Audio Genre Classification using Deep Learning Methods

Anurathi Bala,[1] Joann Rachel Jacob, [1] Mitchell Leahy [1]

[1] Khoury College of Computer Sciences, Northeastern University
DS 5500 Capstone Project - Phase 1, 2023
bala.an@northeastern.edu, jacob.joan@northeastern.edu, leahy.mitch@northeastern.edu

Abstract

Audio genre classification is essential for various music applications, including recommendation systems and content organization. This project investigates the efficacy of different deep learning architectures, namely Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and Vision Transformers (ViT), for audio genre classification. The study utilizes the GTZAN and FMA Small datasets, offering a diverse collection of audio tracks spanning multiple genres.

To perform genre classification, acoustic features such as MFCC and Mel Spectrograms are extracted from the audio files in the GTZAN and FMA Small datasets. These features serve as the foundation for training and evaluating the CNN, ViT and LSTM. Extensive experimentation is conducted, and performance evaluation metrics, including accuracy, precision, recall, and F1 score, are calculated. The CNN and ViT models leverage spatial information present in the spectrograms, while the LSTM model effectively captures temporal dependencies in the MFCC sequences. Experimental results demonstrate that the LSTM model performs well, achieving 80.8% accuracy using MFCCs on GTZAN dataset, the CNN model achieves accuracies of 72% and 48% using Mel Spectrograms on GTZAN and FMA-Small datasets respectively. However, the Vision Transformer model exhibits lower accuracies of 30% and 41% on Mel Spectrograms of GTZAN and FMA datasets respectively and requires further improvement.

Through a comparative analysis of these architectures, considering their strengths and weaknesses, this project contributes to the understanding of deep learning-based audio genre classification and provides insights for future advancements in this field.

Keywords: Vision Transformer, Convolutional Neural Network, Recurrent Neural Network, Mel Spectrograms, Mel-frequency Cepstral Coefficient, Music genre classification, GTZAN dataset, FMA Small Dataset.

Introduction

Audio genre classification is a challenging task in the field of music information retrieval, aiming to automatically categorize audio tracks into distinct genres based on their acoustic characteristics. Accurate genre classification has various practical applications, such as music recommendation systems, content organization, and playlist generation. The task involves extracting meaningful features from audio signals and training machine learning models to classify them into predefined genre categories.

In this project, we delve into the problem of audio genre classification and propose a solution method that combines the use of Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and Vision Transformers (ViT) with different audio feature representations. Specifically, we explore the GTZAN dataset, a widely used benchmark in the field, and the FMA Small dataset, which provides a diverse collection of audio tracks spanning multiple genres.

The primary motivation behind this project is to investigate the effectiveness of deep learning architectures in handling audio genre classification tasks and to explore the impact of different feature representations on classification performance. We focus on two popular feature representations: Mel spectrograms and Mel-frequency Cepstral Coefficients (MFCCs).

Mel spectrograms provide a visual representation of the frequency content of an audio signal over time. They capture both the spectral and temporal characteristics of the audio, making them suitable for CNNs and ViT models, which excel at processing spatial information. On the other hand, MFCCs are commonly used for capturing the spectral characteristics of an audio signal and are well-suited for modeling temporal dependencies using LSTM networks.

Our approach involves preprocessing the audio data by computing Mel spectrograms for the CNN and VIT models. These spectrograms are converted into image-like representations and fed into the respective architectures. For the LSTM model, we extract MFCCs from the audio signals, which are used to model the temporal dynamics of the music.

By employing these different architectures and feature representations, we aim to compare their performance in audio genre classification. We assess the accuracy, precision, recall, and F1 score of each model and analyze their strengths and weaknesses. Additionally, we explore the adaptability of the ViT architecture, originally designed for visual tasks, to audio genre classification by converting the Mel spectrograms into grayscale images and modifying the

ViT model accordingly.

The outcome of this project will contribute to the existing knowledge in audio genre classification and shed light on the suitability of different deep learning architectures with specific feature representations. It will provide insights into the performance characteristics of CNNs, LSTM networks, and ViT models in the context of audio genre classification tasks. Moreover, our findings may pave the way for further research and advancements in the field, such as exploring hybrid architectures and incorporating additional contextual information for improved genre classification.

In the following sections, we will discuss the related work in audio genre classification, present the methodology used for data pre-processing, model training, and evaluation, and present the experimental results and analysis. Finally, we will conclude with a summary of the findings, their implications, and potential future directions for research in this area.

Background

GTZAN Dataset

The GTZAN Music Genre Dataset (Olteanu, A. 2020), obtained from Kaggle, comprises 1,000 song samples, each lasting 30 seconds, spanning 10 conventional music genres: blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, and rock. In addition to the audio samples, the dataset also provides alternative representations in the form of Mel Spectrograms, which we used for data preprocessing. We computed both grayscale and RGB Mel-spectrograms and extracted MFCC features from the audio files. However, the GTZAN dataset has a limitation in terms of its relatively small size, resulting in less than 800 samples remaining after a reasonable train-validation-test split. This limited sample count may impact the classification performance, prompting us to explore alternative approaches. As part of our experimentation, we investigated the use of 3-second audio clips instead of full songs. This approach aimed to capture concise yet representative temporal information, reducing computational complexity while retaining essential audio characteristics for genre classification. To compensate for the limited sample size, we expanded the training data by a factor of ten. This augmentation involved including a variety of samples from the same genres or combining the GTZAN dataset with other similar datasets. By incorporating a larger quantity of data during training, we aimed to enhance the accuracy of our models and mitigate the risk of overfitting. The expanded dataset allowed for a more comprehensive representation of the target genres.

FMA Small Dataset

Additionally, we decided to explore and apply our CNN and ViT models to the Free Music Archive (FMA) dataset (Michael Defferrard et al., 2021). The FMA is an open dataset published by the Free Music Archive, which serves as a free and open online repository of royalty-free music, managed by WFMU. The complete FMA dataset comprises over 100,000 tracks from more than 15,000 artists. Due to computational constraints, we utilized a subset of the dataset called FMA-small, which consists of approximately 8,000 songs evenly distributed across eight genre categories: Hip-Hop, Pop, Folk, Experimental, Rock, International, Electronic, and Instrumental. Each genre category includes 1,000 clips, most of which have a duration of 30 seconds. For this dataset, we computed both RGB and grayscale Mel-spectrograms to facilitate further analysis and model training. We have used the FMA dataset available on Kaggle due to ease of use using Kaggle API (Gupta, S. 2021).

Mel Spectrograms

The Fast Fourier Transform (FFT) is a powerful mathematical algorithm that enables the analysis of the frequency content of an audio signal. By applying the FFT to an audio signal, we can transform it from the time domain to the frequency domain, revealing the specific frequencies present in the signal. To analyze the frequency content of the entire audio signal, the spectrogram is commonly used. The spectrogram is created by performing the FFT on overlapping windowed segments of the signal. This results in a visual representation where the y-axis represents frequency and the x-axis represents time (Downey, 2016).

In order to enhance the perceptual characteristics of the spectrogram, certain transformations are often applied. One common transformation involves converting the y-axis to a log scale. This is because humans perceive pitch and frequency on a logarithmic scale rather than a linear scale. By converting the scale to logarithmic, we can better align the spectrogram representation with human perception.

Another transformation applied to the spectrogram involves converting the color dimension to decibels. Decibels are a logarithmic unit that measures the intensity or magnitude of a sound. By representing the intensity of each frequency component in decibels, the spectrogram can provide a more intuitive representation of the audio signal's energy distribution across different frequencies.

To further align the spectrogram representation with human perception, the Mel scale is often employed. The Mel scale is a perceptual scale of pitch that accounts for the fact that humans are better at detecting frequency differences at lower scales than at higher scales (McVicar et al., 2015). The Mel scale ensures that equal distances in pitch sound equally distant to the human listener. By applying the Mel scale transformation, the resulting representation is called the Mel spectrogram.

Mel spectrograms are valuable for audio classification using deep learning due to their ability to capture perceptually relevant features of audio signals (Virtanen, 2021). By representing audio in the frequency domain using the Mel scale, they align with human perception and emphasize important frequency regions (Piczak, 2015). They provide a compact representation that is robust to noise and variations in audio recordings (Salamon & Bella, 2017). Figure 1 shows sample mel spectrograms obtained from our datasets. Pre-trained deep learning models can effectively learn patterns and dependencies from Mel spectrograms, enabling accurate and transferable audio classification across different tasks and domains. Mel spectrograms serve as informative input features that encapsulate temporal and spectral information, facilitating the learning process for deep learning models in audio classification tasks.

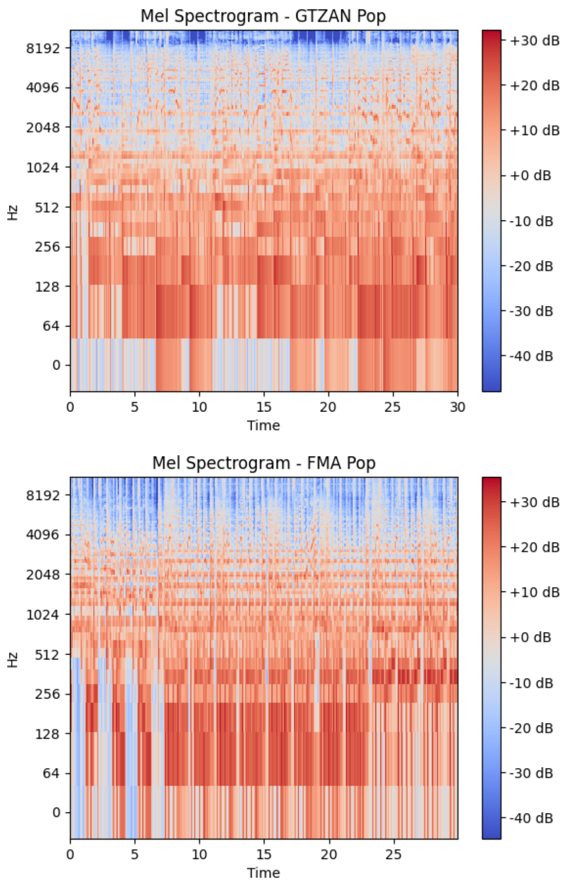


Figure 1: Mel spectrogram representation of Pop genre in GTZAN and FMA Small Dataset

Mel Cepstral Frequency Coefficients

Mel Frequency Cepstral Coefficients (MFCCs) have been widely utilized as a standard feature extraction technique in various audio signal processing applications, including speech recognition, music information retrieval, and audio classification tasks (Logan, 2000; Tzanetakis & Cook, 2002).

MFCCs are a representation of the short-term power spectrum of sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency (Davis & Mermelstein, 1980). This mel scale approximates the human ear's response more closely than the linearly-spaced frequency bands used in the traditional Fourier Transform, thereby providing a more perceptually relevant representation (Stevens, Volkmann, & Newman, 1937).

The process of calculating MFCCs can be broadly summarized in the following steps. First, frame the signal into short frames. The rationale behind this step is that frequencies in a signal change over time, so in most cases it doesn't make sense to do the Fourier transform across an entire signal which causes the frequency contours of the signal over time. It is assumed that frequencies in a signal are stationary over a very short period of time. Second, for each frame, calculate the periodogram estimate of the power spectrum. Third, apply the mel filterbank to the power spectra, sum the energy in each filter. A mel filterbank multiplies the power spectrum with a set of filters. This is meant to mimic human ears' response. Fifth, the logarithm of all filterbanks is taken. Sixth, the Discrete Cosine Transform (DCT) of the log filterbanks is taken. Sixth, 2-13 of these are kept since the most meaningful coefficients exist in this range. This is done for the other respective windows and provides a 2D array of features. The x-axis is time while the y-axis is the set of coefficients the y-axis is the MFCC coefficient number.

The MFCCs have been shown to be highly effective at capturing timbral and textural characteristics of music, and are the de facto standard in most music genre classification tasks (Tzanetakis & Cook, 2002). Figure 2 shows sample MFCC obtained from GTZAN dataset.

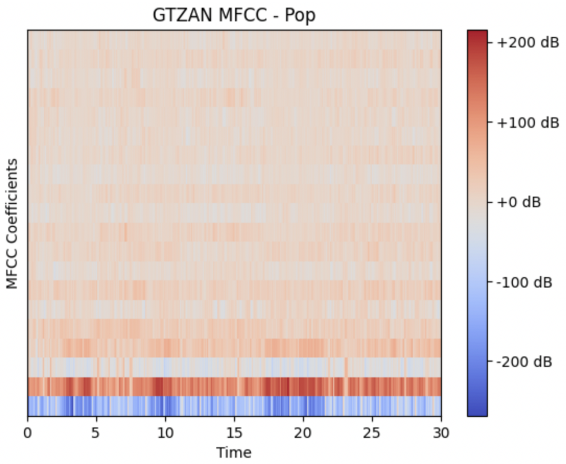


Figure 2: MFCC representation of Pop genre in GTZAN Dataset

Related Work

Previous research in music genre classification has explored various machine learning algorithms, including traditional ones like SVM, KNN, Random Forests, and XGBoost, as well as deep learning techniques such as CNN, RNN, and hybrid CNN+RNN models. CNN models achieved an accuracy of 94% with Mel spectrograms, while LSTM models achieved 74.11% accuracy with MFCCs. Hybrid models like CNN+Bi-GRU and CNN+LSTM achieved accuracies of 89.3% and 87.2% respectively with Mel spectrograms. Vision Transformer models have shown an accuracy of 71% (Artificialis, 2023).

To establish a baseline model, we refer to the work of Jongpil Lee et al. (2017) on "Automatic music genre classification using convolutional neural networks." They propose a deep CNN model with multiple branches designed to capture various levels of musical features using Mel spectrograms as input. Their approach demonstrates high classification accuracy on large-scale music datasets.

Building upon this work, our baseline model adopts a similar CNN approach with Mel spectrograms as input. CNNs excel at learning discriminative features from frequency representations of audio signals. Additionally, we investigate the effectiveness of other models such as LSTM networks and Vision Transformers for audio classification tasks. By considering multiple model architectures, we aim to compare their performance and determine the most suitable model for our specific audio classification problem.

Y. Khasgiwala and J. Tailor (2021) conducted experiments comparing the performance of ViT, CNN, and RNN-LSTM on the FMA-Small dataset using MFCCs. Their results indicate the best test loss and test accuracy were achieved with LSTM, highlighting the potential of Vision Transformer models for music genre classification. This research contributes to the exploration of deep learning techniques in audio analysis.

Furthermore, the research conducted by Jia Dai and Shan Liang focuses on leveraging the sequential nature of audio data for music genre classification. They highlight the limitations of conventional frame feature-based methods that ignore the sequential nature of audio. To address this, they propose the use of LSTM RNN to capture long-range sequential information in music data. Their method involves deriving segment features from LSTM frame features and fusing them with initial frame features for improved representation.

The evaluation of their method on the ISMIR database demonstrates the benefits of incorporating sequential information in music genre classification. The proposed fusional segment feature achieves 89.71 % classification accuracy, a significant improvement over the baseline model using a deep neural network (DNN). This work showcases the potential of LSTMs in modeling sequential information in music data and emphasizes the significance of integrating frame and segment features for enhanced classification performance.

Project Description

Exploratory Data Analysis (EDA) and Pre-processing

We began our project by working with the GTZAN dataset, where we performed EDA to gain insights into the audio data. We visualized sound waves and analyzed various audio features, including zero crossing rate, harmonics, tempo, spectral centroid, spectral rolloff, MFCCs, and Mel Spectrograms. We converted the audio files into MFCCs and Mel Spectrograms to be used in our models.

Since our Vision Transformer (ViT) model did not perform well with the GTZAN dataset, we also conducted similar pre-processing steps on the FMA Small dataset. We converted the audio files into Mel Spectrograms to explore the effectiveness of the ViT model with this dataset. Furthermore, we conducted an evaluation of a Convolutional Neural Network (CNN) architecture using the FMA Small dataset to gain comprehensive insights into the performance of this model in audio file classification.

Both datasets are balanced, GTZAN has 100 tracks in each of the 10 genres, while FMA has 1000 tracks in each of the 8 genres. Some files are corrupted and are removed during conversion to mel spectrograms or MFCCs using librosa library.

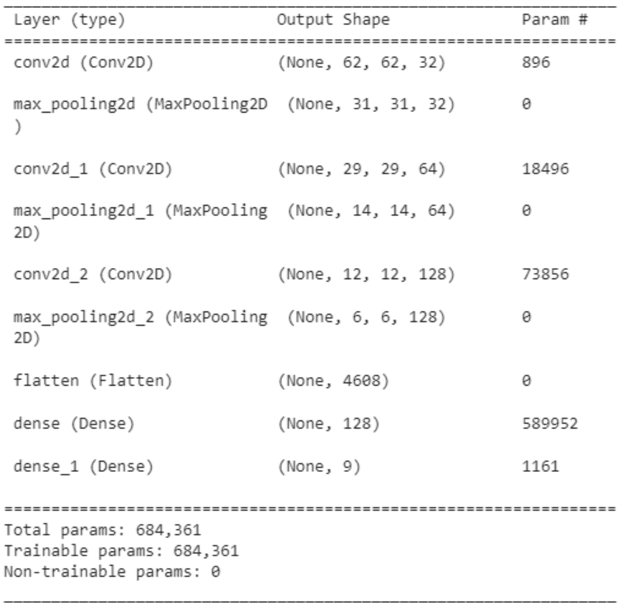
Convolution Neural Network (CNN)

For the CNN model, we designed an architecture that consisted of Convolutional 2D layers, Max Pooling 2D layers, and Dense layers (Figure 3 & 4). The input to the model was Mel Spectrograms represented as images with a size of (64, 64, 3). The images were processed as RGB rather than grayscale, allowing for potential color information to be considered in the classification process.

During the training process, we used a batch size of 32, meaning that the model processed 32 spectrograms in each iteration of training. The dataset was split into a train/test split of 80/20, with 80% of the data used for training and 20% reserved for testing the model's performance on unseen data.

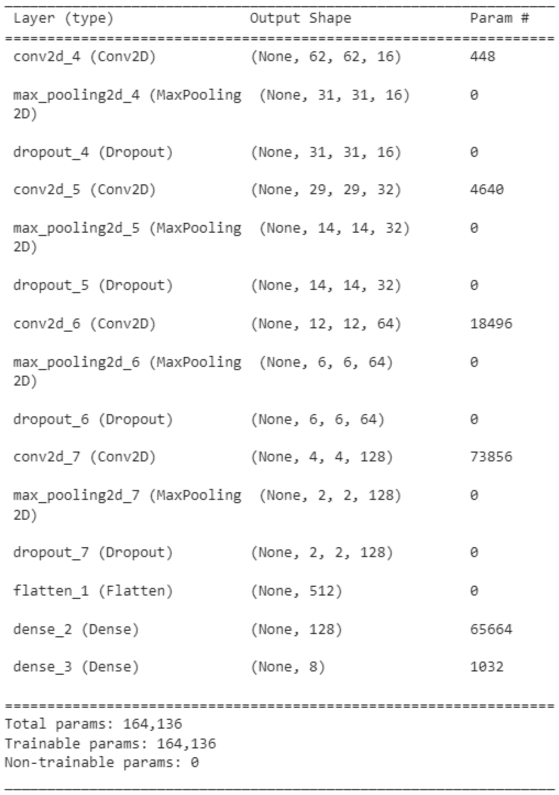
Regarding the GTZAN input, a beneficial approach involved segmenting the 30-second audio files into 3-second files and subsequently transforming them into spectrograms. This method resulted in a tenfold increase in data volume. However, for the FMA dataset, which already consisted of 8000 audio files, such segmentation and transformation steps were deemed unnecessary.

Due to overfitting of the CNN model constructed for the GTZAN model with the FMA dataset, an additional sequence of Convolutional layers and max pooling layers was added. Furthermore, dropout layers were included after each of these sequences.



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 29, 29, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 12, 12, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 6, 6, 128)        0
 2D)

 flatten (Flatten)           (None, 4608)              0

 dense (Dense)               (None, 128)               589952

 dense_1 (Dense)             (None, 9)                 1161

=================================================================
Total params: 684,361
Trainable params: 684,361
Non-trainable params: 0
_________________________________________________________________
```

Figure 3: CNN Architecture - GTZAN



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 62, 62, 16)        448

 max_pooling2d_4 (MaxPooling  (None, 31, 31, 16)       0
 2D)

 dropout_4 (Dropout)         (None, 31, 31, 16)        0

 conv2d_5 (Conv2D)           (None, 29, 29, 32)        4640

 max_pooling2d_5 (MaxPooling  (None, 14, 14, 32)       0
 2D)

 dropout_5 (Dropout)         (None, 14, 14, 32)        0

 conv2d_6 (Conv2D)           (None, 12, 12, 64)        18496

 max_pooling2d_6 (MaxPooling  (None, 6, 6, 64)         0
 2D)

 dropout_6 (Dropout)         (None, 6, 6, 64)          0

 conv2d_7 (Conv2D)           (None, 4, 4, 128)         73856

 max_pooling2d_7 (MaxPooling  (None, 2, 2, 128)        0
 2D)

 dropout_7 (Dropout)         (None, 2, 2, 128)         0

 flatten_1 (Flatten)         (None, 512)               0

 dense_2 (Dense)             (None, 128)               65664

 dense_3 (Dense)             (None, 8)                 1032

=================================================================
Total params: 164,136
Trainable params: 164,136
Non-trainable params: 0
_________________________________________________________________
```

Figure 4: CNN Architecture - FMA

Vision Transformer (ViT)

The Vision Transformer (ViT) model architecture is a variant of the Transformer originally developed for natural language processing tasks but adapted for image classification purposes. In our project, we utilized ViT to address music genre classification using Mel Spectrograms derived from songs.

The ViT model takes advantage of image patches, accompanied by positional embeddings, which are then fed into the encoder architecture of traditional transformers. The model incorporates multiple encoder blocks that leverage the MultiHeadSelfAttention layer to capture global dependencies and extract crucial features from the input sequence. The TransformerBlock combines self-attention with a feed-forward neural network, incorporating residual connections and layer normalization (Figure 5).

The complete ViT architecture, implemented as the VisionTransformer class, involves processing input images in patches, projecting them into a lower-dimensional embedding space, and applying multiple transformer blocks. The model incorporates class tokens and positional embeddings, with the final transformer block's output used for multi-label classification through a multi-layer perceptron (MLP) head. By adapting the transformer architecture and leveraging self-attention mechanisms, ViT effectively addresses image. Our model architecture is shown in Figure 6.

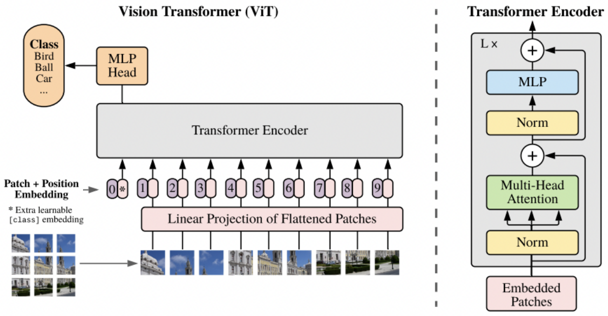
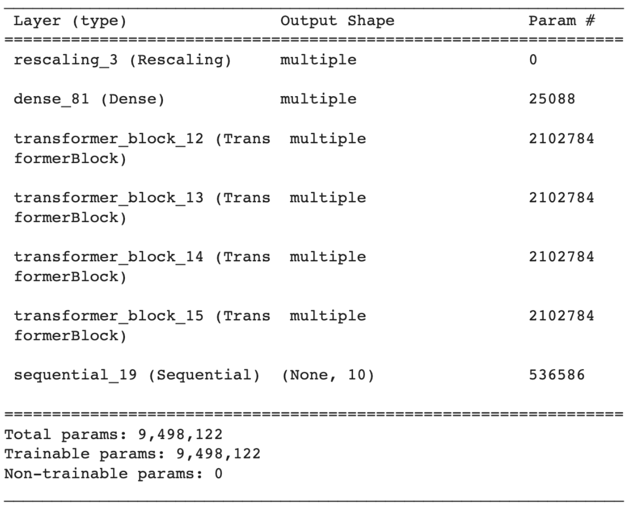


Figure 5: ViT architecture (Dosovitskiy, A et al. 2020)



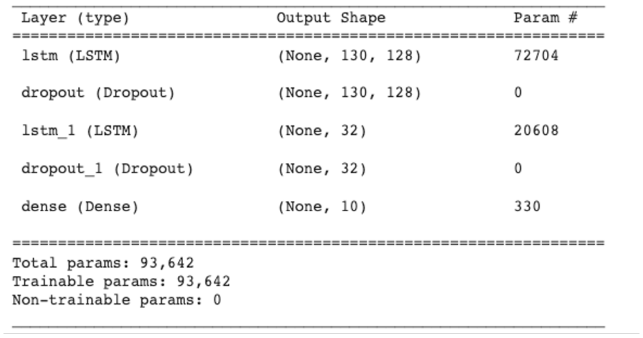
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 rescaling_3 (Rescaling)     multiple                  0

 dense_81 (Dense)            multiple                  25088

 transformer_block_12 (Trans  multiple                 2102784
 formerBlock)

 transformer_block_13 (Trans  multiple                 2102784
 formerBlock)

 transformer_block_14 (Trans  multiple                 2102784
 formerBlock)

 transformer_block_15 (Trans  multiple                 2102784
 formerBlock)

 sequential_19 (Sequential)  (None, 10)                536586

=================================================================
Total params: 9,498,122
Trainable params: 9,498,122
Non-trainable params: 0
_________________________________________________________________
```

Figure 6: Our ViT architecture

Long Short-Term Memory (LSTM)

To preprocess the GTZAN data for the LSTM model, we split each 30-second song clip into 10 3-second clips. MFCCs were then computed using these shortened audio files. We used the librosa library to generate MFCCs with a sample rate of 22050 Hz and a hop length of 512. The purpose of splitting the data into smaller clips was to increase the amount of data and mitigate overfitting issues that affected the original model.

The LSTM model architecture consisted of an LSTM layer with an output size of 128, followed by a dropout layer with a rate of 25 percent. This was followed by another LSTM layer with an output size of 32, a dropout layer with a rate of 50 percent, and a Dense layer with softmax activation to output probabilities for the 10 music genres. We used the Adam optimizer with a learning rate of 0.001 and employed cross-entropy loss as the objective function. The model was trained for 100 epochs with a batch size of 64. Early stopping was implemented based on validation loss, resulting in training termination after 46 epochs (Figure 7).



```
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 130, 128)          72704

dropout (Dropout)            (None, 130, 128)          0

lstm_1 (LSTM)                (None, 32)                20608

dropout_1 (Dropout)          (None, 32)                0

dense (Dense)                (None, 10)                330

=================================================================
Total params: 93,642
Trainable params: 93,642
Non-trainable params: 0
_________________________________________________________________
```

Figure 7: LSTM Architecture - GTZAN

Empirical Results

Convolutional Neural Network (CNN)

GTZAN In order to achieve high test accuracy and avoid overfitting, several changes were made to the original input data and preprocessing steps. Firstly, the image size was reduced to (64, 64, 3), which can help reduce computational complexity while still capturing important frequency information in the spectrogram. Additionally, the audio signal was split into smaller segments to increase the amount of training data available, potentially capturing more diverse audio patterns.

The model's performance was evaluated using the accuracy metric, which measures the proportion of correctly classified samples, and the categorical cross-entropy loss, which quantifies the dissimilarity between predicted and actual class probabilities. Additionally, the model was evaluated on precision, recall, and F1-score.

Overall, the model achieved an accuracy of 72% across all genres, indicating a reasonably satisfactory performance (Figure 8). The macro average F1-score, which provides an overall measure of the model's effectiveness, was found to be 0.71, demonstrating consistent performance across the various genres. Furthermore, the weighted average F1-score was calculated as 0.72, suggesting that the model's performance was generally well-balanced across the dataset.

These empirical results indicate the capability of the CNN model to classify audio files from the GTZAN dataset with considerable accuracy and provide valuable insights into the model's genre classification performance.

The genres ”classical” and ”metal” had relatively lower precision, recall, and F1-scores compared to other genres. The challenges in accurately recognizing ”classical” and ”metal” could be attributed to their distinct characteristics and complexity (Figure 9). Classical music often exhibits intricate compositions and nuanced elements, making it more challenging for the model to capture its unique patterns effectively. Similarly, metal music is characterized by aggressive and intense sounds, which might introduce difficulties in distinguishing it from other genres. These factors likely contributed to the lower performance observed in classifying these specific genres.

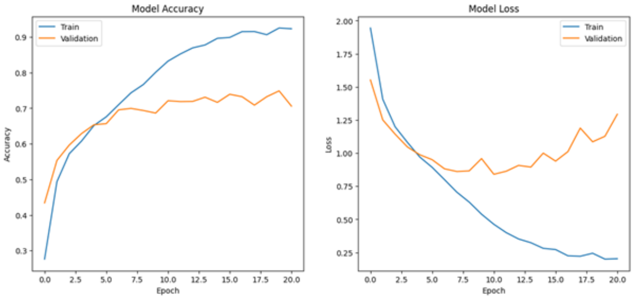


Figure 8: CNN - Accuracy and Loss Curves

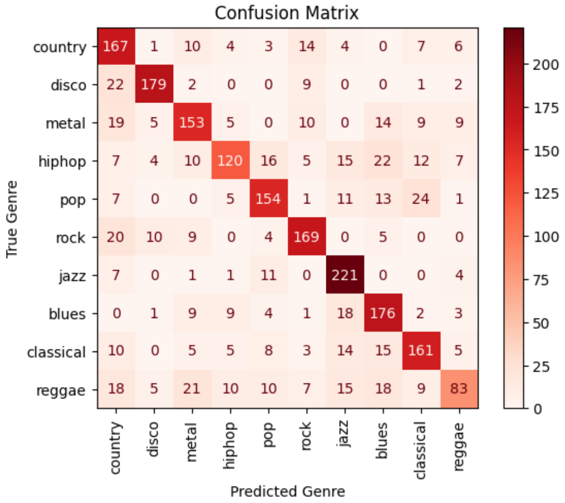


Figure 9: CNN - Confusion Matrix

FMA In the case of the FMA dataset, a consistent image size of (64, 64, 3) was utilized. However, unlike the GTZAN dataset, the audio signal was not segmented into 3-second clips, as the dataset provided ample data for analysis.

To evaluate the performance of the model, several metrics were employed. Firstly, the accuracy metric was utilized, which measures the proportion of correctly classified samples. Additionally, the categorical cross-entropy loss was employed to quantify the dissimilarity between predicted and actual class probabilities. Furthermore, precision, recall, and F1-score were evaluated to assess the model's classification performance.

The overall accuracy achieved by the model on the FMA dataset was 0.48 (Figure 10). The macro average F1-score, which provides an overall measure of the model's effectiveness, was found to be 0.46, indicating moderate performance across the different genres. Furthermore, the weighted average F1-score was calculated as 0.46, suggesting a balanced performance across the dataset.

These results shed light on the model's performance in classifying audio files from the FMA dataset, considering various metrics, and provide insights into the classification capabilities of the model for different genres.

Several genres also had relatively lower precision, recall, and F1-scores. Specifically, the genres ”Hip-Hop,” ”Instrumental,” ”Electronic,” and ”Pop” exhibited lower performance metrics (Figure 11). The challenges in accurately recognizing these genres could be attributed to various factors. ”Hip-Hop” music often contains complex rhythms, intricate lyrics, and diverse production styles, making it challenging to capture its distinctive features accurately. Similarly, ”Instrumental” music lacks vocal cues, which might make it more challenging for the model to identify and differentiate from other genres. The genres ”Electronic” and ”Pop” encompass a wide range of subgenres and stylistic variations, which could introduce additional complexities and result in lower classification performance.

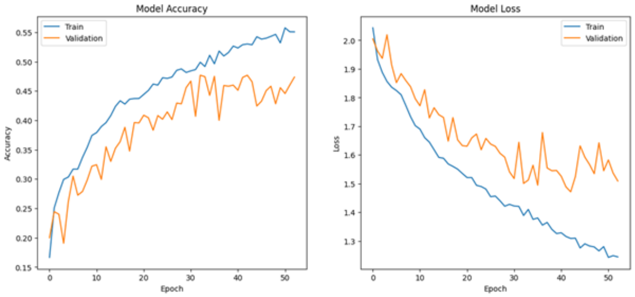


Figure 10: CNN - Accuracy and Loss Curves

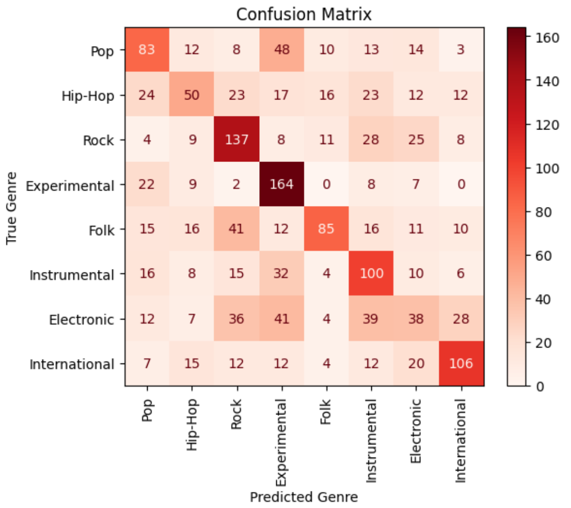


Figure 11: CNN - Confusion Matrix

Vision Transformer (ViT)

GTZAN In our project, we initially utilized the GTZAN dataset and employed RGB and grayscale mel spectrogram images derived from 30-second audio samples. However, despite experimenting with multiple model architectures, we encountered challenges in achieving satisfactory accuracy. To address this, we explored an alternative approach by using shorter 3-second segments extracted from the original 30-second audio, effectively increasing the training data size tenfold. However, even with this augmented dataset, the results did not meet our expectations, with the best accuracy achieved being approximately 30% using our final model architecture with RGB images of size 128 for 30 epochs using learning rate of 3e-4 with learning rate scheduler, Early Stopping and Adam Optimizer with wight decay (Figure 12 & 13).

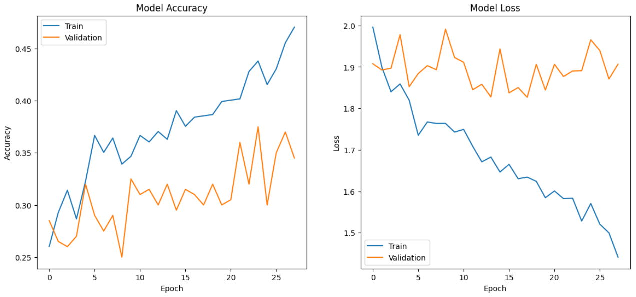


Figure 12: ViT - Accuracy and Loss curves

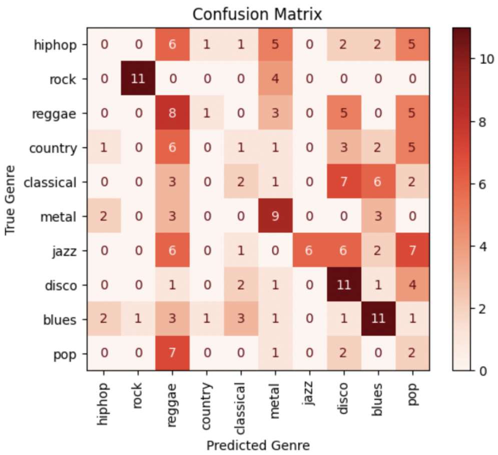


Figure 13: ViT - Confusion Matrix

FMA Furthermore, we conducted experiments using the FMA Small Dataset, applying various model architectures. Among these, the highest accuracy achieved was 41% when experimented with Grayscale images with an image size of 64 with a learning rate of 1e-3 and Adam optimizer with weight decay of 1e-4 for 50 epochs. Notably, the Vision Transformer (ViT) model, which we anticipated to outperform traditional convolutional neural networks (CNNs), did not meet our performance expectations (Figure 14 & 15).

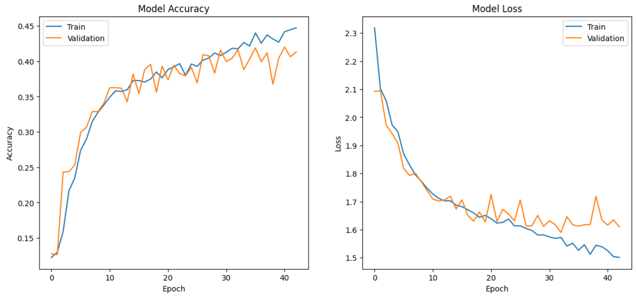


Figure 14: ViT - Accuracy and Loss curves

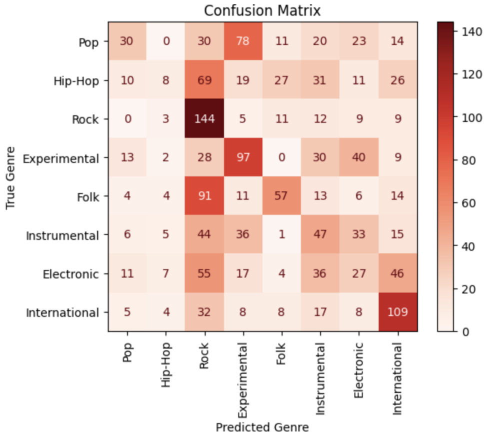


Figure 15: ViT - Confusion Matrix

During our experimentation, we explored three ViT model architectures, varying parameters such as patch size (4, 6, 8, 12, and 16), number of layers (4 and 8), embedding dimensions (256 and 512), and number of attention heads (4, 8, and 12). Additionally, we experimented with image sizes of 64, 72, 128, and 224. Our evaluation metric was accuracy, while we simultaneously monitored validation loss to prevent over fitting.

Given the inherent noise in music data, we employed the AdamW optimizer [13] with a learning rate of 0.001 and weight decay of 0.0001. AdamW resolves the problem of sparse gradients on noisy data by leveraging the properties of the AdaGrad and RMSProp algorithms. The inclusion of decoupled weight decay in AdamW yields improved generalization performance. For gradient calculation during back-propagation, we used the Categorical Crossentropy loss function from the Keras library.

The model was trained for 50 epochs, and it exhibited signs of over-fitting after 30 epochs. To mitigate over fitting, we introduced dropout layers to the model and trained it for an additional 50 epochs. The best test accuracy achieved was 30% for the GTZAN dataset and 41% for the FMA dataset. To gain further insights, we plotted the confusion matrix and obtained the classification report, allowing us to monitor precision, recall, and F1 score.

Our findings indicate that despite our efforts to enhance the accuracy of the models, we encountered limitations and challenges in achieving the desired results. These outcomes may be attributed to the complexity and diversity of music genres, the size and quality of the datasets used, and the suitability of the chosen model architectures for the given tasks.

Long Short-Term Memory (LSTM)

The LSTM model performed well achieving a test accuracy of 80.78% (Figure 16 & 17). This was a sizable improvement from 40% test accuracy with the original data set with 1,000 30-second clips. Shortening the clips and increasing the data set by 10 times its original size caused the model to stop over-fitting. Other than minor tweaks that were made to the dropout layers, the model's architecture stayed the same from the original. Each training example had 130 MFCC feature vectors with 13 coefficients each.

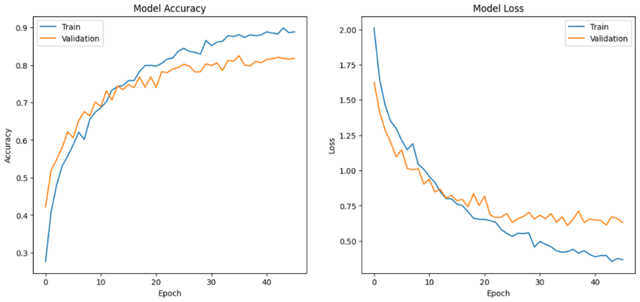


Figure 16: LSTM - Accuracy and Loss Curves

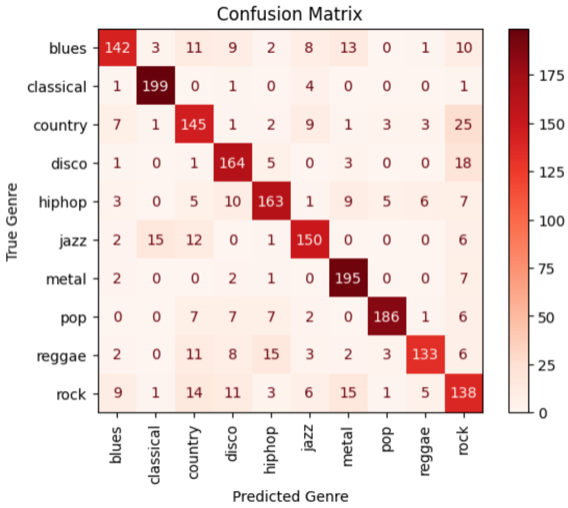


Figure 17: LSTM - Confusion Matrix

As shown in Figure 18, we evaluated the performance of various models on different datasets using specific features and measured their classification accuracy along with precision, recall, F1-score, and accuracy.

For the CNN model trained on the GTZAN dataset using Mel Spectrogram features, we achieved a test accuracy of 0.72, with precision, recall, and F1-score all around 0.72.

The macro average and weighted average F1-scores were also 0.72.

Similarly, when applying the CNN model on the FMA-Small dataset with Mel Spectrogram features, the test accuracy dropped to 0.48, and precision, recall, and F1-score were approximately 0.47. The macro average F1-score was 0.48, while the weighted average F1-score was 0.46.

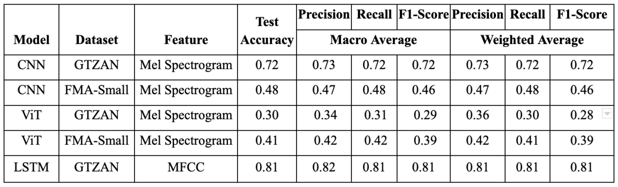
In contrast, utilizing the ViT (Vision Transformer) model on the GTZAN dataset with Mel Spectrogram features yielded lower results, with a test accuracy of 0.30. Precision, recall, and F1-score were 0.34, 0.31, and 0.29, respectively. The macro average F1-score was 0.36, while the weighted average F1-score was 0.30.

Similarly, when applying the ViT model on the FMA-Small dataset with Mel Spectrogram features, the test accuracy improved to 0.41. Precision, recall, and F1-score were 0.42, 0.42, and 0.39, respectively. The macro average F1-score was 0.42, while the weighted average F1-score was 0.41.

Lastly, the LSTM model performed well on the GTZAN dataset using MFCC features, achieving a test accuracy of 0.81. Precision, recall, and F1-score were consistently 0.81. The macro average and weighted average F1-scores were also 0.81.

Based on these results, we can conclude that the CNN model showed reasonably good performance on both datasets when using Mel Spectrogram features, with better results on the GTZAN dataset. The ViT model, on the other hand, struggled to achieve comparable accuracy, especially on the GTZAN dataset. The LSTM model demonstrated the highest accuracy and consistency on the GTZAN dataset when utilizing MFCC features.

Overall, the choice of model, dataset, and feature extraction technique significantly impacted the classification performance, highlighting the importance of selecting appropriate combinations to achieve optimal results in audio classification tasks.



Model	Dataset	Feature	Test Accuracy	Precision	Recall	F1-Score	Precision	Recall	F1-Score
				Macro Average			Weighted Average		
CNN	GTZAN	Mel Spectrogram	0.72	0.73	0.72	0.72	0.73	0.72	0.72
CNN	FMA-Small	Mel Spectrogram	0.48	0.47	0.48	0.46	0.47	0.48	0.46
ViT	GTZAN	Mel Spectrogram	0.30	0.34	0.31	0.29	0.36	0.30	0.28
ViT	FMA-Small	Mel Spectrogram	0.41	0.42	0.42	0.39	0.42	0.41	0.39
LSTM	GTZAN	MFCC	0.81	0.82	0.81	0.81	0.81	0.81	0.81

Figure 18: Comparison of models

Conclusion / Future Direction

Based on the results of this project, several areas for improvement and future exploration have been identified. With more time, we would focus on fine-tuning the model architectures to better suit the characteristics of the dataset. This would involve conducting extensive hyperparameter tuning to optimize the model's performance and potentially achieve better results. Additionally, we would like to experiment with segmenting the Free Music Archive (FMA) dataset into smaller time segments rather than using the entire 30-second clips. This could provide more diverse and granular data for training and potentially improve classification accuracy.

Furthermore, we would explore working with different datasets to evaluate the model's generalization and robustness. This could involve acquiring additional audio datasets or curating our own audio data and assessing how well the model performs across various genres and sources. Additionally, we would consider incorporating Mel Frequency Cepstral Coefficients (MFCCs) as an alternative feature representation and compare their performance with that of Mel spectrograms.

We would aim to try other hybrid models beyond the CNN architecture, to assess their suitability for audio classification tasks. By exploring different model architectures, we could potentially uncover new insights and improve overall classification performance. We would also like to figure out why ViT did not perform well for both the datasets tried using multiple model architectures. We also faced the problem where a result obtained could not be reproduced with ViT architecture though random state was fixed. These are improvements to be made with ViT. Another thing to experiment is to use transfer learning and see if it helps in this domain task.

For future DS 5500 students undertaking similar projects, we would advise to spend ample time on data preprocessing and exploration, as it plays a crucial role in the success of the project. Also, to keep track of the experimentation process and document the choices made, as this facilitates reproducibility and provides a clear record for analysis and reporting. Lastly, to collaborate with peers and seek feedback to gain diverse perspectives and enhance the quality of the project. By building upon the insights gained from this project, future students can further advance the field of audio classification and contribute to the broader domain of data science.

GitHub Repository

https://github.com/joannjacob/Audio-Genre-Classification

Instructions to Download / Run System

- Clone the GitHub repository. This contains all the Google Colab files for the EDA and different models. Files included in the repo are:
  - README.md
  - Preprocessing_&_EDA.ipynb
  - CNN_GTZAN.ipynb
  - CNN_FMA.ipynb
  - VIT_GTZAN.ipynb
  - VIT_FMA.ipynb
  - LSTM_GTZAN.ipynb
- **GTZAN Dataset**
  - Download the GTZAN dataset from https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification.

  - Create a folder in your drive and upload the ”Data” folder downloaded here which contains images_original, genres_original, features_3_sec.csv and features_30s.csv).
  - Use the correct link to your drive in the required cells.
- **FMA Small Dataset**
  - Downloading the data from the original repo and loading it into the drive takes a long time and space since it is 7.2 GB, so instead we are using the FMA Small from Kaggle using Kaggle API FMA Kaggle Dataset - https://www.kaggle.com/datasets/imsparsh/fma-free-music-archive-small-medium.
  - Download your Kaggle API file kaggle.json and upload into the colab by running the cells in the notebook. This is needed to download the dataset into the notebook.
- Upload and open the notebooks in Google Colab.
- Instructions to run each file are given within the notebook.

Clone the GitHub repository. This contains all the Google Colab files for the different models. Upload and open the notebooks in Google colab. Instructions to run each file are given within the notebook.

References

Artificialis. 2023. From Sound to Sight: Using Vision Transformers for Audio Classification. https://medium.com/artificialis/from-sound-to-sight-using-vision-transformers-for-audio-classification-aa6a0293914c.

Choi, K.; and et al. 2018. Content-Based Music Genre Classification Using Deep Neural Networks. In *Proceedings of the 2018 International Workshop on Pattern Recognition*, 85–91. ACM.

Dai, J.; Liang, S.; Xue, W.; Ni, C.; and Liu, W.-J. 2016. Long short-term memory recurrent neural network based segment features for music genre classification. 1–5.

Davis, S.; and Mermelstein, P. 1980. Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4): 357–366.

Dieleman, S.; and et al. 2016. End-to-End Learning for Music Audio Classification using Cross-modal Projection and Deep Neural Networks. *arXiv preprint arXiv:1612.01840*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2020. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.

Downey, A. B. 2016. *Think DSP: Digital Signal Processing in Python*. O’Reilly Media.

emla2805. 2020. Vision Transformer for Music Classification. https://github.com/emla2805/vision-transformer.

Gupta, S. 2021. FMA - Free Music Archive - Small Medium. https://www.kaggle.com/datasets/imsparsh/fma-free-music-archive-small-medium.

Hafizah, S.; and et al. 2021. Music Genre Classification Using Convolutional Neural Network with Residual Network. In *2021 International Conference on Information Management and Technology (ICIMTech)*, 144–149. IEEE.

Keras Examples. 2021. Image Classification with Vision Transformer. https://keras.io/examples/vision/image_classification_with_vision_transformer/. Accessed: June 25, 2023.

Khasgiwala, Y.; and Tailor, J. 2021. Vision Transformer for Music Genre Classification using Mel-frequency Cepstrum Coefficient. In *2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON)*, 1–5.

Logan, B. 2000. Mel Frequency Cepstral Coefficients for Music Modeling. In *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*.

McVicar, M.; and et al. 2015. *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Springer.

Michael Defferrard; and et al. 2021. FMA - A Dataset for Music Analysis. https://github.com/mdeff/fma.

Olteanu, A. 2020. GTZAN Dataset - Music Genre Classification. https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification.

Piczak, K. J. 2015. ESC: Dataset for Environmental Sound Classification. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM ’15, 1015–1018. New York, NY, USA: Association for Computing Machinery. ISBN 9781450334594.

Salamon, J.; and Bello, J. P. 2017. Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification. *IEEE Signal Processing Letters*, 24(3): 279–283.

Stevens, S. S.; Volkmann, J.; and Newman, E. B. 1937. A Scale for the Measurement of the Psychological Magnitude Pitch. *Journal of the Acoustical Society of America*, 8(3): 185–190.

Tian, Y.; Kwong, S.; and Wong, H.-f. 2017. Long Short-Term Memory Recurrent Neural Network-based Segment Features for Music Genre Classification. In *2017 IEEE International Workshop on Pattern Recognition*, 85–88. IEEE.

Tzanetakis, G.; and Cook, P. 2002. Musical Genre Classification of Audio Signals. *IEEE Transactions on Speech and Audio Processing*, 10(5): 293–302.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, ; and Polosukhin, I. 2017. Attention is All You Need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS’17)*, 6000–6010. Red Hook, NY, USA: Curran Associates Inc.

Virtanen, T.; Plumbley, M. D.; and Ellis, D., eds. 2018. *Computational Analysis of Sound Scenes and Events*. Springer Cham, 1 edition. ISBN 978-3-319-63449-4.