Music Genre Classification on the Free Music Archive

Daniel Amin

Data Science Program
The Graduate Center: CUNY
365 Fifth Avenue. New York, NY 10016
damin@gradcenter.cuny.edu

Kiwook Kwon

Data Science Program
The Graduate Center: CUNY
365 Fifth Avenue. New York, NY 10016
kkwon@gradcenter.cuny.edu

Stephanie Luk

Data Science and Engineering Program
The City College of New York: CUNY
160 Convent Ave. New York, NY 10031
sluk000@citymail.cuny.edu

***Abstract*— Semi-automating or automating this process could assist or replace humans in the many different Music Information Retrieval (MIR) tasks. In this project, we classified eight music genres by using various deep learning architectures on a subset of the full FMA dataset. We implemented various Neural Network models: naive dense architecture, Convolutional Neural Network (CNN), VGG16 and trained them over a subset of the Free Music Archive (FMA) dataset. Our best performing model was a simple Dense network trained over MFCC data, followed by a retrained VGG16 architecture on Mel-audio-spectrograms. This experimental result shows that our method achieved the performance which was quite similar to many others related works in this field.**

Keywords—genre classification, FMA, music, CNN, VGG16

I. Introduction

The music industry is one of the biggest industries existing today, ranging from music streaming platforms such as Spotify, Youtube Music, etc. to record labels, music distributors, advertisers, and so on. Most of the time these corporations receive thousands of songs per day to be reviewed for business. Spotify and other streaming platforms need genre classification in order to build their playlists and recommendations; record labels, distributors and advertisers receive demos and need genre classification as well to know how to properly market the music. In some cases, record labels and advertisers also scout for music with potential to be signed and marketed, in this case, usually done manually by humans. Music Information Retrieval (MIR) is a big field concerned with browsing, searching, and organizing large music collections. There are many tasks that can be done such as music genre classification, music recommendation, music generation etc. Many leading companies in this industry have developed the state-of-the-art techniques by running their own research laboratory. For example, Spotify has published research papers robustly.[1] With the massive amounts of music in the world there is a huge push to be able to browse, search, and organize large music collections efficiently.

We chose to focus on the music genre classification task. A music genre is a conventional category that identifies some pieces of music as belonging to a shared tradition or set of conventions [1]. Music can be divided into different genres in many different ways. The artistic nature of music means that these classifications can and are often subjective and controversial, and some genres may overlap. If a deep learning architecture has good success in genre classification, it can automate the process; assisting or replacing humans, provide framework for development and evaluation of features for any type of content-based analysis of musical signals. The goal of this project was to see how well we can classify the genre of a song given the audio file using deep learning. The motivation of these tasks is to improve the automation of these processes in the music industry.

II. Related Works

Convolutional Neural Networks (CNNs) have been successfully utilized in various image classification problems. Spectrograms were introduced by the paper, "Detection, estimation, and classification with spectrograms" in 1980 [2]. This technique converts audio data to image data; thus, it allows the use of CNN although audio data is sequential. In 2015, spectrograms were proposed to use for CNN in practice [3]. There has been increasing amounts of work done that uses spectrograms of music audio, which treats the spectrograms as images to feed into CNN's for music genre classification. However, as mentioned earlier, a noticeable difference between ordinary images and spectrograms is that music has a heavily sequential aspect to it. This could be a reason why existing music genre classifications with CNNs are not able to model long-term temporal information in spectrograms of music data. Recurrent Neural Networks (RNNs) have the ability to model long-term dependencies such as music structure or recurrent harmonies which are significant for music classification [13]. Choi et al. designed a hybrid model combining a CNN and RNN; CRNN [4].

[1] https://research.spotify.com/publications

Some works were done regarding music genre classification using CNNs. One of them was a work by M. Dong in "Convolutional Neural Network Achieves Human-level Accuracy in Music Genre Classification" [5]. In this project, the author used two-layer Convolutional layers for 10-genre music classification, in which the author claimed to reach above 70% accuracy, on par with human level. The author used the GTZAN dataset [6], which was different from ours, but somehow related in which there could be potential for the architecture to work in our case. The author used 64 filters for both Convolutional layers with Maxpooling between them and some regularizations. The features used were Mel-Spectrograms as well which was the same type of spectrogram that we used.

VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford. [7] VGG16 has outperformed some of the other CNN architectures in the music genre classification task. For example, to classify music genres on Audio Set [8], a VGG16 achieved 0.64 accuracy and 0.66 F1 score. [9] It is the highest accuracy and F1 score compared with other deep learning models or feature engineering models. In our best knowledge, a VGG16 architecture has never been used for music genre classification on FMA dataset. Therefore, we conducted an experiment on whether a VGG16 performs well on the FMA for genre classification. We used a pretrained VGG16 that Keras supports as a package.[2] We also implemented different hyperparameters and parameters to the VGG16 architecture; image augmentation, progressive image resizing, and freezing weights.

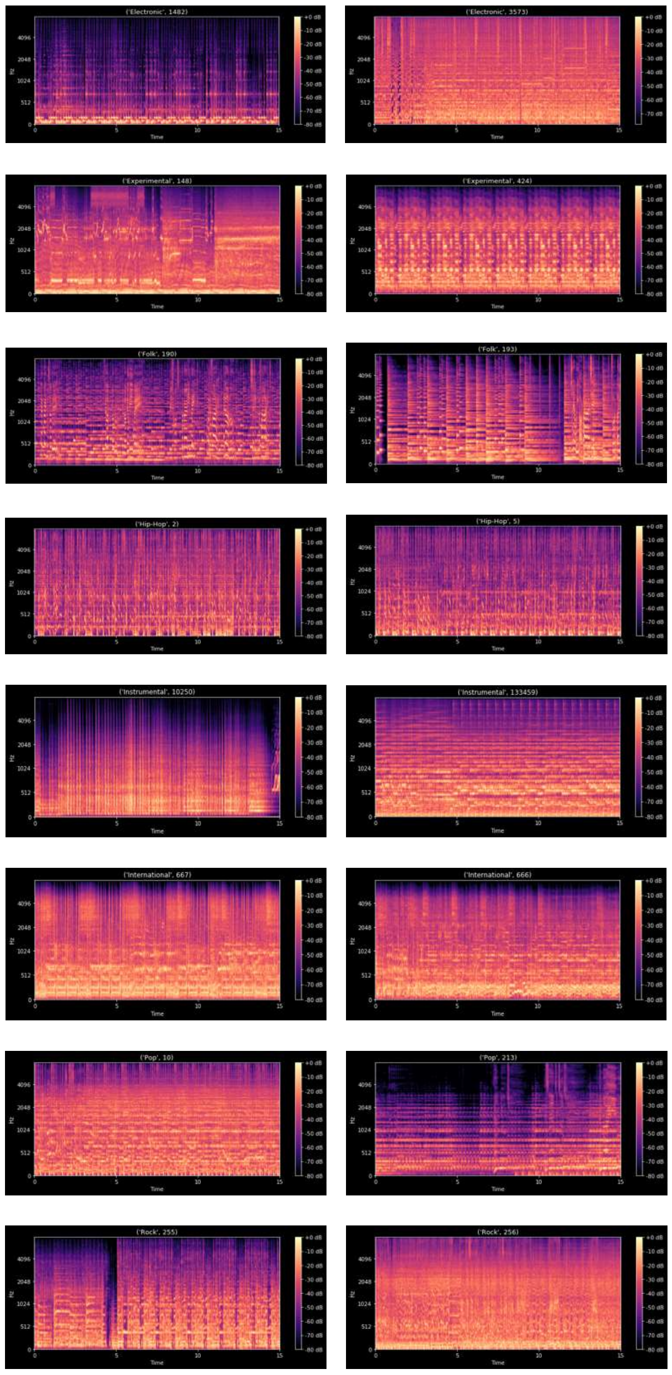
III. Dataset and Features

There are many common datasets used for content-based (ie., MIR, GTZAN, Million Song Dataset (MSD), AudioSet, etc.). We chose the FMA dataset over some of the other ones due to the dataset's essential qualities for a reference benchmark: large scale, permissive licensing, available audio, quality audio, metadata rich, easily accessible, future proof and reproducible.

The Free Music Archive (FMA) is an interactive library of high-quality, legal audio downloads which was introduced in 2017 and launched in 2019[3] with the purpose of offering free access to new music to the public [10]. We used the dataset of the UCI Machine Learning Repository[4]. The dataset is a large collection of public copyright licensed audio files. It has four sizes available:

1. fma_small.zip: 8,000 tracks of 30s, 8 balanced genres (7.2 GiB)
2. fma_medium.zip: 25,000 tracks of 30s, 16 unbalanced genres (22 GiB)
3. fma_large.zip: 106,574 tracks of 30s, 161 unbalanced genres (93 GiB)
4. fma_full.zip: 106,574 untrimmed tracks, 161 unbalanced genres (879 GiB)

For this project, we used the fma_small dataset. It consists of 8,000 audio clips that are 30 seconds each. It consists of eight top genres: electronic, experimental, folk, hip hop, instrumental, international, pop, and rock. The 8,000 tracks are balanced among the 8 genres; so 1000 tracks per genre. The dataset has two main components, the audio clips and the metadata related to each clip. The metadata includes information related to the song (title, album, artist, genre), mel frequency cepstral coefficients (MFCC), and Chroma features.



[2] https://keras.io/api/applications/vgg/#vgg16-function
[3] https://www.freemusicarchive.org/
[4] http://archive.ics.uci.edu/ml/datasets/FMA%3A+A+Dataset+For+Music+Analysis

Fig. 1. Mel-spectrogram examples: Row 1 - 8, electronic, experimental, folk, hip hop, instrumental, international, pop, and rock, respectively

IV. Mel-Spectrogram Generation

In order to feed the audio files into the different models, each audio file was converted into a mel-spectrogram. The fast Fourier transform (FFT) is a powerful tool that allows us to analyze the frequency content of a signal. Music audio signals's frequency content varies over time. To handle this, a short-time Fourier transform is performed; computing the FFT on overlapping windowed segments of the signal. This results in a spectrogram. Humans do not perceive frequencies on a linears scale. To account for this, a mathematical operation on frequencies is performed to convert them to the mel-scale; something humans are better able to understand. A mel-spectrogram is a spectrogram where the frequencies are converted to the mel scale. All audio files were converted to mel spectrograms using Librosa; a Python package for music and audio analysis. Figure 1 below shows some example mel-spectrograms for each genre.

V. Methods

We first conducted an exploratory data analysis (EDA) to explore the relationships of the included audio features. We looked into and got a general understanding of the audio features: chroma, tonnetz, mel frequency cepstral coefficient (MFCC), spectral centroid, spectral bandwidth, spectral contrast, spectral rolloff, root mean square energy, and zero-crossing rate. The audio files were converted into spectrograms, a visual representation of spectrum of frequencies over time, as features for modeling purposes. Then the mel-spectrograms were pickled to make model training and validation much faster.

For the first architecture, we used a simple Dense Network. As discussed before, the dataset included a pre-extracted feature, mel frequency cepstral coefficients (MFCC) which was one form of spectrogram when extracted fully. This included feature, however, was not in form of a full spectrogram, instead, was averaged across frequencies and shaped as a (1 x 140) vector, with the second dimension being the time dimension. With this nature of the data, it was sensible to use a vanilla dense network as a base model to compare with other architectures. The dense network architecture is shown below:

- Input layer with shape (1 x 140)
- Dense layer with 50 neurons
- Dropout with 0.5 probability
- Dense layer with 50 neurons
- Dropout with 0.5 probability
- Output layer with 8 neurons

We used L2 kernel regularizers at all layers excluding the output layer, all also used 'relu' activation function except for the output layer which used softmax. The model used Adam optimizer with 0.001 learning rate and categorical-crossentropy loss. We employed the save best model mode on Keras based on the highest validation accuracy. We did not tune much of the hyperparameters here since it was meant to be a baseline model. Most of the hyperparameters were chosen in which they caused as little overfitting as we could get to be acceptable for a baseline model.

Another architecture that we eagerly tried was a simple Convolutional Neural Network model. This architecture and all the next architectures discussed used Mel-spectrograms that were extracted and shaped as (128 x 400) vectors. The first dimension represented the frequency range while the second dimension represented the time dimension. There were some related works that mentioned how a simple CNN on a Mel-spectrogram "could achieve a human-level accuracy at genre classification" [5]. On this particular work, the architecture was applied on a different dataset. GTZAN, and claimed to achieve above 70% accuracy. Our architecture we used was based on it with several modifications such as number of filters and some regularizations, mainly to reduce overfitting. The architecture was as below:

- Conv2D with 32 filters of (3 x 3) kernels, strides of 1
- MaxPool2D with pool size of (2 x 4)
- Conv2D with 32 filters of (3 x 5) kernels, strides of 1
- MaxPool2D with pool size of (2 x 4)
- Dense layer with 16 neurons
- Dropout layer with 0.5 rate
- Output layer with 8 neurons

All layers used 'relu' activations with L2 kernel regularizers, a softmax activation at the output layer. We also used Adam optimizer with 0.0001 learning rate and the same categorical-crossentropy loss. Many of these hyperparameters were tuned based on many experiments and found to be less overfitting than others.

One of our best architectures was a retrained VGG16. As discussed before, VGG16 was a multi-layer CNN architecture that performed really well on regular images. While spectrograms did not have the same properties as regular images, we felt there might be some benefits in transfer learning, in learning some low-level features. The complete architecture of VGG16 can be seen in figure 2; we used a variant where all of these layers were pre-trained from the ImageNet dataset, which was a well-known "large-scale hierarchical image database" [11]. We froze most of the weights pre-trained but configured the last 4 convolution layers to be trainable. On top of that we added a Dense layer with 16 neurons along with a Dropout layer with rate of 0.3 and an output layer. We were hoping that the pre-trained layers might transfer the low-level features, while the trainable layers would add new feature-extractions relevant to the spectrograms. We realized this might not be the most ideal solution, given how different spectrograms were from images in the ImageNet dataset, however, the results were not bad. In addition to the architecture, we found that augmenting the dataset worked best combined with this model. Some configurations of the augmentation:

- Rotation range = 20 degrees
- Width shift range = 0.2 x width range
- Height shift range = 0.2 x height range

- Horizontal flip = True

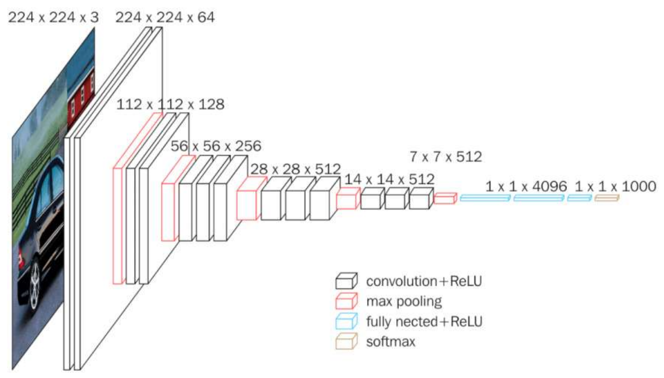


Fig. 2. VGG16 Architecture[5]

One interesting method that we also tried was a progressively-resized VGG16 (Figure 3). This method was used in an article, based on Jeremy Howard's fast.ai deep learning course that states how progressively resizing the images while using transfer learning with CNNs could boost the performance significantly [12]. The idea was that CNNs did care about scaling, so slowly training the network while scaling up helped reduce overfitting. Following the article, we scaled the spectrograms first to 42x100, with the whole VGG16 weights frozen. In other words, we only trained the last dense layers with respect to our data.

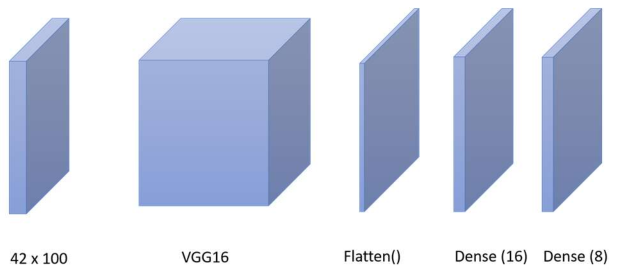


Fig. 3. VGG16 Architecture

Once we saved the best epoch on that training session, we transferred that model for further learning with doubled spectrograms size. With 84x100 spectrograms, we put two new Convolution layers as input layers for the new model with the new size by getting rid of the two first VGG16's Convolution layers (Figure 4).

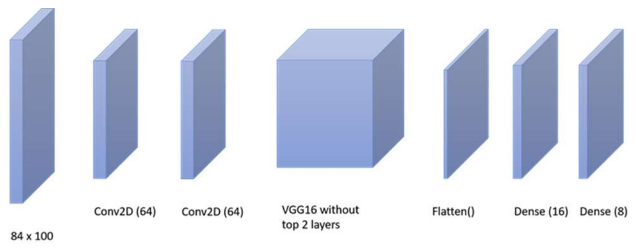


Fig. 4. Progressively-resized VGG16 second step

In this step, we trained two new low-level convolutional layers while freezing the other weights. This was done in hope that the new low-level feature extractions could reduce overfitting. As we worried before, since spectrograms did not work the same way as regular images, this method was not very promising. However, if we were able to redo this process, there were definitely some other things we could try to possibly improve it.

VI. Experiements / Results

We trained four different models with train dataset and validation dataset. The figure below (Figure 5) is one example of the result of our trained models. It shows the progress of the dense network model. As both accuracies are closed, this model was trained well although they were around 0.50 at the end.

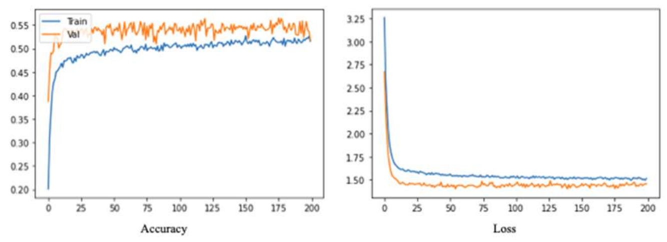


Fig. 5. Dense Network Training

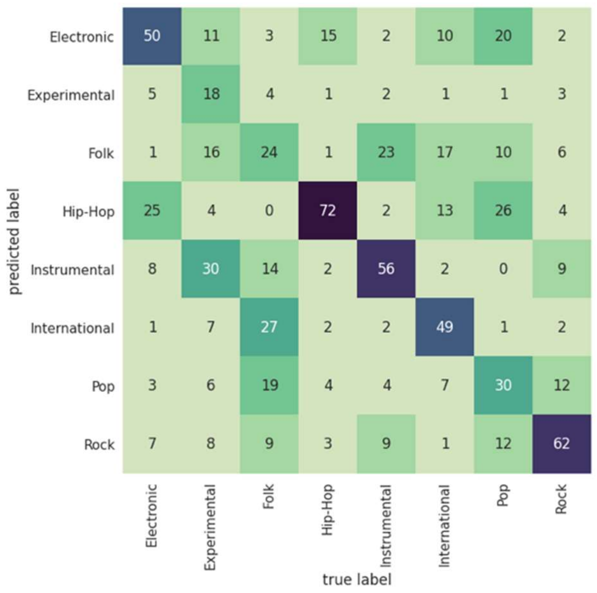
Each model was evaluated on two different datasets: validation dataset and test dataset. In this paper, we show the evaluations metrics on a test dataset because it can propose how each model can be generalized. Table 1 shows the summary of our models and comparisons with others in this field. In our best knowledge, the two models: CRNN and CNN_RNN_parallel are the current best performance models. Each performs almost similar as 0.44 accuracy and 0.44 F1 score. We also compared our models with the baseline model of the author of the FMA dataset. He did not test his model on fma_small dataset; thus, we trained and tested his baseline model on fma_small dataset. Our four models overpowered the baseline model. However, excluding our best model, ours performed less than current state-of-art models. Our best model, Dense Network, was on a par with CRNN or CNN_RNN_parallel. Indeed, the accuracy of our model was slightly high while F1 score was the same.

TABLE I. Benchmakr of Our Models on Test Dataset

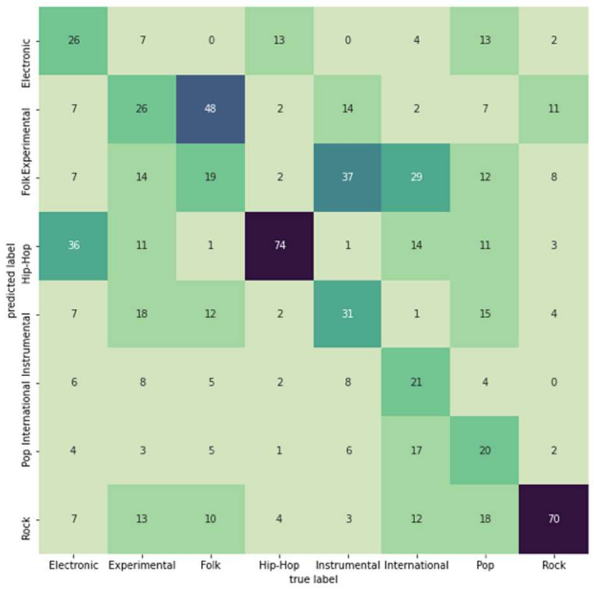
Author	Model	Accuracy	F1 Score
Priya Dwivedi	CRNN	0.44125	0.44
Priya Dwivedi	CNN_RNN_parallel	0.44375	0.44
The FMA author	Baseline	0.12	0.13
Ours	Dense Network	**0.45**	0.44
Ours	CNN with augmentation	0.36	0.34
Ours	VGG16 with augmentation	0.43	0.42

[5] Source: https://neurohive.io/en/popular-networks/vgg16/

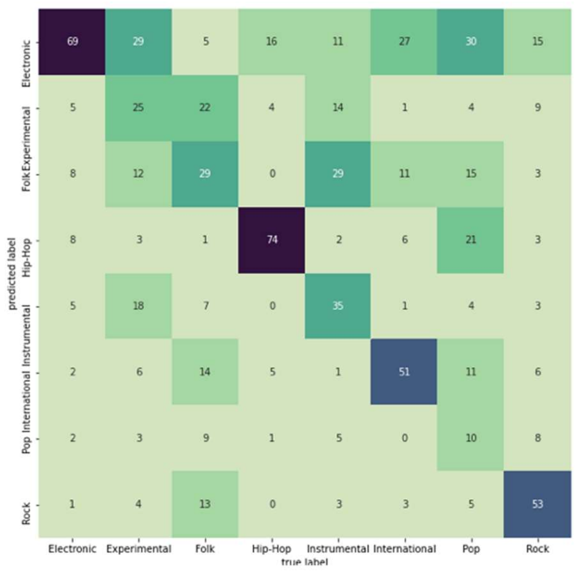
Author	Model	Accuracy	F1 Score
Ours	Progressive VGG16 with augmentation	0.33	0.29



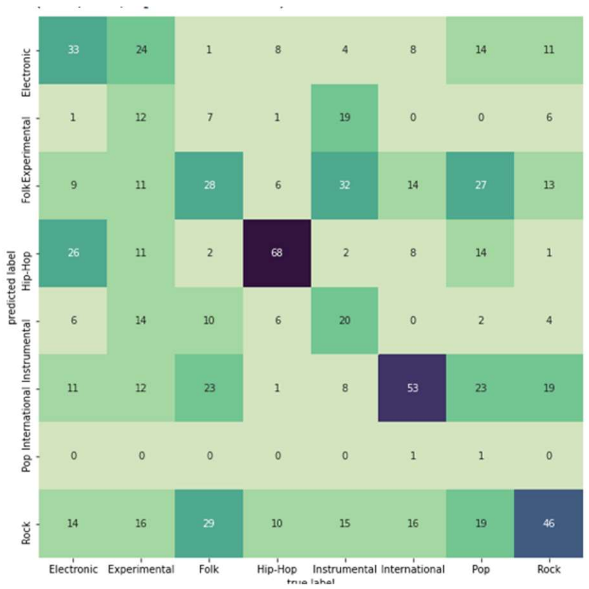
Dense Network



CNN with augmentation



VGG 16 with augmentation



Progressive-resizing VGG16 with augmentation

Fig. 6. Confusion Matrix of Each Model

As you can see in figure 6, all models were able to classify hip-hop well, which means that it is a pretty discrete genre for our models. Another interesting genre is Rock. In some models, Rock performed quite well although not as well as hip hop. One reason why the Dense network performed really well was probably mostly because of the features used. The averaged MFCC turned out to be very effective for this task, even though it was not the full spectrogram. Unfortunately, to use a CNN or other architectures we could not use these features (1 x 140).

Another possible explanation why the CNN architecture was performing not as well as the other related works presented was because mainly those architectures were developed for different datasets. FMA, known for its high variability, might require more feature extractions than simple two-layer CNN, which could be the reason why retrained VGG16 performed way better than it.

VGG16 with augmentation performed quite well. Some ways we could have done it differently, however, were to add more trainable layers, and also hyperparameter tuning. Since we did not have time to do grid search or random search, we believed a proper hyperparameter tuning could boost this model even more, possibly better than the Dense Network. Progressively-resized VGG16, however, did not perform well as expected. One particular reason for this was possibly due to the fact that resizing spectrograms were not the same as resizing images. When spectrograms were resized, there were definitely lost feature information either from the frequency range domain or time domain, which could lead to the features not distinguishable enough for the model to learn. What we could have done differently, however, was to add more trainable layers in the VGG16 in the first step, instead of freezing all of them, therefore, could add new high-level feature representations that were relevant to spectrograms. In the second step, we also would have made the last dense layers trainable, to increase model complexity.

VII. Discussion

There were some challenges and possible reasons for low accuracy. One can be computing power. We did not have enough memory to handle image data of large sizes; more than 25GB. As we upgraded to Colab Pro that allows us to use up to 35GB of memory, we could solve this problem. However, training models is still expensive. For example, when we trained a VGG16 model, we spent approximately 40 minutes for one epoch. Therefore, to reduce training time, it requires a high-performance GPU or TPU. Another possible reason for why the models have low accuracy could be insufficient sample size. Although there were 8,000 mel-spectrograms, 1,000 per genre, it is still a relatively small sample. Using a larger subset or the full FMA dataset may have improved results.

Music genres are categories that emerge from an interplay of cultures, artists, and market forces to characterize similarities between compositions and organize music collections. The boundaries between genres remain fuzzy and up for interpretation, making Music Genre Recognition (MGR) a nontrivial task. In the FMA dataset, the genres were annotated by the artists themselves. Although, they are the best ones to judge what their creations should be labeled as, sometimes it might be inconsistent and motivated by factors such as achieving a higher play count.

The FMA dataset itself may pose some challenges. There may be some confusion between the genre classes. When using the FMA dataset (and medium subset), the best performance reached was a F1 score of 0.63, indicating that there are some challenges with this dataset compared to some other popular music datasets.

VIII. Project Organization

Final scripts: extract_melspec.py, extract_mfcc.py - for extracting spectrograms and pickling them for easy import. Documentations were on the Github Readme file. Some libraries we used include: sklearn, keras (tensorflow), numpy, librosa, matplotlib, zipfile, gc, and pickle.

Key notebooks were:

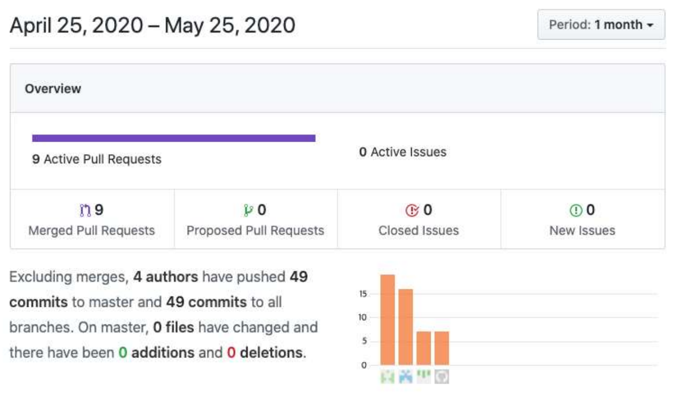
- Basic_arch_mfcc_from_features_csv.ipynb
- Mel-spec-progressive.ipynb
- Extracted_mel_spec_cnn.ipynb
- melspec_vgg (1).ipynb
- Mel-spec_AudioClips.ipynb
- Genre_Spectrograms.ipynb
- EDA.ipynb, crnn_model.ipynb

Basic_arch_mfcc_from_features_csv.ipynb contained modeling from MFCC features csv with regular Dense network. Mel-spec-progressive.ipynb contained modeling for progressively-resized VGG16 and Extracted_mel_spec_cnn.ipynb contained both simple CNN architecture modeling and Augmented VGG16 modeling. All three notebooks contained the final best models along with evaluation metrics.

IX. Teamwork

As a group, we collectively worked on the project and communicated regularly on Slack to discuss our progress, Google Colab (Pro) to execute codes, Google Drive to store datasets, GitHub for Jupyter Notebooks,[6] and a shared Google Docs for the presentations, and final report.

- Daniel Amin contributed to modeling architectures, creating scripts for spectrograms extraction and parts of the presentation and report work.
- Kiwook Kwon contributed to evaluation metrics, two presentations for our workflows, visualization of model architectures, and parts of the final presentation and report.
- Stephanie Luk contributed to the exploratory data analysis, generating playable audio clips and mel-spectrogram plots, some model architecture comparisons, the presentation, and final report.


April 25, 2020 – May 25, 2020

Period: 1 month

Overview

9 Active Pull Requests

0 Active Issues

9 Merged Pull Requests

0 Proposed Pull Requests

0 Closed Issues

0 New Issues

Excluding merges, **4 authors** have pushed **49 commits** to master and **49 commits** to all branches. On master, **0 files** have changed and there have been **0 additions** and **0 deletions**.

Fig. 7. Overview of Commits. Columns L to R: Kiwook, Stephanie, Daniel (columns 3 and 4)

[6] Gitbhub master link: https://github.com/kikwon/fma_project

kikwon
#1
23 commits 2,839 ++ 350 --
swtluk
#2
16 commits 7,622 ++ 1,717 --
daniel-amin
#3
7 commits 4,450 ++ 2,098 --
10
Apr 19
Sat 25
May
Thu 07
Wed 13
Tue 19

Fig. 8. Contributors Statistics

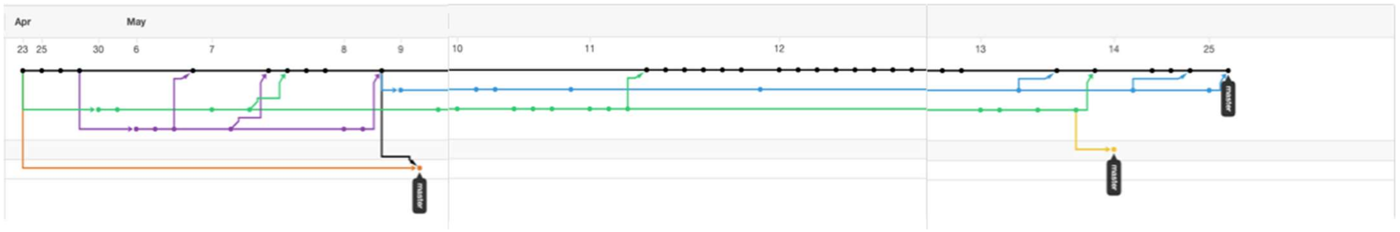


Fig. 9. Workflows. Black: Kiwook, Purple and Blue: Daniel, Green and Yellow: Stephanie

X. Conclusion

In this project, music genre classification is studied using the Free Music Archie small dataset. We proposed simpler and more straightforward approaches (CNNs) and more complex architectures (VGG16). Our experiments show that deep learning models can extract useful features from mel-spectrograms. The best model achieved 0.45 accuracy and 0.44 F1 score, which were quite similar to many other related works in this field. However, based on the model architectures we used, the deep learning models do not necessarily perform better than our baseline Dense model using MFCC features. As stated in our Discussions section, the low accuracy could be due to a small sample size and could be improved by using one of the larger subsets or the full dataset along with access to increased computing power.

References

[1] G. Samson. “Genre”. URL: https://www.oxfordmusiconline.com/grovemusic/view/10.1093/gmo/9781561592630.001.0001/omo-9781561592630-e-0000040599

[2] R. A. Altes. “Detection, estimation, and classification with spectrograms”. The Journal of the Acoustical Society of America. 1980. URL: https://asa.scitation.org/doi/10.1121/1.384165

[3] A. Khamparia, D. Gupta, N. G. Nguyen, A. Khanna, B. Pandey, and P. Tiwari. “Sound classification using convolutional neural network and tensor deep stacking network”. IEEE. 2015. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8605515

[4] K. Choi, G. Fazekas, M. Sandler, and K. Cho. “Convolutional Recurrent Neural Networks for Music Classification”. URL: https://arxiv.org/pdf/1609.04243.pdf

[5] M. Dong. “Convolutional Neural Network Achieves Human-level Accuracy in Music Genre Classification”. URL: https://arxiv.org/pdf/1802.09697.pdf

[6] Bob L. Sturm. “The GTZAN dataset: Its contents, its faults, their effects on evaluation, and its future use”. 2013. URL: https://arxiv.org/pdf/1306.1461.pdf

[7] S. Karen, and Z. Andrew. “Very deep convolutional networks for large-scale image recognition”. ICLR 2015. URL: https://arxiv.org/abs/1409.1556pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

[8] J. F. Gemmeke, D. PW Ellis, D. Freedman, A. Jansen, W. Lawrence, R C. Moore, M. Plakal, and M. Ritter. 2017. “Audio set: An ontology and human-labeled dataset for audio events”. In Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on. IEEE, pages 776–780.

[9] H. Bahuleyan. “Music Genre Classification using Machine Learning Techniques”. URL: https://arxiv.org/pdf/1804.01149.pdf

[10] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson. “FMA: a Dataset for Music Analysis”. URL: https://arxiv.org/pdf/1612.01840.pdf

[11] J. Deng, W. Dong, R. Socher, L. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, 2009.

[12] A. Bilogur. “Boost your CNN image classifier performance with progressive resizing in Keras”. URL: https://towardsdatascience.com/boost-your-cnn-image-classifier-performance-with-progressive-resizing-in-keras-a7d96da06e20

[13] J. Pons, T. Lidy, and X. Serra, “Experimenting with musically motivated convolutional neural networks,” in Content-Based Multimedia Indexing (CBMI), 2016 14th International Workshop on, pp. 1–6, IEEE, 2016