Music Genre Classification

Auburn University
Department of Computer Science & Software Engineering
COMP 5/6600: Machine Learning
Dr. Bo Liu
Final Report

Clayton Haley
cch0045@auburn.edu

Gabrielle Taylor
grt007@auburn.edu

Colin Vande Vijvere
ccv0004@auburn.edu

Korsha Brown
ksb0034@auburn.edu

Abstract

Music classification models are utilized as the basis for the features you see in many digital music platforms. Spotify, Apple Music, Tidal and Pandora are just a couple of the well-known multi-billion dollar companies that are currently taking over the sector. Their applications include playlist generation, recommendation systems, and song search optimization. Spotify is at the top of the market with a net worth of $26 billion. This is because they have placed machine learning at the core of their research. They invest most of their time and resources into machine learning to improve the way users find and listen to music. One of their most recent popular additions is their Discover Weekly Playlist, which features new music based on the listeners activity and also exposes users to new and upcoming talent. Shazam, another popular digital music platform uses music classification to simply identify the title and artist of a song just by "listening" to it. Regardless of how music classification is implemented, genre classification is a vital step in the process to be able to provide the best and most accurate service to their users, which served as the motivation of our project. Our goal is to write an algorithm that can identify a particular song using machine learning.

1 Introduction

Music information retrieval is the interdisciplinary science of retrieving valuable information from music. Individuals demonstrate different preferences in music, yet little is known about the underlying preferences of music. As the amount of music being released on a daily basis continues to rise, so does the need for accurate meta-data required for database management and search/storage purpose rise. In the past decade, this field has grown tremendously due to its many real-world applications.

Music is said to increase productivity, alertness, and concentration which could lead to the ability to enhance certain cognitive networks by the way in which it is organized. Additionally, music in a historical sense, has also been used for social bonding, comfort, motivating or coordinating physical labor, the preservation and transmission of oral knowledge, ritual and religion, and the expression of physical or cognitive fitness. Those who use or study MIR may have a background in psychology, academic music study, signal processing, informatics, machine learning, and computational intelligence. MIR can perform tasks like fingerprinting, pitch tracking, tempo estimation, song form, score alignment and genre classification.

Though being one of the most common ways to manage digital music databases, music classification is considered very difficult, as well as a crucial task. Music genres are difficult to classify due to their subjective and ambiguous nature. This idea has been constantly debated in the field of MIR. Consider the fact that the coined term of the genre for a piece of music came along later. Genres are also being redefined every couple of years and can overlap or become hierarchical without necessarily being a subset of the other. Oftentimes, genres don’t correspond with the sound of music either. You will find they resonate to more of the time, place, or culture of when the musician created it. In our project, we plan to create a machine learning model that is able to differentiate music genres by extracting features from the music dataset and using those features to classify songs.

Feature extraction involves reducing the number of resources used to describe a large set of data. The key to constructing an effective model is properly optimized feature extraction. There are a number of extraction tools available that come in a number of different formats.There are two libraries that are mainly used in python: Librosa and PyAudio. Librosa analyzes audio signals in general, but is a module that is more geared for music. PyAudio can be used to play, record, or steam the audio directly.

Feature extraction can be used to create a playlist of similar genres, serve as the foundation for recommender systems, categorize music, and find trends and discover popular genres to focus marketing and creative efforts.

2 Dataset

The dataset we selected was the GTZAN Dataset which is a well-known music genre classification dataset. It contains a collection of music that consists of data on 10 different genres with one hundred audio clips, each 30 seconds long. Besides that, it also includes one thousand 3 second audio clips. It also contains a visual representation for each audio file for potential image classification. There are 2 CSV files containing features of the audio files. Overall, we found that this is a very good dataset to use. Our only complication with the dataset is that the 30 second clips did not have that many samples, leading to a lower accuracy compared to its 3 second counterpart.

3 Methods

We implemented four methods in our analysis, two of them serving as baseline models, and three of them being a part of our final product. The first of our two baseline models was the K-Nearest-Neighbors algorithm. Because this was just the baseline model, we only did some basic exploration of the data in order to get a model that performs decently on our data. This also applies to our second baseline model, the Random Forest Algorithm.

3.1 Baseline Models

The KNN model was trained on both the 30 second dataset and the 3 second dataset, using the sklearn.neighbors KNeighborsClassifier module. We gave the model minimum and maximum values for K (1,25) then trained the model and tested the model to get accuracy results. The results also gave us the best value for K, which we then used to train and test the models using that specific value to obtain the accuracy, precision, recall and f1-scores.

Similarly to the KNN algorithm, we used Random Forest as a baseline model to get a better understanding of how well the data is learned on the model. The sklearn.ensemble RandomForest Classifier was our method of choice. There were a couple parameters that we had to set first: n_estimators = 100, max_depth = 5. The “n_estimators” parameter identifies the number of trees that we want to build, each having a “max_depth” of 5. We then used the prediction results from this model to generate the metric scores, feature importances, and confusion matrices. As expected the metrics for both KNN and Random Forest were low, so we knew that we had to tune our parameters in order to receive optimal results. However, after further research, we discovered that the KNN will ultimately not perform well on our data, especially our 30 second data set, due to the low number of samples. So, we decided to throw this model out of our final product and choose a higher performing model, the Multi-layer Neural Network Model and a Convolutional Neural Network.

3.2 Models Used for Final Analysis

As mentioned previously, we used three models for our final, in-depth analysis of the GTZAN datasets: Random Forest, Two-Layer Neural Network - Scratch, Two-Layer Neural Network - sklearn, Convolutional Neural Network. Unlike the baseline models, these models required significant amounts of fine tuning so that we could find a model that best fits our data.

3.2.1 Random Forest

The first step in the Random Forest was to define our sklearn classifier. The optimal parameters for our model were n_estimators =100 and criterion = 'entropy'. We adjusted "n_estimators" to higher and lower values, but 100 estimators proved to achieve better results. The selected splitting criteria, entropy (information gain), was chosen solely because this is the splitting method that we are much more familiar with in this course. The next step was to calculate the accuracy for the test data, find the top five important features of the model, and calculate the confusion matrix for the prediction results. These steps were identical for the 30 second dataset and the 3 second dataset.

3.2.2 Two Layer Neural Network - Scratch and Sklearn

The goal of implementing a scratch Two Layer Neural Network was to compare its performance with those of the Convolutional Neural Network and sklearn's MLPClassifier. Our hand-coded neural network was a feed forward network with backpropagation learning. The two activation functions used were ReLU (Hidden Layer) and Soft Max (Output Layer). When training the model on our data, we started with a learning rate of 0.001 and increased it by multiplying by 1.5 (increasing by 50% each iteration) until the learning rate reached a value just less than one. The number of iterations we used was 800. We then plotted the trend of accuracy against the increasing learning rate. The next and final step was to find the optimal learning rate along with its accuracy. The process of the sklearn Neural Network was identical to that of our scratch model. The only difference is that we trained the model using sklearn's MLPClassifier.

3.2.3 Convolutional Neural Network (CNN)

The Convolutional Neural Network was our highest performing model, as expected. First, we had to initialize a Keras Sequential model, with the input shape being the number of features in our data set, and several dense layers with their respective activation function. Instead of only using two layers, we chose a total of four, while hoping for better prediction results. The first three layers used the ReLU activation function and had 512, 256, 64, and 10 units respectively. The loss function was sparse categorical cross-entropy, and the metric we used to measure performance was accuracy. Before fitting the model, we had to adjust our dataset to where there was training, validation and testing data. The validation data is only part of the training set in order to eliminate the chance of overfitting. We chose a batch size of 30, which means that 30 training samples are passed through the model before the internal parameters are updated. The last parameter for the model fitting was choosing the number of iterations, which was 200. We noticed that the model was getting a training accuracy of about 100% and a fairly constant validation accuracy pretty early in the model; therefore, we did not need 800 iterations like the sklearn and scratch models. Next, we needed to generate predictions on the test set and show plots for the training accuracy and testing accuracy. The last step was to create a confusion matrix for the predicted and actual labels.

4 Challenges

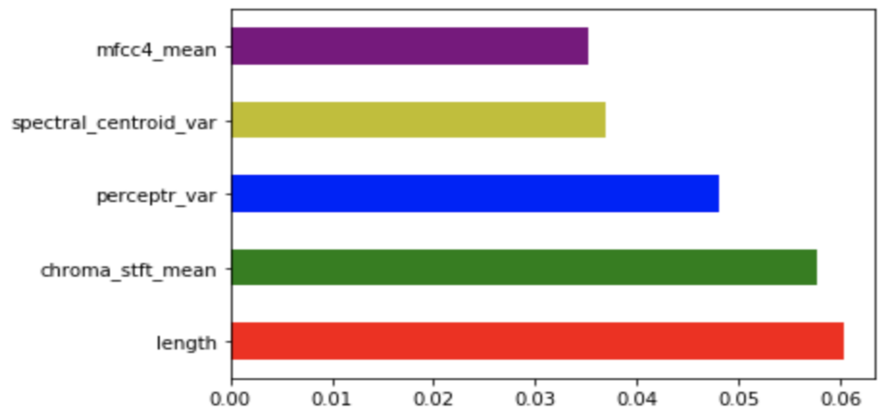
Although our model presented pretty accurate results in some areas, there were some difficulties that we encountered during our analysis that may have prevented us from achieving better scores. Our first and main challenge was the size of the 30 second data set. For each genre, we only had 10 clips, but we had 100 clips in the 3 second dataset. This could easily lead to underfitting on the 30 second clips. The next problem that we discovered was the trend of accuracy against the learning rate in our scratch and sklearn neural networks. As the learning rate increased, the accuracy for our scratch model increased at a fairly smooth rate. On the other hand, the accuracy decreased in an odd manner for our sklearn model. This was concerning although we did see good accuracy for both models. The issue could lie somewhere in the scratch neural network, or it could have something to do with the varying activation functions.

5 Results

5.1 Random Forest Feature Importances and Accuracy

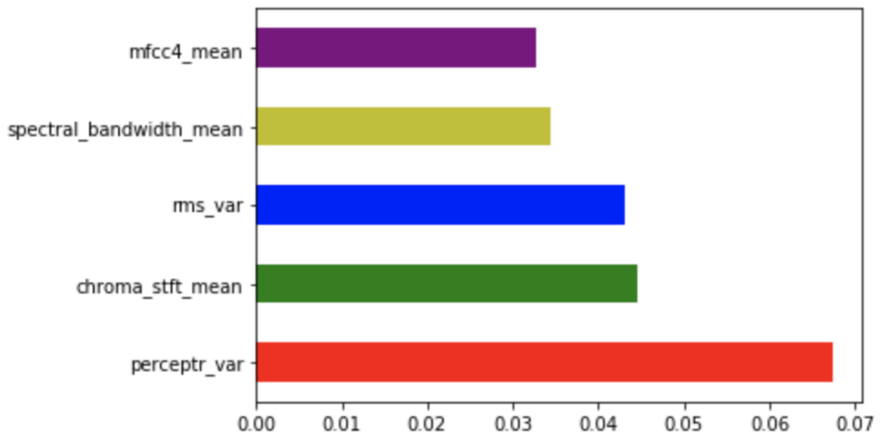
<u>**30 Second Dataset:**</u>

```
Accuracy:  78%
```



<u>**3 Second Dataset:**</u>

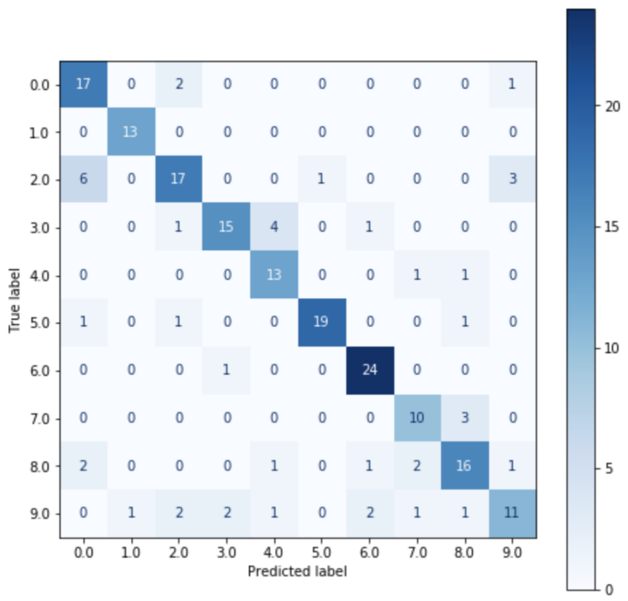
```
Accuracy:  88%
```



The model predicted with 78% accuracy on the 30 second dataset and 88% on the 3 second data set. Above are the top 5 feature importances for our model. This means that these features contributed the most, or were the most significant in predicting the genre of a music clip. As we can see, the important features are different for both models. One feature that caught our eye was "length", and we wondered why it is so important for the 30 second clips but not the 3 second. "Chroma_stft_mean" and "mfcc4_mean" are equally important for both models, but "perceptr_var" is more significant in the 3 second model than the 30 second.

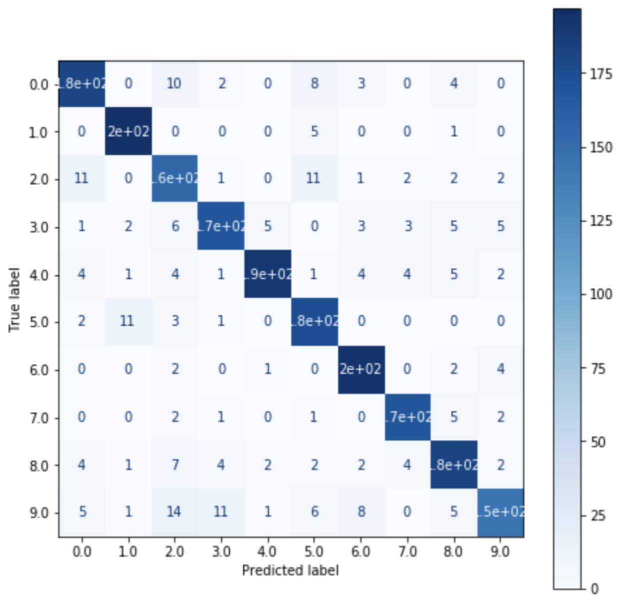
5.2 Random Forest Confusion Matrices

<u>30 Second Dataset:</u>



True label \ Predicted label	0.0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0
0.0	17	0	2	0	0	0	0	0	0	1
1.0	0	13	0	0	0	0	0	0	0	0
2.0	6	0	17	0	0	1	0	0	0	3
3.0	0	0	1	15	4	0	1	0	0	0
4.0	0	0	0	0	13	0	0	1	1	0
5.0	1	0	1	0	0	19	0	0	1	0
6.0	0	0	0	1	0	0	24	0	0	0
7.0	0	0	0	0	0	0	0	10	3	0
8.0	2	0	0	0	1	0	1	2	16	1
9.0	0	1	2	2	1	0	2	1	1	11

<u>3 Second Dataset:</u>



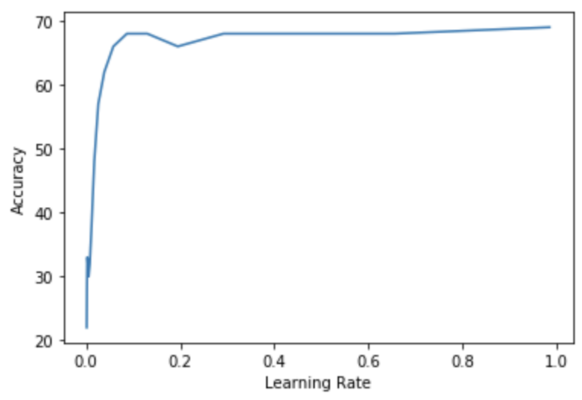
True label \ Predicted label	0.0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0
0.0	.8e+02	0	10	2	0	8	3	0	4	0
1.0	0	2e+02	0	0	0	5	0	0	1	0
2.0	11	0	.6e+02	1	0	11	1	2	2	2
3.0	1	2	6	.7e+02	5	0	3	3	5	5
4.0	4	1	4	1	.9e+02	1	4	4	5	2
5.0	2	11	3	1	0	.8e+02	0	0	0	0
6.0	0	0	2	0	1	0	2e+02	0	2	4
7.0	0	0	2	1	0	1	0	.7e+02	5	2
8.0	4	1	7	4	2	2	2	4	.8e+02	2
9.0	5	1	14	11	1	6	8	0	5	.5e+02

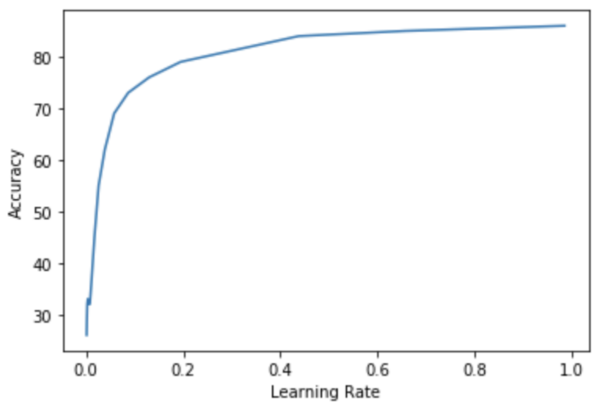
The confusion matrices show how many of each label were classified correctly. It can also show us which labels were misclassified the most. In both matrices we can see that the most misclassified genres are 9 (rock), 0 (blues) and 2 (country). The most correctly classified genres are 7 (pop), 1 (classifical), and 6 (metal).

5.3 Neural Network Scratch Model

```
30 Second Dataset:
Accuracy:  69.0 %
Optimal Learning Rate:  0.9852612533569336
Number of Iterations:  800
```

```
3 Second Dataset:
Accuracy:  86.0 %
Optimal Learning Rate:  0.9852612533569336
Number of Iterations:  800
```



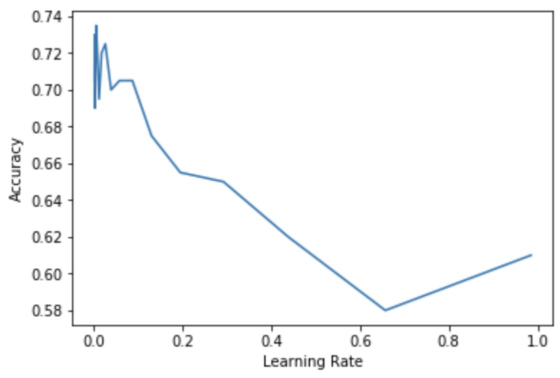


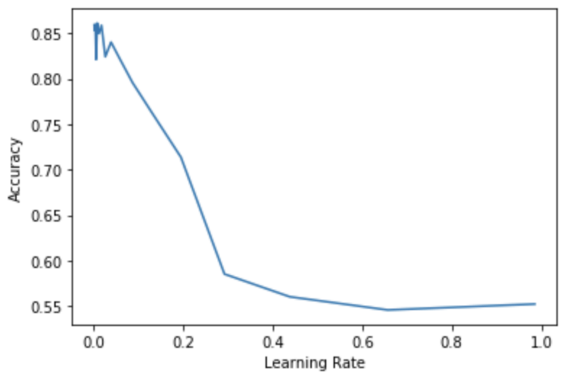
The two plots above show the trend of accuracy as the learning rate increases. The two lines seem to increase at a similar rate; however, the maximum accuracy achieved for both models is significantly different. The 30 second dataset has an accuracy of 69% and the 3 second with an accuracy of 86%. The optimal learning rate for each model is the same.

5.4 Neural Network Sklearn Model

```
30 Second Dataset:
Accuracy:  0.735 %
Optimal Learning Rate:  0.005062500000000001
Number of Iterations:  800
```

```
3 Second Dataset:
Accuracy:  0.8613613613613613 %
Optimal Learning Rate:  0.0075937500000000015
Number of Iterations:  800
```



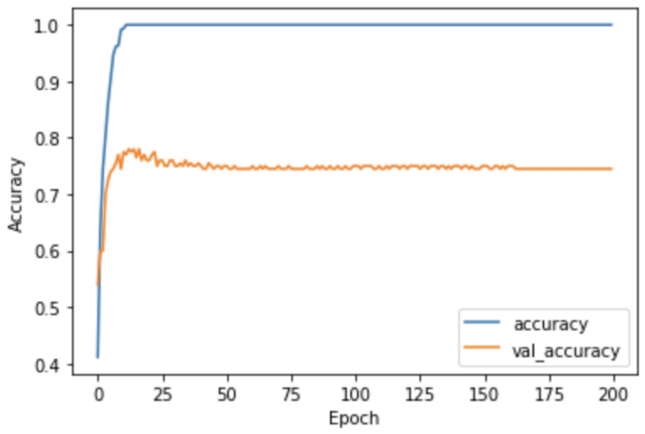


The plots above show the same representation as the previous plots. The only difference is that the accuracy decreases as the learning rate increases. The accuracies for both models appear to be about the same as our scratch model. The optimal learning rate for the 3 second dataset is higher than the 30 second learning rate.

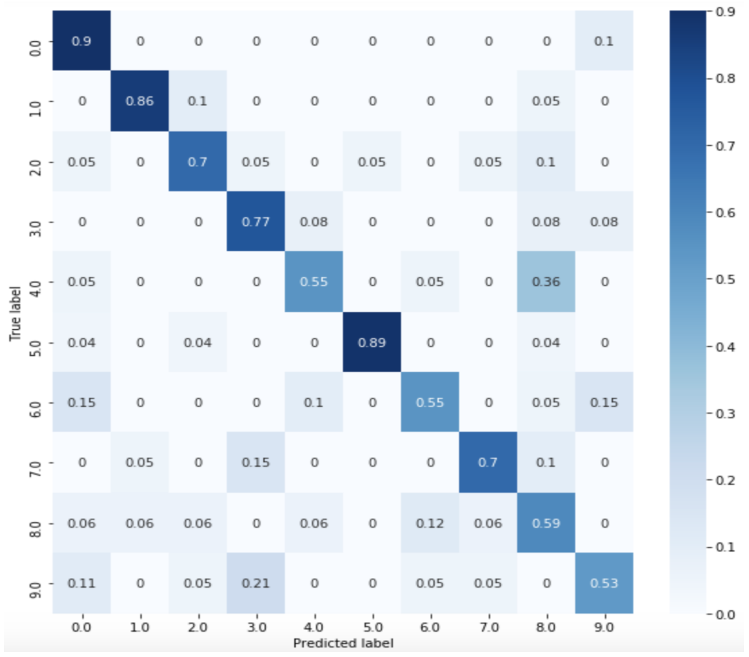
5.5 Convolutional Neural Network (CNN)

__30 Second Dataset:__

```
7/7 [==============================] - 0s 878us/step - loss: 2.3792 - accuracy: 0.7100
```



The plot above shows the trend of accuracy against the number of epochs. As expected, the training accuracy and validation accuracy are almost symmetrical, with the testing accuracy being much lower than the training. But, we do see that model predicted with 71% accuracy, which is almost the same as the scratch and sklearn models. This shows that our scratch model performed very well.

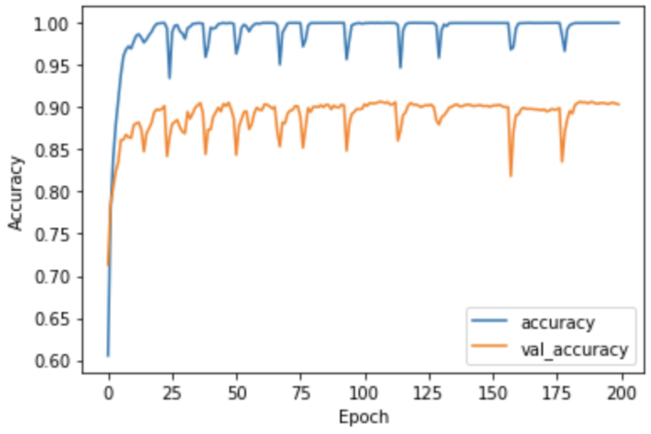


	0.0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0
0.0	0.9	0	0	0	0	0	0	0	0	0.1
1.0	0	0.86	0.1	0	0	0	0	0	0.05	0
2.0	0.05	0	0.7	0.05	0	0.05	0	0.05	0.1	0
3.0	0	0	0	0.77	0.08	0	0	0	0.08	0.08
4.0	0.05	0	0	0	0.55	0	0.05	0	0.36	0
5.0	0.04	0	0.04	0	0	0.89	0	0	0.04	0
6.0	0.15	0	0	0	0.1	0	0.55	0	0.05	0.15
7.0	0	0.05	0	0.15	0	0	0	0.7	0.1	0
8.0	0.06	0.06	0.06	0	0.06	0	0.12	0.06	0.59	0
9.0	0.11	0	0.05	0.21	0	0	0.05	0.05	0	0.53

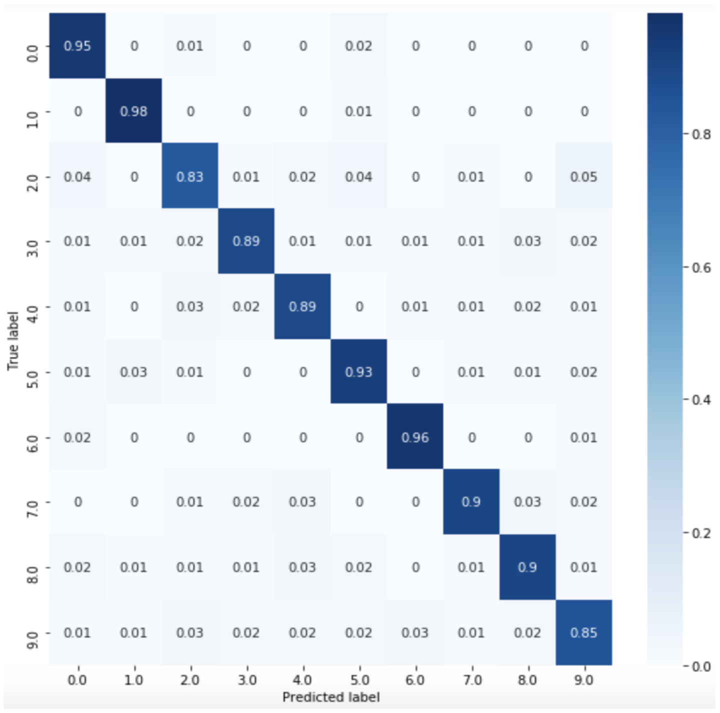
The confusion matrix above is normalized unlike the previous matrices. This means that the numbers are representations of the percentage of the labels that were predicted correctly. We can see that the most misclassified genres are 9 (rock), 6 (metal), 4(hiphop), and 8 (reggae). The most correctly classified genres are 0 (blues), 5 (jazz), and 1 (classical).

<u>3 Second Dataset:</u>

```
63/63 [==============================] - 0s 716us/step - loss: 0.8102 - accuracy: 0.9074
```



The CNN predicted with about 91% accuracy on our testing data. Again, like in our 30 second model, the training accuracy trend is almost symmetrical with our validation accuracy trend, which is very good to see. The total loss for this model is also much lower than the 30 second model.



True label \ Predicted label	0.0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0
0.0	0.95	0	0.01	0	0	0.02	0	0	0	0
1.0	0	0.98	0	0	0	0.01	0	0	0	0
2.0	0.04	0	0.83	0.01	0.02	0.04	0	0.01	0	0.05
3.0	0.01	0.01	0.02	0.89	0.01	0.01	0.01	0.01	0.03	0.02
4.0	0.01	0	0.03	0.02	0.89	0	0.01	0.01	0.02	0.01
5.0	0.01	0.03	0.01	0	0	0.93	0	0.01	0.01	0.02
6.0	0.02	0	0	0	0	0	0.96	0	0	0.01
7.0	0	0	0.01	0.02	0.03	0	0	0.9	0.03	0.02
8.0	0.02	0.01	0.01	0.01	0.03	0.02	0	0.01	0.9	0.01
9.0	0.01	0.01	0.03	0.02	0.02	0.02	0.03	0.01	0.02	0.85

The confusion matrix above is also normalized. Because our testing accuracy was high, we can expect these matrix values to be much higher as well. The most misclassified genres are 9 (rock) and 2 (country). The most correctly classified genres are 0 (blues), 5 (jazz), and 6 (metal).

6 Conclusion

The task of music genre classification is valuable and widely explored. As various techniques increase, the methods for music genre classification expand. We were able to implement four models to classify the music genres and compare. We implemented a RandomForest classifier, a two-layer neural network with sklearn, a two-layer neural network from scratch, and a convolutional neural network. Our CNN had the highest accuracy using the 3 second data features and the RandomForest had the highest accuracy for the 30 second data features. We were able to see that Rock was the most misclassified genre. Rock shares features that are very similar to blues and pop music. Therefore, it tends to be misclassified more. We believe that with more unique features, the classification will improve drastically.

	30 seconds	**3 seconds**
RandomForest	78%	88%
NN Sklearn	73%	86%
NN Scratch	69%	86%
CNN	71%	91%

Each model produced great results. Our neural network model from scratch was almost as successful as the sklearn model. We believe that each of these models are useful solutions for music genre classification. We conclude that using 3 second data features greatly improves the model because the features are more specific than the broad 30 second data features. Additionally, we conclude that the CNN model is the best model for music genre classification between the four models because of the high accuracy. In the future, we hope to test what features help to improve rock classification.

References

Bahuleyan, Hareesh. "Music genre classification using machine learning techniques." arXiv preprint arXiv:1804.01149 (2018). https://arxiv.org/pdf/1804.01149.pdf

Garza, M. (2020, July 22). Mir: Music information retrieval. Retrieved April 24, 2021, from https://tagteamanalysis.com/2020/07/mir-music-information-retrieval/

Haggblade, M., Hong, Y., & Kao, K. (n.d.). Music Genre Classification. Retrieved March 7, 2021, from http://cs229.stanford.edu/proj2011/HaggbladeHongKao MusicGenreClassification.pd

Nanni, Loris, et al. "Combining visual and acoustic features for music genre classification." Expert Systems with Applications 45 (2016): 108-117. https://www.sciencedirect.com/science/article/pii/S0957417415006326?casa_token=B2pSx2FRamwAAAAA:Hc9FRt33-dHo6w0uLFXOPg5siyEKofiOozvb40UGvXTlq07rcq5cxeqi3_WZITi-FJQ2twTtANc0

Srebrenik, B. (2018, December 20). Machine learning and music classification: A content-based filtering approach. Retrieved March 7, 2021, from https://towardsdatascience.com/machine-learning-and-music-classification-a-content-based-filtering-approach-f2c4eb13bade