Machine Learning Final Project: Using a Convolutional Neural Networks to Classify Musical Genres

Alexander Cox, Ava Sato, Ernst-Richard Kausche

Abstract: In this paper we will take a look at two different architectures of convolutional neural networks, namely ZFNet and GTZANet, and their effectiveness at classifying music by genre. We implement these models in tensorflow and examine their performance on the GTZAN dataset, in order to gain insight into the strengths of each one. We conclude from these experiments that while GTZANet was able to marginally outperform ZFNet, though both models display large amounts of room for future improvement.

1. Introduction

Deep Learning in recent years has shown itself to be an incredibly powerful tool, capable of solving many problems that humans are normally tasked with. In this report we will analyze the ability of machine learning architectures to determine genres of music. This problem has a wide range of applications as it could be used to simply categorize music for distributions, or even to make predictions about what a certain user might be interested in musically. We find this problem to be interesting because we are all deeply passionate about music and how it is shared via the internet and we, like many others of our generation, are greatly affected by technological advancements. Particularly we are interested in the future of machine learning as applied to the music industry. For this project we've made use of two different architectures of Convolutional Neural Networks. The first of these is the popular ZFNet architecture which won the ImageNet competition in 2013, where it achieved an error rate of just 14.8% for classifying images. The

second model we've implemented is a custom architecture that we developed as a more simplified alternative directed at our specific problem. To compare their respective performances, we used them to categorize our dataset of 1000 different songs each belonging to one of 10 different genres. Our results have shown that while the model that our team developed was able to significantly outperform the ZFNet architecture, both models have room for improvement.

2. Background and Related Works

Musical genre refers to a style or composition that different groups of songs share. Music is easily categorized into genres by humans, but the massive amount of music on streaming platforms such as Spotify and SoundCloud makes this task better suited for computers. Genre classification has already been tackled using convolutional neural networks, so we turned to existing literature surrounding the implementation of this model to inform our project design. Zhang et. al describes their method of genre classification using convolutional neural networks. They use spectrograms, a visual representation of a song's timbre, as input for their neural network. The architecture of their neural network is able to reduce the total number of layers by using shortcut connections, something that we did not build upon in our model but might be interesting for future attempts. This ultimately increased the accuracy of their model as well as the speed.

Xin et. al examines music classification from a different perspective, this being emotion classification: determining whether a song would make the typical user happy, sad, etc. Xin et. al's study encounters the important issue of feature extraction and the challenges of attempting this when it comes to music classification

Feng et. al further improves upon feature extraction by paralleling CNN and Bi-RNN blocks

3. The Problem

As we are relatively new to machine learning, our ultimate goal was to create a convolutional network for learning's sake. Our benchmark for success was achieving fifty percent accuracy in genre classification. Using convolutional networks for genre classification is extremely relevant to the current technological climate. The amount of digitally stored music is larger than ever and the primary source of digital music is subscription-based streaming services rather than pre-purchased music libraries. On streaming sites, algorithms are used to aid users in finding new music based upon their musical preferences, making it necessary to be able to distinguish between genres on a massive scale. This is the problem we set out to address with our project.

We tackled this problem by designing a convolutional neural network that could take in a spectrogram representing thirty seconds of a song and identify its genre. In doing this we hoped to create a neural network architecture that could in turn be applied other problems in the realms of song and sound classification and be applied to the real-world issue identified previously. As a primary goal of this project was learning (not the machine kind), we felt genre classification to be a feasible milestone on our journey to understand more challenging applications of convolutional neural nets, such as identifying the name of a song given a .wav file or generating new songs with existing songs as a database.

Our task was, at first glance, simple: use TensorFlow libraries and information gathered from previous research on this topic to design and implement a convolutional neural network. With the infrastructure in place all that remained was to train the model on music. We found the GTZAN dataset, named after its creator George Tzanetakis. It consists of 30 second sections of 1000

songs grouped into ten different genres. Each song was stored as a spectrogram, which, as mentioned earlier, is a visual representation of the timbres of a song. In a spectrogram the x-axis represents time and the y-axis represents frequency. The spectrogram can in turn be represented by a matrix with a height of 288, a width of 423, and a depth of 3. The height and width of the matrix account for the number of pixels in the spectrogram, and the depth represents the color value of each pixel. These matrices representing songs are easily fed into a convolutional neural network, which can identify similarities and differences between songs of different genres and use this information to classify new input.

Number of attributes	Number of possible labels	Proportion of each label in data set
373248 (where each attribute is a distinct color value for a distinct pixel)	10 (one for each genre)	10% (there are 100 spectrograms for each genre and 1000 total spectrograms)

Figure 1: Characteristics of Data Set

4. Solution

In order to solve this problem, our group has employed the use of two different architectures of CNNs, namely ZFNet, and our own custom architecture which we will refer to as GTZANet for the remainder of this paper. The ZFNet architecture was designed to recognize specific features of an image in order to categorize it into one of many different image classes. The architecture makes use of five layers of convolutions, where the size of the kernel becomes increasingly small as it narrows in on smaller, more dense image regions. After the convolutional layers, the architecture then flattens the resulting output in order to send it through two “dense” layers, which correspond to the hidden layers of a neural network. Finally the

outputs of these neurons are passed through to a softmax output layer, where each neuron's activation represents the certainty of the corresponding label. This model uses its earlier convolutional layers with larger kernels in order to first identify larger more elementary features of the image, such as a line, before progressively shrinking the kernel size on later convolutions in order to narrow in on more complex features such as part of a face.

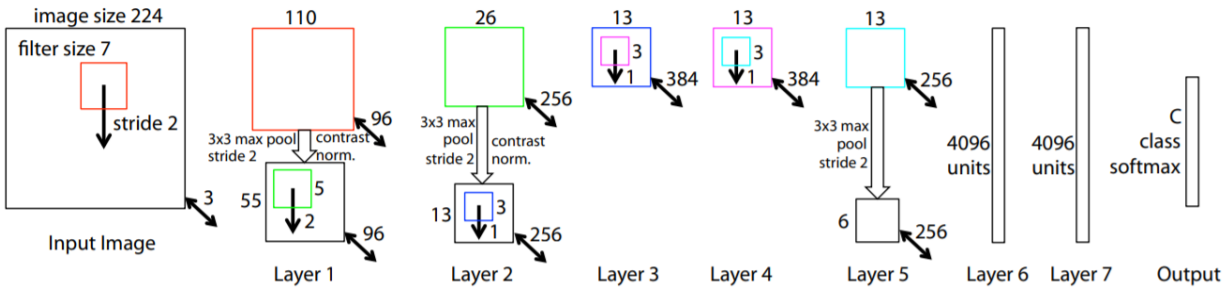

Figure 2: ZFNet Architecture

The GTZANet architecture follows a similar series of sequential layers, but with a few differences focused on optimizing the model for our specific problem. In particular rather than using a larger kernel earlier on, GTZANet only uses 3-by-3 convolutions with a stride length of one for each layer, in order to zero in on more specific aspects of the spectrogram input, as there can be a lot of information packed into small regions of spectrogram diagrams. Additionally this architecture makes use of a max pooling layer after every convolutional layer in order to compress the data output of each convolution. Finally, our model only makes use of only one dense layer, which is the output layer, as the extra hidden layers with 4096 neurons each in ZFNet were very computationally expensive. Between these two key differences the GTZANet model is able to cut the training time roughly in half when compared to ZFNet, and as a result we were able to repeatedly test it in order to optimize some of its hyper parameters.

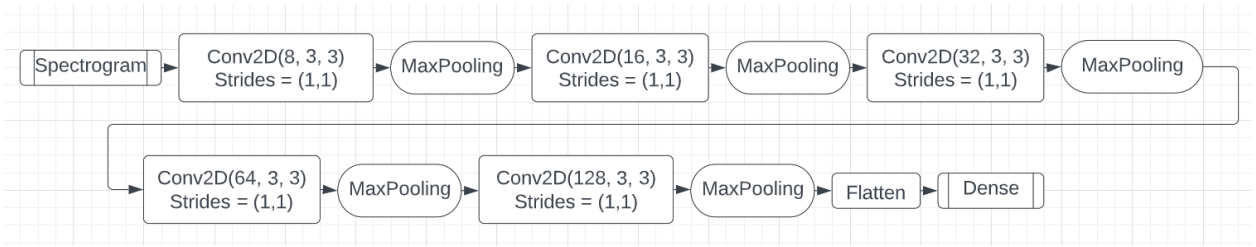

Figure 3: GTZANet Architecture

5. Experiments and Results

In order to test the effectiveness of each of these models, we split our dataset into a training set and a validation set with 20% of the data reserved for validation. Each of the models was implemented in tensorflow using the Sequential class. The models were then each compiled using the SGD optimizer class, which uses gradient descent with momentum, with an initial momentum and learning rate of 0.01 and 0.9 respectively. As a loss function we used sparse categorical cross entropy, which is similar to the categorical cross entropy loss function often used for training CNNs except that it makes use of enumerated classes rather than a one-hot encoded system. Additionally the model was compiled to use both accuracy and top 5 categorical accuracy, which is determined by how often the correct label was in the top 5 class certainties in the output layer, as metrics to represent the models progress over the course of training. Each model was then fit to the dataset using the validation set to track its accuracy over 90 epochs. Additionally, we used the ReduceLROnPlateau callback for training which would reduce our learning rate by factor of 0.1, to a minimum of 0.00001 if our the sparse categorical cross entropy of our validation set stagnated.

In running each of our architectures on the GTZAN dataset over 90 epochs, we observed the following accuracies and top 5 categorical accuracies on the validation set.

	Accuracy	Top 5 Categorical Accuracy
ZFNet	0.4472	0.6533
GTZANet	0.5276	0.6231

Figure 4: Experimental Results

From these results we can see that the GTZANet model was able to achieve a higher accuracy on the validation set than ZFNet. However the confidence intervals for these accuracies, calculated using a z-score of 1.96, were [0.378, 0.516] for ZFNet and [0.458, 0.567], showing a significant overlap between the two intervals. The overlap of these two confidence intervals was likely due to the fact that these two architectures are very similar, with GTZANet being based on ZFNet, and therefore being a child architecture in a way. This result displays that while GTZANet was able to achieve marginally better results, it did not outperform ZFNet in a way that was significant enough for us to conclude that GTZANet is the better model for the task.

The accuracy of the NFNet architecture was also surprisingly low here at only around 45% whereas it achieved roughly 85% accuracy in the ImageNet competition. This was likely due to the fact that the images used in our dataset were not only significantly larger than the ones used in the ImageNet competition but also contained a very different kind of data that features very densely packed regions of information. This points to the fact that if we were to reduce the length of the song excerpts used for the dataset we could potentially see a significant increase in the performance of these models.

Additionally we can see in our results that while the the accuracies were 0.45 and 0.53 for ZFNet and GTZANet respectively, the top 5 categorical accuracy was just 0.65 and 0.62 respectively. In particular we see that GTZANet only had the correct label in the top half of its

predictions around 10% more often than when it correctly predicted the proper label. While it's difficult to speculate why this may be, we believe that this may have been caused by the model overfitting the training data to the problem, thus leading it astray in cases where it didn't have the correct prediction. These results demonstrate that when the model made the incorrect prediction, it was usually very wrong with its certainty for the actual label not even showing up in the top half of certainties roughly 80% of the time it was wrong.

6. Conclusions & Future Work

In this paper we analyzed the effectiveness of two different architectures of CNNs when used for genre classification on spectrograms of music. We observed that the GTZANet architecture performed marginally better, likely due to its reduced kernel sizes and smaller stride length resulting in a greater ability to zero in on smaller regions of densely packed spectrogram data. For future experiments we are interested in seeing how the data may be processed differently in order to achieve better results. One way of doing this would be to split the song files into smaller chunks, perhaps 3 seconds rather than 30, and training our model on spectrograms of these files so that there is less information to digest in each input. Alternatively we can experiment more with different architectures as some of our sources achieved great results through this kind of experimentation. Though in general there is still much room for improvement in our work, going forward we are optimistic that machine learning can be a very successful tool when used to classify musical genres.

Works Cited

Liu, Xin, Qingcai Chen, Xiangping Wu, Yan Liu, and Yang Liu. “CNN Based Music Emotion Classification.” arXiv.org. Harbin Institute of Technology, April 19, 2017. https://arxiv.org/abs/1704.05665.

Feng, Lin, Shenlan Liu, and Jianing Yao. “Music Genre Classification with Paralleling Recurrent Convolutional Neural Network.” Arxiv, December 2017. https://arxiv.org/pdf/1712.08370v1.pdf.

Wyse, Lonce. “Audio Spectrogram Representations for Processing with Convolutional Neural Networks.” Arxiv. National University of Singapore, May 2017. https://arxiv.org/pdf/1706.09559.pdf.

Zhang, Weibin, Wenkang Li, Xiangmin Xu, and Xiaofeng Xiu. “Improved Music Genre Classification with Convolutional Neural Networks.” ISCA. Interspeech, 2016. https://isca-speech.org/archive_v0/Interspeech_2016/pdfs/1236.PDF.