Music Genre Classification Using Convolutional Neural Networks

Kurt Weinheimer
University of Maryland, Baltimore County
Baltimore, Maryland
kweinh2@umbc.edu

Abstract

Music genre classification is an important problem to music streaming companies in the modern era and a fascinating problem for machine learning researchers. Through deep learning methods that implement convolutional neural networks (CNNs), 70% accuracy in 10-Genre classification has been achieved which is equivalent to human level classification. In this paper, the proposed CNN takes in mel-spectrograms to mimic the human auditory system and achieves 85% accuracy in 3 genre classification, 72% accuracy in 4 genre classification and 64% accuracy in 5 genre classification.

Introduction

As musicians create, music genres are constantly being changed and boundaries are being pushed. An artist can easily merge two of their favorite genres, possibly rock and roll and hip-hop and create a whole new genre, rock-hop. Where one genre starts and another ends, can be a difficult idea for humans as well as machines to grasp.

If the genre can be learned however, there are many applications involving music that would benefit from it. Every music streaming site (i.e. Spotify, Apple Music, Soundcloud, etc.) already uses this genre classification to categorize music and suggest possible songs to users to keep them on the platform longer and increase overall use of the site. They most likely use machine learning methods to do this, and in this paper, I will be testing what kind of results I can get from using different convolutional neural network models.

Background

Sound is stored in a computer in the time domain as a waveform. There are a certain number of samples per second (usually 44100 or 22050) where each sample is the amplitude at that time. See Fig. 1. Machine learning models can find some patterns in this domain and the research is ever expanding (i.e. WaveNet for Audio Source Separation [1]), but for this problem we are going to try working with spectrograms as it provides more data about the frequencies that happen at each time step.

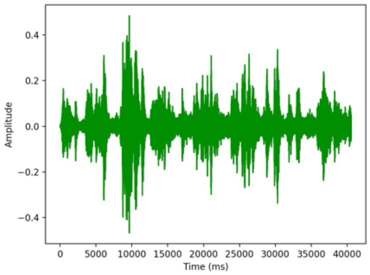

Fig.1 Wav

Spectrograms are created by a Short Time Fourier Transform on a window that slides over the wave/mp3 file usually overlapping with the previous window 50-75%. This function outputs a three-dimensional graph with axes for time, frequency, and amplitude which allows the sound to be viewed as an image with the amplitude axis usually being represented by the color of the pixel. See Fig.2. With all the recent advances in computer vision, this format opens the door to more sophisticated architectures that are already out there.

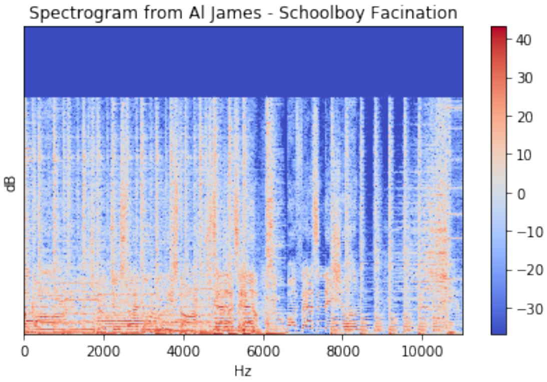

Fig.2 Spectrogram

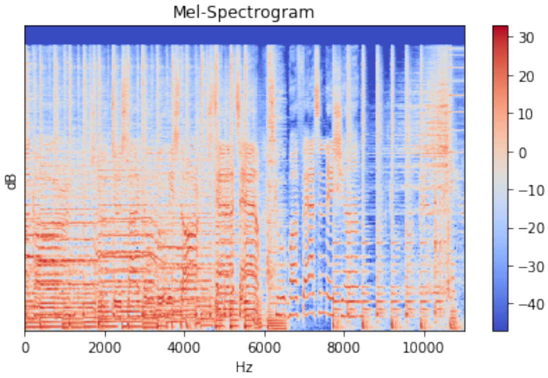

Fig.3 Mel-Scale Spectrogram

To take this one step further, we will scale the y-axis or the frequency dimension logarithmically to create a mel-scale spectrogram or mel-spectrogram. This imitates human hearing which can detect more subtleties in the lower frequencies and less in the higher frequencies. As most sounds occur in these lower frequencies, more patterns should be able to be detected by our models. See Fig.3. The benefits of mel-spectrograms can be seen in detail here [3].

Dataset

To conduct this study, I decided to use the Free Music Archive [5] which is a common dataset used for music genre classification. The small version of the dataset they offer contains 8 different genres with 1000 different mp3 files of 30 second samples. The 5 genres I used for this study were Folk, Hip-hop, Pop, Instrumental and Experimental.

Using the Python package Librosa [4], the mp3 files were transformed to mel-spectrograms that were roughly 10 seconds in length meaning there were 3000 samples per genre. Each mel-spectrogram was made up of 256 timesteps and 256 mel-filter bins. The training set was then comprised of a randomly selected 70% of the samples with the validation set being the remaining 30%.

Approach

The goal of the experiments was to get above 90% accuracy on the classification of the validation set starting with 3 different genres and moving up to 5. I played around a lot with the architecture of the model, but I always tried to keep it simple enough. The architecture seen in Fig. 4 yielded the best results. What is not seen in this are the L2 regularization methods for the convolutional layers. The L2 regularization methods and dropout layers help reduce overfitting and produce better results on the validation set. The models were coded in Python using Keras and Tensorflow with the Adam optimizer and a learning rate of 0.001. Each was run for 50 epochs using cross entropy loss as the loss function.

```
Layer (type)                 Output Shape              Param #   
=================================================================
reshape_3 (Reshape)          (None, 256, 256, 1)       0         
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 256, 256, 32)      320       
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 128, 128, 32)      0         
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 128, 128, 32)      9248      
_________________________________________________________________
max_pooling2d_12 (MaxPooling (None, 64, 64, 32)        0         
_________________________________________________________________
conv2d_13 (Conv2D)           (None, 64, 64, 32)        9248      
_________________________________________________________________
max_pooling2d_13 (MaxPooling (None, 32, 32, 32)        0         
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 32, 32, 32)        9248      
_________________________________________________________________
max_pooling2d_14 (MaxPooling (None, 16, 16, 32)        0         
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 16, 16, 32)        9248      
_________________________________________________________________
max_pooling2d_15 (MaxPooling (None, 8, 8, 32)          0         
_________________________________________________________________
flatten_3 (Flatten)          (None, 2048)              0         
_________________________________________________________________
dropout_5 (Dropout)          (None, 2048)              0         
_________________________________________________________________
dense_7 (Dense)              (None, 100)               204900    
_________________________________________________________________
dropout_6 (Dropout)          (None, 100)               0         
_________________________________________________________________
dense_8 (Dense)              (None, 100)               10100     
_________________________________________________________________
dense_9 (Dense)              (None, 4)                 404       
=================================================================
Total params: 252,716
Trainable params: 252,716
Non-trainable params: 0
_________________________________________________________________
```

Fig. 4 Model Architecture

Results

For 3-Genre classification of folk, hip-hop and instrumental, the lowest validation loss came in epoch 28 with 0.4616 which resulted in a validation accuracy of 85.2%. The loss and accuracy graphs for 3-Genre can be seen below.

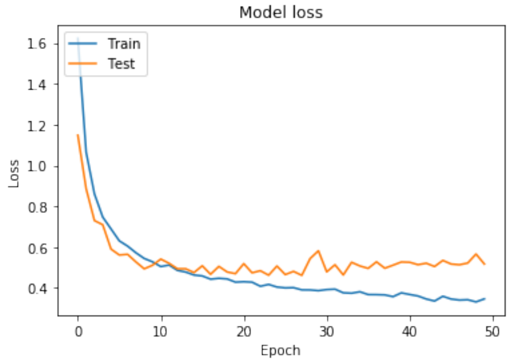

Fig.5 3-Genre Loss

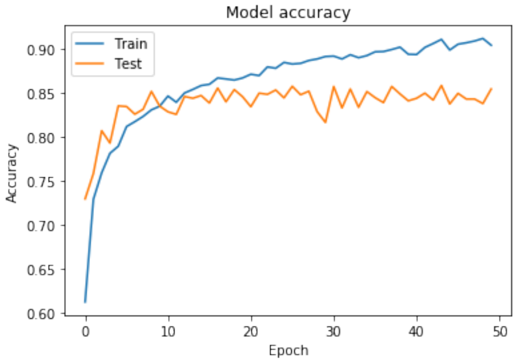

Fig.6 3-Genre Accuracy

For 4-Genre classification of folk, hip-hop, instrumental and pop, the lowest validation loss came in epoch 24 with 0.777 which resulted in a validation accuracy of 72.85%. The loss and accuracy graphs for 4-Genre can be seen below.

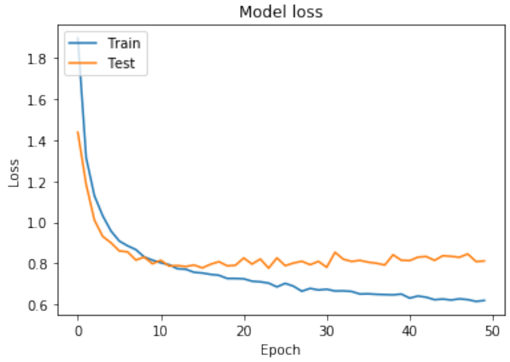

Fig.7 4-Genre Loss

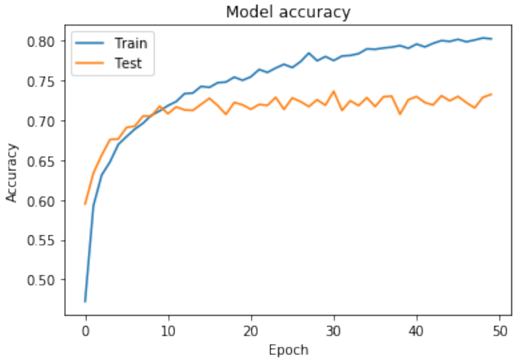

Fig.8 4-Genre Accuracy

For 5-Genre classification of folk, hip-hop, instrumental, pop and experimental, the lowest validation loss came in epoch 31 with 1.0231 which resulted in a validation accuracy of 64.32%. The loss and accuracy graphs for 5-Genre can be seen below.

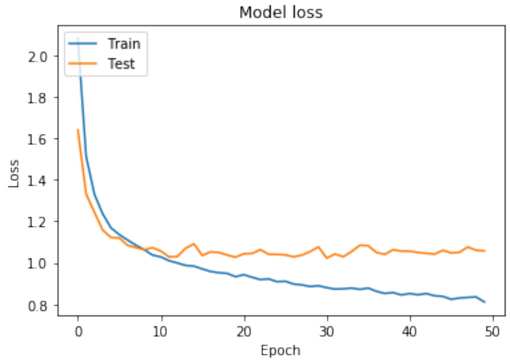

Fig.9 5-Genre Loss

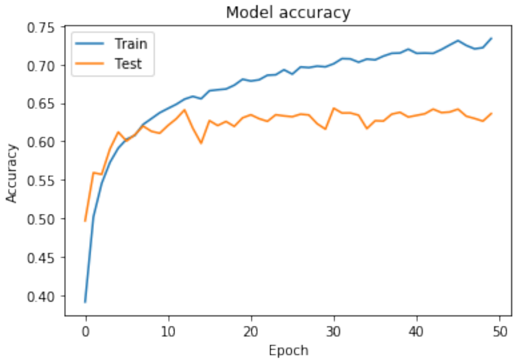

Fig.10 5-Genre Accuracy

Analysis

Compared to other papers, these results are not bad, but they do not measure up to the 70% accuracy achieved for 10-Genre classification by [6]. For a similar study from [2], they achieved 50% accuracy for 8-Genre classification, and it seems that this model would get close to that number if applied to 8 genres. From what I was seeing online for 5-Genre classification, the human level of accuracy is around 70-80% which is about 6% better than the proposed model, but still close.

To improve these results further it would be nice to add in more data which is definitely possible as Free Music Archive offers larger datasets. It would also be interesting to apply state of the art models that perform object classification on ImageNet such as AlexNet and ResNet to this problem to see how they perform in comparison. Another idea could be some method of unsupervised clustering that would allow the model to extract its own genres which may lead to a whole new way of categorizing music.

Conclusion

The benefits from genre classification for music streaming sites is clear in the recommendation systems they implement as well as any other model that works with songs as now the genre can also be taken as a feature. Spotify lists over 1,000 genres on its site which is a pretty hard number to believe [8]. Accurately tagging 1,000 genres and sub-genres of music would seem to be a nearly impossible task for not only machines, but humans as well as Spotify most likely has some human element in the tagging process. Machines are getting close to human level, but at this point there are still strides to be made.

Works Cited

[1] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior: “WaveNet: A Generative Model for Raw Audio”, 2016; [http://arxiv.org/abs/1609.03499 arXiv:1609.03499].

[2] Dwivedi, P. (2019, March 27). Using CNNs and RNNs for Music Genre Recognition. Retrieved December 1, 2019, from https://towardsdatascience.com/using-cnns-and-rnns-for-music-genre-recognition-2435fb2ed6af.

[3] Huzaifah, Muhammad. "Comparison of time-frequency representations for environmental sound classification using convolutional neural networks." arXiv preprint arXiv:1706.07156 (2017).

[4] Librosa - Audio Tools Python Package https://librosa.github.io/librosa/index.html

[5] Michaël Defferrard, Kirell Benzi, Pierre Vandergheynst: “FMA: A Dataset For Music Analysis”, 2016; [http://arxiv.org/abs/1612.01840 arXiv:1612.01840].

[6] Mingwen Dong: “Convolutional Neural Network Achieves Human-level Accuracy in Music Genre Classification”, 2018; [http://arxiv.org/abs/1802.09697 arXiv:1802.09697].

[7] Pandey, P. (2018, December 19). Music Genre Classification with Python. Retrieved December 1, 2019, from https://towardsdatascience.com/music-genre-classification-with-python-c714d032f0d8.

[8] Simmy Richman @simmyrichman. (2015, November 16). Spotify's 1,371 musical genres: How to tell drone folk from skweee. Retrieved December 1, 2019, from https://www.independent.co.uk/arts-entertainment/music/features/spotifys-1371-musical-genres-how-to-tell-drone-folk-from-skweee-a6736971.html.