Using Spectrograms and Convolutional Neural Networks for Music Genre Classification

Gene Tanaka
Department of Computer Science
Stanford University
gtanaka@stanford.edu

Jaime Hurtado-Lopez
Department of Computer Science
Stanford University
jaimeh37@stanford.edu

Abstract

Music streaming services such as Spotify, Apple Music, and Pandora are providing more and more personalized recommendations drawn from user data. In order to match music, engineers need to be able to accurately identify music features. Our project using a dataset of MP3 files to train a convolutional neural network to classify the genre of a song.

1 Introduction

Modern academics argue that grouping music into genres is inaccurate and outdated — categorizations are mostly based on arbitrary conventions. We investigated how accurately convolution networks can identify these "patterns" in music of shared genres. We implemented a CNN model that took in MEL spectograms in the form of a PNG image as input along with correlated one-hot encoded vectors of genres from a CSV file. The model outputs a classification on the expected genre for a given spectogram.

2 Related work

In reviewing the literature related to our project, we found three major papers written on music genre classification. One, written by Keunwoo Choi, was titled *Transfer learning for music classification and regression tasks.* This paper focused on transfer learning, but provided good intuition for how we should edit the architecture that we had built. They used ELU activation functions for their convolutional layers and recommended addding maxpool layers after each convolution layer. They also said that using MEL spectograms were the superior choice in music classification. They had stateof-the-art results by implementing transfer learning. This is an area in which we would have looked further into given more time. Another paper we reviewed was *Urban Sound Classification using Convolutional Neural Networks with Keras: Theory and Implementation* written by Adrian Yijie Xu. This article describes the implementation of a CNN for music genre classification. We took most of our inspiration from this article in terms of our model architecture. They reported excellent results of about 94 percent. In terms of architectures, libraries, and models, we used foundations from librosa, Kapil Varshney, and the paper *Convolutional Neural Networks for Sentence Classification* by Yoon Kim. Librosa is a python library for extracting music features and spectograms from audio files. Kapil Varshney provided a useful tool for printing keras model outputs. The paper by Yoon Kim inspired our intuition for more research on using CNNs for music genre classification since they used them for similar a audio-related project.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

3 Dataset and Features

We found our data on the git hub repository called Free Music Archive. This git-hub page offered various sizes of data-sets of MP3 tracks. It also came with CSV files that contained meta-data such as ID, genres, play counts, etc. This is the data-set that we had most trouble with. Many of the values in the 'genres' column of the file were inconsistent. For example, some included arrays with multiple elements, some included empty arrays, some included empty cells, and some included strings. We manually went into the file to clean it up and make sure ever value was an array with a single element. We used the raw features of png image files. These png image files were of spectograms developed for each MP3 file of our data-set. The number of images for the training/dev set varied between two models. The first time around, it was 1194/398 respectively. We later used more data: 1817/552. The images are appropriate for this task because they are perfect for convolutions extracting a lot of features from each 3 RGB channels.

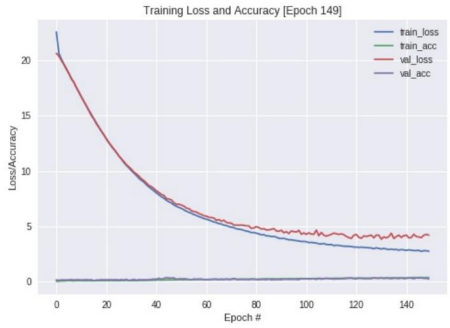
4 Methods

The first model we used consisted of six CNN layers, max pool layers in each, two fully connected layers, and dropout layers in between them. The first model we used had filter sizes of 32, 64, 64, 64, 128, 128. We later increased these to 96, 96, 128, 128, 256, 348. The convolutions layers learn to identify features in the images of the spectograms. This is done by the convolution operation by rearranging information from the image file into different sizes and blocks. This allows the model to use less parameters while at the same time learn different patterns/aspects in an image file. The image file, in our case was the MEL spectogram which shows audio frequency through time. Initially, we did not add any regularization or batch normalization to any inputs. After testing this model, saw very poor results with significant over-fitting. We made several attempts in resolving this. First,we added additional dropouts, after layers. Second, we used L2 regularization on the first convolution layer and the fully connected layers' inputs. Finally, we added batch normalization to each CNN layer. We also noticed that the CSV data had inconsistent values for the genres. Certain values were empty, had an empty array, an array of more than one genre, and some had string values. We thought this would cause issues so we manually cleaned the data so that the genre would only contain arrays with a single element. At the end, we still struggled with over-fitting. We tried switching to an Adam optimizer and increasing the learning rate, but that showed little improvement.

5 Experiments/Results/Discussion

We struggled, mostly with over-fitting. The models without l2 regularization on our fully connected layers were drastically worse than the rest. We tested models with data manipulations. Those with only had 1194 images to train and 398 images to test. We also compared different architectures. Those with less filters actually performed the best. It had 15 percent better accuracy than the next best model.

Figure 1: Best Model, Using Big Dataset, Adam, L2Regularization, and Small Filters



The hyper-parameters we tuned were learning rate drop-orate. We reduced the learning-rate by a factor of two. This resulted in an increase in accuracy by 3. We tested the drop-out rate more because

our biggest problem at first was over-fitting. We added dropout to our CNN layers. At first we used 0.25, but then increased it to 0.5. This resulted in less initial over-fitting.

Figure 2: CNN Model, Adam, Small Filters, Pre-Regularization

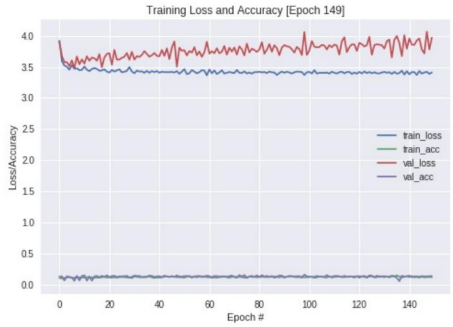
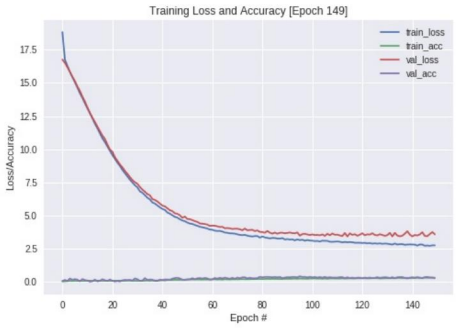


Figure 3: CNN Model, Using Big Dataset, Adam, L2Regularization, and Big Filters



Furthermore, we focused on tuning the architecture, in terms of: the number of fully-connected layers, filter size in CNN layers, and regularization techniques. It turned out that the large filter sizes 96, 96, 128, 256, and 348 resulted in an accuracy 15 percent lower than our architecture. This was the most surprising result to us, since we had reason to believe, from the AlexNet architecture, that bigger filters would be able to detect more features.

Figure 4: Dev-set Loss and Accuracy for Multiple Trained Models

Learning rate = 0.005, unless specified otherwise	Loss	Accuracy
Pre-Data Processing, Large Filters, L2 Reg, Dropout, RLS	8.666	4.5%
Pre-Regularization, Big Filters,Adam	3.838	0.1335
Regularization, Big Filter, Adam	6.754	19.5%
Regularization, Adam, Smaller Filters, learning rate = 0.0001	3.491	34.4%
Big Data	3.505	32.6%

6 Conclusion/Future Work

To add on to our biggest surprise, since the spectogram images are so closely related in appearance, we had strong reason to believe the algorithm would have benefited from detecting more features to be better at distinguishing different spectograms and better classifying genres. Perhaps there was a deeper underlying issue that prevented this from being the result. With more time, we would have looked for cleaner data, researched more literature on multi-genre classification, and trained models on much larger data-sets. Genre classification is to become increasingly more important for streaming services such as Spotify and Apple Music, as they use audio tags and genres for play-list

generations and recommendations for their users. Deep learning models can analyze the fundamental features to an audio-file instead of relying on subjective analysis that has seen controversy over recent norm-breaking songs such as Old Country Road.

7 Contributions

Gene Tanaka: Data manipulation, initial architecture implementation, poster design Jaime Hurtado: Architecture manipulation/research, hyper-parameter tuning, data scraping

References

Tensorflow Keras Matplot

[1]Choi, et al. "Transfer Learning for Music Classification and Regression Tasks." ArXiv.org, 13 Sept. 2017, arxiv.org/abs/1703.09179.

Krizhevskyet al., 2012. ImageNet classification with deep convolutional neural networks

[3] Brian McFee, Matt McVicar, Stefan Balke, Vincent Lostanlen, Carl Thomé, Colin Raffel, ... CJ Carr. (2019, February 13). librosa/librosa: 0.6.3 (Version 0.6.3). Zenodo. http://doi.org/10.5281/zenodo.2564164

[4] Xu, Adrian Yijie, and Adrian Yijie Xu. "Urban Sound Classification Using Convolutional Neural Networks with Keras: Theory and..." Medium, GradientCrescent, 27 Feb. 2019, medium.com/gradientcrescent/urban-sound-classification-using-convolutional-neural-networks-with-keras-theory-and-486e92785df4.

[5] Varshney, Kapil. "How to Plot the Model Training in Keras - Using Custom Callback Function and Using TensorBoard." Medium, Medium, 6 Aug. 2018, medium.com/@kapilvarshney/how-to-plot-the-model-training-in-keras-using-custom-callback-function-and-using-tensorboard-41e4ce3cb401.

[6] Kim, and Yoon. "Convolutional Neural Networks for Sentence Classification." ArXiv.org, 3 Sept. 2014, arxiv.org/abs/1408.5882.

[7] Keras-Team. "Keras-Team/Keras." GitHub, 7 June 2019, github.com/keras-team/keras.