Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Neural Networks and Deep Learning Final Project

Music Genre Classification

Author: Nicolás Ortiz de Zárate

05/07/2023

1 Abstract

Over the last several years, neural networks have shown powerful capabilities in many classification tasks, including music genre classification. In this project, we test three different convolutional neural networks to solve the music genre classification problem and compare their performances. The networks we use are: 1) a 1-dimensional convolutional neural network on the raw audio input, 2) a 2-dimensional model using Mel spectrograms, a Fourier representation of the audio signals, and 3) a combination of the previous two networks that uses both the raw audio and Fourier representations of the data.

2 Introduction

In the last decade, music has become more accessible than ever, particularly with the introduction of music streaming services like Spotify and Apple Music. Part of the appeal of these platforms is not just being able to play the music consumers know and like, but also discovering new music based on the consumer's preferences. To this end, many strides have been taken in the field of Music Information Retrieval (MIR), which is the multi-disciplinary science of retrieving information from music, where music genre classification remains one of the classic research topics.

While differentiating between punk rock and electronic-disco music is a trivial task for humans, automatic classification of music genres remains a non-trivial task in machine learning. The application of Convolutional Neural Networks (CNNs) has proved to be a successful approach to solving the genre classification problem, and they typically utilize 2-dimensional CNNs with spectrograms as input. A spectrogram is a 2-dimensional visual representation of the spectrum of frequencies of a signal as it varies with time. Recently, however, there have been several proposals for using 1-dimensional CNNs on the raw audio input itself.

In this report, we present three different networks that solve the music genre classification problem. First, we use an architecture similar to that used in image classification [3], that takes as input the 1-dimensional raw audio. Then we take the more traditional approach of an architecture with skip layers that uses the 2-dimensional spectrogram representation of the audio files. Finally, we conclude with a newer approach of using both the 1-dimensional and 2-dimensional outputs of the previous models combined together before being sent to a fully connected dense layer [5].

3 Dataset

3.1 Dataset Description

The project was developed using the Free Music Archive music dataset, an open-source dataset used for evaluating a variety of tasks in MIR. The entire dataset is comprised of 917 gigabytes of Creative-Commons-licensed audio from 106,574 tracks, arranged in a hierarchical taxonomy of 161 genres [4]. For the purposes of this project, we just used the small version of the dataset, comprising of 8,000 tracks and 52 genres. Furthermore, we use only the 8 top-level genres for classification, which include the following genres: Hip-Hop, Rock, Pop, Folk, Experimental, International, Electronic, and Instrumental. The 8,000 tracks are split 80/10/10 for the training, validation, and test sets respectively, and each maintains an even balance of genres.

3.2 Data Preprocessing and Preperation

While extensive and for the most part ready to use, the FMA dataset required some preprocessing before being ready to pass through our models. First, we iterated through each track to remove corrupted tracks that were unable to be loaded, in total discarding eight tracks. After being sure the tracks could at least be loaded in, and before being passed through the preprocessing pipeline, the audio files were cut in half to lower the computation demand.

In the audio preprocessing pipeline, we first employ some standard data augmentation techniques, namely the addition of random Gaussian noise and time stretching. Data augmentation has been shown to decrease the likelihood of overfitting and generally helps improve the accuracy of models [2]. After the data augmentation on the raw audio itself, we move on to normalization. First, we transformed the audio signals into their respective Mel spectrograms. Each Mel spectogram is extracted using the Python library librosa, with its respective original frequency, 2048 FFT points, 128 bins, and a hop length of 528. Then we used z-score normalization for both the 1-dimensional audio signals and 2-dimensional Mel spectrograms, where in the latter we normalize across each frequency band.

4 Methodology

4.1 1-D Raw Audio Model

The first model we implement is that of the 1-dimensional model using raw audio signals as input. The model, proposed by Marcel Lederle and Benjamin Wilhelm, consists of 4 blocks, each containing 2 1-dimensional convolutional layers followed by a max pooling layer, a batch normalization, and a ReLU activation. In each consecutive block, the kernel size decreases by 2 while the filter size increases by a factor of 2. After the 4 blocks, we flatten the signal and send it to a fully connected dense layer, where the output is softmaxed to predict from the 8 genre labels. The model is represented in Figure 1.

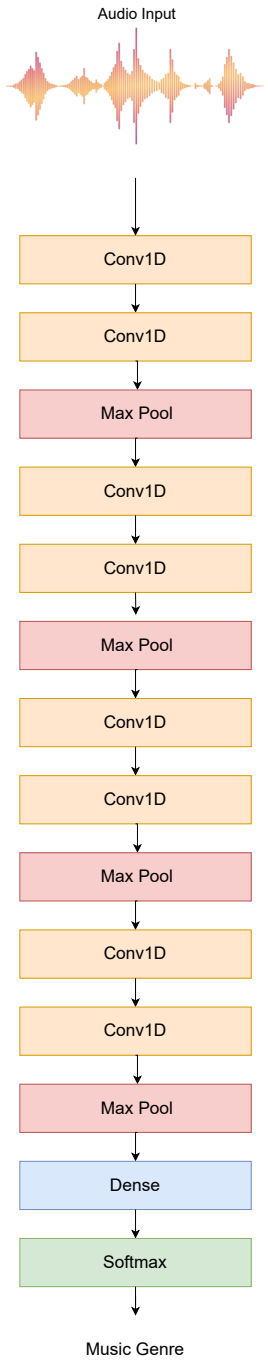

FIGURE 1: 1D CNN Model

4.2 2-D Spectrogram Model

Next, we implement the baseline model for the 2-D spectrograms. In this model, we employ an architecture with skip layers after the main convolutional block. After the residual block, there is a global max pooling and average pooling that are concatenated together before being based through a fully connected dense layer, much like in the 1-dimensional model [6]. The model is represented in Figure 2.

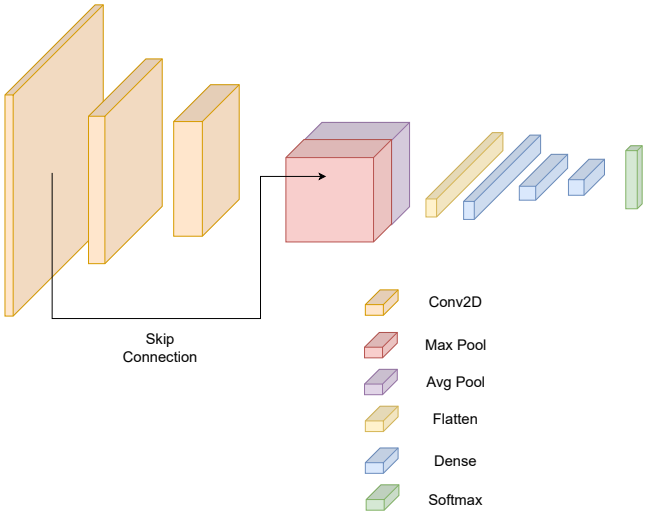

FIGURE 2: 2D CNN Model

4.3 Combined Model

Finally, we implement a baseline combined model that utilizes both previous models. For both the 1-D and 2-D models, we remove the final dense and softmax layers and instead concatenate the outputs of the convolutional blocks together. These concatenated features are then sent to a final fully connected dense layer. The model is represented in Figure 3.

4.4 Implementation

All preprocessing and model creation was done using Pytorch in Python. For all three models, we used the Adam optimizer with a learning rate of 1e-3 and a weight decay factor of 1e-4 for the 1-D model and 1e-5 for the 2-D and combined models. We used a batch size of 16 and trained for 25 epochs. We employed the cross-entropy loss function and trained on the GPUs available in Google Colab. In all models, we use a combination of batch normalization and dropout to avoid overfitting.

5 Evaluation

5.1 Metric

We use a variety of metrics to test and compare our models. Namely, we use accuracy, precision, recall, and F1-score, which are defined as follows [1]:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

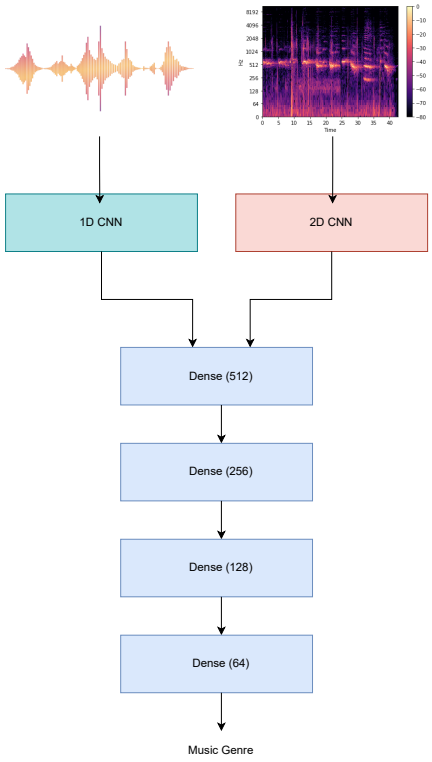

FIGURE 3: 1D CNN Model

$$Recall = \frac{TP}{TP + FN}$$

$$F1Score = 2\frac{Precision * Recall}{Precision + Recall}$$

where TP, TN, FP, and FN are the true positive, true negative, false positive, and false negative rates respectively. As we can see, accuracy refers to how many of the data instances are correctly classified over the total number of data instances. Since our data is entirely balanced, accuracy can be used as a reasonable metric. Meanwhile, precision refers to the positive predictive value in classifying data instances and recall refers to the true positive rate, both of which should ideally be close to 1.

5.2 Results

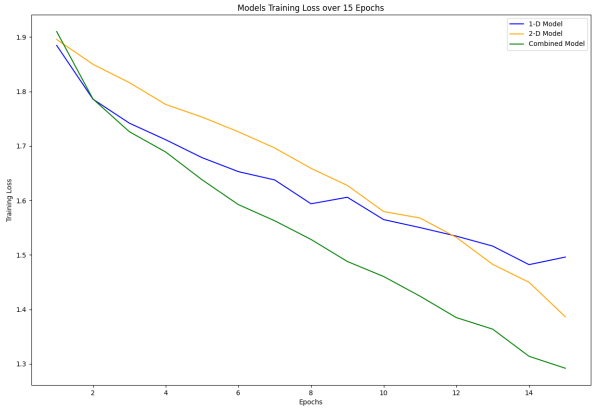

FIGURE 4: Training Loss over Epochs

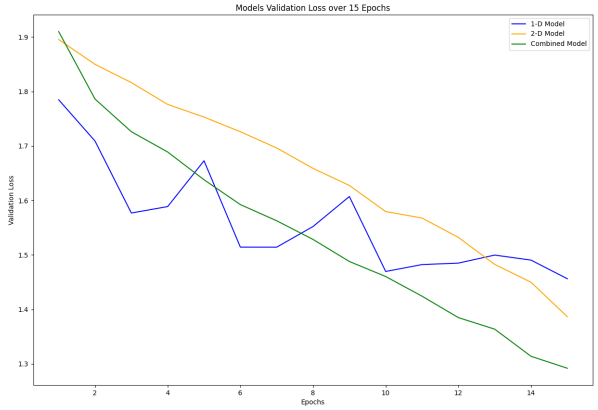

FIGURE 5: Validation Loss over Epochs

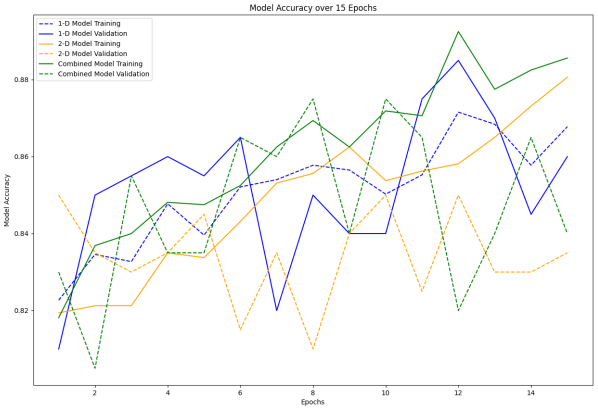

FIGURE 6: Model Accuracy over Epochs

From the results in Table 1, we can see that the best-performing model was the 1-D CNN model, followed by the combined CNN model, and ending with the 2D CNN model. Based on the plots of the model validation loss compared to the respective training loss, we can conclude that both the 2-D model and combined model experienced overfitting, and that is likely why they were not as accurate as the 1-D model.

While the accuracy rates of the models are all above 80 percent, the other metrics show other key insights. Namely, both the precision and recall rates are lower. What this indicates is that the accuracy rates are likely elevated due to the high rates of true negatives, rather than true positives. While having a high accuracy is indeed desirable, subsequent applications of music genre classification would likely depend on higher precision and recall from the model. A successful application, like Spotify's music recommending engine, requires the ability to determine music genres with high precision so that any subsequently recommended music is indeed within

Model	Accuracy	Precision	Recall	F1-Score
1D CNN	0.89	0.69	0.69	0.69
2D CNN	0.85	0.67	0.67	0.67
Combined CNN	0.86	068	0.68	0.68

TABLE 1: Model Performance on Respective Test Sets

the same genre of a user's preference.

6 Conclusions

In this project, we implemented three baseline models: a 1-D convolutional neural network using raw audio signals as input, a 2-D neural network using the Mel spectrogram representations as input, and a convolutional neural network that uses both representations of the audio together. We found that all three models hold a relatively high level of accuracy, with the 1-D model achieving the highest accuracy level, and the other two models slightly overfitting but still achieving moderately acceptable results. Any further work on this project, besides reducing the effects of overfitting on the last two models, would be to focus on achieving better performance in both precision and recall rates. One such approach could be to build deeper networks that would more effectively leverage the power of residual neural networks.

References

[1] MS Windows NT kernel description. Accessed: 2023-07.

[2] Asith Abeysinghe, Sitthichart Tohmuang, John Laurence Davy, and Mohammad Fard. Data augmentation on convolutional neural networks to classify mechanical noise. Applied Acoustics, 203:109209, 2023.

[3] Safaa Allamy and Alessandro Lameiras Koerich. 1d cnn architectures for music genre classification, 2021.

[4] Michaël Defferrard, Kirell Benzi, Pierre Vandergheynst, and Xavier Bresson. Fma: A dataset for music analysis, 2017.

[5] Marcel Lederle and Benjamin Wilhelm. Combining high-level features of raw audio waves and mel-spectrograms for audio tagging, 2018.

[6] Weibin Zhang, Wenkang Lei, Xiangmin Xu, and Xiaofeng Xing. Improved music genre classification with convolutional neural networks. pages 3304–3308, 09 2016.