Michael Haggblade and Yang Hong).

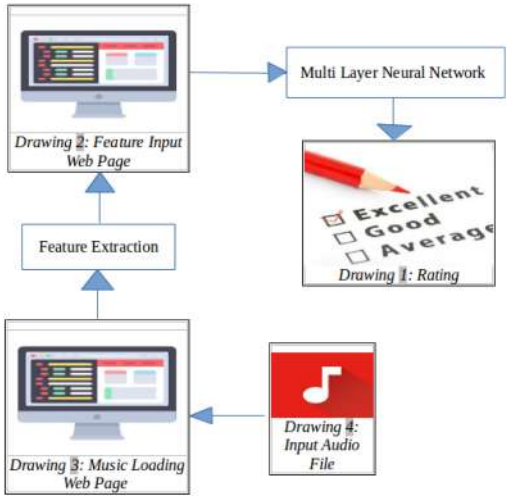
In a research done by Sam Clark, Danny Park and Adrien Guerard, they have used eight summary features, growing neural gas (an algorithm for reducing the computational costs when using a complex neural network) and a neural network for music genre classification. They have considered five classification classes as rap, reggae, classical, and country. Their hypothesis was that the growing neural gas would lead to an improved classification accuracy. This hypothesis was proven to be true with the receipt of nearly 14% increase in training accuracy and a 3% increase in test accuracy (Adrien Guerard, Danny Park and Sam Clark, (2012)) .

In a research done by Cory McKay using neural networks for music genre classification, he has developed a system which can automatically classify MIDI files into hierarchically organized parent genres and sub genres. For his classification he has used primarily twenty original features and achieved an accuracy of 85% for parent genres and 65% for sub-genres(Cory McKay).

There also exist research done by Bo Shao, Dingding Wang, Tao Li, Mitsunori Ogihara in the area of music recommendation. They state that collaborative filtering and content based recommendations are the most popular and recommended methods for music recommendation. But the aforementioned researches see some significant disadvantages of these methods which leads to an ineffective recommendations been made. These disadvantages are, the collaborative method will be effective only if we use a large dataset that consist of user history of user activity on the web. And the content based method lacks the ability to identify user preferences and interests. To overcome these limitation they propose a novel approach which takes into account content features and user access patterns(Bo Shao, Dingding Wang and Tao Li, (2009)).

B. Conceptual Design

The following diagram describes the entire process of determining the rating for a given song.



Here the the inputs are the three features which are

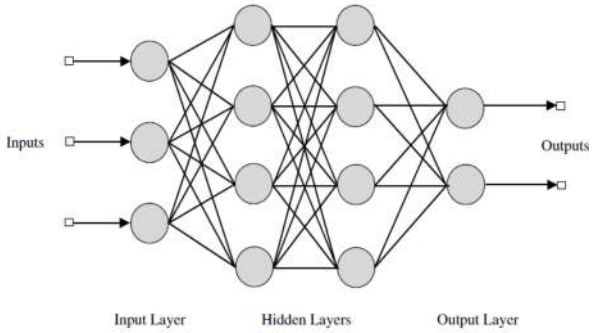


Figure 2: Architecture of the multi-layer neural

tempo, MFCC and harmonic element. The hidden layers do the processing and outputs the relevant label(excellent,moderate,poor).

C. Approach

1.Users: Any artist or music listener who would like to determine the rating of a given song.

2.Inputs: The 3 extracted features from the audio file.

3.Process:There will be a web page which will provide the user with the facility to upload their audio file and then the feature extraction method will run in the back end and display the extracted feature vector on the web page's output area.

Then the user can input this feature values to the neural network available on another page to determine the predicted rating for the input vector.

D. Implemantation

1. Creation of the datasets and preprocessing

The dataset contains 11,000 mp3 music files from a variety of musicians, genres and centuries. This diversified dataset is a good representation of the entire population of Sinhala songs.

First we would extract the features from each of the audio file and save it to a CSV file. So that it would be computationally less expensive compared to processing the audio files for feature extraction during program execution.

```
#feature extraction method
def extract_feature(file_name):
    X, sample_rate = librosa.load(file_name)
    onset_env = librosa.onset.onset_strength(X, sr=sample_rate)
    #beats per minute(tempo)
    tempo = np.mean(librosa.beat.tempo(onset_envelope=onset_env, sr=sample_rate).T,axis=0)
    #captures the specific characteristics of sinhala music
    mfccs = np.mean(librosa.feature.mfcc(y=X, sr=sample_rate, n_mfcc=1).T,axis=0)
    #extracts the harmonic element
    y_harmonic = np.mean(librosa.effects.harmonic(X, margin=3.0).T,axis=0)
    return tempo,mfccs,y_harmonic
```

```
import csv
with open('total_features.csv', 'w') as fp:
    a = csv.writer(fp, delimiter=',')
    a.writerows(total_features)
```

```
from sklearn.cluster import KMeans

def extractFeatures(filename):
    import numpy as np
    features = []
    features = np.array([features])
    for line in file(filename):
        row = line.split(',')
        features = np.append(features, np.array(row[0:3]))
    return np.array(features)

filename="/home/tharindra/PycharmProjects/WorkBench/MusicDataset"
features= extractFeatures(filename)
features=features.reshape(11000,3)

kmeans = KMeans(n_clusters=3)
kmeans.fit(features)

centroids = kmeans.cluster_centers_
labels = kmeans.labels_

for i in range(len(features)):
        print("coordinate:",features[i], "label:", labels[i])
```

Figure 5: Clustering the features using K-Means

After the CSV file is created we would use a K -Means clustering to determine the labels of the dataset. For this we use the scikit-learn library which provides an easy implementation of this algorithm.

Then this dataset would be split into 3 portions as follows 70% as training ,15% as testing and 15% as validation.

Finally the training data and testing feature vectors should be reshaped to be separately fed to the neural network during training and testing.

```
tr_features = tr_features.reshape(7700, 5)
ts_features = ts_features.reshape(3300, 5)
```

And also the labels should be converted in to one hot vectors because we cannot do comparisons with nominal values and also using one hot encoding facilitates ease of processing. Here we use an encoding which is an array of the length equal to the number of classes in our problem and we give an activation to the

relevant position in the array depending on the value of the label. for example if the rating poor has a label of 0 this can be represented as a one hot vector as follows,
[1,0,0] where the position of zero is active to denote the
label. When training and testing the neural network we provide it with the features and one-hot encodes instead of labels.

```
def oneHot(labels):
    import numpy as np
    p = len(labels)
    up = len(np.unique(labels))
    b = np.zeros((p, up))
    b[np.arange(p), labels] =
    return b

tr_encode = oneHot(tr_labels)
ts_encode = oneHot(ts_labels)
```

Now since we have constructed our inputs in a suitable manner for feeding we have to now move on to development of the neural network.

2.Neural network design

The neural network has an input layer, 2 hidden layers and an output layer.

The hyper parameters of the network are as follows,

10 neurons in each hidden layer a learning rate of 0.003 and 600 training epochs.

```
# setting hyper parameters & other variables
training_epochs = 500
n_features = 3
n_classes = 3
n_neurons_in_h1 = 10
n_neurons_in_h2 = 10
learning_rate = 0.01
```

As we are using tensorflow for the construction of the neural network we use 2 placeholders which is a data type that allows us to feed data to the neural network during run time. These are very useful as we can sequentially feed each feature vector along with its corresponding one hot encode to these two separate placeholders.

```
# placeholdr tensors built to store features(in X) and labels(in Y)
X = tf.placeholder(tf.float32, [None, n_features], name='features')
Y = tf.placeholder(tf.float32, [None, n_classes], name='labels')
```

when the input(feature vector) is fed to the first layer of the neural network its multiplied (using matrix multiplication)with the weight matrix which is of size [#features,#neuronsInHiddenLayer1] and provides the ability for each feature to be multiplied by each neuron .The values of the weight matrix are initialised with random values at the start of the training process. After we multiply the weight matrix with the feature vector we add biases to each output from the matrix multiplication. The biases matrix is of size [#neuronsInHiddenLayer1].now this net input is sent through an activation function which will out put a vector of size [#features] which will be the input to the next layer ,the most popular activation function is Sigmoid and we have also used it because it gives an output within the range of 0&1.the Next layer also operates in the same way and progresses to the output layer in which it goes though a Softmax activation function to produce a probability as an output.

```
# network parameters(weights and biases) a set and initialized(Layer1)
W1 = tf.Variable(tf.random_normal([n_features, n_neurons_in_h1]), name='weights1')
b1 = tf.Variable(tf.random_normal([n_neurons_in_h1]), name='biases1')
# activation function(sigmoid)
y1 = tf.nn.sigmoid((tf.matmul(X, W1) + b1), name='activationLayer1')

# network parameters(weights and biases) a set and initialized(Layer2)
W2 = tf.Variable(tf.random_normal([n_neurons_in_h1, n_neurons_in_h2]), name='weights2
b2 = tf.Variable(tf.random_normal([n_neurons_in_h2]), name='biases2')
# activation function(sigmoid)
y2 = tf.nn.sigmoid((tf.matmul(y1, W2) + b2), name='activationLayer2')

# output layer
Wo = tf.Variable(tf.random_normal([n_neurons_in_h2, n_classes]), name='weightsOut')
bo = tf.Variable(tf.random_normal([n_classes]), name='biasesOut')
# activation function(softmax)
a = tf.nn.softmax((tf.matmul(y2, Wo) + bo), name='activationOutputLayer')
```

Figure 10: Construction of the multi-laver neural

Now we define the cost function as cross entropy which is also a popular choice among cost functions.

```
cross_entropy = tf.reduce_mean(-tf.reduce_sum(Y * tf.log(a), reduction_indices=[1]), name='CostFunction'
```

$$H(p, q) = -\sum_{x} p(x) \log q(x)$$

Figure 12: Mathematical formula for cross

where q(x) is the predicted output and p(x) is the expected output. To convert this function into a minimisation function we put a minus in front.

And then we should optimise our cost function during the training process for this we use the Gradient Descent Optimization algorithm. It should be noted that during the training of the network over the 600 epochs we update the weight an biases after each input is fed to the network via a dictionary of feature values and labels. This updating is done by a fraction of the learning rate for this we use the optimization algorithm and we specify the cross entropy to be minimized. Thus

allowing the network to converge to a particular function during the learning process.

```
train_step = tf.train.GradientDescentOptimizer(learning_rate).minimize(cross_entropy
```

Then we also need calculate the accuracy so for that we need to know the number of correct predictions. So we determine the correct predictions by comparing the actual label with the predicted label on the testing features by the network. Then we cast this boolean value to a float to determine the number of correct predictions as a value. then we display the accuracy as the number of correct predictions over the total number of test samples.

```
# compare predicted value from network with the expected value/target
correct_prediction = tf.equal(tf.argmax(a, 1), tf.argmax(Y, 1))
# accuracy determination
accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32), name="Accuracy"
```

E. Evaluation

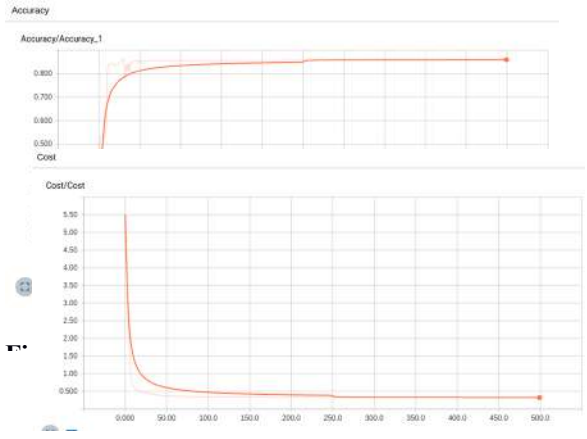
After developing the model we can use the model to make new predictions for that we need to save the desired model and restore it when we need to do prediction through it and the prediction can be made on a new record as follows.

```
with tf.Session() as sess:
    sess.run(initial)
    saver = tf.train.Saver()
    saver.save(sess, "/home/tharindra/PycharmProjects/WorkBench/saveGeetha.ckpt")
    print("Model saved")
    saver.restore(sess, "/home/tharindra/PycharmProjects/WorkBench/saveGeetha.ckpt")
    print("Model restored")
    value =  str(sess.run(((tf.argmax(a, 1), tf.argmax(Y, 1))), feed_dict={X:
[[-105.99996986418589,112.34714673913044,-3.1714756687506451e-07]], Y: [[1, 0, 0]]}))
```

III. RESULTS

A. *Performance Indices*

the main performance indices we consider is the accuracy and the cost. We managed to achieve an accuracy of 50% and a smoothly decreasing curve. The following diagrams illustrate the above explanation. It should be also noted that the following results were obtained after a number of training runs by changing hyper parameters to find this temporary satisfiable solution.



Parameter Configuration	Performance/Accuracy
Training epochs :1000 Number of neurons in hidden layer one:1000 Number of neurons in hidden layer two:500 Learning rate:0.05	Accuracy:81%
Training epochs :1000 Number of neurons in hidden layer one:500 Number of neurons in hidden layer two:500 Learning rate:0.05	Accuracy:80%
Training epochs :500 Number of neurons in hidden layer one:60 Number of neurons in hidden layer two:60 Learning rate:0.05	Accuracy:85%
Training epochs :500 Number of neurons in hidden layer one:60 Number of neurons in hidden layer two:60 Learning rate:0.1	Accuracy:85%

Fig. 18: Changes in performance with respective

IV. SUMMARY & CONCLUSION

According to the extensive research I performed I observed that the currently there is no solution for the aforementioned problem in the context of Sinhala music. So we have not only built a solution for the system but we have optimized the solution and the code in various ways such as using clustering in determining the labels, and using prestored data for feeding input.

ACKNOWLEDGEMENT

its with great honor that I would like to thank MR.Chandrasekara who was grateful enough to offer me a large repository of diverse Sinhala mp3 song collection. Further I would also take this opportunity to thank my supervisor Ms.M.K.A.Ariyaratne for her valuable insights and guiding on this project and also Dr.T.G.I.Fernando for showing me this problem for which I was able to come up with a decent solution.

REFERENCES

Statista. 2017. • *Global music industry revenue 2016 | Statistic*. [ONLINE] Available at: https://www.statista.com/statistics/259979/global-music-industry-revenue/. [Accessed 01 July 2017].

Music of Sri Lanka - Wikipedia. 2017. *Music of Sri Lanka - Wikipedia*. [ONLINE] Available at: https://en.wikipedia.org/wiki/Music_of_Sri_Lanka. [Accessed 01 July 2017].

Digital Music News. 2017. *All Music Can Be Categorized by Just Three Attributes*. [ONLINE] Available at: https://www.digitalmusicnews.com/2016/05/17/music-genres-three-attributes/. [Accessed 01 July 2017].

Aaqib Saeed. 2017. *Urban Sound Classification, Part 1*. [ONLINE] Available at: https://aqibsaeed.github.io/2016-09-03-urban-sound-classification-part-1/. [Accessed 01 July 2017].

A Machine Learning Approach to Automatic Music Genre Classification - Kent Academic Repository. 2017. *A Machine Learning Approach to Automatic Music Genre Classification - Kent Academic Repository*. [ONLINE] Available at: https://kar.kent.ac.uk/24022/. [Accessed 01 July 2017].

Haggblade, M., Hong, Y. and Kao, K. (2017). *Music Genre Classification*. [ebook] p.http://cs229.stanford.edu. Available at: http://cs229.stanford.edu/proj2011/HaggbladeHongKao-MusicGenreClassification.pdf [Accessed 1 Jul. 2017].

Clark, S., Park, D. and Guerard, A. (2012). *Music Genre Classification Using Machine Learning Techniques*. [ebook] Available at: https://www.cs.swarthmore.edu/~meeden/cs81/s12/papers/AdrienDannySamPaper.pdf [Accessed 1 Jul. 2017].

Bo Shao, Ogihara, M., Dingding Wang and Tao Li (2009). Music Recommendation Based on Acoustic Features and User Access Patterns. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(8), pp.1602-1611.

McKay, C. (2017). *Using Neural Networks For Musical Genre Classification*. [ebook] Montreal: McGill University. Available at: http://www.music.mcgill.ca/~cmckay/papers/musictech/GenreClassification1.pdf [Accessed 1 Jul. 2017].