Neural networks for music genre recognition

Nikhil George Titus
University of Massachusetts Amherst
nikhilgeorge@cs.umass.edu

Abstract

Music genre recognition is the task of identifying the genre of the music from the given input file. Music genre recognition is of much interest in the MIR community. This project explores two approaches to music genre recognition for two use cases. One is using a LSTM and taking the weighted average of the probability distribution at each time step. The second model is using convolutional neural networks with global temporal pooling layer. The models were tested out with two data sets: GTZAN and FMA dataset. Accuracy of 77.3% was obtained in the test set of FMA data set.

1. Introduction

Music genre recognition is the task of categorizing music into one of the music genres. This is of much convenience as people tend to like music that belongs to a particular genre. The task is challenging as similar to image recognition we would need a high capacity function to classify the songs. Identifying features that categorize a music to a particular genre is challenging. Historically music genre recognition is done using a set of features as mentioned in [9]. Recently, various approaches using neural networks on the spectrograms of the music file are tried out. This project makes an attempt to classify music into a particular genre based on the mel-spectrograms of the music file input. Mel-frequency cepstrum is obtained by a linear cosine transform of a log power spectrum on a non linear scale called mel scale [1].

Convolution neural networks that are extensively used in image recognition can be used here with the spectrograms too. LSTMs are good at sequences of inputs and hence this can be used for time distributed sequence of the inputs.

2. Related Work

Aaron van den Oord uses mel-spectrograms for music recommendation [10] . This further strengthens our assumption that using mel-spectrograms is a good approach for music genre recognition.

Grzegorz Gwardys and Daniel Grzywczak [4] uses spectrograms and transfer learning using an already trained ILSVRC CNN model. This paper also relies heavily on a lot of feature engineering to achieve an accuracy of 72%

Mel-spectrograms are also used in an article on recommending music on spotify [3].

Convolution neural networks and mel-spectrograms were also tried out successfully by Piotr Kozakowski & Bartosz Michalak[6]

3. Problem statement

The problem statement can be summed up as follows: Given an input music file X_i we need to identify the genre label G_i. We are also given a training set that consists of several music files $(X_1, X_2, .., X_n)$ and genre labels $(G_1, G_2.., G_n)$

4. Dataset

Two data sets where used as part of the project to compare and test the models.

The first data set used was the GTZAN data set. Each genre consists of 100 songs each 30 seconds long. Different genres available in the data set are: blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, rock. Thus a total of 1000 music files were available.

The second data set used was a relatively new data set called the FMA data set [2]. The data set consists of 8000 songs split equally into 8 genres: Hip-Hop, Pop, Experimental, Rock, International, Electronic, Instrumental. Each song had a length of 30 seconds.

The data set is split into training validation and test set in the ratio 70%:20%:10%.

5. Approach

First the data is converted into mel-spectrogram using the librosa python library [7]. This results in a two dimensional array. A sample visualization of a rock song that was

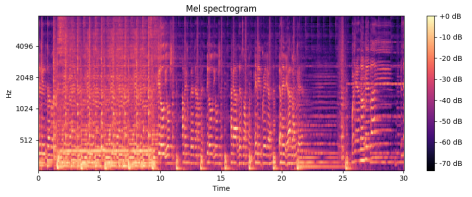

Figure 1. Mel-spectrogram of a rock song in GTZAN data set

taken from the GTZAN data set is shown in figure 1. Y axis represents frequencies and X axis represents time.

Two models where tested out as part of the project. These models are explained below:

5.1. Model 1: Weighted average of the probability distribution of LSTM at each time step

I started off the project by implementing a LSTM based model for genre classification but later used the model by Piotr Kozakowski & Bartosz Michalak [6]. This LSTM based model is of much importance as we can use it in applications where the entire song needn't be fed to arrive at a decision on a particular genre. The primary difference in my model is that I used the weighted average of the probability distribution at each time step. The main reasoning being that the LSTM builds confidence in the genre of the music at each time step. This was mentioned by Piotr Kozakowski & Bartosz Michalak in [6]

The data is fed into a multi layer 1D convolution neural network. A filter size of 256 and filter length of 4 was used. RELU activation and max pooling layers were used. A one dimension convolutional neural network was used as the data is invariant only along the time axis.

Then this data is fed into a LSTM model. The resulting time distributed data is fed into a fully connected layer with softmax activation. The resulting data is a probability distribution of the genre given the input at each time step.

A weighted average of the probability distribution at each time step is taken:

$$P = \frac{\sum w_t P_t}{\sum w_t}$$

where w_t is the time elapsed from the start for the particular distribution ie at time step one the weight is one and at time step 2 the weight is 2 and so on. Thus the last time step will have the maximum weight. The architecture diagram of model 1 is shown in figure 2

5.2. Model 2: Convolution neural network with global temporal pooling

The LSTM model is good if we have to make predictions per time step. It was mentioned by by Kozakowski & Michalak [6] that the authors were able to get much better accuracy with a different architecture. I implemented an

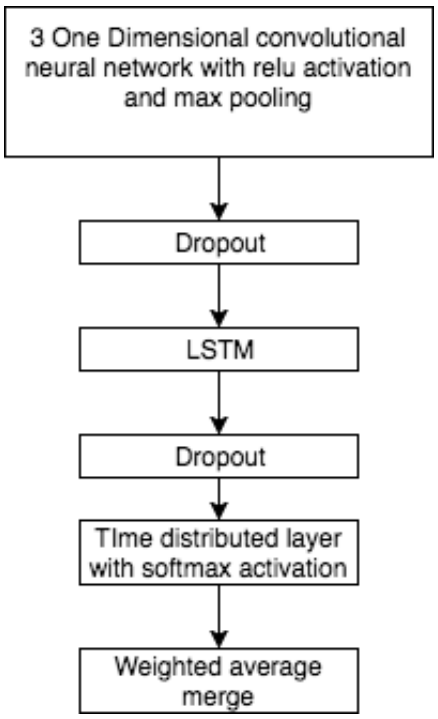

Figure 2. Architecture of model 1

architecture which is very similar to the architecture used by S. Dieleman for latent factor prediction at Spotify [3]. The network consists of 3 1D convolutional neural network with max pooling and average pooling layers as before but after that we have a global average pooling and max pooling layer which are concatenated. I initially started with another l2 layer as mentioned in [3] but I had to drop it as the network was unstable with exploding gradients and hence the l2 layer was removed. The primary reason for having the global temporal pooling layer is to make the outputs invariant along the time axis. For example: A feature of a rock song seen at time 0 is the same as that seen at time 30.

After the max pooling layer we have a dropout layer and an activation layer with softmax activation. Most of the dense layers mentioned in [3] were removed as it was resulting in over fitting of data. Architecture diagram of model 2 is shown in figure 3

6. Experiments and results

The fist step is preprocessing the data to obtain the mel-spectogram. Scripts for parsing GTZAN data set was obtained from [6]. For FMA data set first the meta-data was obtained from pandas dataframe object which was distributed as part of the data. I wrote scripts to map the songs to the particular genre and to get the mel-spectrogram of the resulting songs. It took 5 hours to pre-process the songs of FMA in a p2.xlarge AWS instance.

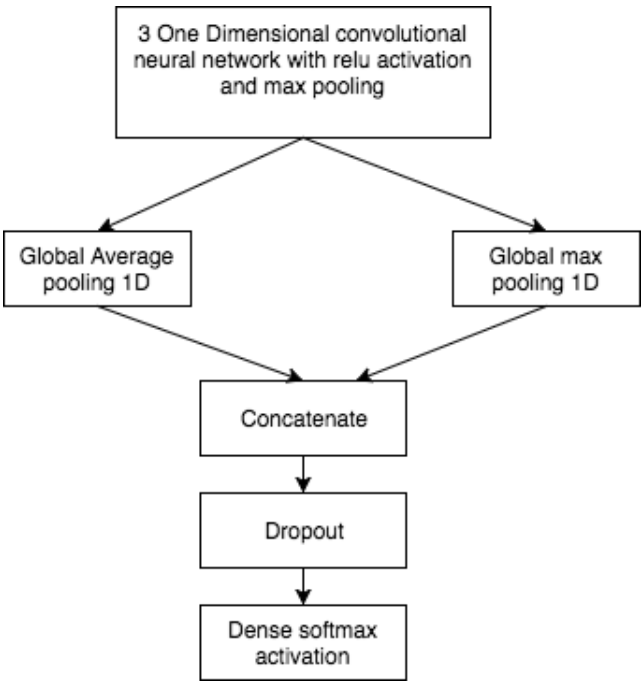

Figure 3. Architecture of model 2

6.1. GTZAN data set classification using model 1

I started my experiments with the GTZAN data set as the data set was small and it was more manageable to train and test the model. I used the code from [6] and made changes to add the time distributed layer. The code was written in keras but I had to write code in tenserflow bypassing the keras layer to get the layer that does the weighted average. Since the data set was small I was using batch size of 64. Rest of the parameters used were same as that of the code provided by Kozakowski et al.

The weighted average method had similar accuracy as that of the simple average method. It had a validation set accuracy of 59.4% and a test accuracy of **60.9%** which is almost equal to that of that of normal average method which had a test accuracy of 62%. The training, validation and test data were split in a similar manner for both the experiments. The training and validation loss of this model can be seen in figure 4 and the training and validation accuracy per epoch can be seen in figure 5.

6.1.1 An empirical analysis of the results of the weighted average and the simple average method

After analyzing the results of probability distribution across time we can see that the LSTM arrives at a particular genre very quickly and it chooses that particular genre with confidence even when the genre is wrong.

We can see in figure 6 the probability distribution of a country song that was correctly classified. LSTM quickly

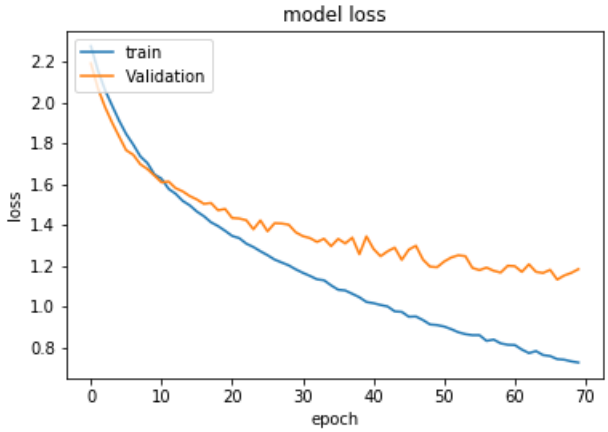

Figure 4. Training and validation loss of model1 on GTZAN dataset

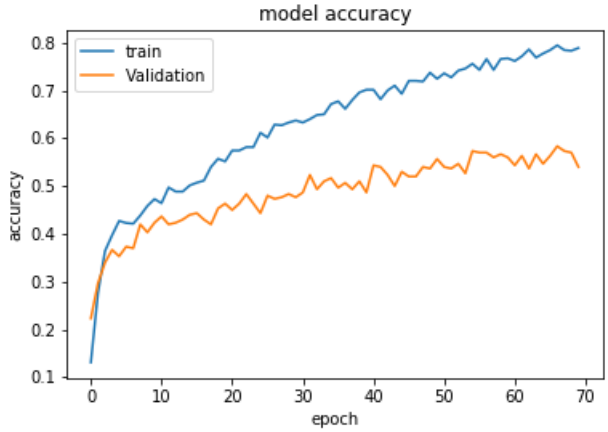

Figure 5. Training and validation accuracy of model1 on GTZAN dataset

builds confidence that the song is a country song and we get very high probability for country genre. Here the weighted average and the simple average would give similar results. Similarly we can see in figure 7 the probability distribution of a song that was incorrectly classified as reggae. Here too the weighted average and the simple average method returns the same results. This pattern was seen in most of the predictions and this is the reason why both the models are giving the same accuracy.

6.2. GTZAN data set using model 2

I coded up the second model as shown in figure 3 using keras. There were 256 one dimensional convolution filters. Drop out layers were added for regularization. The drop out rate was 0.5. Batch size was set to 256. Stochastic gradient descent was used and learning rate update was done using Adam. Learning rate was set to 0.0001. Softmax activation and cross entropy loss was used for training. By using this

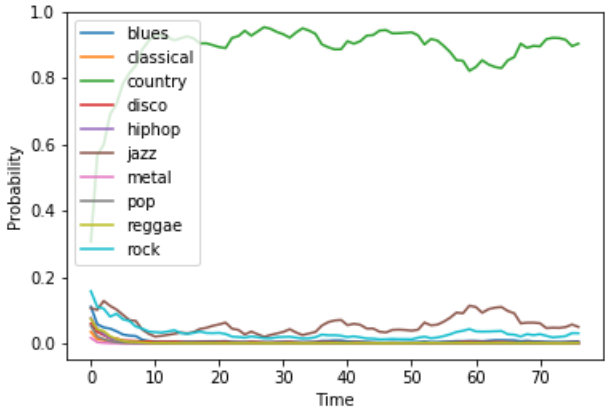

Figure 6. Probability distribution across time steps for a song correctly classified as a country song

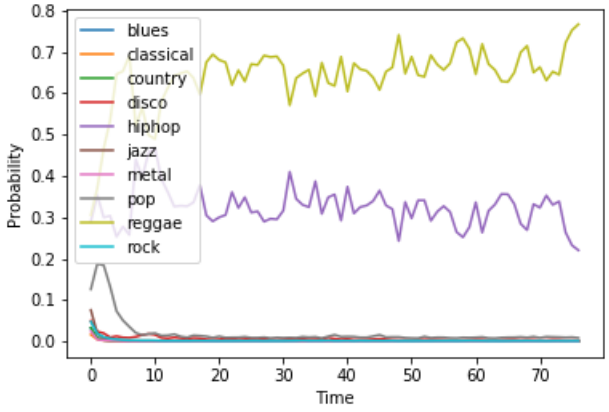

Figure 7. Probability distribution at each time step for a hip-hop song that was incorrectly classified as reggae.

model I was able to get a validation accuracy of 77% and a test accuracy of **74.01%**. The state of the art for GTZAN data set is 84% as mentioned in [5].

6.3. FMA data set using model 2

As I was able to get a high accuracy for model 2 with the GTZAN data set which is a small dataset of 1000 songs, I was ready to run my model on a much larger dataset. The FMA dataset was chosen since it had 8000 songs belonging to 8 genres. The model was trained in an AWS p2.xlarge instance. I was able to get a test accuracy of **77.32%** and a validation accuracy of 75%. This is the best model that was obtained. Figure 8 shows the training and validation loss across epochs. Figure 9 shows the training and validation accuracy across epochs. Figure 10 shows the confusion matrix in the test set. It can be seen from the confusion matrix that genres like experimental and instrumental or electronic and hip-hop are confused a lot.

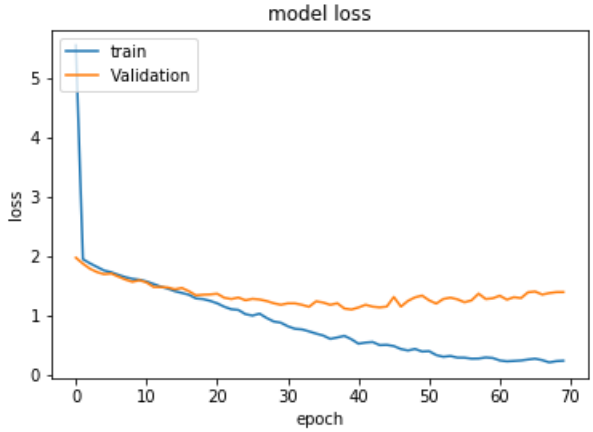

Figure 8. Training and validation loss of model 2 on the FMA dataset

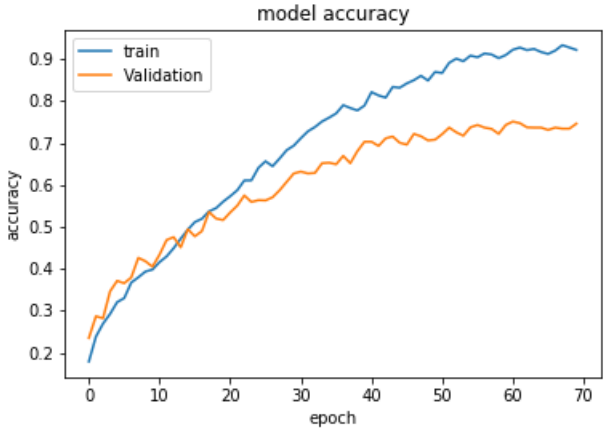

Figure 9. Training and validation accuracy of model 2 on the FMA data set

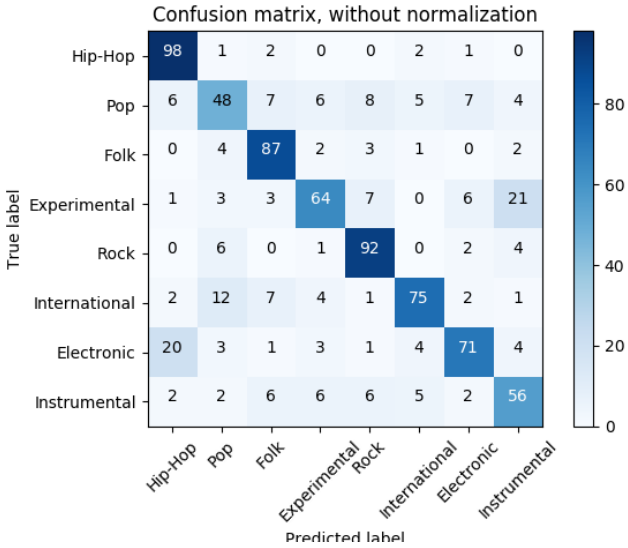

Figure 10. Unnormalized Confusion matrix of model 2 on the FMA data set

7. Conclusion

Two architectures for music genre recognition was explored. One was based on the weighted average of the probability distribution of LSTM and the other was using global temporal pooling layer. The architecture with CNNs and global temporal pooling gives much better accuracy than the LSTM based approach. There are some use cases where we need a continuous probability distribution and the lstm based approach can be used. In future, I would like to explore dilated convolutional neural networks for better representation of the sound signals as suggested by Van den Oord et al [8]. Music generation and recommender systems which are a combination of neural networks and collaborative filtering is something else that I want to explore.

References

[1] Wikipedia. `https://en.wikipedia.org`.

[2] K. Benzi, M. Defferrard, P. Vandergheynst, and X. Bresson. Fma: A dataset for music analysis. *arXiv preprint arXiv:1612.01840*, 2016.

[3] S. Dieleman. Recommending music on spotify with deep learning. `http://benanne.github.io/2014/08/05/spotify-cnns.html`.

[4] G. Gwardys and D. Grzywczak. Deep image features in music information retrieval. *International Journal of Electronics and Telecommunications*, 60(4):321–326, 2014.

[5] P. Hamel. Deep learning in mir demystifying the dark. `https://marl.smusic.nyu.edu/wordpress/wp-content/papercite-data/pdf/ISMIR2013_Deep_Learning_Part2_Hamel.pdf`.

[6] P. Kozakowski and B. Michalak. Music genre recognition. `http://deepsound.io/musicgenrerecognition.html`.

[7] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, pages 18–25, 2015.

[8] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[9] N. Scaringella, G. Zoia, and D. Mlynek. Automatic genre classification of music content: a survey. *IEEE Signal Processing Magazine*, 23(2):133–141, 2006.

[10] A. Van den Oord, S. Dieleman, and B. Schrauwen. Deep content-based music recommendation. In *Advances in neural information processing systems*, pages 2643–2651, 2013.