Machine Learning Project: Final Report

Music Genre Identification and Recommendation System

Name	Reg. No.
Chandu Anilkumar	19BAI1003
Sarthak Vasal	19BAI1050
Harshita Dalwe	19BAI1063
Hasnain Sikora	19BAI1072

Abstract

The more popular aspects of Machine Learning like Natural Language Processing and Image Recognition often overshadow its applications in audio analysis. With most of the global population identifying as enjoyers of audio entertainment, being able to understand and explore one's taste in music can greatly help in widening one's interest in it. The understanding of music begins with the acknowledgment of the genre of a given piece of music.

In essence, music can be boiled down to complex rhythms of notes and percussions with varying speeds and time domains to make melodies and harmonies. With the near infinite number of arrangements possible, the musical universe is vast. For putting things in perspective, a guitar has 6 strings and 24 frets, that makes 144 positions plus 6 open strings to fret on for one tuning setup. In terms of unique notes, a guitar can produce 49 tones including half steps. Standard music studies say there are more than 1500 recognized instruments.

Taking all these numbers into consideration, every musical piece would ideally have a unique and different sound. Although every song has at least a pinch of uniqueness to it, us humans have found liking for certain chord progressions and certain notes. With different permutations of such large numbers, the number of genres that exist in every nook and corner of the world is almost impossible to pinpoint. To avoid such tedious work, it is generally accepted that 10 genres of music cover majority of the songs in the human culture to an extent. These are namely Blues, Classical, Country, Disco, Hip-hop, Jazz, Metal, Pop, Reggae and Rock. Because of such complex possibilities of every song belonging to a specific genre, it is almost impossible for us humans to compute accurately. Through machine learning, we can closely determine the category of a given audio snippet by taking into consideration other audio samples that define the genre. We aim to combine this less explored foray of machine learning with this popular medium of entertainment.

Introduction

Music is one of the most accessible forms of art. It is cherished by most of the population on a regular basis. About 69% of adults aged between 18 and 34 years old reported listening to music every day. New age technology allows every individual to explore and expand their music tastes. Based on the type of music one listens to, streaming platforms target their user's recommendation specific artists and genres. Our project aims to implement a simple machine learning prototype that takes into consideration, the listening patterns of the user as input and classify them into major genres based on valid datasets. This in turn, can be used to recommend music from the same genre, making the system more personalized.

Related works

Lin, Chen, Truong and Chang (2005) proposed using wavelet transform for classification of music as they conferred that its “more natural and effective for describing audio characteristics”. They used wavelets for feature extraction with main goals being sub-band power and pitch. The wavelet transforms were employed to calculate Mel-frequency Cepstral Coefficients (MFCC). The proposed method used SVM and MFCC for classification based on the features extracted.

Ramalingam and Dhanalakshmi (2014), presented a study by applying Discrete Wavelet transform for classifying music and sound and by applying Support Vector Machine learning algorithm to attain optimal class boundary. They compared their performance to Gaussian Mixture Model for different mixtures which showed satisfactory results.

Kour and Mehan (2015), proposed used SVM and BPNN for an automatic music classification system. MFCC was calculated as features to characterize audio content. For the classification of genre from training data, SVM and BPNN were applied.

Ajmera, McCowan and Bourlard (2003), presented a system for speech/music discrimination. They used entropy and dynamism as their main features based on posterior probabilities of speech phonetic class. By using features based on phonetic posterior probabilities they figured how to locate speech segments within audio recognizable by speech recognition system. They employed combinations of HMM with ANN and GMM with MLP. Multilayer Perceptron (MLP) resulted with set of real probabilities which made the system preferable.

Methodology

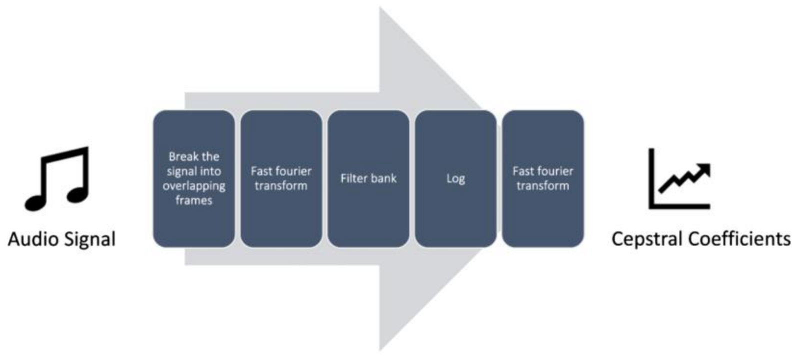
Our project primarily focused on iterating each of the songs from their respective genres and dividing them into segments, thus the feature extraction of MFCC (The Mel Frequency Cepstral Coefficients) was done and the respective genres corresponding to their respective MFCCs were stored in a JSON object for easy retrieval and dataset training.

MFCC

The Mel Frequency Cepstral Coefficients, abbreviated as MFCCs are the representation of the short-term power spectrum of a sound. The MFCC considers human perception for sensitivity at appropriate frequencies by converting the conventional frequency to the Mel Scale. The Mel scale is a scale that relates the perceived frequency of a tone to the actual measured frequency. It scales the frequency to match more closely what the human ear can hear.

MFCCs surpass the human hearing when it comes to identifying small changes in speech at lower frequencies. The range of human hearing is 20Hz to 20kHz. If a human were to differentiate between two tunes of 300Hz and 400Hz respectively and assume the distance between these two however the brain perceived them as a value say 'X', would probably be close to 100Hz. If the same person were to assume the distance of two other tunes at 900Hz and 1000Hz respectively, say 'Y', one can observe that not only X more inaccurate than Y, but it is also observed most human brains perceive Y to be higher than X despite both actual values being a 100Hz. This is where the Mel scale can capture the differences accurately.

The working of MFCC can be easily understood by the following flow chart.



MLP

The perceptron is a simple algorithm intended to perform binary classification. In essence, it predicts whether input belongs to a certain category of interest or not. As the name suggests, A multilayer perceptron comprises of multiple single layer perceptrons. They are at times, commonly called vanilla neural networks.

A deep artificial neural network that consists of an input layer to receive, an output layer that makes a calculated prediction or decision based on the input, and an arbitrary number of hidden layers that work as the computational engines of the MLP. Each node here is a neuron that uses a non-linear activation function, the input node being the only exception. The activation function is a linear function that maps the weighted inputs to the output of each neuron.

On implementation, the accuracy observed for our problem was below our expected values. On further research, we were recommended using the concept of a convolution neural network (CNN)

CNN

Since the very beginning of the concept of artificial intelligence that was rooted back in the 1950s, computer scientists have tried to make machines and systems understand and work on visual and audible data. CNNs were first introduced in the late 1980s but it was not until 2012, when the AlexNet was made that data computing took a giant leap in terms of overall accuracy and efficiency. AlexNet thrived on the concept of a CNN that roughly mimicked the human vision system.

A Convolution Neural Network (CNN) which is also called shift invariant or space invariant artificial neural network is a class of deep neural networks, applied to analysing different media formats such as visual imagery, audio samples, video recognition etc. Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons are mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value.

RNN

Typically, in a Neural Network, all the inputs and outputs generated are independent from each other. But in a Recurrent Neural Network, the outputs from the previous step are taken as the input for the next step. It can utilize its internal state to retain information about performed calculations. When compared to other Neural Network architectures, an RNN reduces the complexity of parameters as the same parameters are used for each input. RNNs can also be used with convolutional layers to increase effective pixel neighbourhood.

Due to the finite precision computation involved, the gradients can ‘vanish’ (tend to zero) or ‘explode’ (tend to infinity) when an RNN is trained through backpropagation. An LSTM (Long Short-Term Memory) is an architecture based on RNN that uses different gates to remember necessary and forget irrelevant data. It tries to solve the exploding and vanishing gradient problems by allowing the gradients to flow unchanged.

Since they can handle inputs in the form of sequences, RNNs are suitable for applications such as handwriting/speech recognition.

SVM

A Support Vector Machine (SVM) is a supervised learning model that can be used for classification, regression and outlier detection. It plots hyperplanes to classify data points. By using 'support vectors', which are data points that influence the orientation of the hyperplane, the margin of the classifier can be maximized to reduce generalisation error.

In cases where the data in not linearly separable, Kernels are employed. These are functions that quantify similarities between observations in order to enlarge the space to accommodate non-linear boundaries between classes. They transform the data to pass a linear hyperplane. Common kernels used include:

- Linear: $\langle x, x' \rangle$
- Polynomial: $(\gamma\langle x, x' \rangle + r)^d$
- Radial Based: $exp(-\gamma\langle \|x - x'\|^2 \rangle)$
- Sigmoid: $\tanh(\gamma\langle x, x' \rangle + r)$

Librosa Library

The Librosa library is a python package which is specifically built for audio analysis.

Librosa's load function will read in the path to an audio file and return a tuple with two items. The first item is an 'audio time series' (type: array) corresponding to audio track. The second item in the tuple is the sampling rate that was used to process the audio. The default sampling rate (sr) used by Librosa is 22050.

The librosa.load function, primarily used here returns the following:

1. The sample rate sr: which means how many samples are recorded per second.

2. A 2D array that consists of:

   The first axis: represents the recorded samples of amplitudes in by determining the change in air pressure in the audio.

   The second axis: represents the number of channels in the audio.

Scikit-learn

Any learning problem considers a set of samples of data, based on which it tries to predict properties of the next given unknown data. Such problems can be classified into 2 major categories.

1. Supervised learning, in which the data comes with additional attributes that we want to predict. This type of a problem can either be a classification, where with the given n number of samples and a given number of categories, the system tries to label the sample to its identified category as accurately as possible or it could be a regression, where the desired output consists of one or many continuous variables where the variables are predicted based on the previously fed information.

2. Unsupervised learning, in which the training data consists of a set of input vectors x without any corresponding target values. The goal in such problems may be to discover groups of similar examples within the data, where it is called clustering, or to determine the distribution of data within the input space. The Scikit-learn concept falls under this working.

It provides a selection of efficient tools for machine learning and statistical modelling including classification, regression, clustering, and dimensionality reduction via a consistence interface in Python. It primarily uses numpy extensively for high-performance linear algebra and array operations.

Framework Used:

Flask

Flask is web framework which is primarily written in python. It is considered to be a microframework which means that it does not require particular tools or libraries. It has no database abstraction layers, form validation or any third-party libraries to provide common functions. However, it does support extensions that can add application features including the ability to take a trained model from jupyter and deploy it into a website.

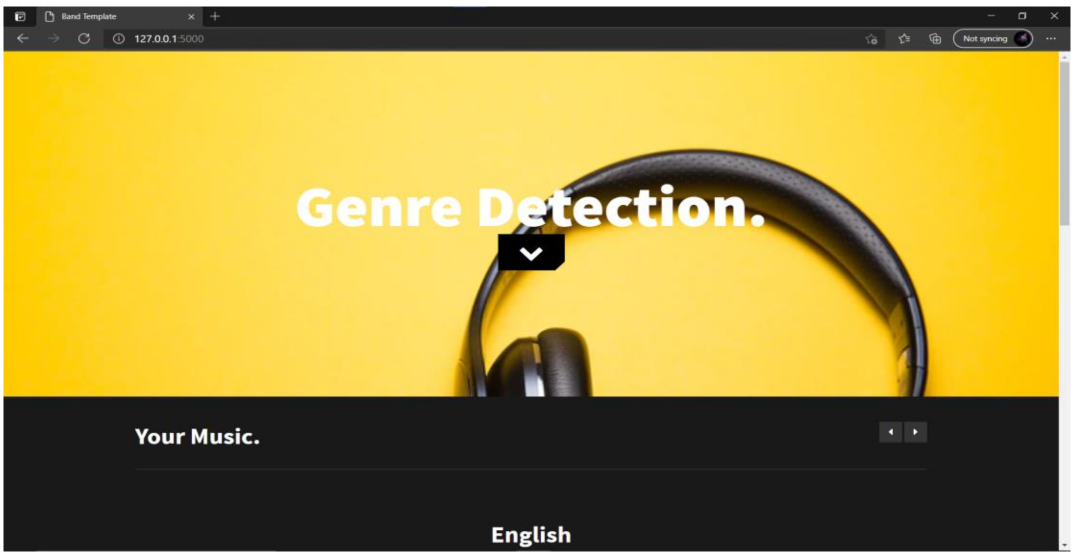
Our aim is to now take the trained model from jupyter notebook. The model that has the highest testing accuracy is chosen such that it provides satisfactory results. We then use a package called h5py to save our trained model as a ".hdf5" file.

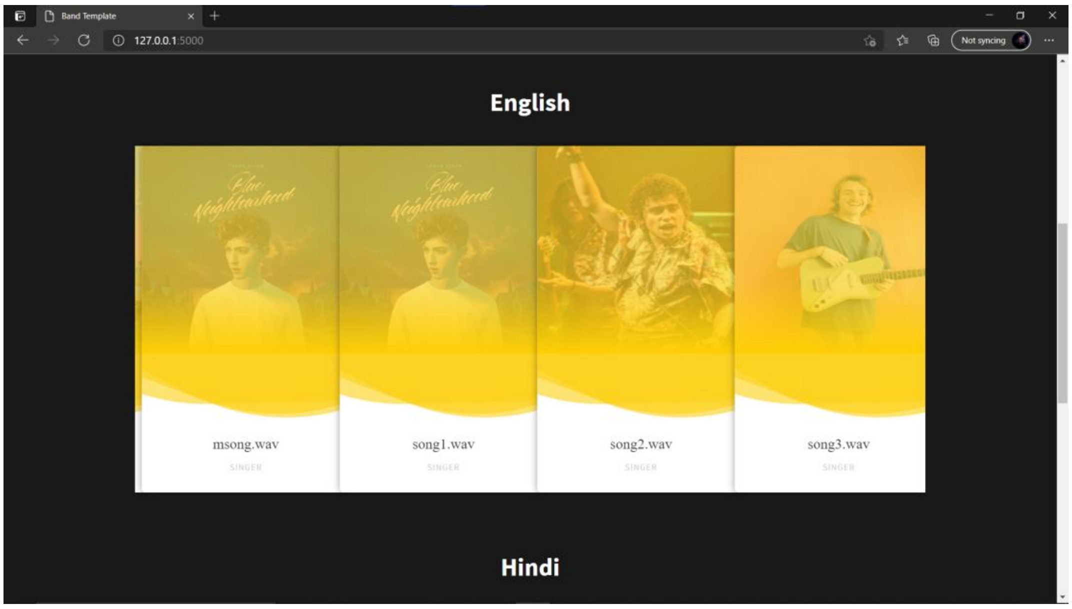
H5py

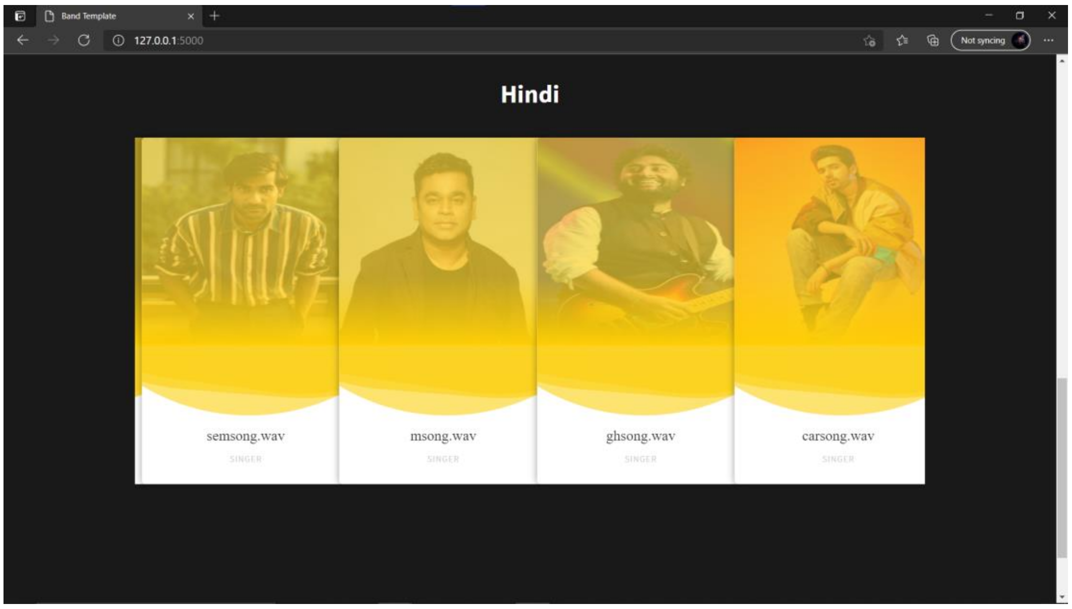
The h5py package is a Pythonic interface to the HDF5 binary data format. It lets you store huge amounts of numerical data, and easily manipulate that data from NumPy. For example, you can slice into multi-terabyte datasets stored on disk, as if they were real NumPy arrays. Thousands of datasets can be stored in a single file, categorized and tagged however you want.

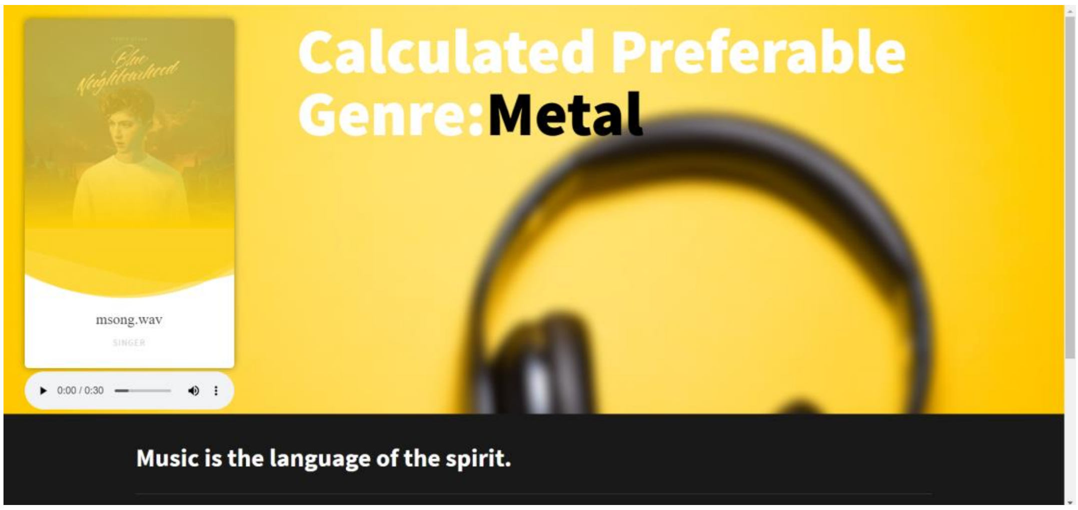
Once the model is saved as a ".hdf5", we then proceed to create a flask application. The flask application uses Keras' load_model function to load the model instance into a variable from a local folder. We then create a function which takes in ".wav" audio files and thus cuts it into segments and calculates and stores MFCC's into an array. This array is then passed down to a predict function which uses the trained model's predict function to give us a genre. This genre is then displayed on the website.

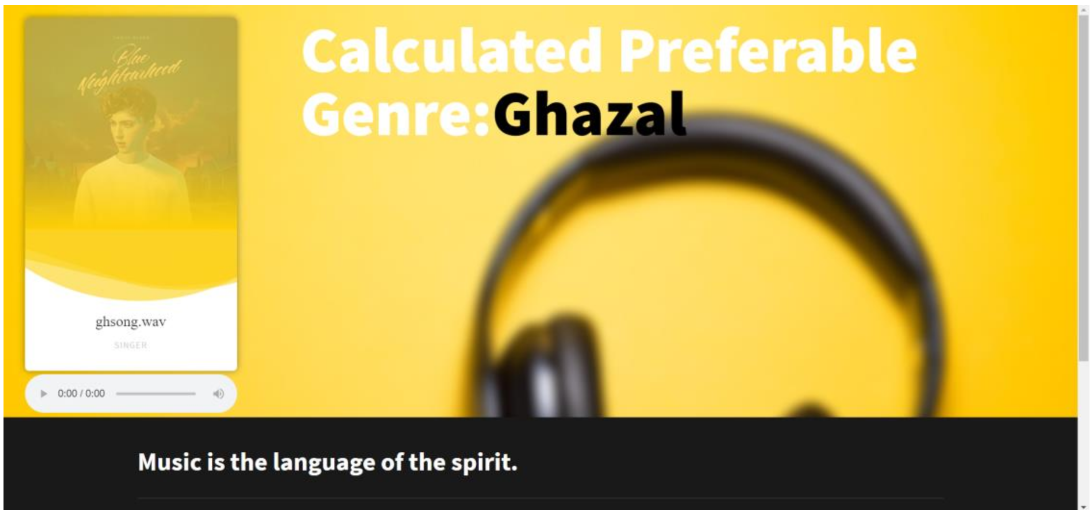
Screenshots




Band Template
127.0.0.1:5000
Not syncing
English
msong.wav
SINGER
song1.wav
SINGER
song2.wav
SINGER
song3.wav
SINGER
Hindi


Band Template
127.0.0.1:5000
Not syncing
Hindi
semsong.wav
SINGER
msong.wav
SINGER
ghsong.wav
SINGER
carsong.wav
SINGER


msong.wav
SINGER
0:00 / 0:30
Calculated Preferable Genre:Metal
Music is the language of the spirit.


Calculated Preferable Genre:Ghazal

ghsong.wav

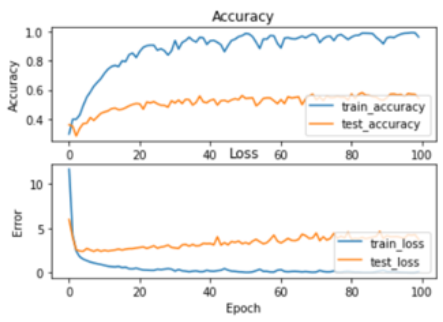
SINGER

0:00 / 0:00

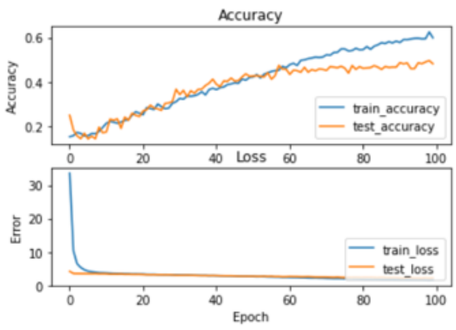
Music is the language of the spirit.

Experiments and results

The initial implementation used a MLP model with the test-train split of 0.35 on the MFCC values collected. This model was further modified by using regularizers and Dropout to reduce overfitting of the data. This model yielded the following results:



MLP with overfitting of data.

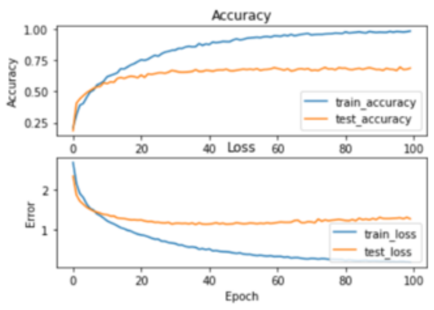


MLP with overfitting removed.

Training Loss: 0.1005 Training Accuracy: 0.9769
Testing Loss: 3.7465 Testing Accuracy: 0.5421

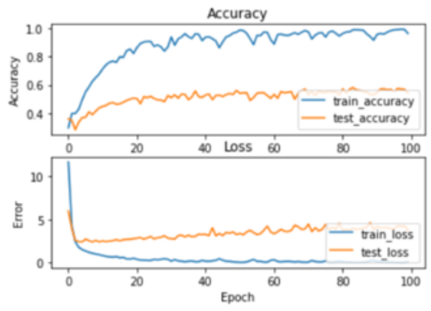
Training Loss: 1.7155 Training Accuracy: 0.5822
Testing Loss: 2.2801 Testing Accuracy: 0.4826

The second implementation replaced the MLP model for a CNN model with the same test-train split of 0.35 and same methods to reduce overfitting. Following were the results from this model:



CNN with overfitting of data.

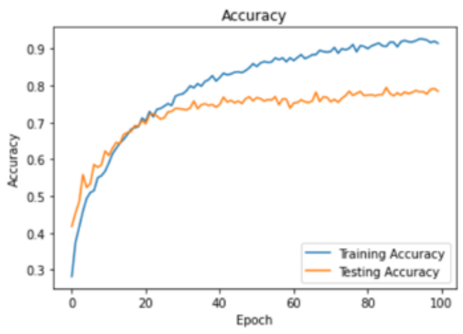
Training Loss: 0.0714 Training Accuracy: 0.9829
Testing Loss: 1.1788 Testing Accuracy: 0.6699

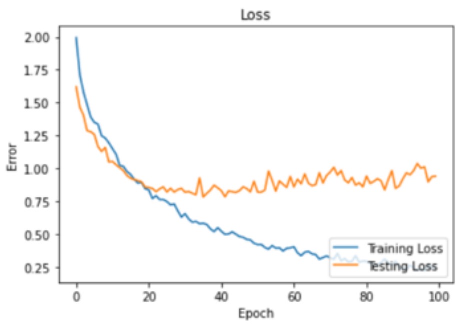


CNN with overfitting removed.

Training Loss: 0.824 Training Accuracy: 0.5845
Testing Loss: 1.117 Testing Accuracy: 0.5237

An RNN model was created by adding an LSTM (Long Short-Term Memory) Layer as the input layer to our existing CNN implementation. This considerably improved the performance of the neural network:





RNN

Training Loss: 0.2789 Training Accuracy: 0.9125
Testing Loss: 0.9411 Testing Accuracy: 0.7843

The final implementation was made with an SVM approach involving four different kernels: Radial Based, Linear, Polynomial and Sigmoid; as well as the LinearSVC class in sklearn (similar to linear SVM, but with better scalability for large number of samples). The following metrics were achieved:

Kernel	**Training Accuracy**	**Testing Accuracy**
Radial Based	0.5965	0.5393
Linear	0.9991	0.4839
Polynomial	0.5442	0.4886
Sigmoid	0.1679	0.1842
Class	**Training Accuracy**	**Testing Accuracy**
LinearSVC	0.9965	0.3624

Comparative study

On comparison of all the implementations achieved across four different architectures, it is observed that the RNN model has the highest value of Testing Accuracy, at 78.4% while also having a Training Accuracy of 91.2%. This performance was evident when the RNN model was able to correctly identify the genres of 9 out of 10 samples given as input during further testing, making it a clear candidate for the project.

The RNN model is followed by the regular CNN model that achieved nearly 67% Testing Accuracy, which reduced to a significantly lower 52.3% after adjustments to remove data overfitting. The MLP model comes in next, with a Testing Accuracy of 54% before overfitting removed and 48% after.

Among the SVM kernels implemented, the Radial Based shows the best Training Accuracy of almost 54%, outperforming both the MLP and CNN implementations with corrections for overfitting. The Linear and Polynomial kernels show similar performance as the corrected MLP model. The LinearSVC model showed a Testing Accuracy of 36.2%, outperforming only the Sigmoid kernel SVM which recorded an extremely low value of 18.4%.

Implementation	**Testing Accuracy**
RNN	0.7843
CNN	0.6699
MLP	0.5421
SVM: Radial Based Kernel	0.5393
CNN without Overfitting	0.5327
SVM: Polynomial Kernel	0.4886
SVM: Linear Kernel	0.4839
MLP without Overfitting	0.4826
SVM: LinearSVC class	0.3624
SVM: Sigmoid Kernel	0.1842

Conclusion

RNNs can use their internal state to process inputs in the form of sequences. This makes them suitable for tasks such as handwriting recognition, speech recognition or in the case of this project, audio processing and recognition.

Due to the high accuracy of the RNN based model during both testing and training phase, it will be implemented in the final version of this project. A second RNN model was created and trained for the classification of Indian genres, which managed to score a remarkably similar Testing Accuracy of 79.3%, and a Training Accuracy of 81.8%.

The implementation is then completed with a user interface in the form of a web application achieved using Flask, for an intuitive and attractive front-end solution.

References

Chien-Chang Lin, Shi-Huang Chen, Trieu-Kien Truong and Yukon Chang, "Audio classification and categorization based on wavelets and support vector Machine," in IEEE Transactions on Speech and Audio Processing, vol. 13, no. 5, pp. 644-651, Sept. 2005, doi: 10.1109/TSA.2005.851880.

Thiruvengatanadhan Ramalingam and P. Dhanalakshmi, “SPEECH/MUSIC CLASSIFICATION USING WAVELET BASED FEATURE EXTRACTION TECHNIQUES”, Journal of Computer Science 10 (1): 34-44, 2014, doi:10.3844/jcssp.2014.34.44

Gursimran Kour and Neha Mehan, “Music Genre Classification using MFCC, SVM and BPNN”, International Journal of Computer Applications (0975 – 8887) Volume 112 – No. 6, February 2015

Jitendra Ajmera, Iain McCowan and Hervé Bourlard, “Speech/music segmentation using entropy and dynamism features in a HMM classification framework”, Speech Communication, Volume 40, Issue 3, 2003