CS 4641 Semester Project: Music Genre Classifier

Introduction

The modern landscape of music is in constant flux. With genre boundaries being pushed consistently [1], and the quantity of available music rapidly increasing [2], classification has become a must-have for music recommendation. To that end, several research projects have taken up the challenge, using various approaches such as convolutional neural networks, deep neural networks, and support vector machines [3][4]. We intend to follow in their footsteps, considering model applicability for this novel goal.

Dataset

GTZAN is a dataset for music genre classification [5]. It includes 10 genres with 100 audio files for each genre. Each audio file has a length of 30 seconds. Each audio file has a visual representation in the form of Mel Spectrograms. The dataset also has 2 CSV files containing features of the audio files with means and variances for each audio file.

Dataset available here

Problem Definition

The surge in music production in recent decades has blurred the lines between music genres [1], creating a blend of music styles. The problem we face is finding music that fits specifically to one genre, or perhaps spans two or more. Thus, we will create a music classifier that will find the genre of input music clips. The motivation behind the music classifier is that it can assist with better recommendation algorithms, especially as it applies to new releases that have not yet had user data associated with them.

Methods

All models and associated implementation code can be found in our github repository in the “neural_networks” directory.

Preprocessing Techniques

We process our data in a number of ways. First, we take the spectrogram data and crop it to a specific rectangular region. We then return this image as a flattened spectrogram for our model. The cropping was needed because the spectrogram data came with a large border area which contained no information. Cropping removes this and removes many wasted parameters. The images are also converted to grayscale. The color in the images does not actually convey any meaning except magnitude and is only for aesthetic and interpretability purposes. Removing this means single pixel brightnesses can be used instead of needing to account for three color channels.

Data augmentation to increase training data quantity was also done by randomly sampling different windows of the spectrogram. This was done equivalently to sampling different time slices from the songs. This was very helpful for us because the original size of our dataset was somewhat small and this greatly increased the number of datapoints for training. In the case of the implemented CNN, using windows of 90 samples (~0.9 seconds) we increased training data by a factor of ~300 when compared to using the full 30-second spectrogram clips (each 30-second clip could have windows spanning 90 pixels starting at any pixel in the spectrogram with sufficient margin).

The Dense Neural Network and Support Vector Machine models both trained on the song metadata which included a large variety of feature extractions. One of the biggest was the use of Mel Frequency Cepstral Coefficients (MFCCs). These are a small set of features that describe the timbre of an audio sample [7].

Other features extracted from the audio data by the dataset and used for these models includes (not exhaustive) chroma, spectroid data, tempo, and harmony. These extracted features are all standardized before being used to train the model.

Neural Network

Our group decided to set up a multi-class classifier, since we have multiple different genres that we are trying to classify. We imported scikit-learn's MLPClassifier, which is a multi-layer perceptron classifier, to predict music genres based on metadata included in the dataset. The reason we chose the MLPClassifier was because an MLP is a neural network, which can model complex patterns in data. This suits our problem since the relationship between features and genres is not linear and multidimensional. Another reason we chose the MLPClassifier was because of its flexibility. We can tune the parameters of the model, like the number of hidden layers, number of neurons per layer, learning rate, and activation function. We can optimize these parameters to find the best model for capturing the features in our spectrogram data.

The model previously had trouble with overfitting using the spectrogram data due to the relatively small dataset and the huge number of parameters in the spectrogram data. This led to an extremely high training accuracy and low testing accuracy. Therefore, we adjusted the model from the midterm report to the final report to use the song metadata instead. This included features such as Mel-Frequency Cepstral Coefficients, spectral data, tempo, etc. The hyperparameters were retuned for this application but the rest of implementation was similar including the use of MLPClassifier. The model ended up using one hidden layer of 32 perceptrons. More perceptrons or layers started to overfit and less was reducing the accuracy so this structure was chosen.

Convolutional Neural Network

Our group chose to use a simple Convolutional Neural Network (CNN) architecture since one common way to look at sound data is as a spectrogram, which is an image where local features can help in classification. CNNs are widely used in computer vision due to their effectiveness in image classification, thus we reasoned they could be quite effective in our case. Our approach is a three tiered architecture with three layers of convolution, activation, and maxpool layers, with an increasing number of channels (applied kernels) each tier (doubling each time). The intent is to have the network learn generalizable features in the earlier layers and then combine them with more specificity later in the network to classify the spectrogram. The spectrogram is fed into the network in segmented randomly-chosen chunks (in length; the height of the spectrogram stays the same) for training. This is a simple random-window data augmentation technique for increasing available training data. In evaluation, every consecutive chunk of the spectrogram is passed into the network, and the output is averaged over all logits before making a prediction. This is intended to "smooth out" the prediction, providing a more accurate result.

Support Vector Machine

Our group decided to use a Support Vector Machine (SVM) using scikit-learn's SVM. We used a support vector machine because it's very effective in high dimensional spaces. This makes it a good choice for multi-class problems such as genre classification. SVM's are also very memory efficient since it uses a subset of training points in the decision function (called support vectors). We preprocessed a csv file containing features from a 30 second clip of audio. This worked well for us since the csv file was small and compact. The feature set was also very helpful with many audio features to help classify genre. One file has for each song (30 seconds long) a mean and variance computed over multiple features that can be extracted from an audio file.

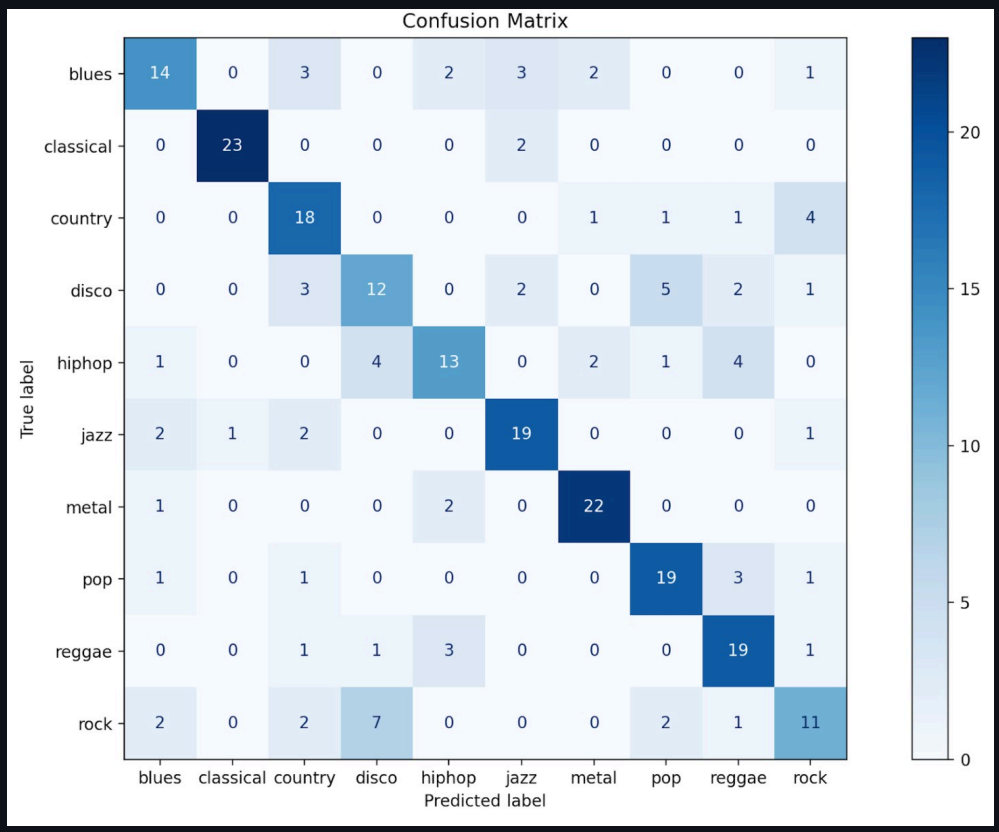
Results and Discussion

Neural Network Analysis

The analysis of the neural network included measures such as accuracy, precision, recall, f1-score, and training accuracy. The accuracy was 0.68. This shows the model is definitely working although the result isn't perfect. Training accuracy is 0.94 which may indicate a bit of overfitting but not complete memorization.

The confusion matrix also indicates that the model struggles most with blues, rock, disco, and hip hop. It was very accurate at classifying classical. Rock and disco were in particular confused with each other a lot. This could either indicate a coincidence with the model or more likely strong similarity between the genres themselves.

	precision	recall	f1-score	support
blues	0.67	0.56	0.61	25
classical	0.96	0.92	0.94	25
country	0.60	0.72	0.65	25
disco	0.50	0.48	0.49	25
hiphop	0.65	0.52	0.58	25
jazz	0.73	0.76	0.75	25
metal	0.81	0.88	0.85	25
pop	0.68	0.76	0.72	25
reggae	0.63	0.76	0.69	25
rock	0.55	0.44	0.49	25
accuracy			0.68	250
macro avg	0.68	0.68	0.68	250
weighted avg	0.68	0.68	0.68	250



Convolutional Neural Network Analysis

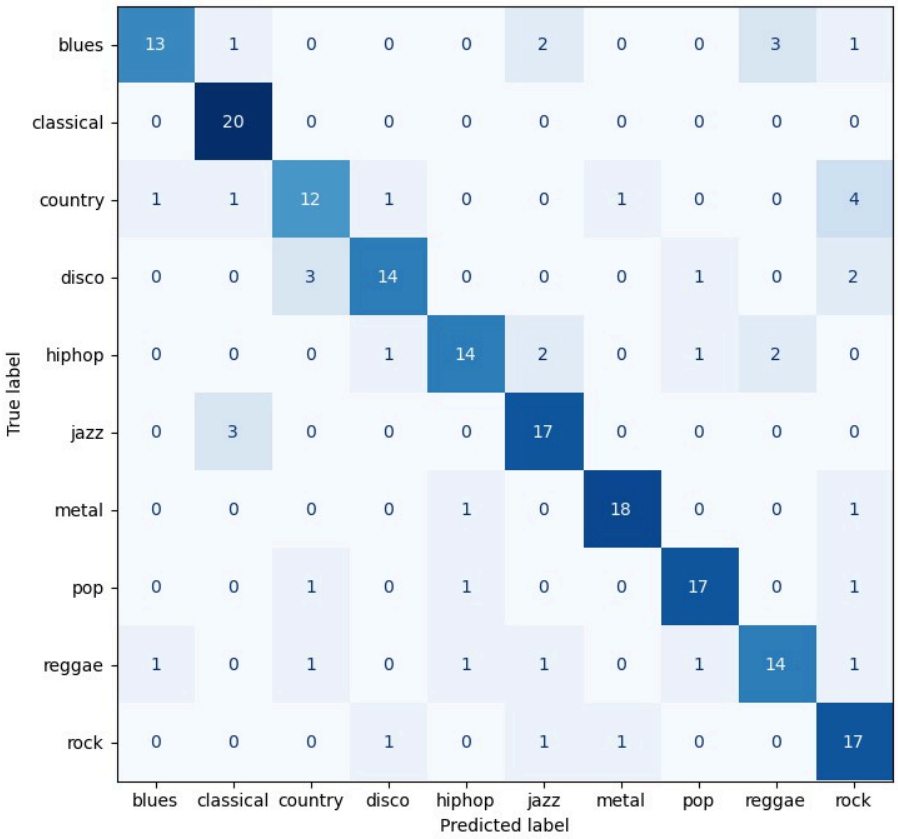
This model was the most successful, and we can see that by analyzing the quantitative measures of the model. The accuracy of the model, which measures the proportion of correctly classified instances, was reported as 0.78. This indicates that the model correctly classifies 78% of the spectrogram images into their respective genres. The average precision score over all classes, which measures the proportion of true positive predictions among all positive predictions, is reported as 0.79. This suggests that when the model predicts a specific genre, it is correct 79% of the time. The average recall score over all classes,

which measures the proportion of true positive predictions among all actual positive instances, is similarly reported as 0.78. This indicates that the model successfully identifies 78% of the actual instances of each genre. The F1 score, which is the harmonic mean of precision and recall, is 0.78. This balanced metric suggests that the model's precision and recall are equally strong.

The confusion matrix generated during the evaluation provides a detailed breakdown of the model's performance across different genres. This was our most successful model, and you can see that as the darkest squares are on the main diagonal, indicating that our accuracy was much better than our previous neural network in our midterm report. One thing we're especially proud of was that classical music was identified perfectly, while metal, hip hop, pop and rock were also identified quite accurately.

Overall, the CNN genre classifier demonstrates a consistent performance with an average accuracy, precision, recall, and F1 score of 0.78-0.79. This performance is indicative of a decently-balanced model that effectively classifies music genres from spectrogram images, although there remains room for further optimization and improvement.

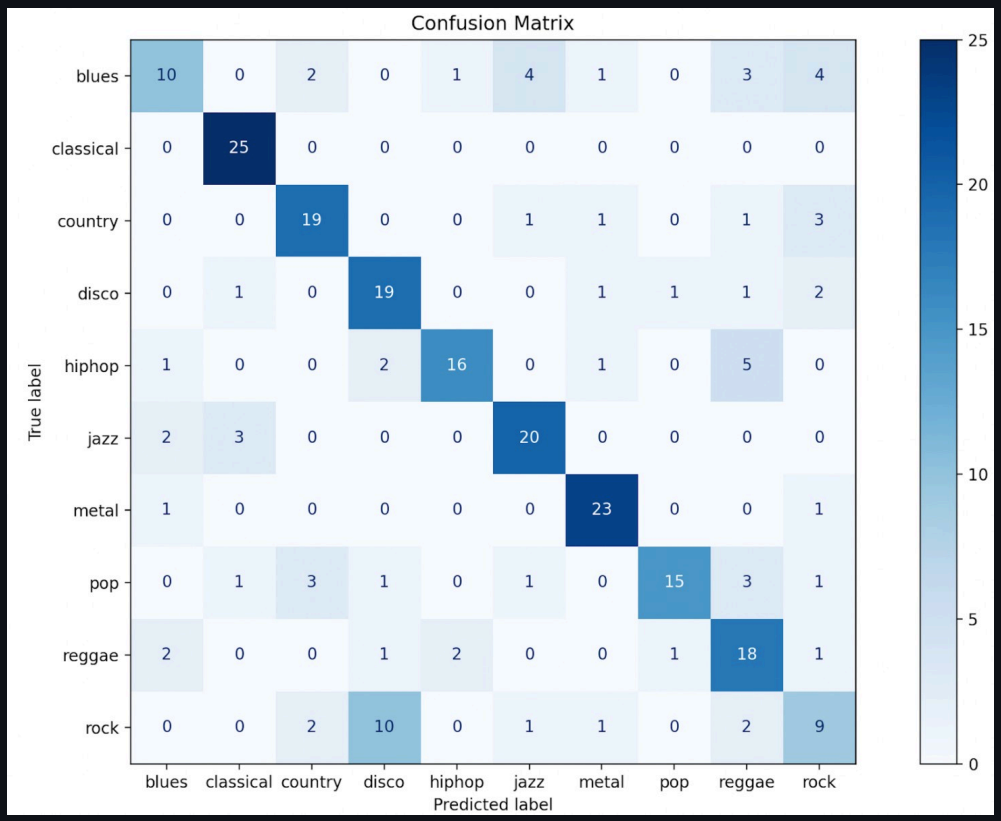
	precision	recall	f1-score	support
blues	0.87	0.65	0.74	20
classical	0.80	1.00	0.89	20
country	0.71	0.60	0.65	20
disco	0.82	0.70	0.76	20
hiphop	0.82	0.70	0.76	20
jazz	0.74	0.85	0.79	20
metal	0.90	0.90	0.90	20
pop	0.85	0.85	0.85	20
reggae	0.74	0.70	0.72	20
rock	0.63	0.85	0.72	20
accuracy			0.78	200
macro avg	0.79	0.78	0.78	200
weighted avg	0.79	0.78	0.78	200



Support Vector Machine Analysis

One metric we used for measuring our support vector machine across our genres, blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, and rock, was accuracy. We achieved an accuracy of 0.696 for classifying genre. While this could be more accurate, it's good enough for our purposes for now. We are planning on making improvements to the model by tuning its parameters. We also have a confusion matrix. The diagonal values represent the correct predictions. For example, classical had 25 correctly classified samples (which are all of the samples), and metal had 23, but rock only had 9, which shows the worst performance. The off-diagonal values represent misclassifications. For example 4 blues songs were misclassified as rock. Another metric we used was precision, which measures how many of the predicted instances for a genre are correct. Pop had the highest precision of 0.88, so 88% of the songs predicted as pop were correct. Another metric we used was recall, which measures how many actual instances of a genre were correctly identified. Classical performed the best here with a recall of 1.00, meaning all of the classical songs were correctly classified. We also had the f1-scores which represent the harmonic mean of precision and recall. Classical performed the best with a score of 0.91, while rock performed the worst with 0.39.

	precision	recall	f1-score	support
blues	0.62	0.40	0.49	25
classical	0.83	1.00	0.91	25
country	0.73	0.76	0.75	25
disco	0.58	0.76	0.66	25
hiphop	0.84	0.64	0.73	25
jazz	0.74	0.80	0.77	25
metal	0.82	0.92	0.87	25
pop	0.88	0.60	0.71	25
reggae	0.55	0.72	0.62	25
rock	0.43	0.36	0.39	25
accuracy			0.70	250
macro avg	0.70	0.70	0.69	250
weighted avg	0.70	0.70	0.69	250



Model Comparison

A quick comparison of the models allows us to see the strengths and weaknesses of each model. Objectively, the Convolutional Neural Network (CNN) model performed the best, while the Neural Network (NN) was the worst, leaving the Support Vector Machine (SVM) model in the middle. The CNN had an accuracy score of 78%, the SVM had a score of 70%, and the NN had an accuracy of 68%. It's important to note that we were able to fine tune and improve the NN model from an accuracy around 20% to 68%. We know that CNN's typically perform better in image and video recognition tasks. Because our model takes in spectrogram data, it makes sense that the CNN model performed the best, in comparison to the SVM and NN. A particular limitation of the NN that we found was that the training accuracy was around 94%, indicating overfitting. Neural Networks were an excellent base model because they allowed us to fine-tune and improve the parameters, but it was prone to overfitting. The SVM performed marginally better than the NN because of its robustness to overfitting, yet it lacks the CNN's strength in learning patterns from the spectrogram input. Overall, the best model was the model most attributed to performing well with image data, which was the Convolutional Neural Network.

Next Steps

As far as next steps are concerned, one major improvement we could make to improve the performance of our models is acquiring additional audio datasets to supplement GTZAN. This would not only make our models more generalizable as we could explore more diverse genres and subgenres, but would also reduce the likelihood of overfitting that we saw in our neural network implementation.

Another step we could take is using dimensionality reduction methods like PCA to improve the efficiency of our models, especially for the neural network and SVM which used the extracted features from the csv. Some of these features may be largely irrelevant for genre classification, so it would be interesting to identify which features are most impactful. This, along with general improvements to our models such as more sophisticated architecture and use of techniques like dropout or regularization could significantly improve the performance of our models.

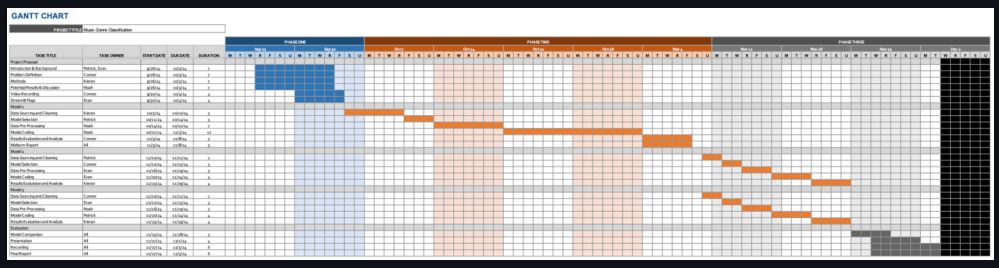
With improvements to the current model, it could be used internally as a music library organizer, efficiently automating the process of grouping songs together. Taking the project further, there is obvious potential to delve into song similarity and implement a recommendation system using the best performing model (CNN). This would be an exciting practical application that is definitely in the realm of possibility.

Contributions

Contribution Table

Name	Final Contributions
Connor	- CNN Analysis - Comparison of Models
Evan	- CNN Implementation - Streamlit - Gantt Chart - Methods: CNN
Kieran	- Report Presentation - Next Steps
Noah	- NN Implementation and Updates - Methods: Preprocessing - Methods: NN - NN Analysis
Patrick	- SVM Implementation - Methods: SVM - SVM Analysis

Gantt Chart



References

[1] "The evolution of music: Exploring the music evolution timeline over the past 20 years," Your Ghost Production, Accessed: Oct. 2, 2024. [Online]. Available: https://yourghostproduction.com/the-evolution-of-music-exploring-the-music-evolution-timeline-over-the-past-20-years/

[2] C. Willman, “Music Streaming Hits Major Milestone as 100,000 Songs are Uploaded Daily to Spotify and Other DSPs,” Variety, Oct. 26, 2022

[3] N.M. Stephen, “Music Genre Classification,” M.S. thesis, California State University, Northridge, Northridge, CA, 2023, Accessed: Oct. 1, 2024. [Online]. Available: https://scholarworks.calstate.edu/downloads/73666b68n

[4] Patil, S.A., Pradeepini, G. & Komati, T.R. “Novel mathematical model for the classification of music and rhythmic genre using deep neural network”. J Big Data 10, 108 (2023), Accessed: Oct 3, 2024. [Online]. Available: https://doi.org/10.1186/s40537-023-00789-2

[5] “GTZAN Dataset - Music Genre Classification,” Kaggle, Accessed: Oct. 1, 2024. [Online]. Available: https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification

[6] R. Kundu, “Confusion Matrix: How To Use It & Interpret Results [Examples],” Accessed: Nov. 8, 2024. [online]. Available: https://www.v7labs.com/blog/confusion-matrix-guide

[7] E. Deruty, “Intuitive Understanding of MFCCs,” Medium, Accessed: December 3, 2024. [Online]. Available :https://medium.com/@derutycsl/intuitive-understanding-of-mfccs-836d36a1f779