CHARLOTTE

ITCS 5156-Applied Machine Learning

Final Project

Music Genre Classification

Hareesh Bahuleyan. 2018. Music Genre Classification using Machine Learning Techniques. arXiv:1804.01149v1

Submitted By

Sanket Revadigar (801203510)

Instructor

Prof. Minwoo Lee

INTRODUCTION

The massive rise of unstructured data, including music data, drives researchers to look for new ways to manage it. Indexing, categorization, and clustering that works. Data in the field of music these tasks are in the focus of Music Information Retrieval in this situation. The first studies were conducted in the 1960s of the twentieth century, but significant progress can be seen at the turn of the twenty-first century: in 2000, the International Society of Music Information Retrieval (ISMIR) was founded, and in 2005, the Music Information Retrieval Evaluation eXchange (MIREX) competition for MIR tasks was launched.

The difficulty of organizing, categorizing, and summarizing music materials on the web is a hot topic in autonomous music information retrieval right now. On-line music databases such as mp3.com and Napster are examples of such endeavors. One of the most essential features of these online databases' genre structures is that they are defined by human professionals as well as amateurs (such as the user), and that the labeling process is time-consuming and costly. Because defining a specific definition of a music genre is extremely difficult, and many music sounds sit on genre boundaries, music genre classification is now done primarily by hand. These difficulties arise from the fact that music is an evolving art form, with performers and composers influenced by music from other genres. However, because they are formed of similar types of instruments, have similar rhythmic patterns, and have comparable pitch distributions, audio signals (digital or analog) of music belonging to the same genre have certain properties [7]. This shows that automatic musical genre classification is a possibility.

PROBLEM STATEMENT

People typically carry their favorite tunes on their smartphones. Songs come in a variety of genres. We can deliver a categorized list of songs to the smartphone user using deep learning algorithms. Deep learning techniques will be used to construct models that can classify audio files into different genres. We will analyze the performance of our trained model after it has been trained. Blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock are some of the genres.

My project is an expansion of previous work, in which I will utilize a different dataset to develop a second model, and I have used the first model from previous work as a reference to create the

second model. The prior research combined the dataset with the other dataset to construct a multi-model, but I simply used the second dataset to create the new model and compare it to the others.

MOTIVATION AND CHALLENGES

Categorizing music files according to their genre is a difficult task in the world of music information retrieval. Categorizing music files according to their genre is a difficult problem in the world of music information retrieval. It would be beneficial for audio streaming services such as Spotify and iTunes, as well as users, to be able to automatically classify and tag music in their libraries based on genre.

- It's human nature to want to make sense of our surroundings and experiences. This is the most basic reason for music classification: it allows us to comprehend what we hear and describe it to others.
- There is a growing worry around the world that music classification restricts creative freedom. People perceive it as putting new music into old boxes, or, even worse, as shoving creations out of those boxes because they don't match a pre-existing sound. This issue is logical from the artist's standpoint, as disgruntled fans frequently use genre comparisons to communicate their dislike for new music.
- Although genre classifications provide clarity and identification for music journalists, musicians, and A&R departments, the most important reason for categorizing music on a human level is to improve listening experience.

There are two major challenges with this problem:

- Musical genres are ill-defined terms. So much so that people frequently disagree over a song's genre.
- Extracting distinguishing features from audio data that may be fed into a model is a difficult undertaking.

SUMMARY OF THE APPROACH

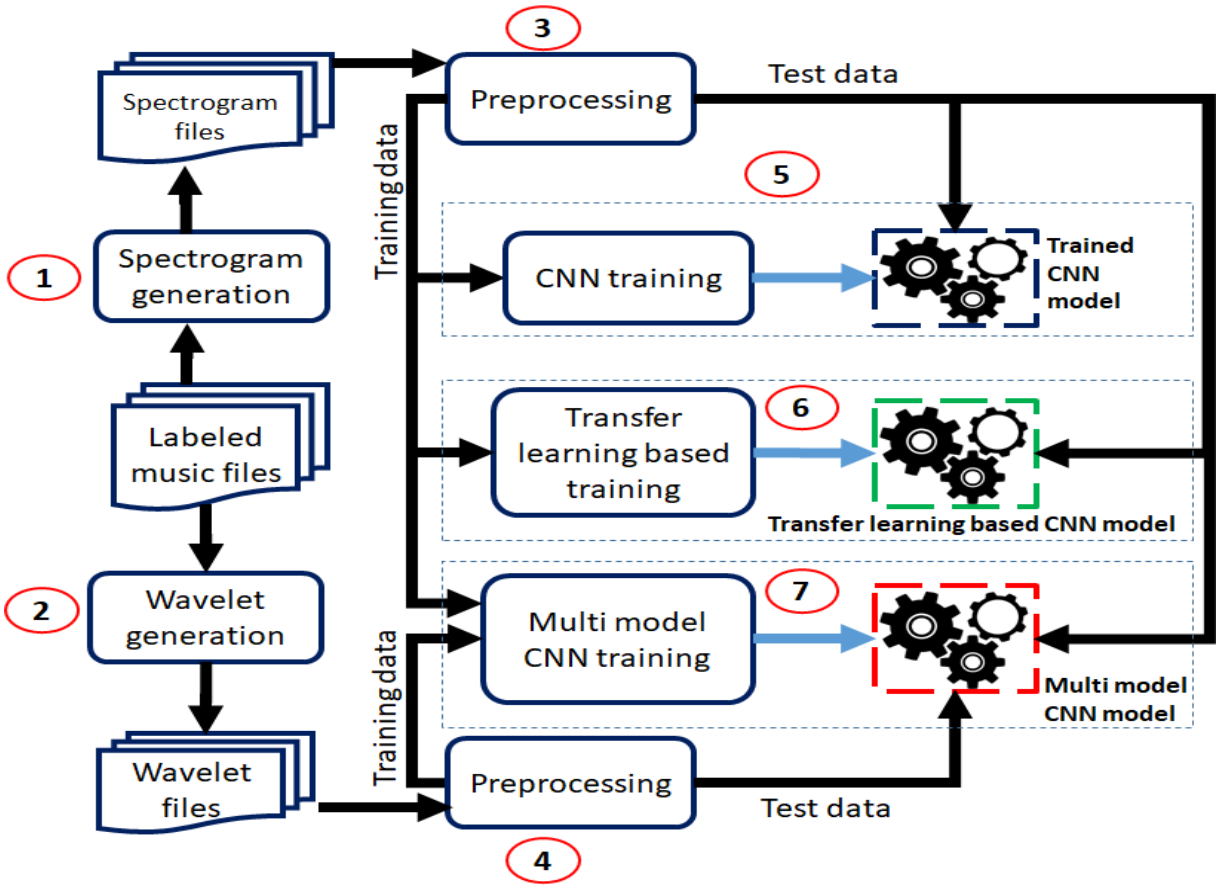

Figure 1

The Figure 1 depicts a high-level summary of the task genre classification process. Three different deep learning models were created. However, I utilized the initial CNN model and used Spectrograms to generate the first model. The second CNN model was built using the Wavelets data. Steps for generating training and testing data using general picture preprocessing like each image is of size (256, 256, 3). The 1st model has obviously performed better with accuracy of 52% as it was created by experts and the model I have created with different dataset is not very accurate sitting at around 32%.To convert the audio files into proper spectrograms and wavelets, I have used librosa library as its very will know library to handle audio files. And in the original work also they clubbed Spectrogram and Wavelet dataset to create the third model which didn't improve the accuracy from the 2nd model, so it makes sense why using the wavelet data alone yields poor results.

BACKGROUNDS AND RELATED WORK

SURVEY OF THE LITERATURE

Lam Hoang. 2018. Literature Review about Music Genre Classification. In Woodstock ’18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY. ACM, New York, NY, USA: - Listed significant Neural Network techniques such as Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN). And the usage of the dataset GTZAN is explained properly and why is so popular and accessible.

Hareesh Bahuleyan. 2018. Music Genre Classification using Machine Learning Techniques. arXiv:1804.01149v1 [cs.SD]: - Different types of classifiers are used like Logistic Regression, Random Forest, Gradient Boosting, Support Vector Machines. As these are ML classifiers, features were chosen manually for audio files central Moments, RMSE, Tempo, MFCC, etc.

Lonce Wyse. 2017. Audio spectrogram representations for processing with convolutional neural networks. arXiv preprint arXiv:1706.09559: - Discussed about Spectrograms and their usage in the Convolutional Neural Networks and shows why Spectrograms are ultimate way to classify the audio files.

PROS AND CONS OF THE SURVEY

Pros

- Leant about the GTZAN dataset and its usage.
- Leant about the Convolutional Neural Network.
- Learnt about the data preprocessing of the audio files that will be suitable for the model.
- And also learnt about how CNN stacks up against other ML classifiers.

Cons

- The code for few papers are not given
- Understanding the ML features is very difficult and very time consuming for the system
- Accuracy is still low even using spectrograms along with CNN.
- Processing audio files is still difficult and time consuming.

RELATION TO OUR APPROACH

I learned a lot about CNN and Spectrograms, as well as how the dataset can be used in the new model in general. The work I've chosen is extremely comparable to the publications; for example, the work I'm doing now uses Convolutional Neural Networks with the GTZAN dataset, and the dataset processing is very similar utilizing the librosa package. The spectrograms in the papers are similar to the ones I'm using in this project. All I needed to do was learn more about Wavelets and how they represent audio.

METHOD

DATASET

The GTZAN Genre Collection was the dataset I used .This dataset was utilized in a well-known genre categorization work published in 2002. There are 100 songs in each of the ten genres (blues, classical, country, disco, hip hop, jazz, metal, pop, reggae, and rock) (each 30 second samples). They were all.wav files. The GTZAN dataset is proposed in the bulk of research papers on this topic. GTZAN, which is frequently seen in Kaggle, is a collection of 1000 music snippets, each lasting 30 seconds. There are 100 audio samples in each of the ten genres in this collection, including blues, classical, country, disco, hip hop, rock, metal, pop, jazz, and disco. The GTZAN dataset has been referenced in over 100 published CS papers on the same issue, and it is widely regarded as one of the most well-known public datasets for music genre recognition. The data is split into 60:40. The audio file are converted into images and stored in train and test folders.

A spectrogram is a visual way of representing the signal strength, or “loudness”, of a signal over time at various frequencies present in a particular waveform. Not only can one see whether there is more or less energy at, for example, 2 Hz vs 10 Hz, but one can also see how energy levels vary over time.

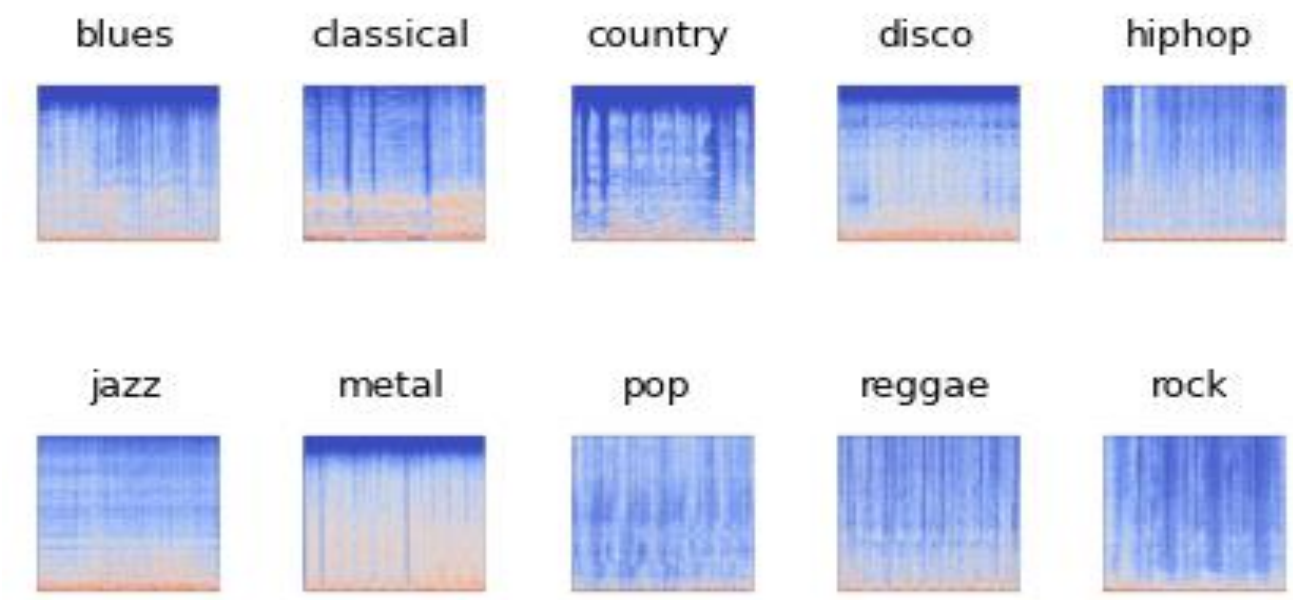

Figure 2 Spectrograms

A wavelet is a wave-like oscillation with an amplitude that begins at zero, increases or decreases, and then returns to zero one or more times. Wavelets are termed a "brief oscillation". A taxonomy of wavelets has been established, based on the number and direction of its pulses. Wavelets are imbued with specific properties that make them useful for signal processing.

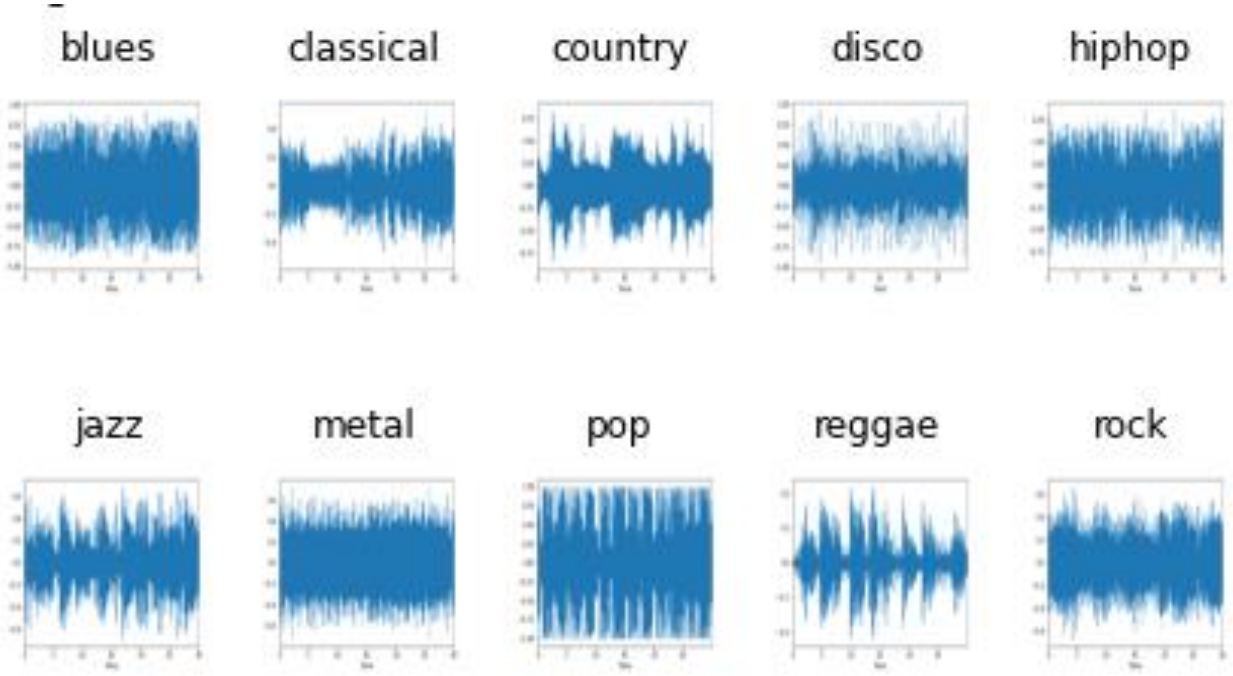

Figure 3 Wavelets

CONVOLUTIONAL NEURAL NETWORK

This method has been utilized extensively in a number of research publications on music genre classification. Several spectrograms obtained from audio recordings are used as inputs, and their patterns are extracted into a 2D convolutional layer with appropriate filter and kernel sizes. Because of the model's efficacy in identifying visual details, spectrogram is cited in CNN. Used the GTZAN dataset to extract the audio files into various types of spectrograms using Librosa library. Those spectrograms were taken as binary inputs of a 2D CNN model, which were formed using Keras library. The layers were also created using TensorFlow library. A 2D convolutional layer was presented at input shape 256x256x32. It contained a 2D NumPy array from the inputs that would be passed to the max-pooling layer, which would then operate a matrix that was half the size of the input layer. After preprocessing the data, created the first deep learning model. Then constructed a Convolution Neural Network model with required input and out units. The final architecture of our CNN model is shown in Figure 04. I use only spectrogram data for the training and testing. CNN model is trained for 500 epochs with Adam optimizer at a learning rate of 0.0001. Categorical cross-entropy as the loss function.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 256, 256, 32)      896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 128, 128, 32)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 128, 128, 32)      9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 64, 64, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 64, 64, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 32, 32, 64)        0
_________________________________________________________________
dropout (Dropout)            (None, 32, 32, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 65536)             0
_________________________________________________________________
dense (Dense)                (None, 128)               8388736
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1290
=================================================================
Total params: 8,418,666
Trainable params: 8,418,666
Non-trainable params: 0
_________________________________________________________________
```

Figure 4 CNN Model

LIBRARIES USED

Librosa- librosa is a python package for music and audio analysis. It provides the building blocks necessary to create music information retrieval systems. Used to convert dataset.

TensorFlow- TensorFlow is an end-to-end open source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in ML and developers easily build and deploy ML powered applications.

Keras- Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages.

Matplotlib- Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

Seaborn- Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

NumPy- NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

Sklearn- Simple and efficient tools for predictive data analysis · Accessible to everybody, and reusable in various contexts · Built on NumPy, SciPy, and matplotlib.

OpenCV- OpenCV provides a real-time optimized Computer Vision library, tools, and hardware. It also supports model execution for Machine Learning (ML).

Pickle- The pickle module implements binary protocols for serializing and de-serializing a Python object structure. “Pickling” is the process whereby a Python object hierarchy is converted into a byte stream, and “unpickling” is the inverse operation, whereby a byte stream is converted back into an object hierarchy.

EXPERIMENTS

EXPERIMENTAL SETUP

There are 2 .py files (1 for spectrogram and 1 for wavelet) to convert the audio files to images dataset, to run these .py files I have used Spyder IDE. I have used Jupyter notebook to perform Exploratory Data Analysis, Model Training and Model Evaluation, which is present in the main.ipynb file. GTZAN dataset is used which is around 1.3 GB. There are different libraries used for this experiment as mentioned above.

TEST RESULTS

Spectrograms

Figure 05 shows the training and validation losses and model performance in terms of accuracy for the spectrogram the one which is included in the previous work.

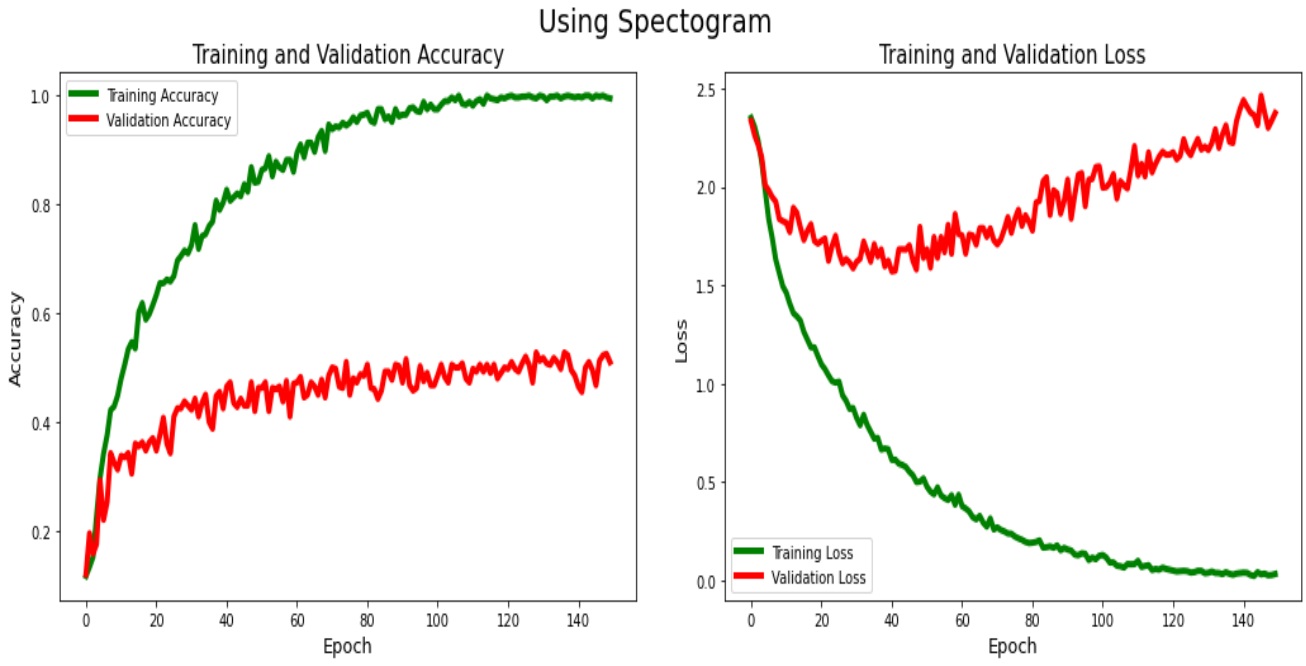

Figure 5

As we can see that I have set to run the model for 150 epochs which takes around 1 hour to run, the training accuracy is almost at 100%, and the testing/validation accuracy is above 50%, which

is not great but a right direction for audio files and its processing. And also training loss is decreased with the increase in epochs and strangely validation loss decreases up to 50 epochs and then increases until 150 epochs.

Figure 6 shows the confusion matrix of the CNN model for Spectrogram on the test data. CNN model was able to classify “classical” genre music with the highest score. CNN performed worst for “Rock” and “reggae” genre music.

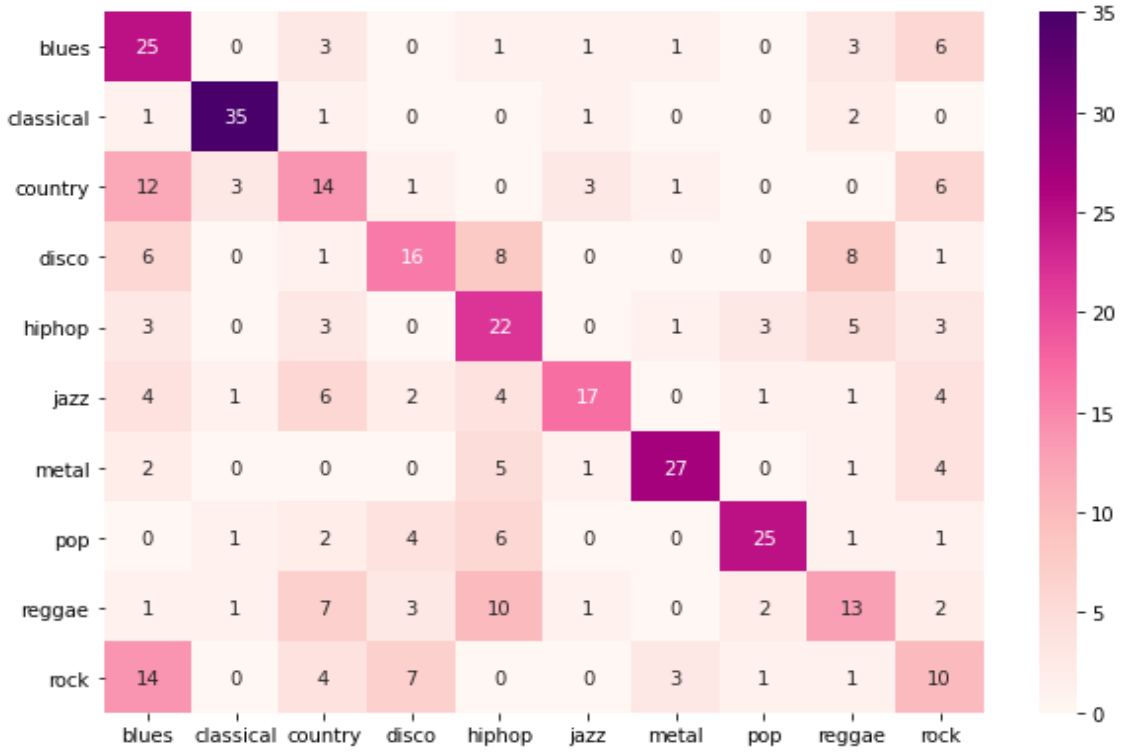

Figure 6

Wavelets

Figure 7 shows the training and validation losses and model performance in terms of accuracy for the spectrogram the one which is included in the previous work.

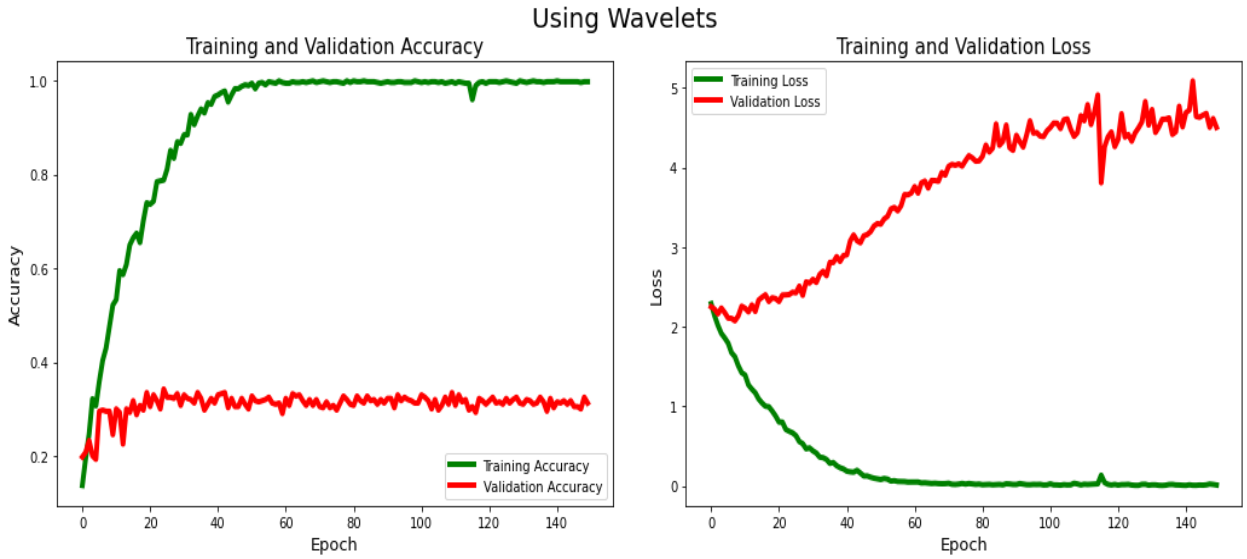

Figure 7

As we can see that I have set to run the model for 150 epochs which takes around 1 hour to run, the training accuracy is almost at 100%, and the testing/validation accuracy is above 30%, which is not great but a right direction for audio files and its processing. And also training loss is decreased with the increase in epochs and strangely validation loss decreases up to 10 epochs and then increases until 150 epochs.

Figure 8 shows the confusion matrix of the CNN model for Wavelets on the test data. CNN model was barely able to get 50% in jazz and pop, and it performs very badly in rest all the genres. So we can conclude that Spectrograms are way better than wavelets to classify the genres.

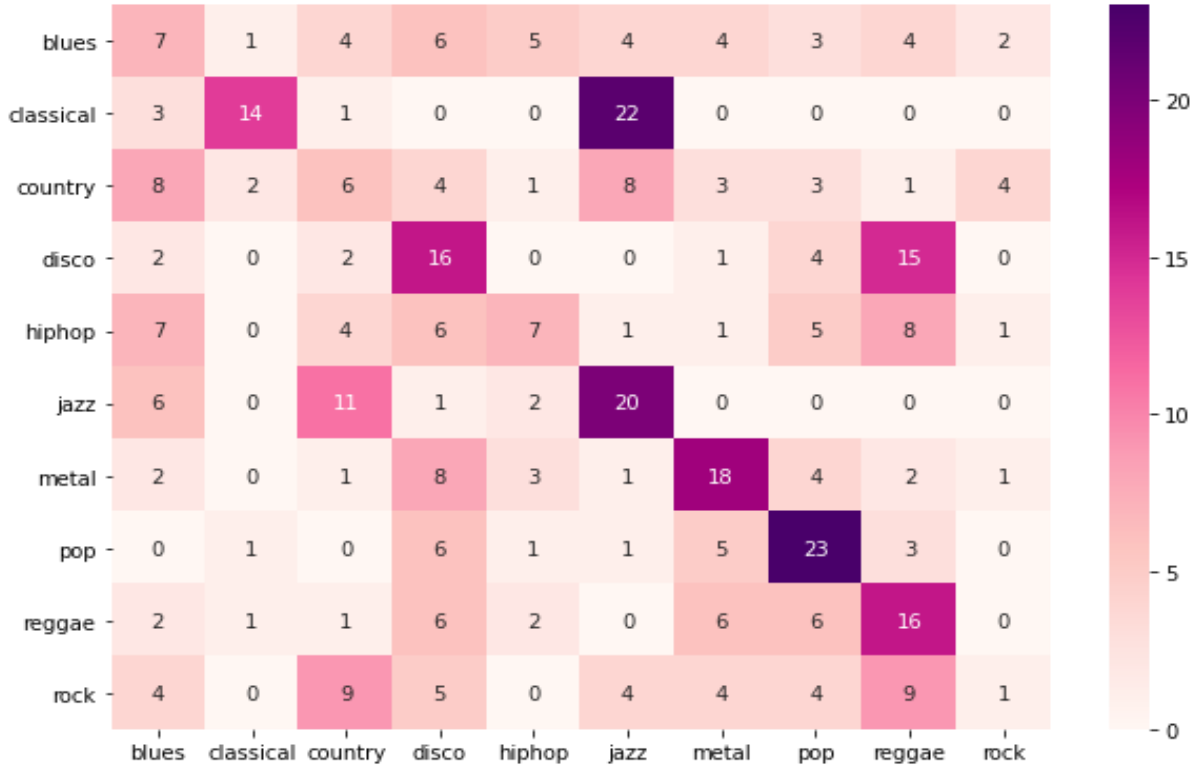

Figure 8

DEEP ANALYSIS

Comparison

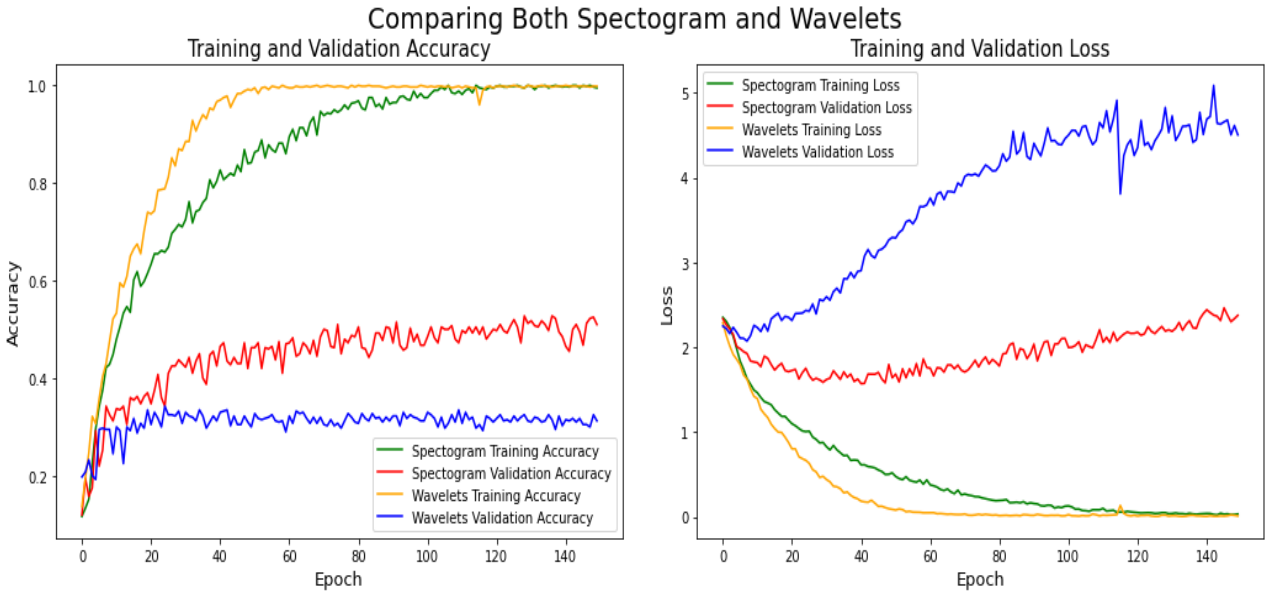

Figure 9

The 2nd model was created by me, using a different dataset i.e. Wavelets, and as expected it performs very poorly. The 1st model which was the previous work, used Spectrograms performed reasonably well around 52%, but my model is able to score only 32%, so it can be improved in many ways by changing the CNN model layers and trying all other possibilities. This topic is still new in ML field and will improve in the near future, and many new libraries might be introduced to handle audio files. Converting the audio files to image is a little time consuming. We can also see in the comparison chart that spectrogram is a better option than wavelet. And also the training accuracy for wavelets reaches 100 % more quickly than spectrogram. The validation loss is more in wavelets which explains why the prediction is worse through wavelets.

The flow of work, It was quite fun by the end, but tricky in the beginning, as it was difficult to find the proper code that I had planned to use, The repo did not explain properly on how to convert the audio dataset into images, so it take a long time to figure out on how to convert the dataset. Many of the syntax were wrong or deprecated as the versions for different libraries were changed so I need to figure out and modify the code to make the code execute properly.

CONCLUSION

This study presents a successful application of CNNs to a MIR task such as Genre Recognition. First of all, I have discussed the type of datasets most researchers selected for their papers. As can be seen, the majority of those papers implement genre classification via GTZAN due to its popularity and accessibility. Second of all, Convolutional Neural Network (CNN) and how it can be adapted to this topic. Each researcher has built his/her result by using a wide range of techniques on a regularly proposed dataset; thus, we cannot come to any conclusion about identifying the most optimum architecture. I personally learnt a lot about the vastness of the music industry and its genres and how important it is to the industry as well as the customers. The spectrograms and wavelets was quite interesting topics as I had never heard those terms before, so learning about them and how they represent the data was quite fun, and to play with results by the end of the project was also very fun.

CONTRIBUTIONS

Previous Work link- https://github.com/sawan16/Genre-Classification-using-Deep-learning

https://www.analyticsvidhya.com/blog/2021/06/music-genres-classification-using-deep-learning-techniques/

I have used the 1 of the 3 models that were presented in this repo, and created 1 more model using different dataset (Wavelet) as a standalone dataset, whereas in the previous work the 2 dataset were clubbed together and 3rd model was created, Using the new dataset alone was not good, as the accuracy was very low compared to Spectrogram dataset, and there might be several ways that I am not aware off which can increase the accuracy of both the models, especially the 2nd model using Wavelets. The previous work used 60:40 split in the data for training and testing, so I created a new dataset with a split of 70:30 but it turns out, the accuracy fell down by 7-8% with this new split so I reverted back to old 60:40 split. Proper explanation was not given on how to convert the audio files into images and the code was also incomplete, so I figured out the rest of the code and also created the image dataset which was needed for CNN model. And also as this repo is 2-3 years old, many of the syntax were changed, so I updated the code so that is was supporting the newer versions of the libraries, and I observed that the model was set to run for 500 epochs which was actual not yielding any valuable results, so I set the new Epoch value to 150 which was the sweet spot to get better accuracies and have a little headroom to stay over there. The comparison chart was not there in the original work, so I created the comparison chart on own with proper styling and colors.

REFERENCES

[1] Hareesh Bahuleyan. 2018. Music Genre Classification using Machine Learning Techniques. arXiv:1804.01149v1 [cs.SD]

[2] George Tzanetakis, Perry Cook. 2002. Musical genre classification of audio signals. IEEE Transactions on speech and audio processing 10(5):293– 302.

[3] Lonce Wyse. 2017. Audio spectrogram representations for processing with convolutional neural networks. arXiv preprint arXiv:1706.09559

[4] Lin Feng, Shenlan Liu, and Jianing Yao. 2017. Music genre classification with paralleling recurrent convolutional neural network. arXiv preprint arXiv:1712.08370 (2017).

[5] Deepanway Ghosal and MF Kolekar. 2018. Musical genre and style recognition using deep neural networks and transfer learning. In Proceedings, APSIPA Annual Summit and Conference, Vol. 2018. 12–15.

[6] Athulya KM et al. 2021. Deep Learning Based Music Genre Classification Using Spectrogram. (2021).

[7] Macharla Vaibhavi P Radha Krishna. 2021. Music Genre Classification using Neural Networks with Data Augmentation. (2021).

[8] https://github.com/sawan16/Genre-Classification-using-Deep-learning

[9] https://www.analyticsvidhya.com/blog/2021/06/music-genres-classification-using-deep-learning-techniques/#h2_6