From Liszt to Justin Bieber: An Audio-Based Music Genre Classifier

Author Yuelong Li, Author Olivia Wang, Author Yuting Wang

Abstract

Convolutional Neural Network is one of the most robust tools in dealing with images in machine learning and deep learning tasks. This project focuses on the implementation of music genre classification based on Convolutional Neural Networks. Good music genre classifier helps with music recommendation systems and automated genre classification for music APPs, and it may facilitates music composition. After processing the original audio signals into Mel Spectrogram, this project constructs the CNN structure based on pytorch and improves the model using autoencoder. After 30 epochs of training, the CNN-based classifier obtained a performance with a test accuracy of 50% and can successfully classify a music's genres by percentage (e.g.[Jazz 30%, Rock 70%]).

1 Introduction

Various Music Applications provides multiple choices for music lovers. Music recommendation systems using Machine Learning models can predict users' taste and feed them with the music they would possibly love. As determining genre is the first step of music recommendation system, better model could be built based on a good genre classifier.

The inspiration of this problem comes from our group members' interest: listening to music. We want to choose among different Machine Learning algorithms. Ideally, precisely classify different music into 10 genres: Classical, Hip hop, Country, Rock, Metal, Blues, Pop, Jazz, Reggae,and Disco. In a journal called "Music Genre Classification with Python", Professor Parul Pandey elucidates the robustness of CNN by highlighting that "Single-layer Neral Networks trained with back-propagation algorithm constitutes an excellent example of a successful Gradient-Based Learning technique" (1). Compared with the original audio wave signal recognition, this project corresponds to a single-class classification problem by using the converted data: Mel Spectrogram, which makes the audio signal recognition possible to deal with and generates a more precise result.

2 Dataset

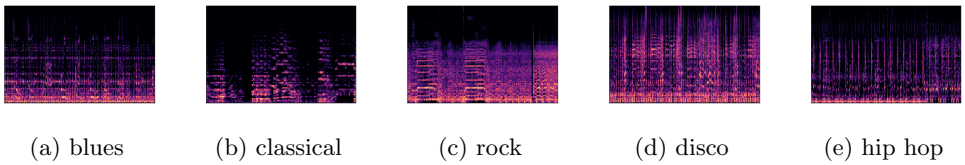
2.1 Dataset Overview

For dataset selection, we chose the GTZAN dataset from Kaggle. It is a benchmark dataset that was used in over 100 published articles. It includes 1000 sound excerpts of 30 sounds. Each audio file has a corresponding image file, which is a visual representation of Mel Spectrogram. All music are classified into 10 common music genres: Classical, Hip hop, Country, Rock, Metal, Blues, Pop, Jazz, Reggae and Disco.

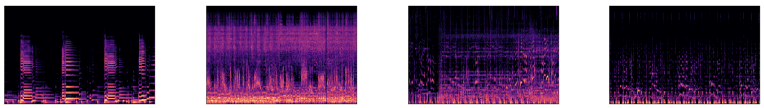
Genre	Classical	Hip hop	Country	Rock	Metal	Blues	Pop	Jazz	Reggae	Disco
Size	100	100	100	100	100	100	100	100	100	100

2.2 Mel-Spectrograms

Mel Spectrograms are two-dimensional graphical representations of audio signals. Mel Spectrogram differentiates from regular spectrogram by its frequency spacing on y-axis. This spacing better approximates the hearing scale for human ears–where lower frequencies are emphasized and higher frequencies are compressed. Below are the graphs for different genres in the GTZAN dataset.



(a) blues (b) classical (c) rock (d) disco (e) hip hop



(f) jazz (g) metal (h) pop (i) reggae

Figure 1: Mel Spectrograms of Different Genres

3 Solution

3.1 Preparing data

3.1.1 Packing data

We first read data from the .png files into corresponding matrices and labels. The initial matrix representation of the spectrograms are of sizes $288 \times 432 \times 3$, with the last dimension representing the RGB channels. These data matrices, amounting to several gigabytes, however, are too large to be handled swiftly, during the training phase, so we needed to reduce the data sizes using encoding.

The method for encoding that we ended up using is average pooling. By sliding a averaging window of 4 by 4 across the $288 \times 432 \times 3$ image with step size 4, we convolve it down to the shape of $72 \times 108 \times 3$, effectively reducing its size from 839 KB to 23 KB. The compression rate of 16 times can prove very important during the training stage. As to the information loss, the MSE loss is measured to be less than 0.01, and visually the compressed spectrogram still looks quite similar to the original.

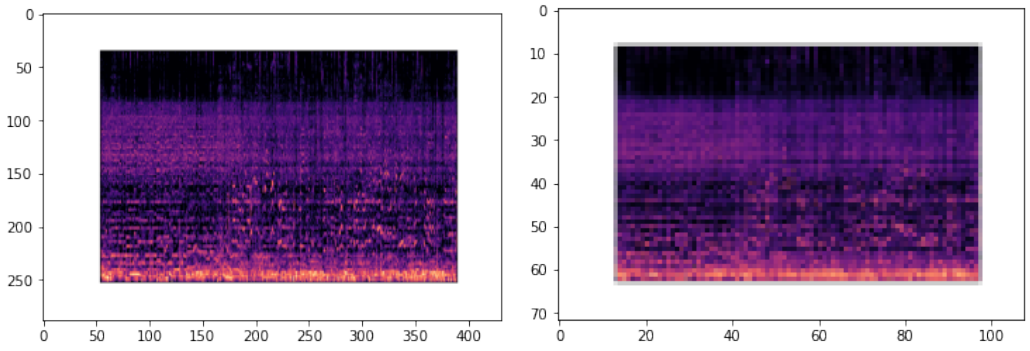


Figure 2: Original data vs encoded data

Since the music excerpt data aren't very abundant, we decided to use the training set and the validation set interchangeably. We split the pictures into train and test sets with respective sizes 800 and 200, randomly sampled from the encoded dataset. We then further transpose the matrices in each data set to match the order of (channel, height, width) and convert them into tensors for compatibility with PyTorch. The final result is two arrays of tensors each with size 800*3*72*108 and 200*3*72*108.

3.2 Convolutional Neural Network

We chose convolutional neural network, and the reason behind is very intuitive. CNN is very commonly used for image classification problems. By looking at the Mel Spectrogram data, we found that different genres have recognizably different patterns, even though human are not able to classify the genres. It seems like Mel Spectrograms should offer enough information to determine the music genres.

For structure of our CNN Model, we refer to LeNet-5 structure proposed by Yann LeCun. However, because the image size is different from the model. We changed the LeNet model to better fit our data. The final network structure is an transformed model from LeNet. Our CNN model consists of two parts:(i) a convolutional encoder consisting of two convolutional layers; and (ii) a dense block consisting of two fully-connected layers; The architecture is summarized in figure 3.

We designed our CNN model specifically so that it can handle the rectangular shape of the initial inputs in the convolutional layers, and we added only one fully connected linear layer at the end because this turned out to be the structure that gives the fastest training convergence and results as opposed to deeper models.

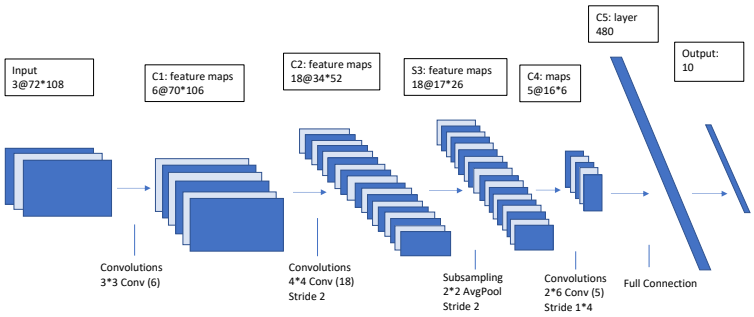


Figure 3: CNN Model Architecture

3.3 Training

3.3.1 Loss function

To define the loss function, we used cross entropy, defined as:

$$l(f, y) = \sum_{1 \le i \le n}^{1 \le j \le 10} I_{y_i}(j) \log(f(x_i)_j).$$

Where f is the model, X_i representing the matrix of the $i-$th spectrogram, and $f(x_i)$ gives the 10 element vector corresponding to the probability of the song being one of the each genre, and y_i gives the label corresponding to the $i-$th data. The cross entropy in this case has the special advantage of rewarding the clearly distinguished predictions, and it's also positive definite with exactly matching distributions corresponding to its minimum.

3.3.2 Batch Training

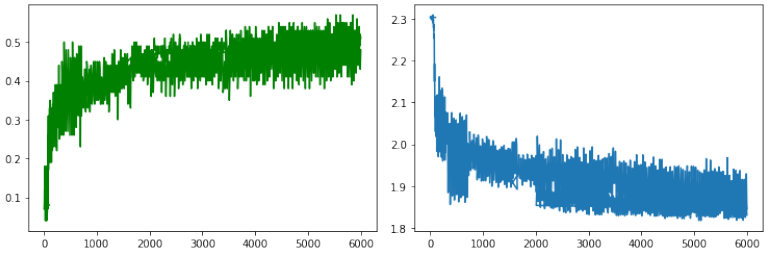
For the training part, it turned out that applying the gradient descent on our entire training set of 800 music excerpts would still take an extremely long time. So we decided to use batch training, by separating the data set into smaller batches for each epoch. The training set was thence sampled into 8 batches each with size 100. We used gradient descent with momentum of 0.9, and trained for 1800 epochs. During the training process, we also constantly reduced the learning rate after a few hundred epochs, to adapt to the lowering training loss as the parameters approach the minimum point. We used learning rates as follow:

1. 0-100 epoch: 0.01
2. 100-150 epoch: 0.006
3. 150-200 epoch: 0.005
4. 200-300 epoch: 0.004
5. 300-1300 epoch: 0.0045
6. 1300-1800 epoch: 0.003.

Eventually, we were able to reduce the training loss, measured by cross entropy, from 2.3 to 1.83, and increase the training accuracy from 0.08 to 0.62, namely 62 percent. The test accuracy was increased to 0.45 (45 percent).

Sample output of classification:

```
[['metal 0.0%', 'disco 100.0%', 'pop 0.0%', 'classical 0.0%',
'rock 0.0%', 'blues 0.0%', 'hiphop 0.0%', 'reggae 0.0%', 'country 0.0%', 'jazz 0.0%']]
['disco']
disco00026.png
```



(a) Training accuracy (b) Training loss

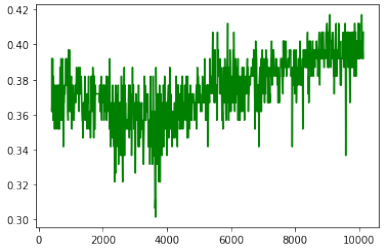
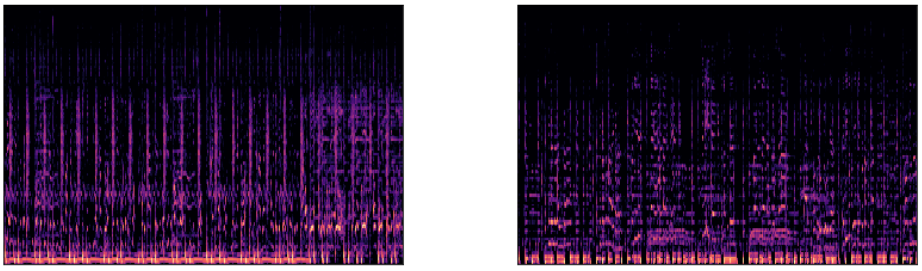


Figure 5: Testing accuracy

4 Results and Discussion

4.1 Results and conclusion

From the confusion matrix, the model performance is genre-dependent. For genres like Metal and Classical, it achieves a high accuracy, potentially because Metal has very dense sound tracks that are easily distinguishable, while what makes classical music rather identifiable remains unclear. One of the observable flaw is that the neural network always mistakens hiphop as reggae even after repeated training. It could be explained by the fact that some genres of music have similarity regarding rhythm and melody, and we could indeed see the similarities in their respective spectrograms. Potential solutions to this could be training the neural network on the original spectrogram data that hasn't gone under compression, however this would require significantly longer training time and more memories than what usual computers can supply. We could also make improvements by feeding the neural network other sources of outputs that capture more details about the sound tracks that spectrogram somehow omits. In general, our network performs well in terms of accuracy in predicting the top two possibilities of genre.



(a) hiphop00003 (b) reggae00004

Figure 6: Comparison between two randomly sampled spectrograms of hiphop and reggae

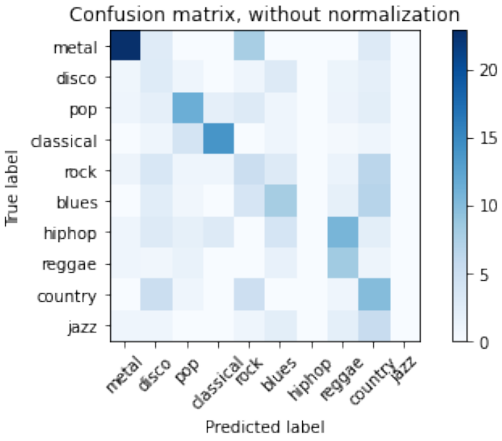


Figure 7: Confusion Matrix

4.2 Limitations

Our model's training accuracy is 50%, and testing accuracy is only 40%, which is lower than the expectations. However, if taking the top two categories to be the output, the accuracy can reach 70%. Thus, if the goal is to suggest a subset of categories that the music belongs to, the results could be more reliable. Since our training data size was reduced from 839 KB to 23 KB through the convolution, for the purpose of training time reduction, information loss was inevitable, which caused prediction to be less accurate. Using more memory or offloading the computation to GPU with training hours the model could possibly solve this problem. Although the predictions are average compared to other models, the results as percentages for each genre are useful for further recommendations. Since music can have more than one genre, or multiple music elements from different genres, this model provides useful result when it comes to multi-label music genre classification.

4.3 Future Work

Regarding the future work, the first part includes solving the problems as described above. To improve the current model, one way is to use auto-encoder. Using auto-encoder instead of the average pooling currently used for data pre-processing, we may further reduce the information loss when compressing data by capturing the non-linear and non-local relationships embedded in the patterns of the Mel-Spectrogram data. A propossed model would be to flatten the 288*432*3 individual matrices, put them through several fully connected linear layers with tanh activation function, reducing the number of nodes down to 72*108*3, and bring the dimension back up with more linear layers. In the end, transpose the data to the 288*432*3 tensor as original and compute the MSE loss. With this implementation, we could potentially improve the training accuracy by capturing the nuances in the Spectrograms that would necessarily require higher image resolution for the input.

Also, we can add more amount of data to train our model, and we may split the data into three sets, such as train sets, validation sets, and test sets. Furthermore, the data form should not be restricted to only one genre that is Mel Spectrogram because when we use Mel Spectrogram as the only input, two genres, which are hip-hop and reggae, will have very high similarity. Thus, other forms of data input can better distinguish hip-hop from reggae. The dataset of greater various forms will be included for the further training and optimization. This part would perfect the existing experimental structure along with a better result.

On a more practical level, we could implement the current version of our neural network for tasks that it is acute in. For instance, since the model is good at classifying music genres, the team will extend its performance to multiple applications(e.g music recommendation systems, automated genre classification for music apps, facilitating music composition, and so on). We also plan on building a server based version of the neural network, so that people can conveniently upload their audio-tracks to a web-based AI to identify its genre.

5 Conclusion

The experiment demonstrates the outstanding performance of Convolutional Neural Network structures in analyzing pictures and carrying out different tasks. Considering that audio files are intrinsically time sequences, it may also be beneficial to utilize RNN to generate more precise predictions of genre.

References

[1] V. Pathak and L. Iftode, "Byzantine fault tolerant public key authentication in peer-to-peer systems," *Computer Networks*, vol. 50, no. 4, pp. 579–596, 2006. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2005.07.007

[2] R. Guerraoui, "Genuine atomic multicast in asynchronous distributed systems," *Theoretical Computer Science*, vol. 254, pp. 297–316, 2001.