Individual Report—Music Genre Classifier

Bill Sun (1003039838) Word count: 489

I have contributed to the music genre classifier by building the code for preprocessing and for the training and validation, as well as setting up the infrastructure for training and carrying out the training of models.

For preprocessing, I first wrote the code that organizes the songs, turns mp3 files into NumPy arrays, cuts a 10-second sample from each song, and balances the dataset with an equal number of samples from each genre. This was integrated with Robert's work of downloading songs using the Spotify API, audio normalization, label creation and encoding, and train-validation data split. I then wrote code to perform Fourier Transform and encode audio to Mel-frequency cepstral coefficients (MFCC) on short intervals of audio. Before the Spotify API was implemented, I also collected songs from my personal playlist and manually sorted them by genre to form an initial data set.

For the models, even though I did not code the initial model (1D CNN), I participated in the training of that model, which proved to be a failure. I then coded up the second model (2D CNN) and integrated it with the Fourier Transform code that I wrote earlier. I trained the model using three different kernel configurations (as discussed in Section 4.2 of the Final Report) and reported the results. I also set up the Google Cloud infrastructure when it was found out that my computer did not have enough computing power to handle the training (it crashed multiple times training on the second model). Since the second model keeps underfitting, both Robert and I tried various ways of improving the results (such as adding or removing convolutional / linear layers, adding batch_norm (), adding dropout ()), but nothing seemed to work. Eventually, we decided to abandon this model and find a more reliable solution online.

We were on the verge of giving up until I came across a GitHub repository / report that performs music genre classification after applying MFCC to the audio. Since the report stated that MFCC was very effective, I decided to give it a try, and implemented MFCC to data preprocessing. Upon Robert's recommendation, I created a multi-layer GRU RNN model compatible with the MFCC data and began training. Immediately, the training and validation accuracies shot up to over 80%, far more than what the previous two models have ever accomplished. I then tweaked the hyperparameters on the RNN for better performance, and collected the graphs and best model, which were used for the report and demonstrations. Finally, it was up to Robert to finish the testing infrastructure and carry out the tests on a new set of songs (test data).

Overall, this project was a lot of trial-and-error. I learned to be patient and always be open to new ideas, and never give up. It also taught me about the nature of being a machine learning engineer, which can involve using various software tools (such as cloud) and references.