BitByteBeat: The Music Genre Classification Bot

Group 15: Siu Ye [40032209], Johnny On [40137434],
Chelsie Ng [40071692], Tyler Shanks [401307528]

Machine Learning Course Project
COMP 432
Andrew Delong

Gina Cody School of Computer Science and Software Engineering
Concordia University
Canada
December 6 2021

1 Abstract

We perform music genre classification manually using our own understanding of music. Methods that take into account conventional algorithmic approaches have not yet been developed since the distinctions between genres tend to be subjective and ill-defined. By observing and making predictions based on these ill-defined patterns, machine learning can observe and make predictions given enough audio data, of which large amounts can easily be harvested from online music. In this project, we aim to develop a music genre classifier that will accurately predict a song's genre by using a deep learning approach. Specifically, we will train two different models using two types of neural network architectures, namely, Convolution Neural Networks (CNN) and Recurrent Neural Networks (RNN). Python is used with a machine learning package: Tensorflow.keras to build our two models. We will compare and determine our best model through each of their accuracies.

2 Introduction

2.1 What are music genres?

Music genres are essential tools for improving our understanding and enjoyment of music. A genre can be a powerful tool when searching for similar music or to highlight innovative musicians. As a tool for understanding and discussing artists' creations, genres are important because they are among the few and most valuable we have available. This way of categorizing sounds is flexible and descriptive, which can dramatically increase our ability to comprehend, recognize, and enjoy music. There are two ways to define a genre: the density of similar characteristics and the sparsity of distinctive features: dense clusters of productions form a music style, whereas pronounced differences define its edges [1]. Nevertheless, there are gray areas in the classification of music genres that may cause overlap between categories [2]. This is true whether reviewing a broad range of genres or an individual artist's discography.

2.2 Can we detect music genres?

In spite of our varying levels of musical knowledge, it is very hard to explicitly define what a music genre a song belongs to just with our human ears [2]. What makes jazz sound like jazz? And how can you tell the difference between hip hop and the blues? Computers or machines can use machine learning to build interpretive patterns based on data that is fed into them. Based on the data that is fed into the computer, machine learning can automatically create analytical models [3]. The power of machine learning principles comes into play when it comes to extracting music signatures from each genre [4]. We are then able to extract trends and patterns from a large dataset [4]. A number of techniques have been developed to help classify music genres based on how computers are used as new music equipment is being developed and deployed into our daily lives.

2.3 How to achieve our goal?

Our main goal is to be able to make accurate predictions on a song's genre. To achieve this, we will explore ways on how neural network architectures and splitting the training and testing data among groups affect our results and see if there are any benefits to do so. Python is used because there exists a collection and code stack of various open-source repositories and it is simple to understand [5]. The Librosa Python module analyzes audio signals and is designed to be used for tracking music signals [6]. It includes all of the parts to build a Music information retrieval (MIR) system. This module is very well documented and includes numerous examples and tutorials. Tensorflow is a Python-based scientific computing package. It is an open-source machine learning library used with the keras library designed to handle automatic differentiation libraries, which is helpful for implementing neural networks [7].

2.4 Which dataset will we use?

The project's data uses the GTZAN dataset [8] most frequently for evaluation. This dataset set includes a variety of recording-related sources (personal CDs, radio, microphone recordings), files were collected from 2000-2001. In particular, there are 10 genres, each with 100 audio files, all lasting 30 seconds each [8]. Using this dataset, we extract audio features, namely Mel-frequency Cepstral Coefficients (MFCC), from the raw data using the Librosa library. MFCCs are a small set of features that summarize the spectral envelope (often described in terms of timbre) [9]. They are frequently used in MIR. Since we cannot directly use the audio file as an input for our models, we need to preprocess it and then use the data in the GTZAN dataset. This means extracting useful features from the audio signal. Therefore, MFCC feature extraction is one of the ways to extract useful information from the signal because it defines the brightness of a sound [9]. It can also be used to calculate the timbre (quality) of the sound [10]. Then, a training set with category labels is provided for the purpose of training and evaluating the performances of the classifiers. A test set with no category labels is given to test the trained classifiers against unseen data.

Everything will be and can be run from a Jupyter notebook. The initial step is to import all libraries needed for our project, regarding music genre classification. From the GTZAN dataset, we are able to load all songs in it. Once the dataset is loaded and saved via the dataset path, save_mfcc is defined to perform the MFCCs extraction and save them into a .json file along with their respective genres. We divided each sample songs into different segments of 30 seconds and create our MFCCs. This ensures we have even more data to play around with. Once the data is ready and has been preprocessed, we can create both of our models for music genre classification.

2.5 Which models best suit our case?

Convolution neural network is based on visual images. Its algorithms predict future datasets based on how they process and interpret visual images. CNN image classifications take an input image, process it and classify it under certain categories [11]. Computers see an input image as an array of pixels and it depends on the image resolution. By analyzing images and recognizing information in images, machines can successfully analyze and classify future data [12]. A CNN can classify music genres through the study of spectrograms (visual graphs of musical frequencies) which, in turn, enables machines to identify the category of particular types of music [11]. Each input image passes through a series of convolution layers with kernels, pooling, fully connected layers and applies a softmax function to classify an object with probabilistic values between 0 and 1.

Recurrent neural network learns how to remember sequences of information. Their unique characteristic is their memory, which draws on information from past inputs to influence current inputs and outputs [11]. A pattern usually emerges when dealing with audio information from songs. These deep learning algorithms are commonly used for ordinal or temporal problems [12]. As far as the RNN is concerned, it is a little different from traditional neural networks. It stores and uses the past data as information to predict future outcomes [11]. The output from recurrent neural networks depends on the previous elements within the sequence, unlike traditional deep learning models which assume inputs and outputs are independent.

3 Methodologies

3.1 Preprocessing

An MFCC is a 2-D array. One dimension represents time while the other dimension represents the different frequencies. By analyzing the .json file created from the save_mfcc method, we can observe the following. There are 9986 mfccs in total where each mfcc is mapped to a specific music genre. A genre is labelled as an integer from 0 to 9 because there are 10 different genres in our dataset (i.e. ['pop', 'metal', 'disco', 'blues', 'reggae', 'classical', 'rock', 'hiphop', 'country', 'jazz']).

Also, by looking at the distribution of our genres, we can see that it is evenly distributed. This is because the dataset contains 100 sample songs for each music genre. Note that, one song was removed in the jazz genre as jazz0054.wav was corrupted and could not be read. We can observed the genre distribution through our plot_histogram function.

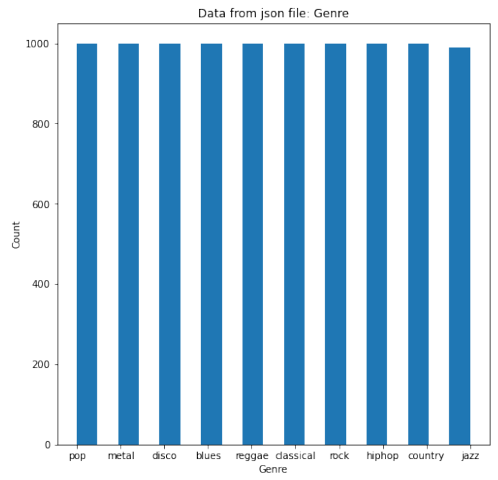


Figure 1: Distribution of music genre data

Also, the visual representation of each genre through a spectrogram which shows how quickly the frequencies themselves are changing over time. The first axis is frequency while the second axis is time.

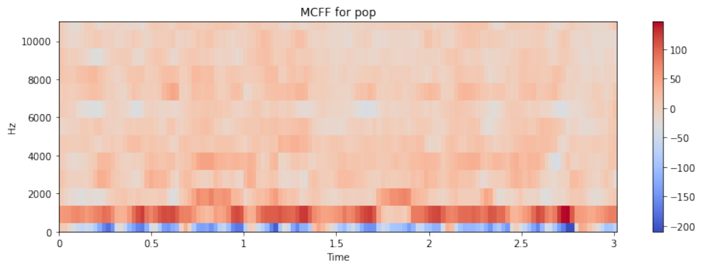


Figure 2: MFCC example

3.2 Data Splitting

To have the data ready for training, we randomly split $(\boldsymbol{X}, \boldsymbol{y})$ into three parts, with no overlap. We separate $\boldsymbol{X}$ and $\boldsymbol{y}$ into 60% training, 20% validation and 20% testing.

3.3 Training with CNN

Our CNN model uses multiple layers. It is of sequential type. The input goes through these layers (specifically three convolution layers, one dense layer and one output layer) and at the end, the result gets flattened out and normalized into either one of ten possible results. Once the model has been created, the test data is fitted into it. When observing testing accuracy, we obtain the following result: 72.37%

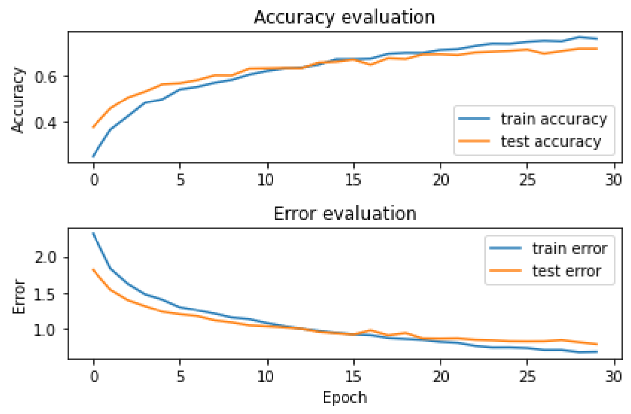


Figure 3: CNN accuracy/error evaluation

3.4 Training with RNN

Our RNN model uses multiple layers as well. It is of sequential type. The input goes through these layers (two LSTM layers, one dense layer and one output layer). The result gets flattened out and normalized into either one of ten possible results of music genres. Once the model has been created, the test data is fitted into it. When observing testing accuracy, we obtain the following result: 59.11%

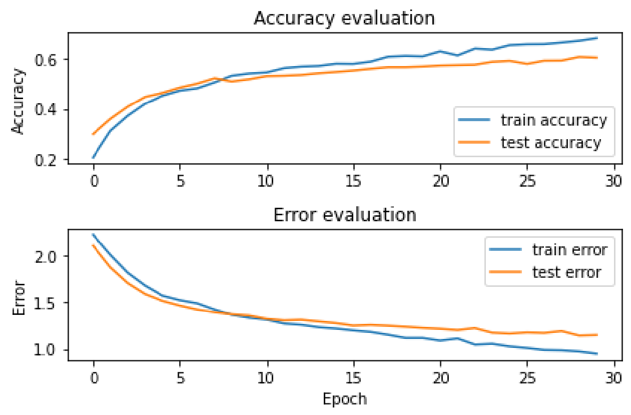


Figure 4: RNN accuracy/error evaluation

4 Evaluation

In this section, we will draw out conclusions from the models trained and compare them. As seen in section 3.3 and 3.4, we can conclude that our best model is CNN with 72.37% accuracy.

To further evaluate the models, we plotted their confusion matrix and classification report.

4.1 Confusion Matrix

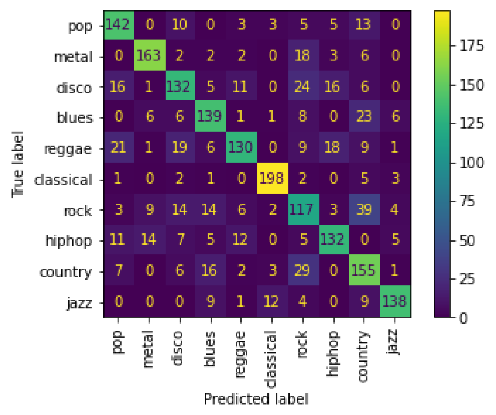


Figure 5: CNN confusion matrix

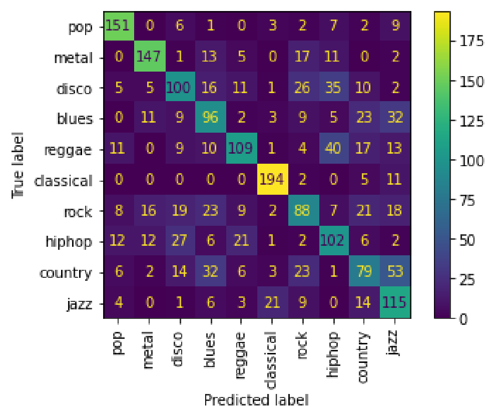
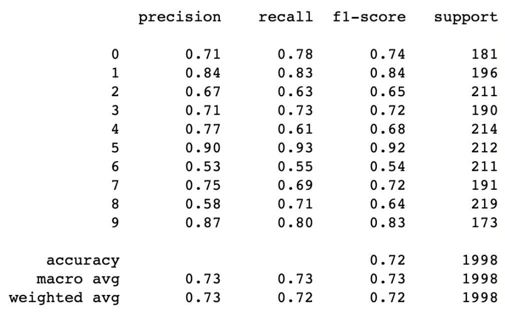


Figure 6: RNN confusion matrix

As we can see, from figure 5 and 6, the CNN made less errors than the RNN. Both has more accurately predicted classical music. Our CNN model misclassified rock as country the most while our RNN model misclassified country as jazz the most.

From the diagonal of both confusion matrix, the CNN model performed way better.

4.2 Classification Report


```
              precision    recall  f1-score   support

           0       0.71      0.78      0.74       181
           1       0.84      0.83      0.84       196
           2       0.67      0.63      0.65       211
           3       0.71      0.73      0.72       190
           4       0.77      0.61      0.68       214
           5       0.90      0.93      0.92       212
           6       0.53      0.55      0.54       211
           7       0.75      0.69      0.72       191
           8       0.58      0.71      0.64       219
           9       0.87      0.80      0.83       173

    accuracy                           0.72      1998
   macro avg       0.73      0.73      0.73      1998
weighted avg       0.73      0.72      0.72      1998
```

Figure 7: CNN classification report

```
              precision    recall  f1-score   support

           0       0.77      0.83      0.80       181
           1       0.76      0.75      0.76       196
           2       0.54      0.47      0.50       211
           3       0.47      0.51      0.49       190
           4       0.66      0.51      0.57       214
           5       0.85      0.92      0.88       212
           6       0.48      0.42      0.45       211
           7       0.49      0.53      0.51       191
           8       0.45      0.36      0.40       219
           9       0.45      0.66      0.53       173

    accuracy                           0.59      1998
   macro avg       0.59      0.60      0.59      1998
weighted avg       0.59      0.59      0.59      1998
```

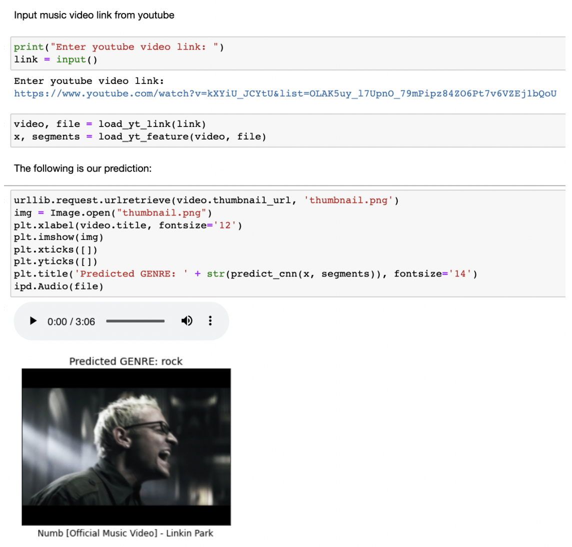
Figure 8: RNN classification report

Looking at the classification report from figure 7 and 8, our CNN model have higher precision, recall and f1-score. Our CNN model has on average 0.73 for all three metrics while our RNN model has 0.59, 0.60, 0.59 respectively.

To conclude, the CNN model had better precision and recall than the RNN model.

5 Conclusion

Finally, we are quite satisfied with the accuracies achieved. We were also able to create an interactive way for a user to use our model. The user would be able to copy and paste a YouTube link to the jupyter notebook itself and we would predict its genre with our best model (see figure 9).



Input music video link from youtube

```
print("Enter youtube video link: ")
link = input()
```

```
Enter youtube video link:
https://www.youtube.com/watch?v=kXYiU_JCYtU&list=OLAK5uy_l7UpnO_79mPipz84ZO6Pt7v6VZEj1bQoU
```

```
video, file = load_yt_link(link)
x, segments = load_yt_feature(video, file)
```

The following is our prediction:

```
urllib.request.urlretrieve(video.thumbnail_url, 'thumbnail.png')
img = Image.open("thumbnail.png")
plt.xlabel(video.title, fontsize='12')
plt.imshow(img)
plt.xticks([])
plt.yticks([])
plt.title('Predicted GENRE: ' + str(predict_cnn(x, segments)), fontsize='14')
ipd.Audio(file)
```

Figure 9: YouTube prediction

In the event that we plan to further improve BitByteBeat, we agreed that we would require substantially more training data, both in the form of more genre types as well as overall dataset size. Genres change over time and our dataset currently has old songs. A major improvement would be to update our dataset with more recent songs. With this we could classify a larger set of songs more accurately.

In addition we also discussed the prospect of training it with other models like KNN or adding other audio features like volume, energy, pitch, etc along with the MFCC. We could also try applying feature selection techniques like Principal Component Analysis to identify optimal features to further improve results.

We have also looked into switching from the Keras framework of Tensorflow to the PyTorch framework. It could have been pleasant to compare the training speed of the model between the two frameworks and potentially see one of them being faster. It could have been interesting to analyze which model is better through its accuracy and testing between both CNN and RNN models depending on the framework used.

That being said, we can only talk about the 'what if' and 'why not' now because of all the knowledge we gained throughout the project. We indeed learned a lot and are very happy to have reached this level of completion. More specifically, it was really fun to try out our model on different YouTube songs and have a good laugh about it. It was a great introduction to Machine Learning for all of us!

References

[1] K. Kosina, “Music genre recoginition.” http://www.music.mcgill.ca/~ich/classes/mumt611_08/similarity/KosinaMusicGenreRec.pdf, 2002. (Accessed Nov. 26, 2021).

[2] N. Ndou, R. Ajoodha, and A. Jadhav, “Music genre classification: A review of deep-learning and traditional machine-learning approaches.” https://ieeexplore.ieee.org/document/9422487/authors#authors, 2021. (Accessed Nov. 27, 2021).

[3] R. Adragna and Y. H. Sun, “Music genre classification.” https://www.eecg.utoronto.ca/~jayar/mie324/musicgenre.pdf, 2018. (Accessed Nov. 27, 2021).

[4] L. Gui, Z. Gu, and T. Liu, “Music genre classification via machine learning.” http://cs229.stanford.edu/proj2017/final-reports/5244969.pdf, 2017. (Accessed Nov. 27, 2021).

[5] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.

[6] C. R. D. L. D. P. E. M. M. E. B. McFee, Brian and O. Nieto, “librosa: Audio and music signal analysis in python.” https://librosa.org/doc/main/index.html, 2015. (Accessed Nov. 14, 2021).

[7] M. Abadi and al., “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015. Software available from tensorflow.org.

[8] A. Olteanu, “Gtzan dataset - music genre classification.” https://www.kaggle.com/andradaolteanu/gtzan-dataset-music-genre-classification/activity, 2020. (Accessed Nov. 14, 2021).

[9] M. Haggblade, Y. Hong, and K. Kao, “Music genre classification.” http://cs229.stanford.edu/proj2011/HaggbladeHongKao-MusicGenreClassification.pdf, 2011. (Accessed Nov. 27, 2021).

[10] H. C. Ceylan, N. Hardala, A. C. kara, and F. Hadalac, “Automatic music genre classification and its relation with music education.” https://files.eric.ed.gov/fulltext/EJ1302228.pdf, 2021. (Accessed Nov. 26, 2021).

[11] P. Dwivedi, “Using cnns and rnns for music genre recognition.” https://towardsdatascience.com/using-cnns-and-rnns-for-music-genre-recognition-2435fb2ed6af#:~:text=A%20spectogram%20is%20a%20visual,state%20at%20time%20t%2D1., 2018. (Accessed Nov. 14, 2021).

[12] S. A. S. a. S. Ahlam Wahdan, “Using cnns and rnns for music genre recognition.” https://www.researchgate.net/publication/349804849_Classifying_Audio_Music_Genres_Using_CNN_and_RNN, 2021. (Accessed Nov. 14, 2021).