Max Eduardo Lazarini Wienandts

Project: Music Classification.

Introduction:

Problem Statement: Can a song be accurately classified into 10 different genres solely based on its audio file using machine learning techniques?

There are several possible use cases for this project. The most direct use cases are to classify folders, and to classify new songs in streaming services. Moreover, once a song is classified in a genre, it can be used as an input in a recommendation model. What is more, it may be possible to use the resulting model of this project as a base model in a transfer learning technique to solve any classification problem related to audio, i.e., if the audio from a heart beat indicates any disease.

The data:

The data used is the GTZAN Dataset. Available in <https://opihi.cs.uvic.ca/sound/genres.tar.gz> and <https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification>.

Each audio file has 30 seconds and they are classified in 10 different music genres: blues, classical, country, disco, hip-hop, jazz, metal, pop, reggae, rock. There are 100 samples for each genre but for jazz that has only 99 samples.

Approach:

For each audio file was saved the plot of the waveform signal and the MEL spectrogram. In the jupyter notebook, there are an example of these plots for one song for each of the genres.

Leland Roberts in https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53 describes a spectrogram as a way to visually represent a signal’s loudness, or amplitude, as it varies over time at different frequencies. He adds that the MEL transformation is a unit of pitch such that equal distances in pitch sounds equally distant to the listener.

It is important to notice that the raw audio signal would generate a vector with more than 650,000 features for each song. Working with these features would require a more powerful hardware. Hence, techniques such as CNN together with LSTM will not be tested.

For all the models was used 649 songs for training, 150 for validation, and 200 for test.

First, using only the MEL spectrogram as input, I tried a CNN model with 64 filters, followed by a max pooling layer, a flatten layer, a dense layer with 128 units and SeLU as activation function, and a dense layer with 10 units with softmax as output. The loss function used is the sparse categorical cross entropy, and the optimizer is Adam with a learning rate of 0.00001. This model has 179.440.778 parameters. With 108 epochs, this model resulted in a validation accuracy of 0.6133 and training accuracy of 0.9985. It took almost 7 minutes to run these 108 epochs. In the jupyter notebook can be found a plot of the training and validation accuracy by epoch for all models.

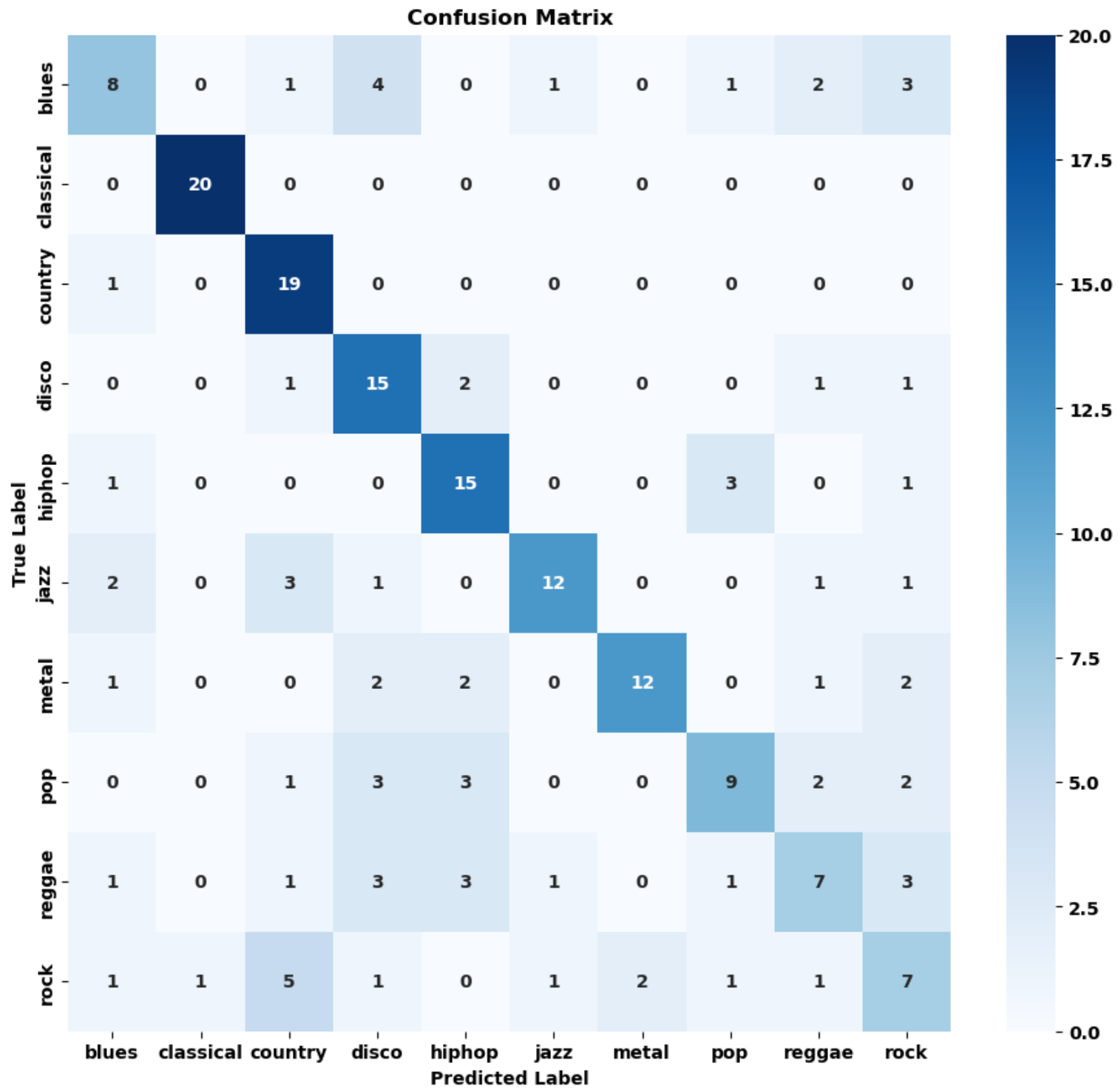
It is clear that this model has an overfit problem. Therefore, to increase the validation accuracy and decrease the overfit, I tested adding dropout layers, bias and kernel regularization for the CNN layer, decrease the number of filters, and develop completely new architectures. At total, I tested 4 different architectures using only the MEL spectrogram as input. The best result was “model 4” that uses only 16 filters, but it reads the input three times using different CNN layers with different kernel size (further information available in the jupyter notebook). Nonetheless, the improvement was minimal. The validation accuracy increased only to 0.6200 and the training accuracy remained higher than 0.99, and it took almost 12 minutes to run 128 epochs.

Considering that just changing the architecture was not enough to solve the overfitting, I tried a model (model 5) with 2 inputs, the MEL spectrogram and the waveform signal. Both inputs pass through 2 CNN layers with 16 filters and, after flattened, they are concatenated. Unfortunately, the overfit remained and the accuracy did not increase.

One possible reason for none of these approaches solve the overfitting problem is the lack of data. Hence, I tried using transfer learning with pre-trained weights and adding some new dense layers to fine tune the model. The models that I tried are Resnet50 and EfficientNetB2. The model with the best performance was EfficientNetB2 with a validation accuracy of 0.6533 and a training accuracy of 0.8259. It took more than 20 minutes to run.

Results:

The confusion matrix for the EfficientNetB2 is:



As we can see, the model was capable of perfectly identify the classical genre. In addition, only one music from the genre country was mislabeled. Moreover, the genre disco and hip-hop had a satisfactory accuracy of 75%, and the genres jazz and metal had an accuracy higher than 50%. On the other hand, the model struggled in accurately classifying the blues, pop, reggae, and rock genres, with accuracies ranging from 35% to 45%.

It is interesting to notice that this model mixed some blues songs as disco, and some rock songs as country. No prevalent genre mixing was observed for pop and reggae.