Genre Classification via Album Cover

Jonathan Li
Department of Computer Science
Stanford University
johnnyli@stanford.edu

Di Sun
Department of Computer Science
Stanford University
disund@stanford.edu

Tongxin Cai
Department of Civil and Environmental Engineering
Stanford University
joycectx@stanford.edu

Abstract

Music information retrieval techniques are being used in recommender systems and automatically categorize music. Research in music information retrieval often focuses on textual or audio features for genre classification. This paper takes a direct look at the ability of album cover artwork to predict album genre and expands upon previous work in this area. We use transfer learning with VGG-16 to improve classifier performance. Our work results in an increase in our topline metrics, area under the precision recall curve, of 80%: going from 0.315 to 0.5671, and precision and recall increasing from 0.694 and 0.253 respectively to 0.7336 and 0.3812 respectively.

1 Introduction

Genre provides information and topic classification in music. Additionally, the vast majority of music is written collaboratively and with several influences, which leads to multi-label (multi-genres) albums, i.e., a song and its album cover may belong to more than one genre. Music genre classification, firstly introduced by Tzanetakis et al [18] in 2002 as a pattern recognition task, is one of the many branches of Music Information Retrieval (MIR). From here, other tasks on musical data, such as music generation, recommendation systems, instrument recognition and so on, can be performed. Nowadays, companies use music genre classification, either to offer recommendations to their users (such as iTunes, Spotify) or simply as a product (for example Shazam). Identifying and classifying music genres is the foundation toward these tasks.

Deep Learning techniques have proved to be successful in extracting features and patterns from the large dataset. In the early time of the application of machine learning techniques, some scholars apply a Growing Neural Gas (GNG) to perform categorization and specialized neural networks for prediction.[9] Later, much work has been done on machine learning models for genre classification by lyrics, such as Naive Bayes, Support Vector Machines, XGBoost, etc[7]. More recently, audio tracks are also relied on for genre classification, taking advantage of visual representations of the spectrograms (e.g. MFCC[1]) and Convolutional Neural Networks (CNNs) [3, 12, 15] as following. However, fewer studies explore image-based classification via album cover.

CS230: Deep Learning, Spring 2019, Stanford University, CA.

The importance of the genre classification via album cover is that the visual style of the album cover art mostly reflects the overall tone of the album or perhaps portrays a visual representation of the album's topic. Certain musical genres are known to be associated with a certain album cover art style[8]. For example, metal albums tend to use darker colors with unusual fonts for the album title, and ambient albums tend to have more abstract, computer-generated designs. Although these styles are recognizable to most human observers, since the genre classification is a subjective task and high human effort required[4], we want to figure out whether machines can also determine them and even in better performance.

In this project, we are interested in building a classifier for Album Genre using Album Cover artwork. Building upon the work of a previous student in CS230, we are taking a novel approach to this problem in an attempt to improve model performance. The input to our algorithm is the 31,471 album covers from MuMu dataset and each of their many genre labels. We intend to use a VGG-16 model pre-trained on ImageNet data to conduct transfer learning via Keras instead, which, we hypothesis, will be more predictive of the output, the album genre. In this paper, we showcase the results from both the new model and the baseline model trained on music album images.

The rest of this paper is organized as follows. Section 2 outlines the existing approaches in the literature for music genre classification. Section 3 is an overview of the dataset used in this study as well as some basic statistics and their implications. The proposed models and the implementation details are discussed in Section 4. The results are reported and analyzed in Section 5, followed by the conclusion and future work in Section 6.

2 Related work

Some works published in recent decade show that investigations related to musical genre in MIR community are not exhausted, and they still remain as an active research topic. Alexnet, ResNet pre-trained on Imagenet, and other CNNs [7, 5, 8] attempt image-based music genre classification but the overall results are poor. Some scientists use multi-modal approaches, like Joint equal contribution (JEC), to seek for higher accuracy.

Oramas et al[10], in the visual representation part of their study, chose ResNet-101[5] with pre-trained weights learned in The ImageNet Large Scale Visual Recognition Challenge (ILSVRC)[14], and fine-tuned on the music genre labels from the approximately 30k cover art derived from the MuMu (Multimodal Music) dataset. Their results indicate that the model achieved a 0.7343 auc score, which-while lower than the other modalities like audial and textual features in isolation- still demonstrates the possibility for model to learn to differentiate genres from cover art. Lee et al [8] implement several neural networks, both Alexnet completely trained on their dataset (Alexnet) and pre-trained on ImageNet data and implemented via transfer learning (Alexnet, Resnet18, Resnet34, and Resnet152). They concluded that the too general labels with many diverse artists and single-label data yield poor performance of their models. Besides using CNNs, Passant [11] identify music genres from Spotify album covers using Clarifai's deep learning API. He first tag artists through album covers and then bridge the gap between artist-tags and genre-tags by using "tops-n songs" playlists from Spotify. With what he found, he sees the possibility for more exploration of album artwork in regards to genre via deep learning.

Multi-label classification has attracted attention in the context of MIR. Auto-tagging has been studies from a multi-label perspective using traditional ML approaches[20, 16, 17] as well as deep learning approaches[2, 13]. However, there are few people focusing on multi-label genres classification.

3 Dataset and Features

Our dataset is the MuMu: Multimodal Music Dataset [10]. This dataset has multi-label genre annotations that combine information from the Amazon Reviews dataset and the Million Song Dataset (MSD). The Amazon Review dataset comprises of millions of album customer reviews and album metadata gathered from Amazon.com. The Million Song Dataset is a collection of metadata and precomputed audio features for a million songs. The MuMu dataset has 147,295 songs, across 31,471 albums. For these albums, there are 447,583 customer reviews from the Amazon Dataset. We are concerned with the 31,471 albums and each of their many genre labels.

We first filter our dataset to include only albums with the top 20 major genres,"Pop, Rock, World Music, Dance & Electronic, Jazz, etc.", each of which represents at least 1% of the whole dataset. After doing this, we find our dataset has 17,689 albums remaining. The label distribution is shown in Figure 1. Then, we split the dataset into an 80/10/10 train, dev, test split. Finally we resize the images to 224x224 resolution.

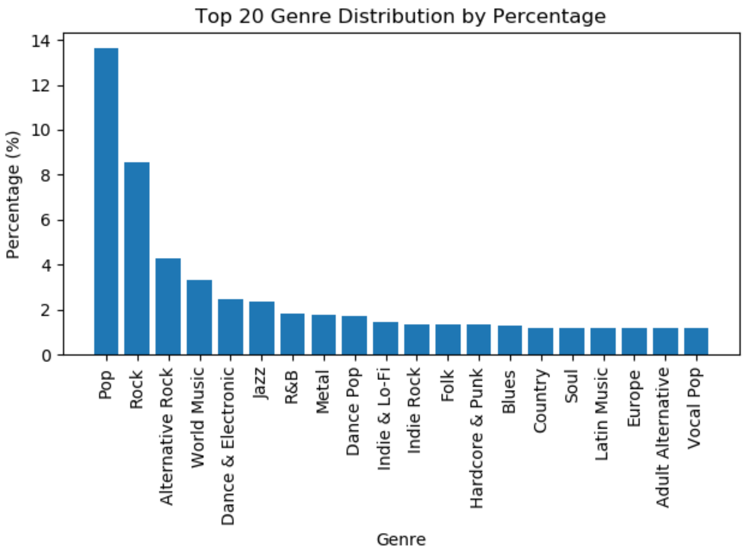


Figure 1: Distribution of Top 20 Genres by Percentage Before Filtering.
00

4 Methods

4.1 Transfer Learning

We transitioned to a transfer learning approach with the hopes that it would help make up for the fact that our dataset only includes 14000 images to train on. We decided to use VGG16 with weights trained from imagenet preloaded into the convolutional blocks, randomly initialized fcc layers, and the last layer resized to be of size 20 (the number of genres we're observing).

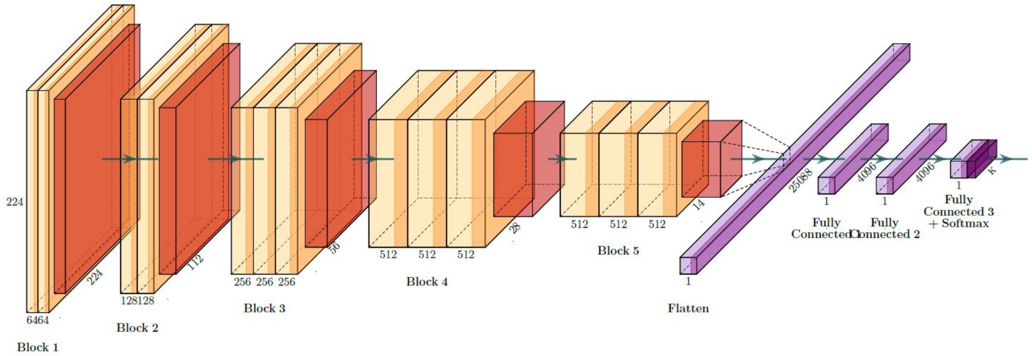


Figure 2: The architecture of the VGG16 convolutional neural network[19]

We utilize Binary Cross Entropy Loss to predict class probabilities, defined as follows:

$$\mathfrak{L} = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i)) \tag{1}$$

5 Results and Discussion

5.1 Baseline

Our baseline model is adapted from a past student project [6] which itself was adopted from a project by Oramas et. al. [10]. We keep the same model architecture and train it for 20 epochs on our filtered dataset. This model is designed as a large CNN with 6 convolutional blocks of 2D convolution, batch normalization, ReLU Activation, and max pooling. In addition to the convolutional layers, there are 3 fully connected layers with batch normalization, ReLU, and dropout following. The last layer is an output layer with one node for each of the genre labels in our dataset. We use sigmoid cross entropy loss to predict class probabilities independently. After running our baseline, we find that on our test set the area under the prc curve is 0.383 and area under the roc curve is 0.916. Precision and Recall with thresholds of 0.5 are 0.694 and 0.253 respectively.

5.2 Transfer Learning

The metrics we observed in our experimentation are as follows: AUC of the ROC curve, AUC of the PR curve, precision, and recall. Precision and Recall are measured with a threshold of 0.5.

We use Adam as an optimizer with Binary Cross Entropy Loss. Input image shape is (224,224,3). All hyperparameters are the Keras defaults. Our initial weights are loaded in using the imagenet argument in the VGG16 Keras class. We have frozen the first 12 layers of the model (the first 3 convolutional blocks). Finally, we train for 10 epochs. Our training results can be seen in Figure 3 above. Performance on the test set was measured using area under the ROC curve, which was 0.7844, and area under the Precision recall curve, which was 0.5671. Precision and Recall with thresholds of 0.5 were 0.7336 and 0.3812 respectively.

5.3 Analysis

From this we can see that our new model shows a significant improvement in both precision and recall. Higher precision means the model tends to be more selective when categorizing an album's genre, but this is most likely acceptable since for this multi-label problem, we care more that whatever genre is picked is correct.

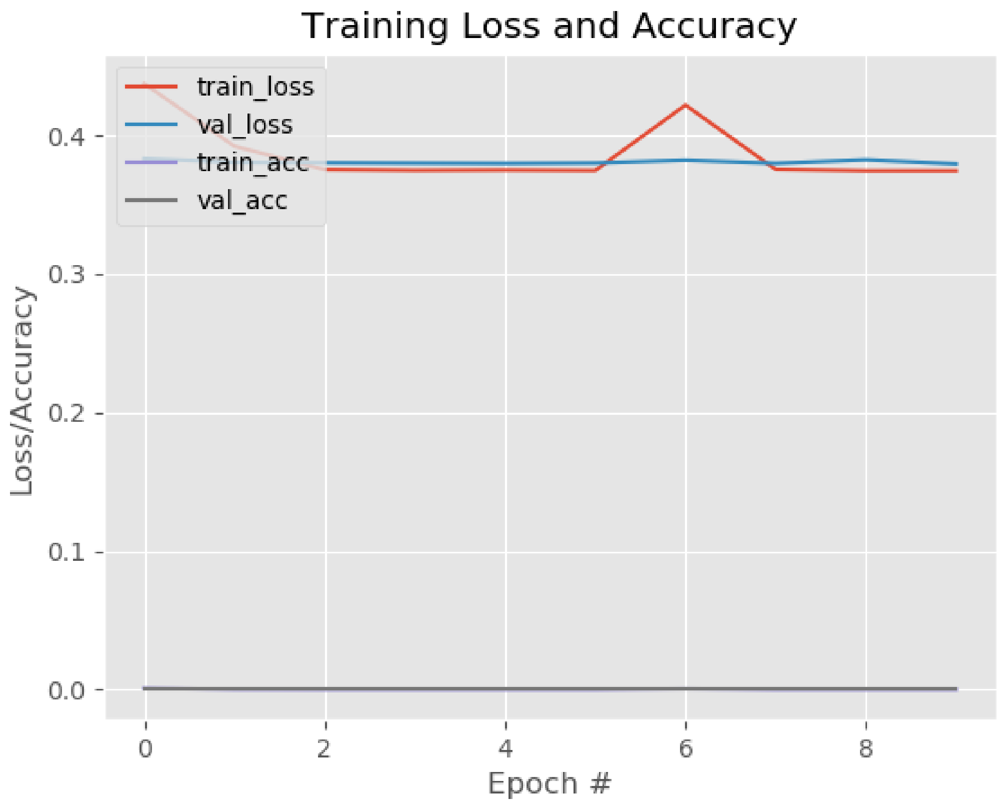


Figure 3: Training Loss

6 Conclusion and Future Work

Ultimately our work results in an increase in our topline metric, area under the precision recall curve, of 80%: going from 0.315 with our baseline to 0.5671. Precision and Recall in our baseline with thresholds of 0.5 were 0.694 and 0.253 respectively. With the transfer learning model, we are able to increase these to 0.7336 and 0.3812 respectively.

Future work can continue to improve on the performance of this project by further experimentation with model architecture. The current dataset's images are mostly from older and more obscure artists and albums. Higher quality data in the form of more modern and more popular and well designed album covers can be collected for the model to train on.

7 Contributions

Jonathan Li helped write reports, build and train models, and collect data. Tongxin Cai conducted literature research, helped with report writing and video report making. Di Sun helped research, pre-processing data, modify and run models.

References

[1] Hareesh Bahuleyan. Music genre classification using machine learning techniques. *arXiv preprint arXiv:1804.01149*, 2018.

[2] Keunwoo Choi, George Fazekas, and Mark Sandler. Automatic tagging using deep convolutional neural networks. *arXiv preprint arXiv:1606.00298*, 2016.

[3] Sander Dieleman, Philémon Brakel, and Benjamin Schrauwen. Audio-based music classification with a pretrained convolutional network. In *12th International Society for Music Information Retrieval Conference (ISMIR-2011)*, pages 669–674. University of Miami, 2011.

[4] Robert O Gjerdingen and David Perrott. Scanning the dial: The rapid recognition of music genres. *Journal of New Music Research*, 37(2):93–100, 2008.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] Christian Koenig. Classifying album genres by artwork. 2017.

[7] Akshi Kumar, Arjun Rajpal, and Dushyant Rathore. Genre classification using feature extraction and deep learning techniques. pages 175–180, 2018.

[8] Nathan Lee and Robert Baraldi. Cse 546 final paper predicting musical genre from album cover art.

[9] Rachel Lee, Ryan Walker, Lisa Meeden, and James Marshall. Category-based intrinsic motivation. In *Proceedings of the ninth international conference on epigenetic robotics*, volume 146, pages 81–88, 2009.

[10] Sergio Oramas, Oriol Nieto, Francesco Barbieri, and Xavier Serra. Multi-label music genre classification from audio, text, and images using deep features. *CoRR*, abs/1707.04916, 2017.

[11] Alexandre Passant. Identifying music genres form spotify album covers using clarifai's deep learning api and google prediction. `https://medium.com/@apassant/identifying-music-genres-form-spotify-album-covers-using-clarifai-s-deep-learning-api-and-google-61d40799fb87`, 2015.

[12] Jordi Pons, Thomas Lidy, and Xavier Serra. Experimenting with musically motivated convolutional neural networks. In *2016 14th international workshop on content-based multimedia indexing (CBMI)*, pages 1–6. IEEE, 2016.

[13] Jordi Pons, Oriol Nieto, Matthew Prockup, Erik Schmidt, Andreas Ehmann, and Xavier Serra. End-to-end learning for music audio tagging at scale. *arXiv preprint arXiv:1711.02520*, 2017.

[14] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[15] Christine Senac, Thomas Pellegrini, Florian Mouret, and Julien Pinquier. Music feature maps with convolutional neural networks for music genre classification. In *Proceedings of the 15th International Workshop on Content-Based Multimedia Indexing*, pages 1–5, 2017.

[16] Klaus Seyerlehner, Gerhard Widmer, Markus Schedl, and Peter Knees. Automatic music tag classification based on block-level. *Proceedings of Sound and Music Computing 2010*, 2010.

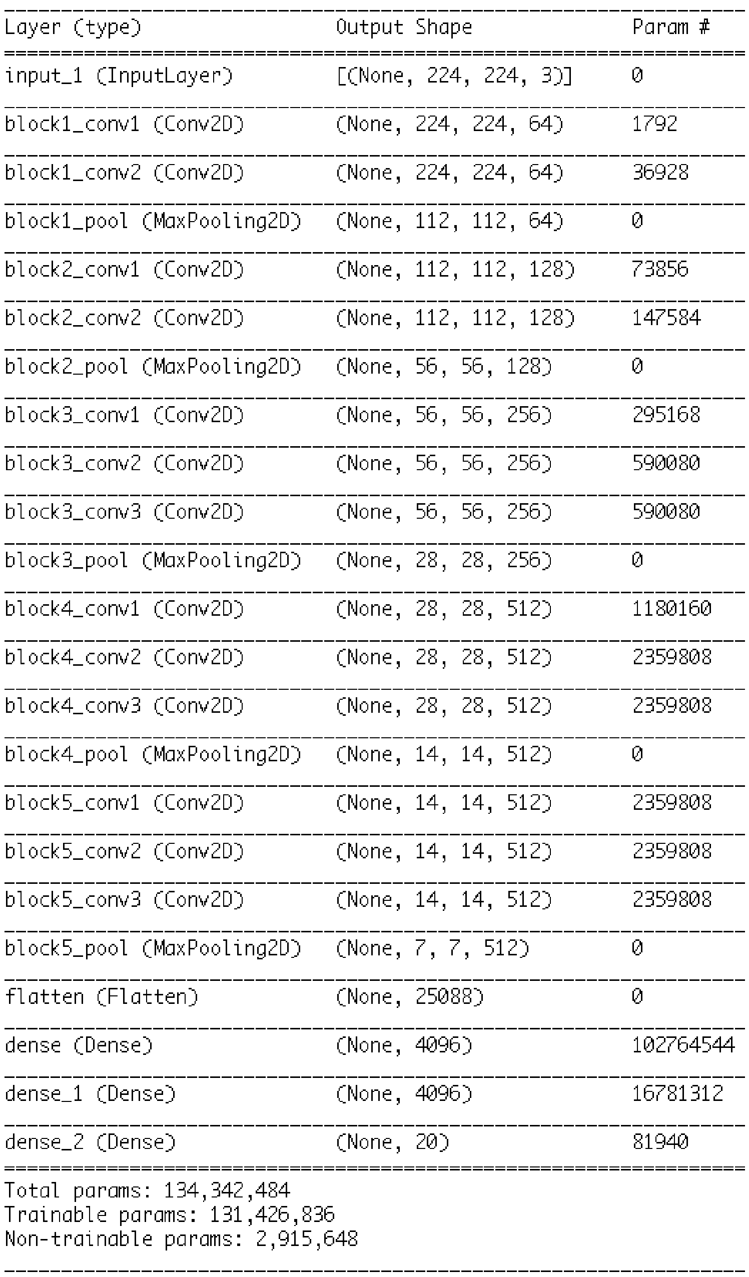
[17] Mohamed Sordo et al. *Semantic annotation of music collections: A computational approach*. PhD thesis, Universitat Pompeu Fabra, 2012.

[18] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on speech and audio processing*, 10(5):293–302, 2002.

[19] Grega Vrbancic, Milan Zorman, and Vili Podgorelec. Transfer learning tuning utilizing grey wolf optimizer for identification of brain hemorrhage from head ct images. In *Computer Science Research Conference*, volume 57.

[20] Fei Wang, Xin Wang, Bo Shao, Tao Li, and Mitsunori Ogihara. Tag integrated multi-label music style classification with hypergraph. In *ISMIR*, pages 363–368, 2009.

8 Appendix



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 224, 224, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 112, 112, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 56, 56, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 56, 56, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 56, 56, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 28, 28, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 28, 28, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 28, 28, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 14, 14, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 14, 14, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
dense (Dense)                (None, 4096)              102764544
_________________________________________________________________
dense_1 (Dense)              (None, 4096)              16781312
_________________________________________________________________
dense_2 (Dense)              (None, 20)                81940
=================================================================
Total params: 134,342,484
Trainable params: 131,426,836
Non-trainable params: 2,915,648
_________________________________________________________________
```

Figure 4: Model Architecture with three frozen Conv blocks