Artificial Neural Network Audio Classification with Multi Layer Perceptrons

Roman Bellisari
CPE 646: Pattern Recognition and Classification (Spring 21)
Stevens Institute of Technology: 1 Castle Point Ter, Hoboken, NJ 07030
rbellisa@stevens.edu

Abstract

Abstract Audio files are one of many forms of unstructured data that can be used to solve a variety of problems. Within the field of machine learning, making class predictions with unstructured data such as audio files often requires significant amounts of preprocessing to achieve satisfactory results. The objective of this audio classification analysis is to utilize artificial neural networks to output a 10-class probabilistic classifier corresponding to selected music genres: Blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae rock.

1 Introduction

Introduction Audio classification is a very hot topic in the field of machine learning and spans a wide variety of applications across industries. Audio manipulation and classification exists at the intersections of signal processing, machine learning and physics. Machine learning professionals can use audio classification for speech detection, comparing urban construction sounds or even identifying emotion in speech.

1.1 The Dataset

The first step in generating a comprehensive audio classification system is to identify a dataset that properly models the problem. For the task of music genre classification, the key characteristics that define an optimal dataset are audio quality consistency, uniform audio file properties and logical predefined genre classifications. One dataset that fits all the needs of this problem is the GTZAN Genre Collection dataset compiled by George Tzanetakis [1]. The dataset includes 10 genres with 100 unique songs per genre, amounting to 1000 total tracks. All tracks are in lossless digital .WAV format and are exactly 30 seconds long. One additional key feature that makes the Marsyas dataset favorable for the task of audio classification is the standardized sample rate of 22050 Hz. This is useful when extracting features from the dataset, as it does not allow for over or under sampling of certain tracks.

1.2 Preprocessing Methodology

Machine learning models are unable to adequately model unstructured audio data on its own. To allow for proper adaptation to machine learning implementations, the raw digital audio signals must be converted into numerical values for the model to analyze. The audio vectorization process was implemented using various transforms to obtain Mel Frequency Cepstral Coefficients (MFCCs) for each sample. MFCCs are optimal for genre recognition and classification problems, because it covers perceived frequency in comparison to actual measured frequency. Because humans are able to identify subtle changes in pitch at low frequencies much more effectively than changes at high frequencies, the MFCC scale used for projects where class outputs are exposed to human subjectivity.

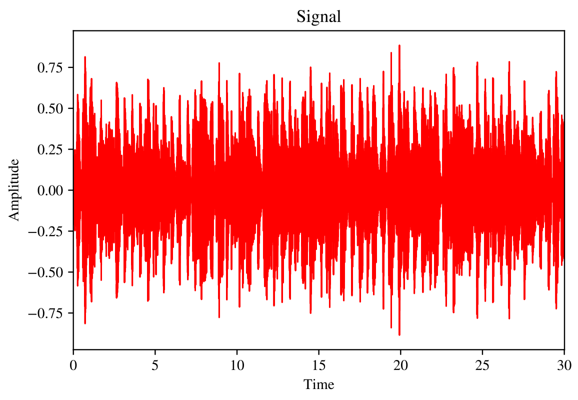

Figure 1: Example of Librosa Plot

2 Methodology

2.1 Technologies Used

Several numerical and analytical libraires were implemented within Python to develop a probabilistic classifier to address the music genre classification task. Numerical engines Numpy and Pandas were utilized to perform linear algebra computations and multi-dimensional matrix manipulation tasks [2,3]. All audio preprocessing tasks were performed using the Librosa library [4]. Performance analysis and data visualizations were implemented via Matplotlib to provide helpful insights on the dataset and the models performance [5]. Scikit Learn was used for dataset manipulation to split the data into training and testing samples [6]. Keras TensorFlow was utilized for model construction and serves as the foundation for the classification predictions [7].

2.2 Preprocessing Implementation

The first step towards music genre classification is to obtain a uniform dataset that the machine learning model can utilize. In order to extract these values, useful information must be obtained from each audio file within each genre folder. For every sample window in each audio file, the MFCC was obtained through a series of Fast Fourier Transforms (FFTs), Short Time Fourier Transform (STFTs) and Log Spectrogram analysis.

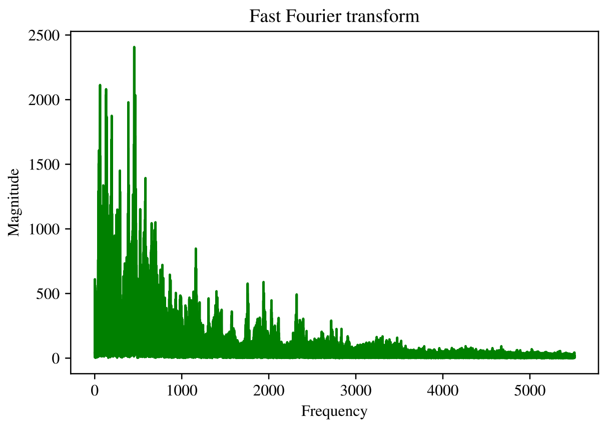

Figure 2: Example of Librosa Fast Fourier Transform

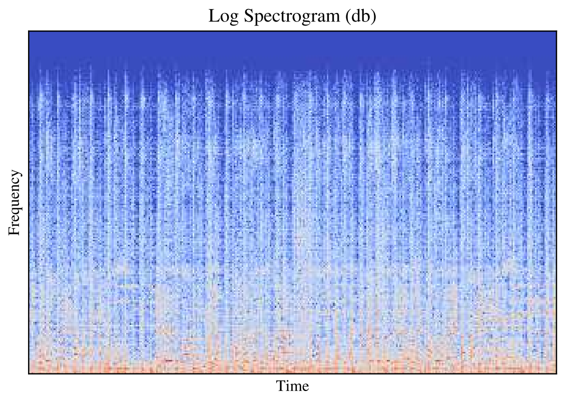

Figure 3: Example of Librosa Log Spectrogram

MFCCs provide data for frequency, magnitude and time. Each MFCC was computed with 13 unique features, a value commonly used with MFCC analysis [9]. This data is stored in a 13-dimensional array with lengths equal to the samples per track divided by the total number of MFCC windows. The total number of samples per track is calculated as the sample rate multiplied by the total track duration.

$$(22,050Hz \times 30sec) = Samples/track \qquad (1)$$

Another advantage of using the MFCC feature values compared to the log spectrogram variables is the issue of dimensionality. MFCC features are limited to 13 predetermined dimensions whereas the log spectrogram outputs with thousands of dimensions. When performing computationally intensive tasks such as model training, the increased is a key consideration as it plays a major role in computational complexity.

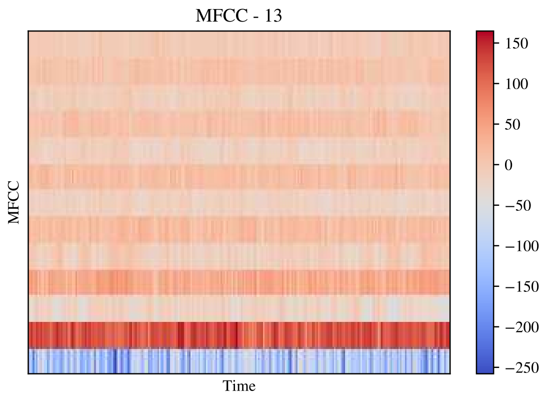

Figure 4: Example of Librosa MFCC Output

2.3 Exploratory Data Analysis

Once all of the audio data has been converted into useful numerical values, a model can be used to identify and predict certain trends within the training dataset. Before training the model, the total dataset was split into training and testing subsets using Scikit Learn.

Because the extracted MFCC audio data exists in high dimensions, each sample in the training dataset contains dimensions of 130 by 13. Performing exploratory data analysis through dimensionality reduction, components of each sample were extracted using the principal component analysis (PCA) technique. It is evident that the data is not linearly separable, further narrowing down the model selection process.

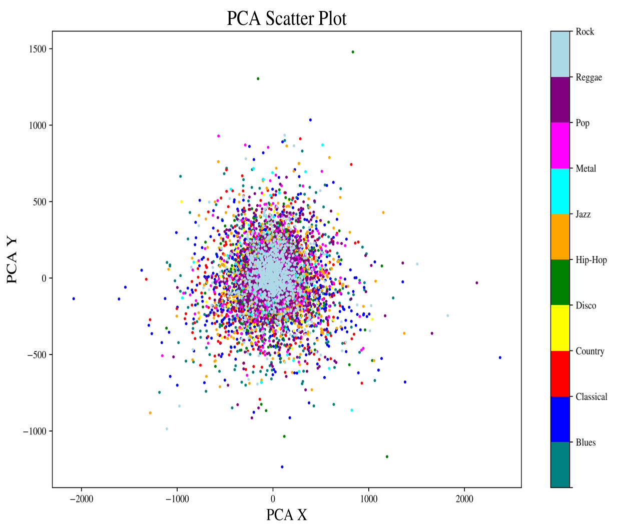

Figure 5: PCA 2D Decomposition

2.4 Model Selection

Given the high input dimensions and the desire to output a multiclass probability distribution, a neural network design was selected as the primary model. In efforts to reap the benefits from the artificial neural network (ANN) architecture, a multilayer perceptron (MLP) model was implemented with 4 layers using Keras. The model weights were fit using the backpropagation algorithm to find the optimal weights for the training data. The dimensions of MLP were the following 1690 for input layer, 1024 for layer one, 512 for layer two, 256 for layer three, 64 for layer four, and finally the output layer has 10 nodes corresponding to the probabilistic genre classifications. To evaluate the performance of the model, the sparse categorical cross entropy loss function was compared with the accuracy of the training and validation datasets. These variables help identify model overfitting and allow for the optimization of certain parameters.

3 Machine Learning Techniques

3.1 Computational Complexity

In efforts to reduce the total computational complexity and time required for parameter optimization, a reduced dataset was used. This dataset is identical to the full dataset but with one song per genre compared to 100, the time required for model training was reduced by a factor of 100.

3.2 Parameter Configuration

In efforts to reduce the total computational complexity and time required for parameter optimization, a reduced dataset was used. This dataset is identical to the full dataset but with one song per genre compared to 100, the time required for model training was reduced by a factor of 100.

After evaluation of the models performance on both the training and validation dataset, several techniques were implemented in efforts to improve performance. Firstly because of the extremely quick convergence, the learning rate was modified allowing the model to converge more gradually. The default Keras learning rate of 0.01 was reduced to 0.0005 through trial and error. The relationship between convergence speed and total epochs inverse. Another reason to decrease learning rate and increase total weight updates was to obtain more fluid learning curves. With a larger learning rate and smaller epochs variable assigned, the loss and accuracy curves converge too rapidly to accurately visualize.

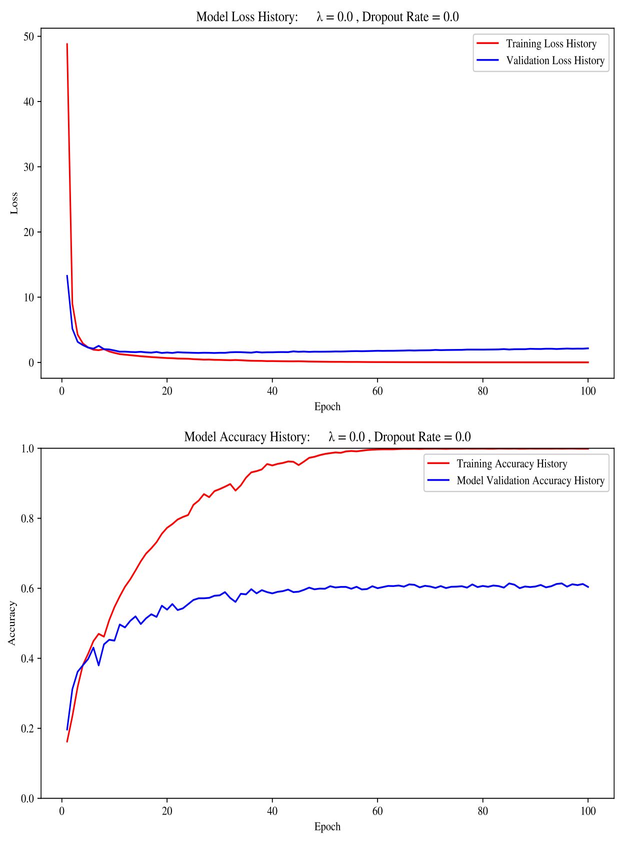

Figure 6: Standard Learning Functions

3.3 MLP Optimization

3.3.1 Hyperparameterization

Upon the successful learning rate configuration, the training and validation error must be addressed to determine the models performance on unseen data. When validation loss far exceeds training loss, the model is defined as overfit. There were several measures introduced to combat model overfitting. Firstly, a kernel regularizer was implemented to penalize values straying too far from the desired target. A L2 or ridge regularization coefficient was introduced as lambda with a default value of 0.01. After loss and accuracy plots were used to compare the convergence rates, it was determined that the final value for the lambda regularization coefficient was 0.001.

$$L_2 Regularization = \lambda \sum_{i=1}^{n} \theta_i^2 \tag{2}$$

3.3.2 Dropout Layers

In addition to L2 regularization, the dropout layers were also introduced after the three dense model layers to fight against overfitting. The Keras default dropout rate of 0.3 was changed to 0.2 due to the strong improvement with the L2 regularization term. Lastly, in order to fully defend against model overfitting, the final technique that was implemented was early stopping. Because the total number of weight updates or epoch updates was chosen to be 100, this leaves enough room to stop updating the weights if the training loss function varies from the validation loss function by a certain value.

3.3.3 Activation Functions

Given the unique architecture of ANN's, another issue that needs to be addressed is the vanishing gradient problem arising from the backpropagation algorithm. Because traditional hyperbolic or sigmoid activation functions such as softmax have gradient derivatives that evaluate very close to zero towards the limits, this vanishing gradient problem becomes apparent with these activation functions. In order to avoid this issue, the Rectified Linear Unit (ReLU) activation function was utilized as the activation for the three dense layers with the softmax function used only for the 10-class output layer.

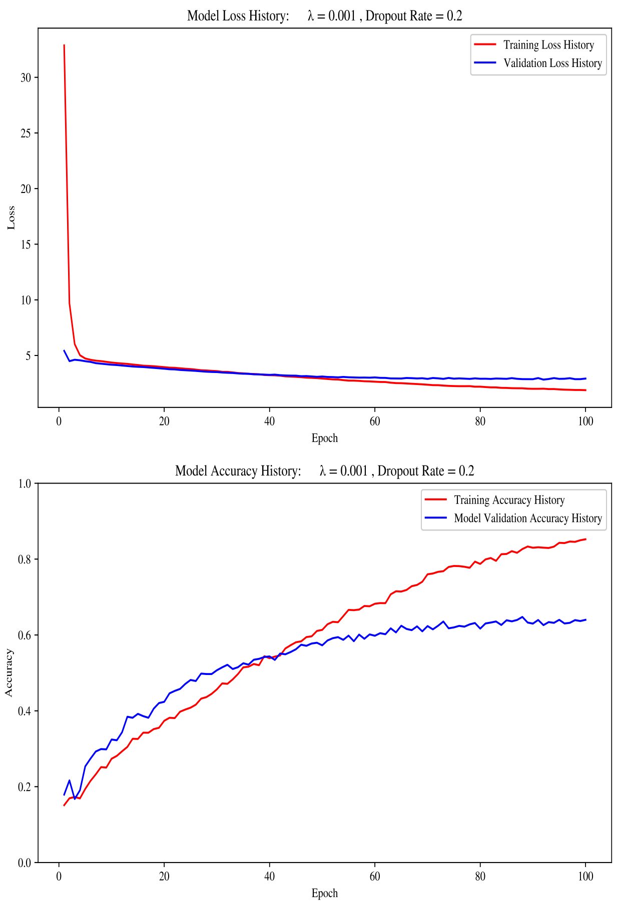

Figure 7: Regularized Learning Functions

4 Results

4.1 Classifications

After optimizing the model to reduce error and overfitting, the final accuracy on the validation dataset was 64%. Given the probabilistic output, the obtained validation accuracy is favorable due to the fact that a multiclass probability output provides more information than just a single class output. For example, it is more beneficial to know that a misclassified song has high class distributions of pop, country, and rock rather than just knowing the single class of the highest output.

4.2 Cross Class Confusion Matrix

The covariance matrix in figure 9 highlights this finding by visually representing common output class combinations that are misclassified. The defined linear trend across the confusion matrix indicates strong single class performance, whereas lighter areas away from this intersection indicate increased misclassifications. The confusion ma-

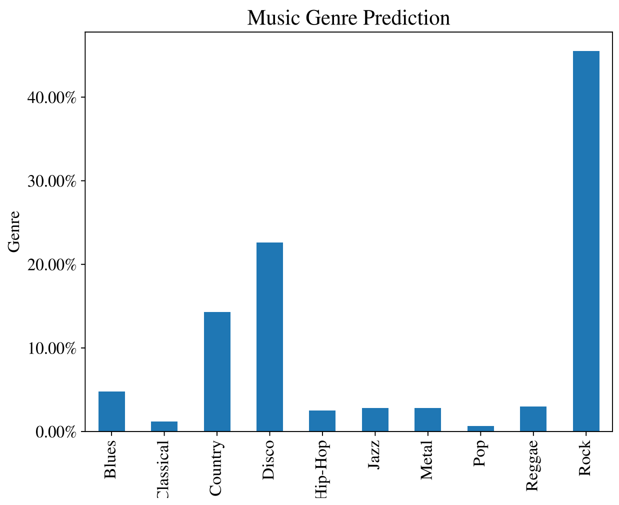

Figure 8: Multiclass Probability Distribution Output

trix also reveals which genres have the strongest performance on this model. It is evident that metal and classical music have the best performance metrics whereas country, disco, hip hop and rock have the weakest performance measures in comparison.

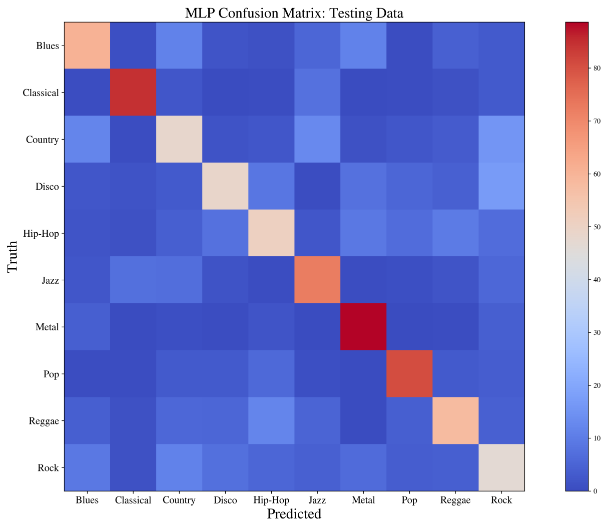

Figure 9: Validation Confusion Matrix

5 Future Considerations

5.1 Model Architecture

Given a larger timeframe and additional resources, there are several areas that could be explored to improve performance. By introducing more data into the training set, this may help to improve long term model performance on both seen and unseen data. Another obvious solution to improve performance is to increase the total MLP layer depth. Given additional time and computational recourses, this may be an effective method to improve overall performance.

5.2 Witholding Data

One popular solution to performance enhancement within audio processing is to remove certain classes with high similarities to other classes. With this specific dataset, the selection of songs included in the pop and rock genre varies widely. Because rock music often has similarities to pop, hip hop, disco or country, the misclassifications are inevitable. One option to solve this problem is to remove this genre altogether, creating a 9 class probabilistic classifier and removing these samples from the training data.

5.3 External Rescources

Although this may not be ideal if this class is a desired output, another possibility is to introduce ensemble learning with different models/features. Although it would drastically complicate the analysis, one example of additional features that may be beneficial is Spotify's Developer API feed. Spotify provides numerical values corresponding to certain song characteristics such as danceability, Valence, Energy and Tempo [8].

References

[1] George Tzanetaki Marsyas GTZAN Genre Collection,
`http://marsyas.info/downloads/datasets.html`

[2] Numpy Documentation,
`https://numpy.org/doc/`

[3] Pandas Documentation,
`https://pandas.pydata.org/docs/`

[4] Librosa Documentation,
`https://librosa.org/doc/latest/index.html`

[5] Matplotlib Documentation,
`https://matplotlib.org/stable/contents.html`

[6] Scikit Learn Documentation,
`https://scikit-learn.org/stable/`

[7] Keras TensorFlow Documentation,
`https://keras.io/api/`

[8] Spotify Developer API Documentation,
`https://developer.spotify.com/discover/`

[9] MFCC Feature Selection

https://www.researchgate.net/post/why_most_of_
reasearchers_consider_13_MFCC_coefficients_for
_analysis_of_speech_or_images