COMP4107 Final Project
Music Genre Classification Using CRNNs and the FMA Dataset

Aidan Crowther, 100980915
David Zilio

December 17, 2018

1 Introduction

FMA: A Dataset for Music Analysis, is a data-set that is provided by Cornell University. This dataset consists of 106,574 songs, of 161 varying genres, and is offered for free, with the permission of the included artists. Using a reduced version of the dataset, and examining the work of other with this dataset, we implemented our own solution to categorize this data.

2 Dataset

The FMA_small dataset utilized consists of 8000 30s audio files, with 8 balanced genres. This data is supplied in conjunction with metadata files, containing information regarding song genre, artist name, album name, etc... We broke this dataset into 7200 songs for training, and 793 songs for validation.

- Training set: 7200 Songs, 8 balanced genres
- Testing set: 793 Songs, 8 balanced genres

Note that the testing data set has 7 fewer songs than expected due to known file corruption existing within the reduced size FMA datasets. All excluded songs are listed in a ReadMe within the submission.

3 Utilized Libraries

A number of libraries were utilized in order to achieve the task of categorizing music by genre, these include:

- Keras/Tensorflow: Keras is a high level neural networking API, being run on top of Tensorflow. This library allows for fast development in Tensorflow, and provides hooks and callbacks to more easily pull out network data, and implement tools such as early stopping and tensorboard.
- Librosa: Librosa is an audio and music analysis tool built in python. It allows for advanced audio analysis, and in our case, is utilized to generate MEL spectrograms of the input data, so as to present it in a way that the neural network can understand.

4 Methodology

4.1 Data Preprocessing

The data provided within the FMA_small dataset is in a form that is not particularly useful for neural network to process. So, in order to provide this data in a form that is useful, we must first process it. We do this by utilizing librosa to generate MEL spectrograms of the 30s audio clips, utilizing 64 "MELs", or frequency bands[1]. This format presents audio data as a chart of frequency vs time, with higher values meaning that there is more active energy at that point. These spectrograms are then logarithmically scaled, to account for the logarithmic nature of sound, and is then normalized. These spectrograms are then saved as a 2D matrix of data, which is combined with its one-hot encoded genre classification within a dictionary. These dictionaries are then collected, and stored into 10 "pickle" encoded files, each consisting of $\approx$ 800 samples.

4.2 Data Augmentation

Due to the relatively small dataset utilized ($\approx$ 900 samples/genre), we need to artificially augment our data in order to prevent network underfitting, and improve the learning rate. This was accomplished by splitting each spectrogram matrix into 4 identically sized components, each of these components is equivalent to $\approx$ 7.5s of audio, and should provide sufficient data to classify the song genre, while also allowing using to increase training data size from 7200 samples to 28800. This also had the benefit of reducing the network complexity, by presenting the network with smaller overall samples to convolve.

4.3 Network Model

For any song to be classified, the features we intend for the network to detect are the frequencies utilized in conjunction, and how they relate to later time points, in order to extract genre classification based on the "shape" of the music. To this end, we made use of CNNs in order to match patterns within the input data, as well as LSTMs to help with vanishing gradients, and to build classification paths using the data from the CNN.

The network consists of, 4 convolutional layers, with kernels of size 8x(6x6), 16x(4x4), 32x(3x3), and 32x(2x2), feeding into 2 LSTM layers, and then 2 dense layers in order to then feed into the 8 output neurons to determine a classification[4]. As there are 8 classes to match, we utilized a shuffled data set, in batches of 128 in order to ensure that there is at least one entry of each classification per batch.

5 Training

Our network was originally trained utilizing 8KHz samples, with 32 MELs, and classifying utilizing this data; however, we noticed that the processing time to render more accurate spectrograms was not significantly higher, and would exclusively result in better results, so it was determined to increase this resolution. It was also noted that the genre classification data extracted from column 41 of the metadata file was not correctly balanced, and resulted in poor performance of the network, as the overall balance of files was far exceeding 40% for a single category. This was verified by utilizing the "testPickle.py" program, which lists the balance of genre data across all training sets. This issue was resolved by switching to using the genre data in column 40, and resulted in significantly better, and more reliable results.

Following this, a number of network models were tested, consisting of more/fewer CNN layers, adding/removing RNN layers, and increasing/decreasing the number of dense layers. These modifications were utilized in order to optimize the learning of our network, and were each given 20 epochs to determine roughly where their optimal accuracy would lay. The results of this testing gave us our final network model, with our best achieved accuracy being $\approx 48\%$. As our goal was to achieve $\approx 45\%$ accuracy, we believe this to be relatively successful attempt, albeit not as accurate as other models found online.

6 Testing

The network was checkpointed whenever it achieved a better validation accuracy, with this checkpointed model being saved, and exposed utilizing "getClassification.py" in order to allow either; a single mp3 to be classified, with the output being the predicted genre; or a pickle file containing encoded mp3s, where the output is the accuracy of the network at predicting the data in that set. The network will also generate a spectrogram image of every mp3 file it processes for debugging purposes, and will automatically scale the provided file down to a randomly selected 7.5s sample[2]. The usage for this testing program is

```
python getClassification.py [songToClassify].mp3

#or

python getClassification.py [dataSetToValidate].pkl

#if you wish to observe the CNN activations for an input file, run using

python visualize.py [songToClassify].mp3
```

The pickle files utilized for our testing can be found at http://www.aidancrowther.com/Data.zip as this folder was omitted in the submission due to large file size. An example file is included for testing (test.mp3).

7 Possible Improvements

7.1 Dataset Improvements

Our utilized dataset consists of small sample lengths, and a sparse population of only ≈ 1000 samples per genre. This was due to our use of the FMA_small dataset, which was inspired by it's use of balanced genre classes, and a small download size (8GB vs 900GB). Although this proved to be functional for our task, this lack of significant training data is likely a contributor to our low accuracy, and is a probable cause for our observed data overfitting as time went on. As such, a larger set of samples, along with an increased sample rate and number of MELs for the generated spectrogram should likely provide better results; albeit at the cost of significantly more time, and processing resources.

Other dataset issues include the fact that many of the songs utilized in this training sample have multiple genres associated with them. For the case of simplified training, we utilized the most heavily influential genre as the classifier for any given song; however, this does have to effect of reducing our perceived accuracy by having the network be marked as incorrect for guessing another subgenre that the song may have. An example of this behaviour can be seen using our "test.mp3" file, which is a song with the genre of "Electronic Rock"; despite our network almost always marking it as one of these two genres, it would still receive poor performance for this. As such, we believe that future efforts in the field of music classification should utilize more distinct genres, songs with only one, or very few distinct genres, and possibly utilize an alternate classification system that allows for the network to classify songs with multiple genres.

7.2 Model Improvements

A significant cause of inaccuracy in our generated model was its tendency to overfit data. We combated this by utilizing recurrent network layers, and dropoff ratios in order to prevent neurons from building extensive dependencies on each other for classification. This had the effect of directly reducing the learning rate per epoch, but also resulted in network accuracy $> 40\%$. By then combining these improvements with early stopping for our network, we were able to generate a model that provided a good fit for our data, while reducing the impact of overfitting as we had previously seen. This also helped by reducing the occurrences of vanishing gradients with our ADAM optimizer as we approached convergence, allowing for our network to better deal with local minima.

Along with these optimizations to the model, some other potential improvements could potentially be found in simplifying the model, such as by reducing the number of convolutional layers, as these do leave fewer sources of data per filter for the network to base classifications on. This was a trade off that we decided to accept, as having more convolutional layers allowed for us to perform more feature extraction, while also reducing the network complexity as compared to having more connections between the LSTM and CNN layers. This allowed to to achieve acceptable classification performance in a reasonable amount of time.

7.3 Kernel Improvements

Our utilized network has 4 kernels of varying size, each hoping to find different patterns within our input data. These kernel sizes were modified and tweaked as network modifications took place; however, there was no significant time put into optimizing these kernels beyond a reasonably performing layout. It is possible that by utilizing kernels on the first layer that scan all 64 MELs at once, and only a few components of the time scale may provide better pattern recognition for the change in frequencies used, and their power, over time. Given more time to develop this system, much focus would be placed on the optimization of these kernels in order to further increase the performance of our network.

Our kernel choices may have also had a negative impact on our classification accuracy due to their relatively large size. This may have contributed to overly reducing the dataset, and possibly obscuring important features that may have otherwise lead to improved classification. As such, we believe that with more time for network training on a network with smaller kernels, and possibly less pooling utilization, classification accuracy could potentially be improved to exceed 50% classification.

8 Conclusion

The reduced size FMA_small dataset that we utilized presented a number of interesting challenges for the development of our network. Between corrupted/malformed data, and ambiguous column definitions in the metadata, a number of issues arose while developing our network. However, it was not so difficult as to be impossible, and presented a useful opportunity to further knowledge in the field of neural network design and optimization.

Our network made use of a CRNN, and achieved classification accuracy of $\approx 48\%$, which is better than our intended accuracy goal, however, it is not as accurate as some other similar system appeared to be able to achieve. We believe this to be in part caused by the small number of samples provided by our dataset; however, we do also feel that a not insignificant factor in this poor classification is due in part to the genres selected for use in the FMA dataset. "Instrumental" for example is a genre that may be difficult to classify, as the only major separator from others is the lack of vocals, and depending on where in the song we are, there may be insufficient data to correctly distinguish between whether a song is on an instrumental solo, or simply has no vocals, but is not classified as instrumental. Other issues raised specifically by the music within our dataset include the fact that a majority of the songs within our dataset are of multiple genres ($> 60\%$ are classified as having International classification), this leads to the network having a perceived performance for certain songs, where the network may classify using one of the alternate genres of this song. These kinds of distinctions between genres may have played a part in our poor performance, and in future, it may be advisable to utilize more distinct musical genres than those in this dataset.

As for the small sample size of our dataset, our data augmentation scheme seemingly served relatively well, with an improved learning rate, and as far as we could determine from runs with and without augmentation, resulted in higher overall classification accuracy as compared to the non augmented data. From what we were able to find online regarding the training of neural nets on auditory data, it appears that samples between 2-5s are typically used, and as such our scheme should allow for sufficient feature extraction, while improving our overall network accuracy. Potential improvements for this augmentation system may include the use of randomization of the data splitting, while also potentially implementing a K-fold technique in order to improve the selection and distribution of data samples such that we classify using more distinctive musical features.

These samples may have also presented more useful features had we utilized a higher sampling rate for our preproccessing system, along with an increased number of MELs in order to improve the quantity of data presented to the network, potentially allowing it to identify more characteristic features within the spectrogram for different genres, as well as allowing for more output data to be presented after passing through all CNN kernels and pooling layers. These improvements to the network would definitely present a good starting point since the time to process with these improved properties is not significantly longer, and as such should likely be run with these parameters from the beginning. Although we will make available our utilized dataset should the particulars of our decoding be found to be helpful at all.

This leads us to conclude that music genre classification is a problem that is relatively well suited to the tasks of neural networks, achieving a relatively high accuracy, while a human may have trouble at times distinguishing some of these features. With future improvements in the data utilized, as well as the model constructed, we believe that this problem still has room to increase accuracy before stagnating.

9 Figures

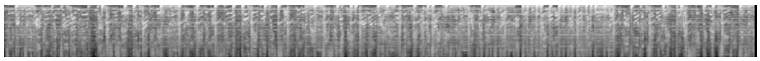

Figure 1: A generated spectrogram

```
Validation accuracy:  47.8562421185372 %
Estimated genre classification:  Rock
```

Figure 2: Results returned by getClassification.py

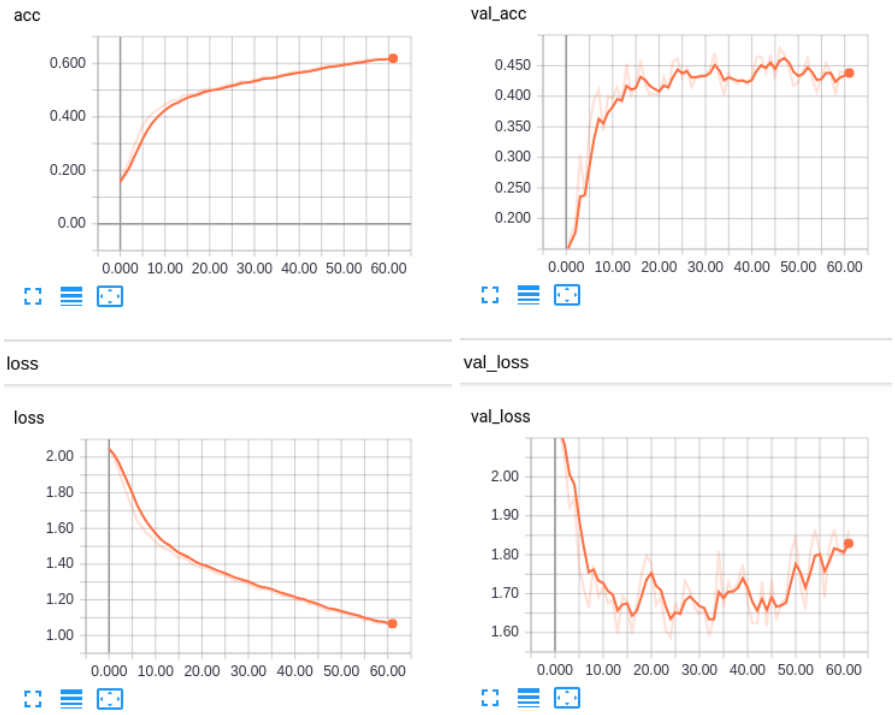

Figure 3: Training and Validation loss/accuracy

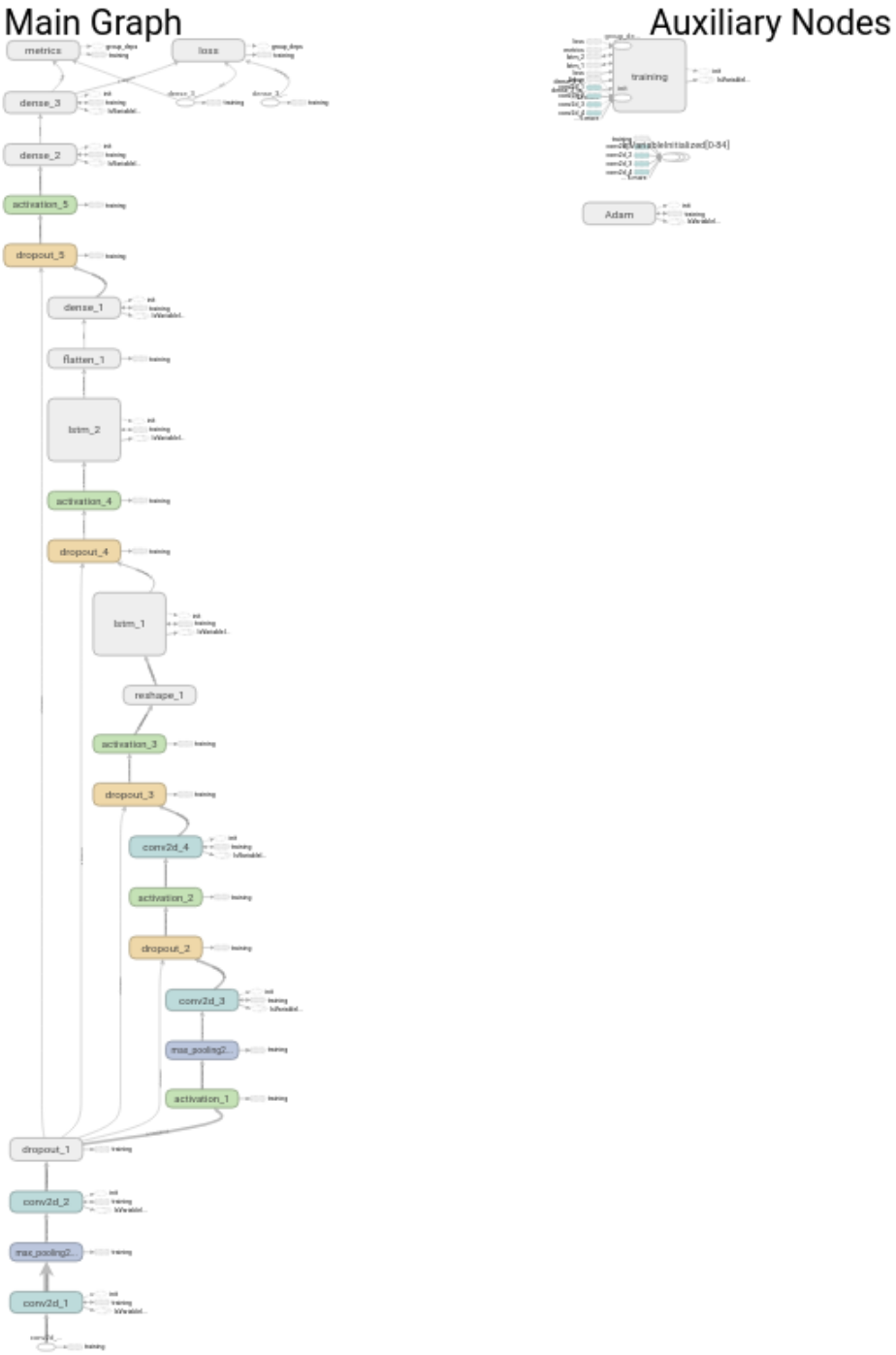

Figure 4: The resultant network model

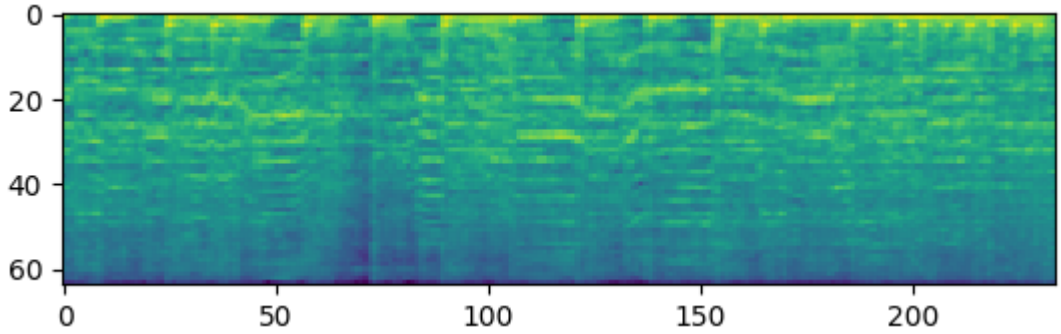

Figure 5: Generated spectrogram for a test file from visualize.py

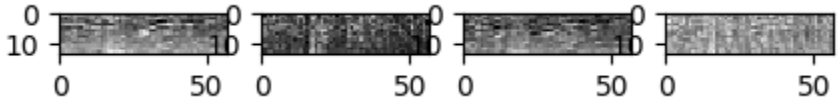

Figure 6: Sample of filter activations from the first convolutional layer

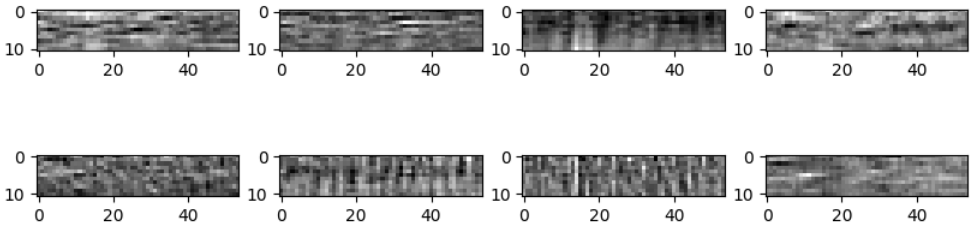

Figure 7: Sample of filter activations from the second convolutional layer

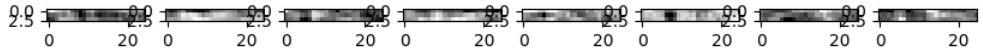

Figure 8: Sample of filter activations from the third convolutional layer

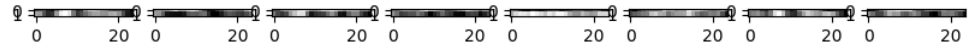

Figure 9: Sample of filter activations from the fourth convolutional layer