MUSIC GENRE CLASSIFICATION FINAL REPORT

Samir Hossain
Student# 1008272101
samirm.hossain@mail.utoronto.ca

Rohin Marok
Student# 1007930107
rohin.marok@mail.utoronto.ca

Zaid Husseini
Student# 1007898287
zaid.husseini@mail.utoronto.ca

Somesh Karthi
Student# 1007935558
somesh.karthi@mail.utoronto.ca

ABSTRACT

This document is the final report for team 4's Music Genre Classification deep learning project —-Total Pages: 9

1 INTRODUCTION

Throughout the ages, music has been prevalent in the transferal of knowledge and experiences in cultures worldwide. It is used as a vessel of storytelling and entertainment that plays a crucial role in the development of society by encouraging creativity and self-expression.

With over 200 million songs available on major streaming platforms and even more songs played in the real world, it is evident that there is a myriad of this media form for us to consume and interpret [1]. In order to truly appreciate the beauty of music, users have to be able to browse and filter this art form to their own needs to find a message that inspires them.

Thus, our team has been motivated to develop a project involving music genre classification. Our goal is to create a recognition model that is able to determine and label a music genre to a specific inputted song. This idea of categorizing music allows the human brain to associate different types of music with their preferences based on prior experience.

The muddled line involved in genre classification stems entirely from the human brain and our way of viewing music as an art form and our perspective of the world as a whole. If we don't entirely know the answer ourselves, how will a computer do it for us?

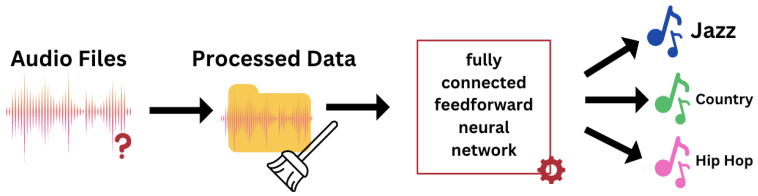

Figure 1: Inputted Audio files get classified into genre

This idea of music classification coming from the brain is exactly why deep learning will be used for this project. As already illustrated, music classification comes from us listening to a lot of music and thus making our assumptions and genres based on what is familiar and what is not. A model will have to do the same through an artificial neural network that mimics the human brain. There is no binary answer for genre classification, but creating a model in this "human-like" nature is what will allow it to be useful. Deep Learning is specifically used for high-complexity decision-making and uses neural networks to learn from data and optimize its choices the more it trains.

2 Background and Related Works

1. Automatic Musical Genre Classification of Audio Signals by George Tzanetakis and Perry Cook (2002)

This paper is one of the earliest works focused on automatic music genre classification. Tzanetakis and Cook proposed a method for classifying musical genres based on features extracted from the audio signal itself, such as timbre, pitch, and rhythm.

2. Music Genre Classification using Machine Learning Techniques by Pádraig Cunningham, et al. (2005)

This research uses machine learning techniques like decision trees and nearest neighbors to classify music genres. It covers the data preprocessing steps as well as the algorithmic aspects involved in genre classification, providing a broad overview of the methodologies.

3. Deep Learning for Music Classification by Yann LeCun (2014)

This work highlights the use of deep neural networks, especially Convolutional Neural Networks (CNNs), for the task of music genre classification. It demonstrates the advantages of using deep learning techniques over traditional machine learning algorithms for this specific task.

4. Genre Classification of Spotify Songs using Lyrics, Audio Previews, and Album Artwork (2017)

Though not entirely focused on the audio processing aspect of genre classification, this research paper introduces further real-life context that can be applied to determine genre from songs, including lyric analysis and album artwork analysis.

5. An Industrial-Strength Audio Search Algorithm (2006)

This research paper evaluates an industry standard song identification algorithim used by popular app Shazam. It dives deep into the mathematical models behind the signal and sound processing involved with identifying songs.

3 Data Processing

To prepare the data for our project, we meticulously gathered a dataset of 1000 songs, distributed evenly across ten genres: blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, and rock. These songs were sourced from YouTube and the Free Music Archive and then converted into WAV format for uniformity and ease of processing. Organizing these files into genre-specific folders enabled a structured approach to processing.

The preprocessing of this data involved several crucial steps. Firstly, we utilized the ‘AudioSegment.from_file()‘ function from the ‘pydub‘ library to load each song. To create uniform-length samples, each song was trimmed to produce two versions: a 30-second clip and a 3-second clip. For songs shorter than the desired length, we used ‘AudioSegment.silent(duration=pad_duration)‘ to pad them with silence, ensuring consistency in duration across the dataset. This process was scripted to automatically adjust the duration, as seen in the code snippet provided in the previous response. Once trimmed, these audio segments were exported in WAV format using ‘song_30sec.export(trimmed_song_path, format=’wav’)‘.

For the 30-second trimmed versions of the songs, we generated spectrograms using the ‘wavfile.read‘ and ‘spectrogram‘ functions from the ‘scipy‘ library. These spectrograms were visualized and saved using the ‘matplotlib‘ library, which facilitated the generation of a visual representation of the audio data’s frequency spectrum over time.

Feature extraction was a key part of our data preprocessing. We employed various functions from the ‘librosa‘ library to extract audio features such as chroma_stft, rms, spectral_centroid, spectral_bandwidth, rolloff, zero_crossing_rate, harmony, perceptr, and tempo, along with 20 mel-frequency cepstral coefficients (MFCCs). The mean and variance of each feature were computed using ‘np.mean()‘ and ‘np.var()‘, providing a comprehensive dataset for each audio sample[2].

The processed data was organized into two pandas DataFrames, ‘df1‘ and ‘df2‘, corresponding to the 30-second and 3-second audio clips, respectively. These DataFrames included columns for each

extracted feature, the filename, and the genre label, and were later saved as CSV files for easy access and use in model training and validation.

To facilitate effective model training and evaluation, we split our dataset into training, validation, and testing sets in a 70:15:15 ratio. This split ensured that our model had a substantial amount of data for training while also providing adequate and separate datasets for validation and testing purposes. This division is critical for assessing the model's performance and generalizability on unseen data, thereby ensuring a robust evaluation of its classification capabilities.

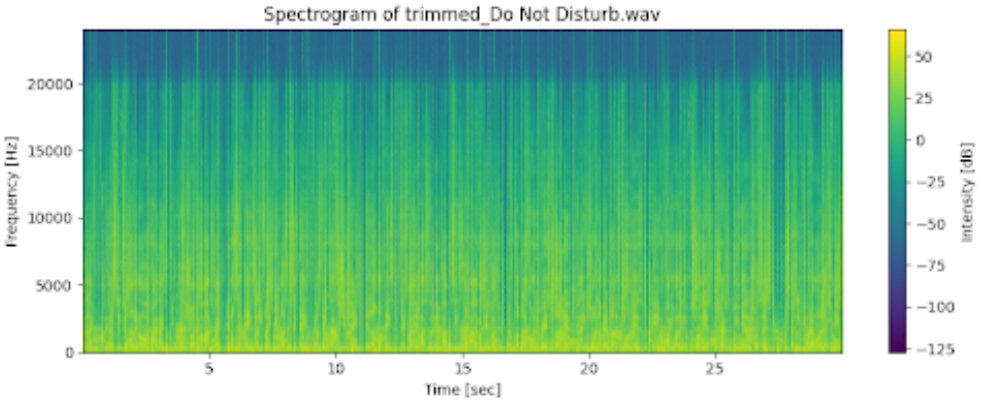

Figure 2: An example of a Mel Spectrogram for a hip-hop song

4 Model Architecture

```python
class MusicGenreClassifier(nn.Module):
    def __init__(self):
        super(MusicGenreClassifier, self).__init__()
        self.network = nn.Sequential(
            nn.Linear(in_features=X_train.shape[1], out_features=512),
            nn.ReLU(),
            nn.Dropout(0.2),
            nn.Linear(in_features=512, out_features=256),
            nn.ReLU(),
            nn.Dropout(0.2),
            nn.Linear(in_features=256, out_features=128),
            nn.ReLU(),
            nn.Dropout(0.2),
            nn.Linear(in_features=128, out_features=64),
            nn.ReLU(),
            nn.Dropout(0.2),
            nn.Linear(in_features=64, out_features=10)
            # Removed the nn.Softmax layer
        )

    def forward(self, x):
        return self.network(x)
```

Figure 3: Python Code showing Architecture

The final neural network model is a fully connected feedforward neural network designed for music genre classification. After trying several models and using different types of inputs we selected this model due to its performance in comparison to its peers. Here's a concise description of its architecture:

Input Layer: The first layer is a fully connected linear layer. The number of input features is dynamic, based on the number of features in the training dataset. This flexibility was created as it

allows the network to adapt to different sizes of input feature sets which made testing and evaluating different sets of data inputs easier.

Hidden Layers and Activation Functions:

First Hidden Layer: 512 neurons, followed by a ReLU activation function.

Second Hidden Layer: 256 neurons, followed by a ReLU activation function.

Third Hidden Layer: 128 neurons, followed by a ReLU activation function.

Fourth Hidden Layer: 64 neurons, followed by a ReLU activation function.

Each of these layers is followed by a ReLU (Rectified Linear Unit) activation function, which introduces non-linearity to the model, enabling it to learn more complex patterns in the data.

Dropout Layers: After each ReLU activation, a dropout layer with a dropout rate of 0.2 is applied. This means 20% of the neurons are randomly dropped out during training. This technique helps in reducing overfitting by preventing complex co-adaptations on training data.

Output Layer: The final layer is a fully connected layer with 10 neurons, corresponding to the number of music genres to be classified. This layer outputs the raw scores (logits) for each genre.

Forward Pass: The forward method defines the forward pass of the network, applying the sequential model to the input data.

Softmax Activation (Excluded): Notably, the nn.Softmax layer is excluded from the network. In many applications, especially those involving classification, the final softmax activation is often applied externally, particularly when calculating loss during training (e.g., using nn.CrossEntropyLoss which internally applies softmax).

This architecture, with its sequence of linear layers and non-linear activations, dropout for regularization, and a final linear layer for classification, appears to be well-suited for tackling the problem of music genre classification based on extracted features from audio data.

5 BASELINE MODEL

For our project, we initially established a baseline model using a simple Convolutional Neural Network (CNN). This baseline was pivotal in guiding the project's initial direction. Our Simple CNN, designed for image classification, was adapted to process mel-spectrograms derived from song data, aiming to predict music genres [**Figure 2**]. We were able to create this model quite quickly due to its simple architecture, which was very beneficial to us as it allowed us to rapidly establish a foundational understanding of our data and the problem at hand. This initial model acted as a springboard, enabling us to iterate and refine our approach more efficiently, and focus on more complex models and techniques to improve accuracy and performance in subsequent stages of our project.

The architecture of our Simple CNN is as follows:

First Convolutional Layer (conv1): It processes input with 4 channels using 16 filters of size 3x3, maintaining spatial dimensions through a stride and padding of 1.

ReLU Activation (relu1): Applied post conv1, it introduces non-linearity, enhancing the model's learning capacity.

Max Pooling (pool1): With a 2x2 window and a stride of 2, it reduces spatial dimensions, aiding in computational efficiency and overfitting control.

Second Convolutional Layer (conv2): This layer increases the depth from 16 to 32 channels, using the same kernel size, stride, and padding as conv1.

ReLU Activation (relu2): It's applied again for added non-linearity post conv2.

Max Pooling (pool2): Mirrors pool1 in functionality.

Adaptive Average Pooling (avg_pool): Ensures a uniform output size of 1x1 for each feature map, making our network adaptable to varying input sizes.

Fully Connected Layer (fc): Transforms the flattened output of avg_pool into final classification outputs, matching the number of music genres (num_classes).

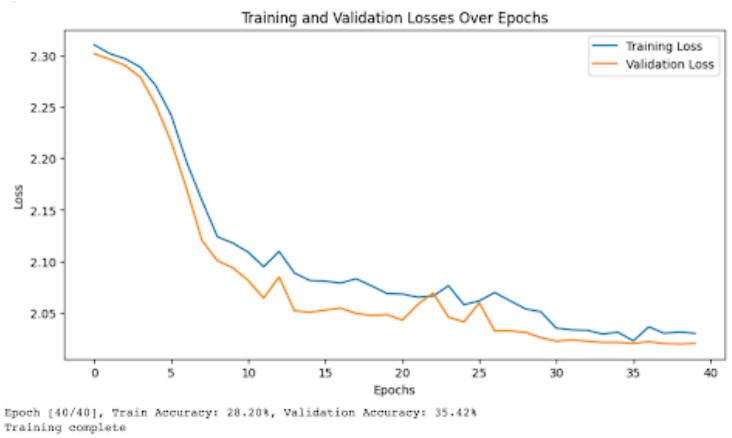

Figure 4: A graph of training and validation loss measured over 40 epochs

This CNN model, though basic, was crucial as our starting point. It achieved a validation accuracy of 42%, indicating its limited yet foundational capability in genre classification. The graph above shows the training results of our baseline model which clearly indicate the model's limited potential, as after 15 epochs it was barely able to reduce loss and the loss was never able to dip below a loss of 2. We then progressed by building on this model to a more complex ResNet architecture, which improved our validation accuracy to 70% but from there we were unable to further improve our CNN implementation so we had to reconsider our choice of model. However, the SimpleCNN remained our comparative baseline, highlighting the advancements made with our other models. This approach aligns with common practices in AI research, where baseline models provide essential benchmarks for progress evaluation.

6 Quantitative Results

Reflecting on the training process of our neural network models, we found that training loss is an essential measure of a model's fit to the training data. The sharp decrease in training loss observed in the graphs on the next page indicates that our model is learning effectively. [Figure 5] Our baseline model exhibited a persistent training loss above 2, signifying a struggle in learning. In stark contrast, our enhanced model showed a significant reduction in training loss, indicating a much more capable learning process.

When considering validation accuracy, which is the proportion of correct predictions, we were pleased to see that our model achieved an impressive 84% validation accuracy, far surpassing the baseline's 42%. The percent increase in accuracy from our baseline model to the actual model was 100%. This high level of accuracy suggests our model's ability to generalize and accurately predict on data it hasn't encountered before. However, we did notice a leveling off at this point, which could suggest that we've reached the upper limits of what the model can learn with its current architecture and the complexity of the dataset at hand.

The architecture of our improved model, which includes multiple hidden layers and dropout layers, is evidently more adept at capturing the nuanced patterns within our data, a capability that the simpler baseline CNN model lacked. This architectural complexity is mirrored in the enhanced validation accuracy we've achieved.

Drawing from these insights, it's clear that our model's sophisticated architecture and the implementation of regularization techniques have yielded quantitatively superior performance compared to the baseline model. Yet, the plateau in validation accuracy at 84% hints that further gains may necessitate not just architectural tweaks but also broader changes, such as more innovative feature engineering, data augmentation strategies, or the exploration of alternative model families to advance our model's performance even further.

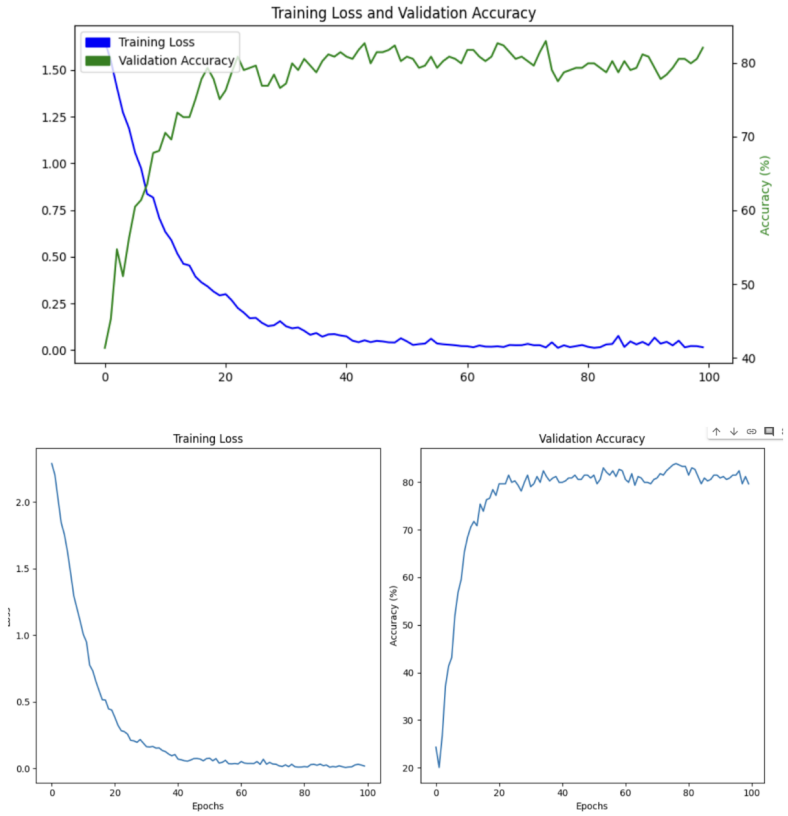

Figure 5: Final Model Training Loss vs. Validation Accuracy

7 Qualitative Results

In providing qualitative insights into our model's performance, we present a sample output showcasing the model's classification process for a well-known track. As illustrated in figure 6, the model outputs a set of logits corresponding to each genre, which it then translates into a ranking of the three most probable genres. For the iconic "Stayin' Alive" by the Bee Gees, the model accurately identifies "Disco" as the top genre, with "Pop" and "Blues" following. This correct identification underscores the model's capability to capture the defining characteristics of a genre throughout a song.

However, the model's performance nuances become more apparent in songs where genre characteristics are not uniform across the track. For instance, certain hip-hop songs that sample or incorporate elements from classical, rock, or blues genres can lead to skewed predictions. These cases highlight the challenges of genre classification in the presence of cross-genre influences and affirm the need for a robust and diverse training dataset. Moreover, the model has shown a tendency to mix up genres with close similarities, such as metal and rock or blues and jazz. Yet, it consistently ranks the correct genre within the top predictions, indicating that while it may not always pinpoint the exact genre, it understands the broader musical context.

Overall, the model excels when songs exhibit consistent genre characteristics throughout their length, affirming the quantitative findings where the model achieved a high accuracy rate. The sample outputs were deliberately chosen to represent both the model's strengths in identifying clear genre-specific features and its limitations when dealing with songs that blur genre lines.

These qualitative results, particularly the model's occasional missteps with hybrid or cross-genre songs, provide a valuable context for understanding its quantitative performance. They also guide

future model refinements, suggesting a focus on enhancing the model's ability to parse and weigh diverse musical elements within a single track.

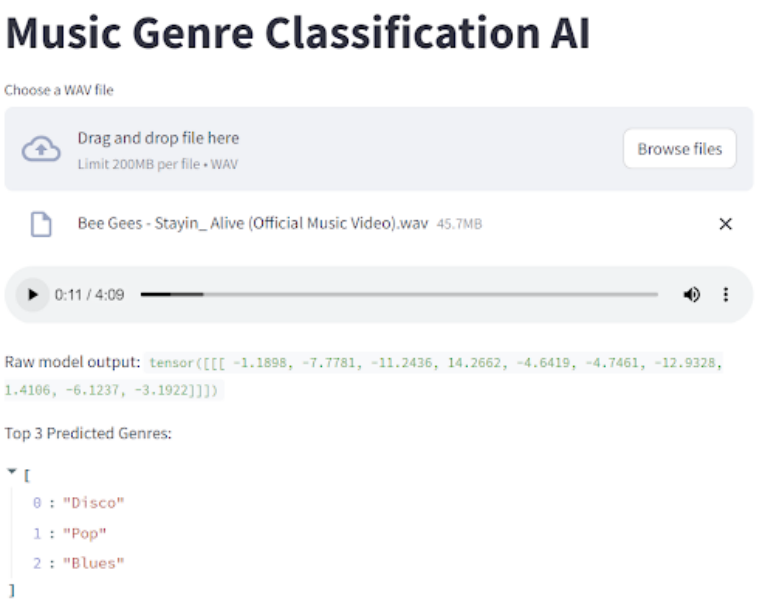

Figure 6: Screenshot that illustrates the model's output and its evaluation using a new song

8 Model Evaluation with New Data

In the development of our music genre classification model, we strategically reserved 15% of our original dataset as a hold-out test set to evaluate the model's performance. This set was specifically chosen to ensure no overlap with the data used during the training and validation phases, providing an unbiased assessment of the model's predictive power.

Furthering our commitment to a robust evaluation, we curated an additional dataset composed of 100 songs, with 10 distinct tracks representing each of the 10 genres. Careful consideration was given to select not only different songs but also different artists from those present in the training and validation datasets. This was a critical step to ascertain that the model was learning to recognize the characteristics of genres, not just memorizing particularities of the artists' sounds.

We then ensured that the new music data was processed consistently in relation to the training data. This was done to guarantee the results were a good representation of the model's performance on new data. Inconsistencies in data processing can introduce bias or noise, leading to incorrect classifications or predictions. This consistency is key to ensuring that the model's performance accurately reflects its real-world effectiveness.

Upon testing the model, it demonstrated that it was adept at classifying genres on the unseen data. This was quantified by our evaluation metrics, where the model achieved an accuracy of 80.23%, a precision of 79.96%, recall of 80.02%, and an F1 score of 79.99%. Additionally, our model was designed to predict the top three most probable genres for each song as shown in figure 6, a feature that acknowledges and leverages the subtle distinctions and overlaps between genres, such as between jazz and blues or rock and metal.

The effectiveness of the model is visually represented in Figure 7, our confusion matrix, which captures the model's predictive accuracy across ten musical genres. For each song in our test set, we extracted 10 random 3-second samples and ran them through our model, ensuring a comprehensive evaluation. The rows of the matrix represent the true genres, and the columns represent the model's predictions for these samples. Numbers along the diagonal show the count of correct predictions for each genre, totaling 1000 tests, while the off-diagonal numbers represent instances where the

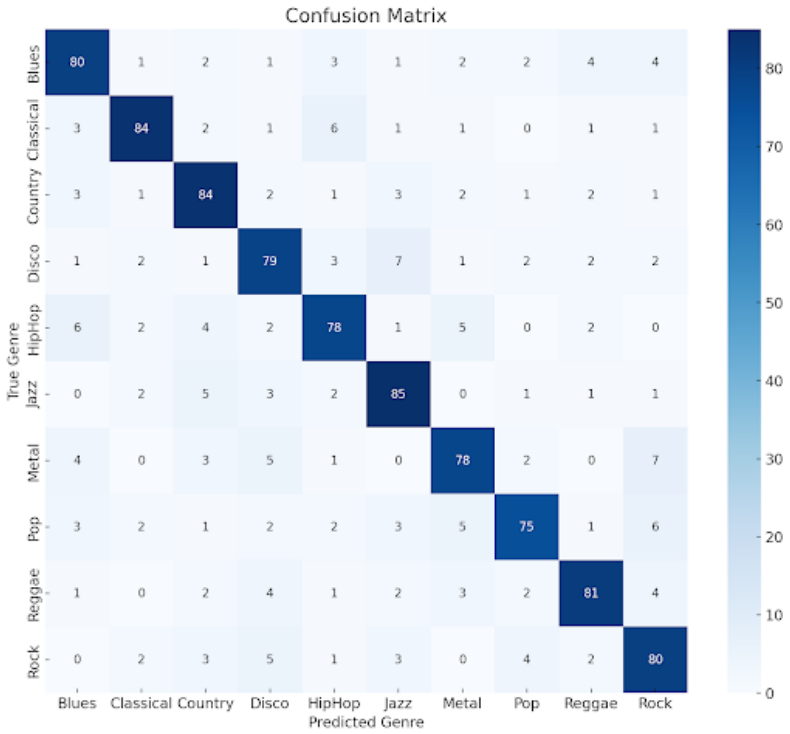

Figure 7: Confusion Matrix of the Music Genre Classification Model.

model's predictions did not match the true genre. The intensity of the darker shades corresponds to higher counts, signifying strong performance in genres such as jazz and metal. Additionally, this matrix confirms the model's capacity to distinguish between genres with similar acoustic features, like rock and metal, and points to areas for potential refinement, particularly in the differentiation of genres such as pop and reggae.

Our iterative and thorough approach to model evaluation on new, carefully curated samples has provided a reliable measure of its applicability in real-world scenarios. The impressive performance on these samples, verified by both quantitative metrics and visualizations in Figure 7, confidently demonstrate the model's readiness for practical music genre classification applications.

9 Discussion

The model's overall performance, in terms of the new/unseen data, exceeded the team's expectations. Our final model achieved an accuracy of approximately 80.23%, precision of 79.96% and an F1 score of 79.99% demonstrates its ability to classify a wide selection of songs across different genres. The confusion matrix in figure 7, reveals a high degree of accuracy in most genres, with it performing exceptionally well in genres such as jazz and metal where the songs have very distinctive features. For example, metal songs can be characterized by fast tempos, much distortion and aggressive rhythms. These elements are very well captured by the features measured in our CSV file for the songs (spectral patterns, centroid, chroma features, etc.). As such, a model trained on a dataset with a rich set of features can learn the 'signatures' of these distinctive genres more easily than those of more homogenous or overlapping genres.

Bearing this mind, we can extend this understanding to discuss why our final model performed much better than our baseline model. The baseline CNN was trained on image classification of spectrograms generated from 30 second and 3 second clips of the original samples. Spectrograms provide a visual representation of a song's different frequencies plotted over time, which is valuable, however they fail to capture other important components such as dynamics and timbre. Alternatively, the CSV files used to train the fully connect network contained many highly characteristic elements

of each song. The fully connected network's ability to understand these more complex and subtle features underscores the importance of comprehensive feature extraction in audio analysis. By capturing aspects such as chroma (pitch), spectral centroid (brightness), and MFCCs (timbre), the network could appreciate a fuller picture of the music, leading to more accurate genre classification [3]. These features likely provided the network with the necessary information to discern not just the loudness but also the quality and type of sounds that characterize different music genres.

From this, we learned the importance of choosing the correct features upon which to train our model that would go beyond surface-level characteristics. The success of our neural network suggests that a well-built model equipped with an extensive dataset can effectively navigate different audio clips to identify genres even in spite of similarities between genres. Going forward we may further explore feature selection and engineering on the original samples to enhance the model's sensitivity. This would allow for the model to better understand the nuances between similar genres and thus improve accuracy.

10 Ethical Considerations

There are various legal implications regarding the data collection for this model. Accessing the audio files of songs that are intellectual property of larger music labels and corporations requires permission, as the data is not openly sourced to us. In fact, songs sold on music distribution platforms like Spotify, Apple Music, and Deezer are not allowed to be downloaded as compressed formats due to these ownership laws, as doing so violates the terms and policies of all three platforms. This relates to the fact that most of the data used in this project has not been organically generated by us, and is rather purchasable digital products owned and created by others.

11 Project Sophistication

Overall, it can be stated that our project is highly sophisticated with respect to the course content. The data processing and collection stage required intensive research on what extracted features in an audio file can be exploited in a machine learning model [4]. Through this, we were able to completely transform the data into something usable, which was a process that goes beyond the complexity of prior labs. In addition, the design choice and implementation of a fully-connected feed-forward neural network has had a lot of consideration put into it to produce the most optimal sound processing model possible.

References

[1]"15 Most Popular Music Streaming Services." *Yahoo! Finance*, Yahoo!, finance.yahoo.com/news/15-most-popular-music-streaming-213711428.html. Accessed 1 Dec. 2023.

[2] N. Senavirathne, "Music genre prediction using audio features," Medium, https://medium.com/@navodas88/music-genre-prediction-using-audio-features-96dea4e71ad3.

[3] M. Lyons, "Music Information Retrieval," Medium, https://medium.com/@meganlyons_79212/music-information-retrieval-8a3facacf88e (accessed Dec. 1, 2023).

[4]Raval, — BY Param. "Solved Music Genre Classification Project Using Deep Learning." *ProjectPro*, www.projectpro.io/article/music-genre-classification-project-python-code/566#:~:text=From%20the%20several%20methods%20to,liveliness%2C%E2%80%9D%20etc.%20to%20describe. Accessed 1 Dec. 2023.