Music Genre Detection Using LSTM and Augmented Audios

Achyut Karnani
Student id: 33243387
ak19g21@soton.ac.uk

1. Introduction

In the following paper, implementation of 2 algorithms of neural networks are discussed to perfrom audio classification. We use (i) an LSTM model (ii) GAN to generate augmented audio and implementing the same LSTM architecture as before. The data used for this project is sourced from [1] website. The audio files here are collected from sources like music radios, CDs, audio recordings, etc. It consists of 100-30 seconds audio clips(.wav files) of 10 music genres. Genres are: blues, classical, country, disco, hiphop, jazz, metal, pop, reggae and rock. The audio files are split into 3 sec clips for more training set. The dataset was randomly split in to training set(70%), validation set(20%) and testing set(10%).

2. Feature Extraction

In order to perform feature extraction and audio transformation, librosa library was used [2]. Most prominent features, Mel Frequency Cepstral Coefficients (MFCC) were extracted from the audio files to represent our feature set. They are obtained by splitting the audio files into short frames, applying filter bank to power spectrum of each frame, taking logarithm on the filterbank energies and then applying discrete cosine transformation on it.Different dataset for 20 and 128 MFCC features were created respectively. For extraction the hop length was set to 255 and number of samples was set to 128 [4]. The obtained mfcc features were then mean-centered and standardised.

3. LSTM-1

The first Neural Network model chosen is a combination of two stacked lstm layers followed by two dense layers. The input to the first LSTM layer is of size (259,20) or (259,128) depending on the number of mfcc features. The output from it is of size 128 units in sequence which is fed to the second LSTM layer. The output from the second LSTM layer is 128 units but not in sequence is given to a Dense layer with 64 units.And the output of the first Dense layer is given to a dense layer with 10 units. The first Dense layer has relu as its activation function and the second one has softmax in order to do classification. We chose Adam as our optimiser and Sparse Categorical Cross Entropy as the loss function to train the models. We choose accuracy as our metric to evaluate the model performance.

4. GAN-LSTM

For the second algorithm, WaveGAN was implemented in order to generate augmented audio files [5]. And its resulting audio files were used to train a model for genre classification. WaveGAN uses CNN architecture to generate audio signals. The code for implementation of WaveGAN is sourced from [3] work. During training, the generator tries to produce real audio signals while discriminator tries to distinguish between fake and real samples. Separate GAN models were generated for each genre. GAN architecture implemented was trained on 30-seconds original clip for 50 epochs to generate 50 audio samples of 4 seconds. The GAN is updated 5 times for a particular batch. These 4 seconds samples are then trimmed to 3 seconds. Feature extraction is implemented on it and is then passed through the lstm architecture for training a classifier. The hyperparameters, architecture, optimisers and loss function for this is the same as LSTM-1.

4.1. Performance

LSTM-1 when trained using 20 MFCC features achieved a training accuracy of 68.56%, validation accuracy of 61.53% and testing accuracy of 62.26%. When trained with 128 MFCC features, it achieved a training accuracy of 82.89%, validation accuracy of 74.92% and testing accuracy of 72.6%. The models didn't overfit nor underfit. As shown in Fig.1, The training on 20 MFCC features was increasing, while the training began to overfit after the 39th epoch when trainied with 128MFCC features.

GAN-LSTM when trained with 20 MFCC features achieved a training accuracy of 100%, validation accuracy of 90.26% and testing accuracy of 26.8%. When trained with 128 MFCC features, it achieved a training accuracy of 90.75%, validation accuracy of 100% and testing accuracy of 13.82%. These numbers indicate that the model is overfitting. This poor performance can be attributed to multiple factors. One potential reason is the small dataset used for the classifier(50-per genre), as well as the fact that the generated audio files contained mostly noise. The noise from the GANs can be attributed to training on small dataset as well as less training for less number of epochs. This can be improved by increasing the number of epoch and by increasing the training dataset.

References

[1] GTZAN Dataset - Music Genre Classification https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification.

[2] librosa — librosa 0.10.0 documentation.

[3] tfworldhackathon/scripts at master · harrystuart/tfworldhackathon.

[4] Work w/ Audio Data: Visualise, Classify, Recommend.

[5] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial Audio Synthesis, Feb. 2019. arXiv:1802.04208 [cs].

A. ILLUSTRATIONS AND TABLES

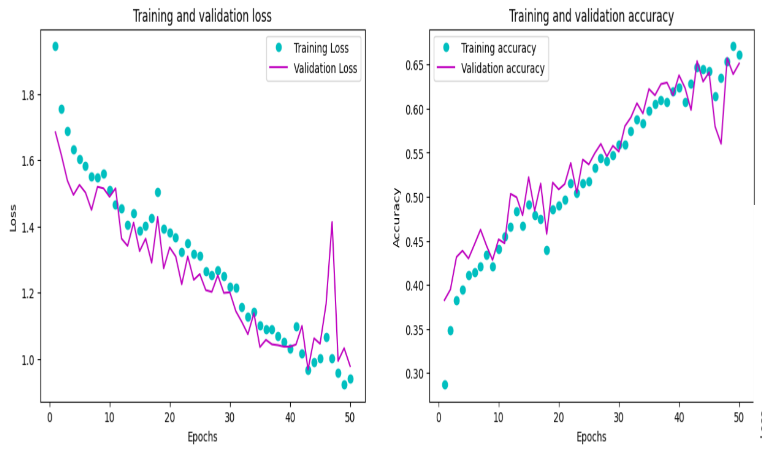

Figure 1: LSTM-1 for 20 MFCC features

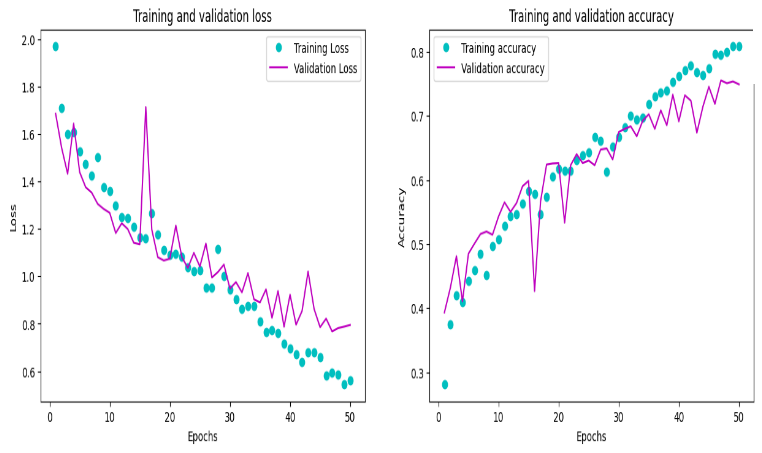

Figure 2: LSTM-1 for 128 MFCC features

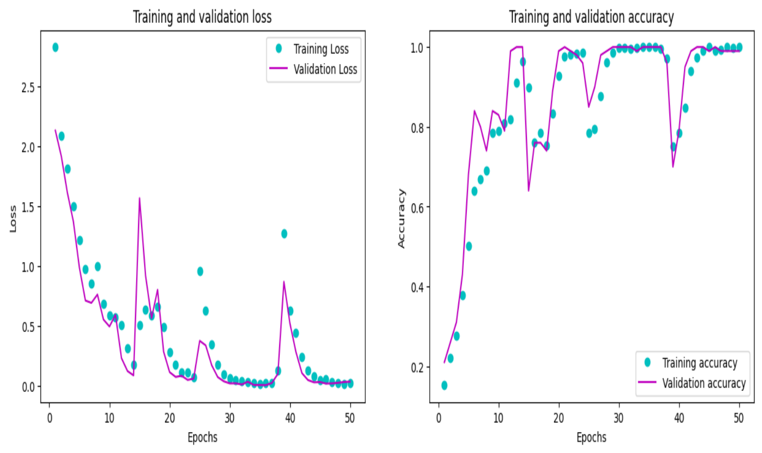

Figure 3: GAN-LSTM for 20 MFCC features

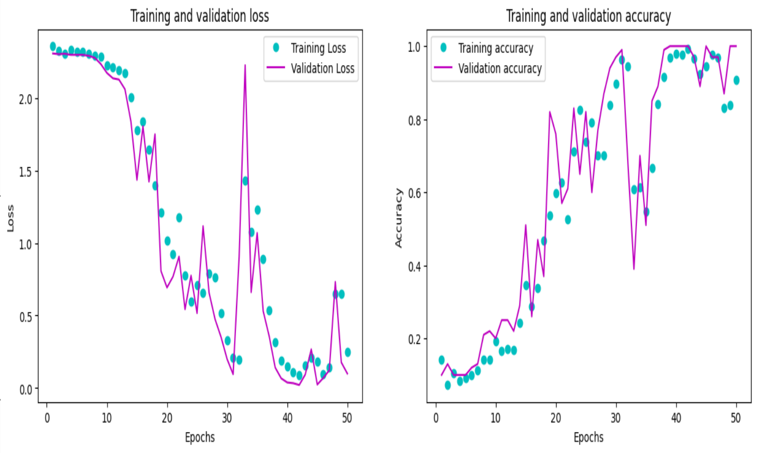

Figure 4: GAN-LSTM for 128 MFCC features

Model	Train_Acc	Val_Acc	Test_Acc
LSTM-1(20) :	0.6856	0.6153	0.6226
LSTM-1(128) :	0.8289	0.7492	0.7267
GAN-LSTM(20):	1.000	0.9026	0.2680
GAN-LSTM(128):	0.9075	1.0	0.1301

Table 1: Results on training and testing set