Convolutional Neural Network for audio genre classification

Justin Kavalan
Georgia Institute of Technology
Atlanta, GA 30332
justinkavalan@gatech.edu

Arvind Krishna
Georgia Institute of Technology
Atlanta, GA 30332
akrishna39@gatech.edu

Josh Bishop
Georgia Institute of Technology
Atlanta, GA 30332
jbishop45@gatech.edu

Benjamin Fuentes
Georgia Institute of Technology
Atlanta, GA 30332
bfuentes3@gatech.edu

Abstract

We outline a strategy to achieve the dual objective of (1) assessing the evolution of a genre of music over time, and (2) to generate the music that best represents the genre. Although, we clearly lay down the steps to achieve our objective, we fall short due to unresolved issues that we identify in published works that we leverage in our project. We present the resolution of several such issues, and discuss our attempts to develop a CNN deep learning model for genre classification using raw audio features.

1. Introduction

Our objective is to quantify and compare the evolution of different genres of music, and generate the most representative music corresponding to a given genre. We planned to use an already published model on genre classification, and then perform back-propagation to tune inputs with fixed weights, in order to maximize the score for a given genre. This will give us the *'input music'* which maximizes the score of a given genre. Such *'input music'* will be the "most representative" music of that genre. We will compare the score of the highest scoring track in the data with the score of the most representative music for that genre, for a given period of time. This will let us know about the evolution of a genre over time, as explained later.

We extract MFCC [1] (mel frequency cepstral coefficients) features [7] from the audio and use multi-layered CNN models [6] for genre classification. We use the FMA (Free Music Archive) dataset [3] to classify audio tracks into genre. We run the model on a couple of datasets: (1) *'fma_small'* containing 8000 audio tracks and 8 genres, and (2) *'fma_medium'* containing 25,000 audio tracks and 16 genres. One important distinction is that the distribution of genres in *'fma_small'* was balanced, while in *'fma_medium'*, the genres were not evenly distributed.

Once we tune the inputs to maximize MFCCs, it helps us answer two key questions: (a) How has a given genre evolved over time? (b) What does the most representative audio sound like for a given genre?

To answer (a) we compare the difference between the score of the most representative audio obtained by back-propagation, and the highest scoring audio in the data. We compare this difference over different genres and time frames. The difference in the score will be a measure of how close the best (highest scoring) music is to the most representative music for that genre and time period. If this difference in score has changed over time, then we will conclude that the genre has evolved over time.

To answer (b) we will convert the *input* corresponding to the maximum score to audio, for a given genre. This conversion can be done using the *LibROSA* package. As MFCCs do not contain the complete information of the audio, the converted audio will require some smoothing to eliminate irregularities. The audio representing a genre can have several uses. For example, it can be used by music-sharing platforms to let the user choose from different genres of music, after listening to the genre-representative audio. It can help in music instruction, where students need to learn the subtle differences between different genres of music.

As far as we know, this is a novel objective, and this has not been attempted by anyone before. Although there has been considerable work done to classify audio into genres ([5], [1],[7],[3]), most of this work used features that are summary statistics of audio signals such as mean, variance, standard deviation etc., over the entire length of the signal (i.e., over the length of the audio). However, such

[1] MFCCs are commonly used for MIR (Music Information Retrieval)

summary statistics cannot be converted back to audio. Convolutional neural networks (CNN) have been found to be useful in some of the recent works ([6],[2]) to classify audio into genres, using raw audio features (feature values at different time stamps). Thus, we use a CNN model to extract useful information from raw audio features for genre classification.

Though our strategy is well laid out above, we found several unresolved issues with the data ([3]) that took us quite a while to resolve. The CNN model that we adapted had also been trained on a very small dataset, with a very small classification accuracy (less than 30% for 10 genres) for test data. The model did not perform well for our data. As such, while this project progressed, our objective slowly became less ambitious as we realized our prerequisite model was inadequate. In this sense, the objective of our work, and as a result the experiments and results, has differed from our initial and overall objective. The objective of our experiments and result section is to simply produce a good enough genre classification model that we can use to perform some basic analysis related to our original objective. This problem space is less novel, as genre classification is normally done by hand (e.g. playlist curation, album publishing), but still has application in music recommendation engine and labelling music archives.

The many different adaptations of our model to achieve an acceptable performance, as well as analysis in line with our initial objective, are presented and analyzed in the subsequent sections.

2. Approach

There are a few datasets available that were considered for this task. GTZan and Million Songs dataset (MSD) are commonly used, but both of them were of limited use for this project. GTZan only has 100 songs/genre, which is an insufficient amount of data for a deep learning model. MSD, on the other hand, has 1 million songs, but it only includes their metadata, not raw audio files. We opted to use the Free Music Archive (FMA) dataset, which has a sufficient number of tracks (at least for the more popular genres), audio metadata, and raw audio files.

We used the baselines.ipynb notebook from the github code repository associated with the FMA paper [3] for data pre-processing, feature extraction, and model building. We adapted several models based on the CNN model described in section 3 of the "baselines" notebook.

The first issue encountered was due to corrupt data files. Contrary to information in the FMA paper and documentation, some of the audio files in the dataset are less than 30 seconds long [3]. We identified and removed such corrupt files from the data. Afterwards, we resolved several deprecated lines of code to pre-process the data, extract raw audio features, and train the model.

However, the baseline CNN model and variants of the baseline model still suffered from poor classification accuracy. We observed a low training accuracy, which was close to the accuracy of random classification. The first hypothesis was that the poor classification accuracy occurred due to a lack of feature information, (this was also noted as a potential problem in [6]). We anticipated that such an issue could be due to an oversimplified model, a large number of classes, or incorrect values of hyperparameters such as learning rate and decay. Thus, we carried out an exhaustive set of experiments to analyze the cause of low training accuracy, and we present this analysis and results in the next section.

3. Experiments and Results

We began our study with an exploratory analysis, using 'analysis.ipynb' from [3] as a baseline. Since our analysis primarily concerns genre, the dataset was immediately inspected for genre distribution, as well as to determine if tracks could be labelled with multiple genres. Figure 1 and Figure 2 confirmed that a single track can be (and often is) labelled with multiple genres in the entire fma dataset. The actual genre labels in Figure 1 are imperceptible at this scale, but the cross correlation matrix gives a visual indication of how sparsely or densely different genres in the dataset may overlap with other genres. Further, Figure 2 shows that a single track may be labelled with as many as 10 genres, and it's quite common for a track to have anywhere from 2-4 genre labels. This complicated our analysis greatly, since most of our models only classify each track with the highest scoring genre. When we back-propagate over a genre's features, we are implicitly back-propagating features of other genres that are highly correlated with that genre. This confusion was definitely responsible for some degradation in the results.

Afterwards, some analysis was conducted to determine whether, on average, there is a meaningful distinction between the genres in the *'fma_small'* dataset. We analyzed this somewhat naively by constructing the average time-series mel spectrogram for each of the eight genres. Inspecting these spectrograms by eye, it's difficult to recognize a meaningful distinction (they look very similar to Figure 3), so we selected the top genre, 'Electronic', as the baseline genre, and subtracted every other average genre spectrogram to get a difference image. By averaging, we do eliminate much of the time-varying behavior that is evident in individual tracks, especially if the genre has an even distribution of many different time signatures. Interestingly, the difference image of Hip-Hop's average mel spectrogram still displayed meaningful time-varying behavior. This indicates a certain degree of stationarity in Hip-Hop time signatures that is less evident in other genres. Other meaningful differences, such as a greater magnitude of bassy tones in the

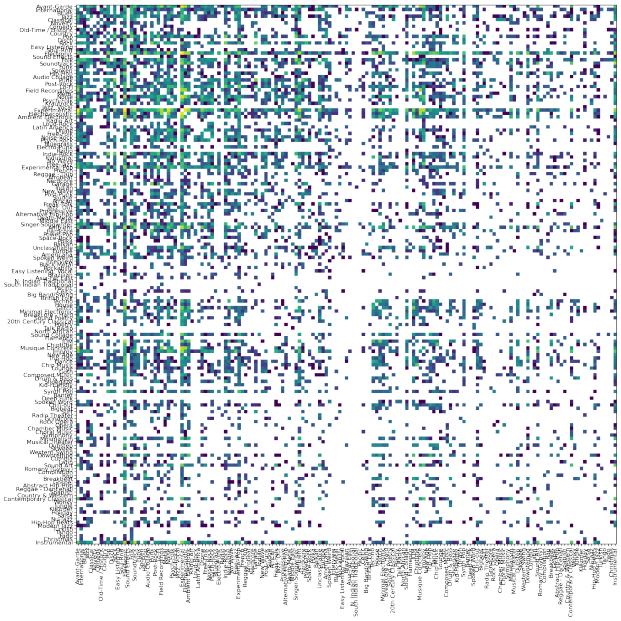


Figure 1: Cross Correlation of Genres in FMA Dataset

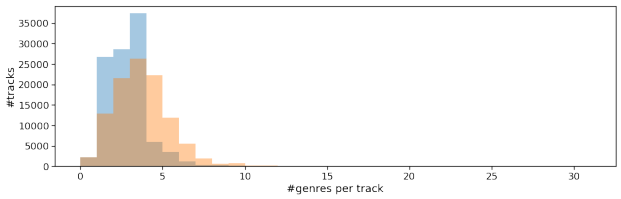


Figure 2: Distribution of # of Genres / Track. Most popular subset of genres in blue, all genres in orange.

Experimental (Figure 13a), Instrumental (Figure 13c), and Rock (Figure 13f) genres, validated our choice of MFCC features as descriptors for genre classification.

The baseline model was composed of 3 convolution layers with a fully connected layer on the output. The inputs to the model were MFCC spectrograms generated over the raw audio, generated using the steps described in section 2. The model classifies tracks as belonging to one of eight genres for *'fma_small'* and one of sixteen genres for *'fma_medium'*. The architecture was based on [6] and is displayed in Figure 5. The hyperparameters for the convolutional layers were chosen from the results of their experimentation. We used a standard cross entropy loss function and stochastic gradient descent for optimization. The hyperparameters for these functions were further tuned based on features of the training curves.

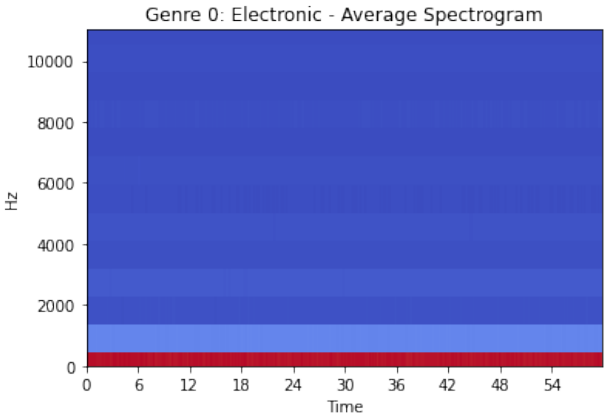


Figure 3: Average spectrogram for baseline genre, Electronic. This baseline was used to produce the difference images for the other top seven genres. Hip-Hop's difference image is displayed below in Figure 4, and the rest of them are in Appendix A.

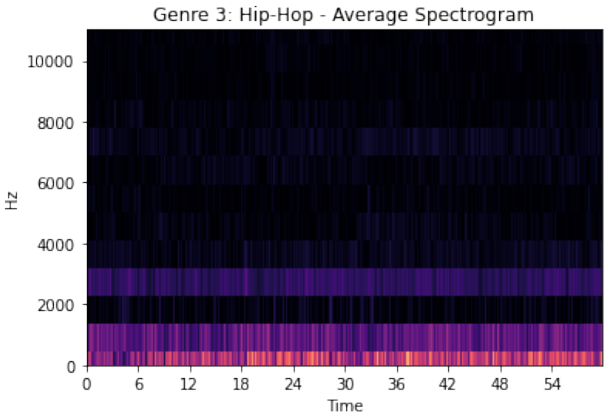


Figure 4: Difference image between average mel spectrograms for Hip-Hop and Electronic.

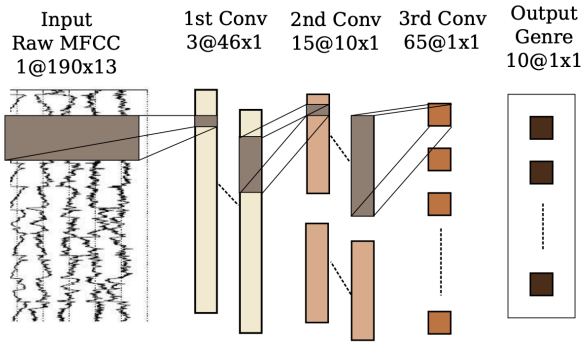


Figure 5: Baseline Model Architecture [6]

The baseline model performed poorly on *'fma_small'*,

consistently giving approximately 12.5% training accuracy. When we downloaded and ran it on *'fma_medium'*, it produced an accuracy of 28%. This was an improvement, but this model was highly susceptible to local minima. This was remedied by increasing the learning rate and decay to relatively high values of .1 and .5, respectively. Initially, we thought our sub-par performance might be due to a lack of samples and feature information because of the improvement when using *'fma_medium'*. Consequently, we performed two experiments by adding additional feature sets to the input data. A chromagram was added in the first trial (which emphasized the rhythmic features) and both chromagram and tonality features were included in the second trial. This produced a training accuracy of 2.5% and 2.0% respectively. In order to make our features more explanatory, we tried our model on segmented audio features as audio segmentation [4] is know to be effective in extracting useful information. However, even with audio segmentation, the model only achieved a training accuracy of 13%.

Investigation into our dataset and model revealed that the most common genre in *'fma_medium'* occurred with 28% frequency, which exactly matched the accuracy we were getting. When verifying the predictions on the model, the model was simply outputting the most common genre. This suggests our model was under capacity and unable to detect any patterns in the dataset. This suspicion was further reinforced by earlier behavior, such as the model performing worse when given more features, as well as difficulties in getting the model to over-fit on training data, even without regularization and with a small training size of 100 samples. The inability for these models to over-fit can be seen in Figure 6 and Figure 7.

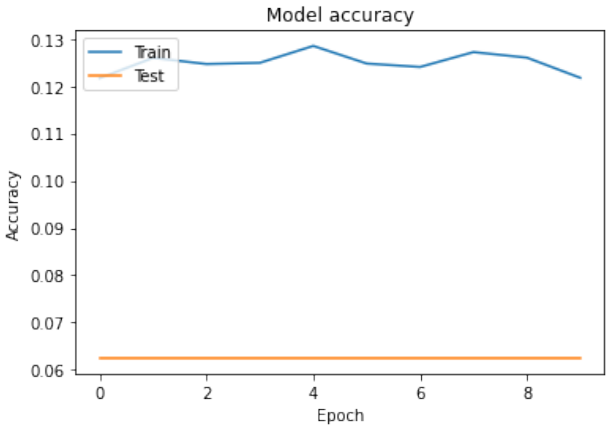


Figure 6: Training and validation accuracy using more convolutional layers

We first added a convolutional layer with 100 filters and increased the filter size of (1, 20) to try to capture a larger time window, since our samples were longer than the original dataset this model was used on. This did not yield any improvement, producing a training accuracy of 12% and a validation accuracy of 11.6%. Further experimentation with stride length, number of filters, and kernel size of convolutions did not yield any notable increase in accuracy.

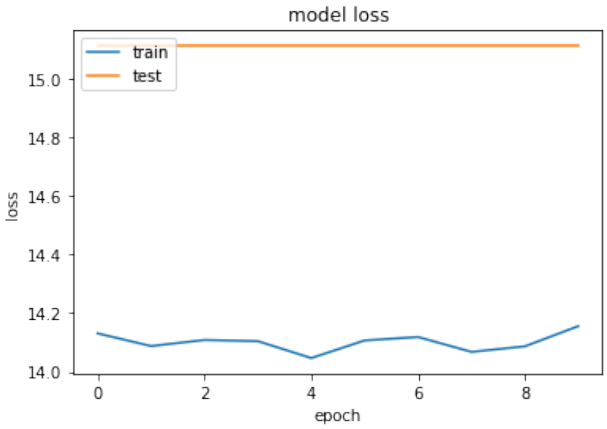


Figure 7: Training and validation loss using more convolutional layers.

We also attempted to address the capacity issues by removing 5 genres from our dataset. Ideally, this would reduce the complexity in categorizing the different genres as this reduces the overlap between genres which could be potentially confusing to the network. However, this produced a training accuracy of 33%, which is equivalent to randomly guessing and still could not overfit.

The most improvement was seen by adding two fully connected layers after our convolutions were flattened of size 512 and 256 and changing the kernel size of the convolution layers. This was the first model to successfully overfit on 1000 samples, producing a training accuracy of 99% and a test accuracy of 28%, comparable to the original paper this model was based on. Now that we successfully overfitted our model, dropout layers were added to reduce overfitting and increase our validation accuracy. However, this only yielded an insignificant improvement of 1% in our validation accuracy. The training history for this experiment can be see in Figure 8 and Figure 9.

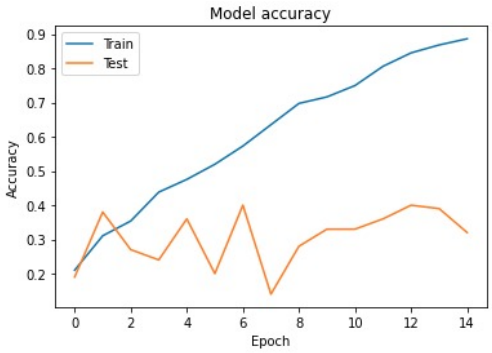


Figure 8: Training and validation accuracy with fully connected layers

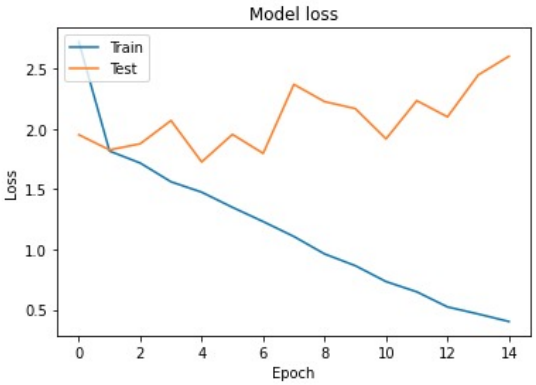


Figure 9: Training and validation loss with fully connected layers

Our explanation for this improvement was that the fully connected layers allowed the network to generalize observations from the overall features. As the convolutional layers encoded temporal patterns, it did not pick up on differences between genres over the scope of an entire track.

This previous model was then applied to *'fma_small'*. A first attempt was using the model without the dropout layer, just to be sure that the models capacity was able to handle a larger dataset. After 5 epochs, the model was able to reach 99% in train accuracy and almost 30% in validation showing that the model was able to performed on *'fma_small'*. However, the validation accuracy was almost the same in the last three epochs and the loss was increasing, facing a possible overfitting. Figure 10 and 11 show the training history for this attempt. A second attempt was done using Dropout layers. After tuning the learning rate and decay parameters, the model was able to reach a 91% accuracy on training data and 32% accuracy on validation after 20 epochs. We finalize this model to answer the questions we mentioned in our objective.

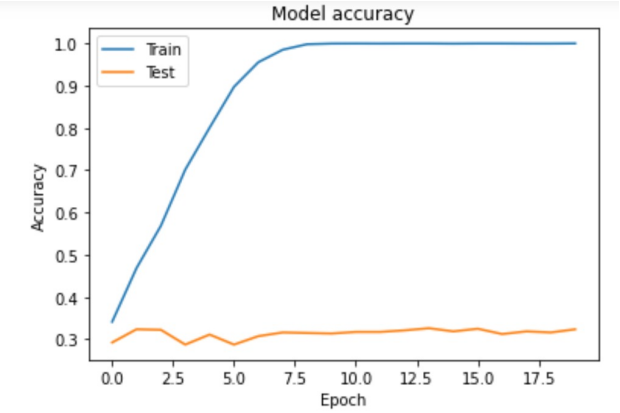


Figure 10: Training and validation accuracy with fully connected layers on *'fma_small'*

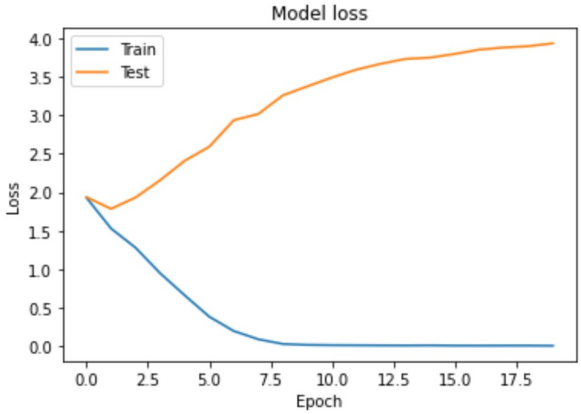


Figure 11: Training and validation loss with fully connected layers on *'fma_small'*

4. Backpropagation: Quantifying genre representation in the data

We use the finalized model to backpropagate with fixed weights to find the MFCC features that maximize genre scores. The feature that maximizes the score of a given genre corresponds to the most representative audio for that genre. We compute the difference of the highest scoring feature in the data to the most representative feature for each genre (obtained by backpropagation). Figure 12 shows the difference in these scores. We observe that the 'International' genre is the most well represented genre in the dataset, as the score corresponding to an audio is very close to the maximum score possible for that genre. On the other hand, the 'Pop' genre is the least well represented as the highest scoring audio in 'Pop' scores much less than the maximum score.

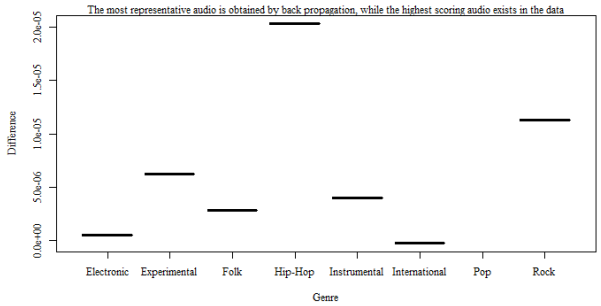


Figure 12: Distance of the highest scoring audio from the most representative audio

5. Conclusion

References

[1] James Bergstra, Norman Casagrande, Dumitru Erhan, Douglas Eck, and Balázs Kégl. Aggregate features and adaboost for music classification. *Machine learning*, 65(2-3):473–484, 2006. 1

[2] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. Convolutional recurrent neural networks for music classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2392–2396. IEEE, 2017. 2

[3] Michaël Defferrard, Kirell Benzi, Pierre Vandergheynst, and Xavier Bresson. Fma: A dataset for music analysis. *arXiv preprint arXiv:1612.01840*, 2016. 1, 2

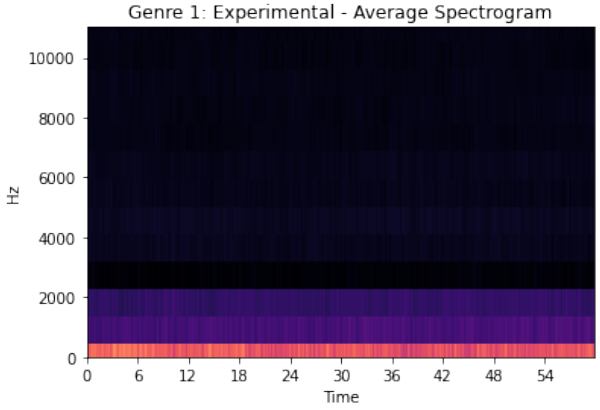
[4] Theodoros Giannakopoulos, Aggelos Pikrakis, and Sergios Theodoridis. A novel efficient approach for audio segmentation. In *2008 19th International Conference on Pattern Recognition*, pages 1–4. IEEE, 2008. 4

[5] Tao Li and George Tzanetakis. Factors in automatic musical genre classification of audio signals. In *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No. 03TH8684)*, pages 143–146. IEEE, 2003. 1

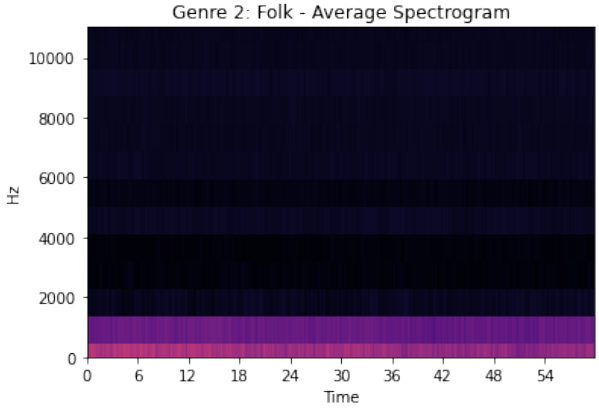
[6] Tom LH Li, Antoni B Chan, and Andy HW Chun. Automatic musical pattern feature extraction using convolutional neural network. *Genre*, 10:1x1, 2010. 1, 2, 3

[7] Paul Mermelstein. Distance measures for speech recognition, psychological and instrumental. *Pattern recognition and artificial intelligence*, 116:374–388, 1976. 1

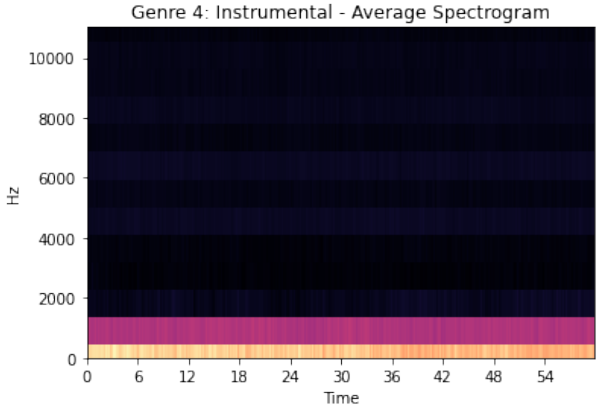
Appendix A. Genres: Time Series Spectrogram Difference Images



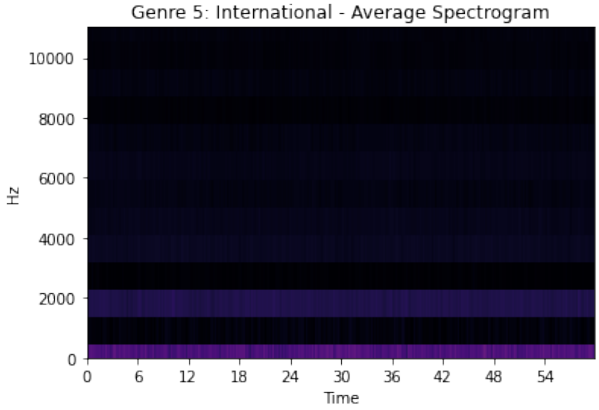
(a) Difference image between average mel spectrograms for Experimental and Electronic.



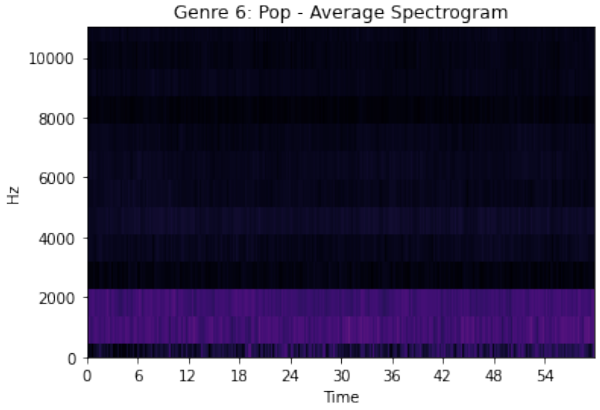
(b) Difference image between average mel spectrograms for Folk and Electronic.



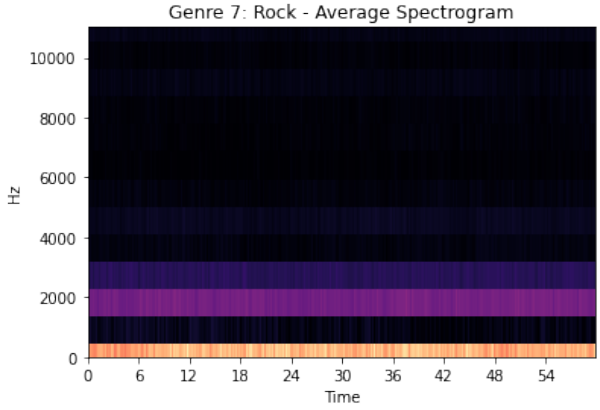
(c) Difference image between average mel spectrograms for Instrumental and Electronic.



(d) Difference image between average mel spectrograms for International and Electronic.



(e) Difference image between average mel spectrograms for Pop and Electronic.



(f) Difference image between average mel spectrograms for Rock and Electronic.

Figure 13: Per-Genre Average Time-Series Mel Spectrogram Difference Images Relative to “Electronic”

6. Work Division

Student Name	Contributed Aspects	Details
Justin Kavalan	Data Creation and Implementation	Scraped the dataset for this project and trained the CNN of the encoder.
Arvind Krishna	Implementation and Analysis	Trained the LSTM of the encoder and analyzed the results.
Josh Bishop	Exploratory Data Analysis	Conducted spectrogram analysis and prepared report figures.
Benjamin Fuentes	Implementation and Analysis	Trained the LSTM of the encoder and analyzed the results.

Table 1: Contributions of team members.