INVESTIGATING AUDIO ARTIFICIAL INTELLIGENCE

AYSHA ARAIEN
PREET PARMAR
LETIAN REN

MATH 509

Contents

1 Introduction

1.1 Background

This project was designed to investigate the strategies that music applications such as Spotify or Apple Music would use to process and analyze audio data. When playing music from either of their products the user can notice that their "shuffle" features in-app seem to have a pattern, such as similar beats per minute (BPM) songs playing in succession or repetition of the same artist. Is this a coincidence or are there algorithms which drive this behaviour?

Thus, the motivation, or in other words, problem statement of this project is to investigate audio artificial intelligence. This will be accomplished by 2 goals:

1. Create a genre classification system using a neural network that performs at a good accuracy (60% or above).
2. Create a recommendation system using descriptive music data to output similar songs based on an input song.

These models will be able to further investigate if genres can be successfully classified and whether this information can be used to create patterns in recommending songs.

1.2 The Data

As the project unfolded, it became apparent that two separate data sources needed to be used in generating the models as the two models required different features that were not available together in a dataset that did not demand computational resources. Although the "Free Music Archive" dataset contains fields that could be used to generate both models, it is too resource intensive to run locally [2].

The "GTZAN" dataset was used for the genre classifier that uses MFCC features [7]. This is a widely used dataset for audio classification consisting of 1000 mp3 tracks labelled by genre. Using this reduced some of the preprocessing work necessary to input the data into a model as the tracks were pre-sorted into individual directories titled by genre.

A music dataset titled "Music Dataset 1950 to 2019" containing 28732 rows was used to develop the recommendation system [5]. This dataset is quite descriptive, and since the recommendation system assesses for song similarity, a dataset that contains as much information about the songs included as possible is of utmost importance.

2 Genre Classification Model

2.1 Features for Neural Network

Mel-Frequency Cepstral Coefficients (MFCC) are one of the most commonly used features in audio classification along with spectrograms. MFCC's were selected rather than spectrograms as input features in the genre classifier because they are easier to calculate and less resource-intensive to process.

To calculate the MFCC's mathematically heavy processing is involved. Although, since the Python package "Librosa" already contains all the calculations, it is used rather than manual mathematical computation. A high-level diagram of the process is shown below.

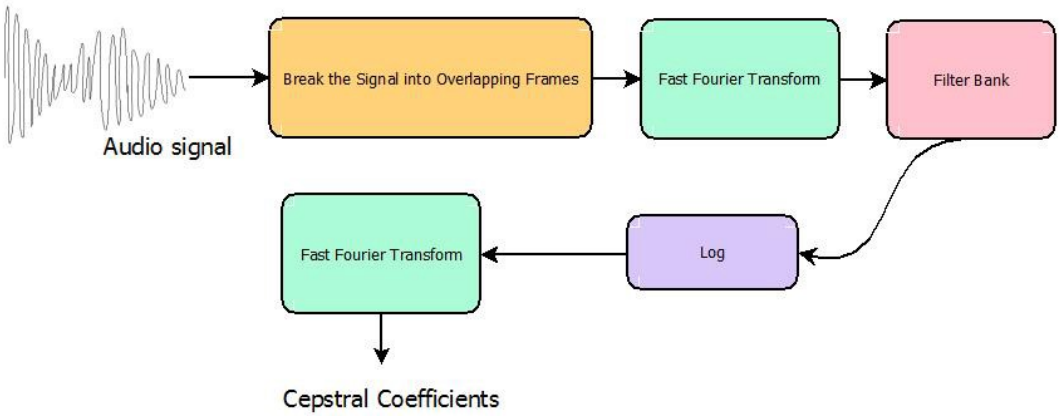

Figure 1: Calculating MFCC's [6]

2.2 Formulation of Mathematical Model

A neural network was developed using the Python package "Keras" that takes MFCC features as input. These features were split into a train and test set with a test size of 35%. The formulation of the model layers we used in the network were as follows:

- **Long-Short-Term Memory Layer** - Used as an input layer; good for sequential data
- **Dense Layer** - Changes weights according to prior layers
- **Dropout Layer** - Helps with regularization by reducing overfitting

The model accuracy and validation score were fine-tuned through an iterative approach. Layers within the network were modified with each iteration such that the accuracy of the model improved. For example; the input layer was modified from a dense layer to a long-short-term-memory layer which improved the accuracy and validation scores by $\tilde{5}$%.

Activation functions define the output of a given neuron in a layer. In a neural network, neurons will output the result of the activation function as input into the next layer. The default, a rectified linear unit (ReLU) was used as the activation function in the hidden layers. A softmax activation function was chosen for the output layer since it transforms input into a probability between zero and one [4]. Therefore, the neuron which has the highest output in the final layer will be selected to classify the song sample.

Loss functions are used to minimize the error and optimize the neural network [1]. A sparse categorical cross entropy loss function was used in the model because each sample only belongs in one class.

2.3 Solution of the Problem

The neural network was built using the Python package "Keras". Though there are guidelines in building networks, such as including dropout layers to reduce overfitting, it can be difficult to tell

which architecture will result in good accuracy. Therefore, the layers in the network were modified iteratively and the network was run multiple times with different architectures before final model selection.

The final neural network was constructed using 5 layers. An LSTM layer of 128 neurons takes in the input matrix of MFCC features (130x13 matrix of vectors). The hidden layers were dense layers of 128, 64 and 48 neurons. Dropout layers were introduced for regularization and to reduce over-fitting. A dropout layer of 0.2 was added after the input layer, and dropout layers of 0.4 were added after each of the final two hidden layers. The output layer was 10 neurons, for 10 genres to be classified. Therefore, the neuron in the final layer that produces the highest weight is selected to classify the genre of the input. The model was run for 50 epochs and results were generated by creating a confusion matrix and classification report.

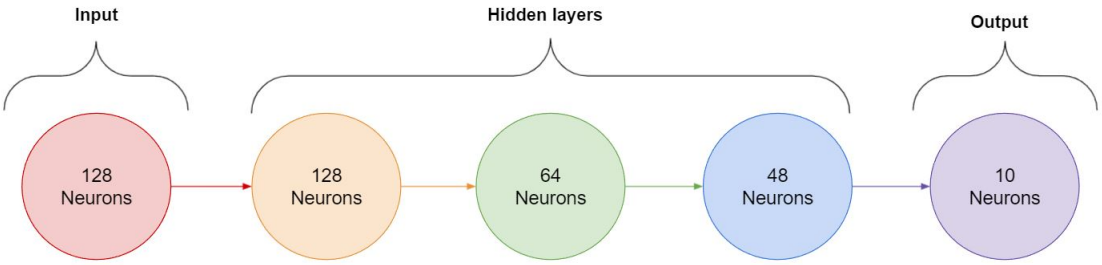

Figure 2: Model Architecture

2.4 Interpretation of Results

The genre classifier performed at a 66% accuracy in classifying the 10000 samples into genres. This result is relatively good, given the amount of data available and the number of classes (10). Genre is also quite abstract – there aren't stringent boundaries between genres. For example; some light metal songs can sound like rock songs.

As seen in Figure 3, a good amount of song samples were correctly classified across the diagonal. Interestingly, it appears from the confusion matrix that the metal song category was by far the most wrongly classified category. 59 country song samples were classified as metal song samples, and 71 reggae song samples were classified as metal song samples. Additionally, the reggae song category seemed problematic to classify. Blues and classical music were the top 2 easily classified categories as compared to the others present in the dataset.

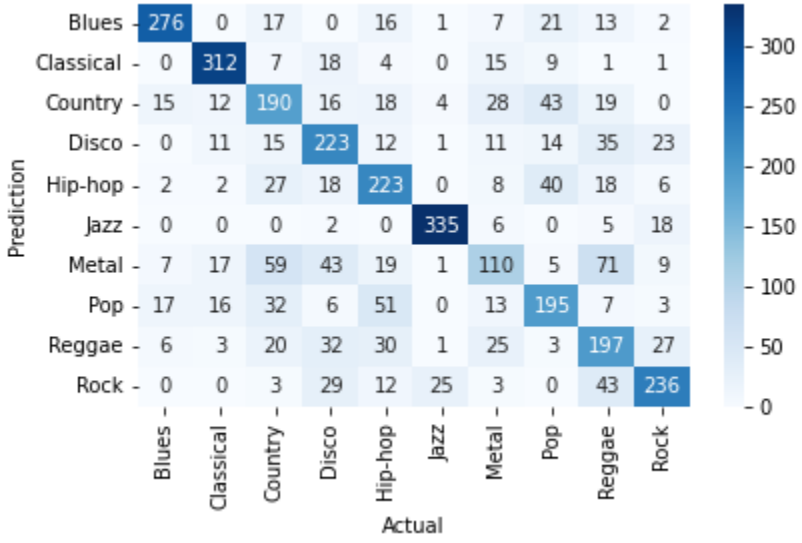

Figure 3: Confusion Matrix

The sensitivity and specificity of the model were relatively similar. This means that the model is nearly equally as likely to result in a false positive prediction as it is to result in a false negative prediction. In the context of this project, the goal should be to minimize both sensitivity and specificity. As seen from Table 1, the model accuracy was fairly good at 72%.

Model	Sensitivity	Specificity	Precision	Accuracy	Validation
Genre Classifier	66%	65%	66%	72%	66%

Table 1: Classification Report

3 Recommendation System

3.1 Features for Recommendation System

The following text fields in the "1950 to 2019 Music" dataset pertaining to songs were as follows:

- Artist Name
- Genre
- Topic - This was the theme of the song. Some examples of potential topics are happy, sad, or romantic.
- Lyrics

3.2 Formulation of Mathematical Model

The purpose of a recommendation system is to output a list of songs based a song that is inputted by the user. Recommendation systems may be constructed in a multitude of ways:

1. A list of items ranked by similarity to the input in terms of features in the dataset.

2. A list of items ranked by similarity to the input item that other user's have liked or ranked – also known as collaborative filtering.
3. A combination of both of the previous methods (1 & 2).

For this project, the first method was selected. This is because, in the context of a "smart shuffle algorithm", an output of songs that are similar to the input song in terms of descriptive features such as artist name, track title, album, etc. should be ranked by similarity and the top track in that rank-list will be the next track in a user's playlist.

3.3 Solution of the Problem

The fields from Section 2.1 were used to construct the recommendation system. Following standard protocol for NLP data processing, noise was removed from the data and then tokenization and vectorization was performed. Cosine similarity was then used to calculate the similarity of all songs to all other songs in the dataset. The recommendation algorithm then takes any input song that appears within the dataset, gets its index within the cosine similarity matrix, and outputs the rank list of songs that are similar to it from that matrix.

3.4 Interpretation of Results

It is difficult to evaluate the effectiveness of the recommendation system, so the rank-list of the top 5 songs similar to "It's The Most Wonderful Time of The Year" by Andy Williams were analyzed. It appears that the system is working. The system recommends the most similar song by title, which is a cover of the original song. It also recommends "High Time" by Paul Jones which has the word "time" in the title and was also released in the 60's. Following that is a holiday song by Paul McCartney, which is also related to our input song.

Rank	**Song Name**	**Artist Name**
1.	It's the Most Wonderful Time of the Year	Andy Williams
2.	It's the Most Wonderful Time of the Year	Jonny Mathis
3.	High Time	Paul Jones
4.	Wonderful Christmas	Paul McCartney
5.	Take Your Time	Jefferson Starship

Table 2: Top 5 Reccomended Songs

4 Critique of the Model

Though the two models could be improved and were limited, they satisfactorily answer the problem statement and accomplish the goals which were to primarily investigate the techniques used in audio artificial intelligence (AI). The genre classifier performed quite well given the resources and time. It accurately classified 72% of input data, which is decent considering there are 10 classes and the dataset was small. The recommendation system also performed decently well, giving very similar songs to the input data upon listening. Although, it is possible to improve the models.

4.1 Limitations and Improvements

The genre classification model did not have a very good validation accuracy (66%), even after correcting for overfitting. More training and test data could increase the validation accuracy and overall fit of the model, although this requires access to more computational power. The architecture of the network is most likely not optimal, and could be researched further to develop a better model.

The recommendation system could be largely improved by the addition of collaborative filtering, which is the addition of user rankings into the model. Then, the model could implement what songs the user has previously liked that are similar to the input song. To do this, datasets with user rankings must be readily available. The Spotify API may be a good place to look. Changing the code to remove duplicate titles to an input song would most likely improve the model since user's would not like to listen to the same song again. Currently, the tags used are lyrics, artist name, genre and topic. If more data could be found or collected that has more fields, such as album or emotion evoked, the system could be more accurate.

The 10000 samples of songs used in the neural network portion are not full songs – therefore some of the nuance is lost between genres. For example, the network could have been fed an instrumental and melodic bridge from a metal song which would sound like rock giving the model more information for decision making.

Moreover, this project was completed under resource constraints. All code was run in local Jupyter notebooks and Google Collab was only consulted towards the end of the project. However, when the recommendation system was run on Google Collab, it would inevitably crash. Computational power was limited throughout the duration of the project, so larger datasets like the "Free Music Archive" were not utilized, and instead smaller datasets in "GTZAN" and "Music Dataset 1950 to 2019" were chosen.

Therefore, more data and running the models in a remote environment would improve the systems greatly.

4.2 Future Directions

Ideally, the two systems were intended to be merged by using the output of the genre classifier as a feature in the recommendation system model. This was an unrealistic approach due to 2 reasons. First, the real world likely only uses a recommendation system as a shuffle algorithm, and audio classification is used by companies like Shazam, where songs need to be identified by audio input [3]. Second, genre is a readily available field in most music datasets, so the application of the full system would not be useful.

The realization that the full system would not have a good application to the real world was pivotal. After the two systems were separated, their usefulness became more clear. Therefore, possible next steps would be to work on each system individually and in more depth. The genre classifier could be fleshed out into an audio recognition system similar to the company Shazam [3]. Another direction could be fully developing a 'shuffle algorithm' using the recommendation system that incorporates more descriptive features from data and collaborative filtering to output the most similar song to an input.

References

[1] Algorithmia. *Introduction to Loss Functions*. URL: https://algorithmia.com/blog/introduction-to-loss-functions. (accessed: 2021.12.08).

[2] Michaël Defferrard. *MDEFF/FMA: FMA: A Dataset for Music Analysis*. URL: https://github.com/mdeff/fma. (accessed: 2021.11.10).

[3] Jovan Jovanovic. *How does Shazam work? Music Recognition Algorithms, Fingerprinting, and Processing*. URL: https://www.toptal.com/algorithms/shazam-it-music-processing-fingerprinting-and-recognition. (accessed: 2021.12.08).

[4] Nick McCullum. *Deep Learning Neural Networks Explained in Plain English*. URL: https://www.freecodecamp.org/news/deep-learning-neural-networks-explained-in-plain-english/. (accessed: 2021.12.08).

[5] Luan Moura. *Music Dataset: Lyrics and Metadata from 1950 to 2019*. URL: https://data.mendeley.com/datasets/3t9vbwxgr5/2. (accessed: 2021.11.25).

[6] Pratheeksha Nair. *The Dummy's Guide to MFCC*. URL: https://medium.com/prathena/the-dummys-guide-to-mfcc-aceab2450fd. (accessed: 2021.12.07).

[7] Andrada Olteanu. *GTZAN Dataset - Music Genre Classification*. URL: https://www.kaggle.com/andradaolteanu/gtzan-dataset-music-genre-classification. (accessed: 2021.11.25).