CSCI 4622

Music Genre Classification

Bao Nguyen, Drew Hoffman, Jishnu Raychaudhuri

Spring 2022

Contents

1 Problem Space

We are trying to predict the genre of various music files to provide more insight as to why certain songs belong to certain genres. We are going about this by testing different popular music classification algorithms and seeing how they perform. We believe that by testing different algorithms we will have a better understanding of how to make an effective classifier.

We think this as an important problem to be solved because the differences between genres in music is often not understood very well. Different experts might disagree on the genre classification of a specific song that seems to toe the line between genres. We hope that the differences and the characteristics between genres will become much clearer. This may also lead to people better understanding why they enjoy and respond to certain genres of music. We also want to see which genres tend to have more crossover and what algorithms are effective at capturing this.

Having a machine learning algorithm to classify songs into their respective genres could be very beneficial, considering how music is consumed in the modern age. Streaming services have an incredible repository of music and usually cater to a wide audience who enjoy a wide variety of music genres. If a certain user wants to listen to music of a specific genre, the benefits of using an algorithm for classification becomes clear. Using an algorithm would be much more efficient than having a group of human beings listen to each and every track and then decide upon a classification.

Here are a few assumptions of how certain song characteristics and dataset features will be more predictive than other features (such as existing prediction song features) and potentially find their impacts on each genre:

- Discrete time features (beats, tempo, etc.) in the music provide a metric to predict genres
- Spectrogram consists of frequency ranges that tie to specific types of music

2 Approach

We used three approaches and each built a classifier using the following algorithms:

1. **K-Nearest Neighbors (KNN):** KNN is a supervised machine learning algorithm that is primarily used to solve classification problems. The model is trained by supplying it with training data. The model then saves the supplied features of the training data for future predictions. When input data to be predicted is provided to the KNN model, the model calculates the k nearest neighbours from the training data to the input data according the features of the input data. The input data is then classified as the majority class among the computed nearest neighbours. The KNN classifier is a popular classifier model as it's very easy to implement and has only one parameter, k.
2. **Convolutional Neural Network (CNN):** This is a Deep Learning neural network, which is commonly used in visual recognition and classification. These classifiers are a form of the multilayer perceptron that we covered in class. Due to the fact that these networks are fully connected networks, they are especially prone to overfitting. The dataset that we are using has a set of 30 second snippets from songs from each genre. For this, the .wav music files were first broken up into 3 second chunks. We found that by breaking the song into individual pieces prior to training, the accuracy obtained by the classifier was much higher, as there were far more samples to use as training data and it helped prevent overfitting. Then, the 3 second intervals were converted into spectrograms using the librosa library of python. A spectrogram is a visual representation of the different sound frequencies and wavelengths of an audio recording. By doing this, we are able to take advantage of the Convolution Neural Network's ability to classify based on an image. We could then use the spectrograms for the visual analysis and classification. We generated 300 spectrograms for each music genre for a total of 2700 samples. Having 20% being removed for validation, we then ran through 25 epochs using a 64 sample batch size.

3. **K-Means Clustering:** k-Means is an unsupervised method assigning random initiated points in the data as centroids. This method then calculates repetitively to optimize the centroid positions. Once the centroids are stabilized (no significant changes after a calculation), the clustering suceeds.

   k-Means' goal is to find groups in the data and partition the data into the number of clusters k. For cluster $C_i = \{x_1, x_2, ...x_{m_i}\}$ and by applying the Euclidean distance method, the intra-cluster variance $V(C_i)$ is defined as:

$$V(C_i) = \sum_{j=1}^{m_i} ||x_j - \mu_i||^2$$

   for $\mu_i = \frac{1}{m} \sum_{i=1}^{m_i} x_i$ to be the centroid of cluster C_i.

   Hence, the objective for k-Means is:

$$\min_{C_1, C_2 ... C_k} \sum_{i=1}^{k} V(C_i) = \min_{C_1, C_2 ... C_k} \sum_{i=1}^{k} \sum_{j=1}^{m_i} ||x_j - \mu_i||^2$$

   for k to be the number of clusters and every x_i is assigned to the cluster of the centroid nearest to it.

   Additionally, unlike regular k-Means methods, k-Mean++'s centroids are initialized and assigned slightly differently: first centroid is chosen uniformly, and the subsequent centroids are randomly picked with the probability proportional to d^2 (the square of the distance between the current centroids and data points).

3 Data

For each of the models we used the GTZAN music genre classification dataset. This is the data that is commonly used to train and develop music classification algorithms. This dataset contains songs from 10 genres with 100 samples of each genre for a total of 1000 samples. The genres included in the dataset are blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, and rock. The dataset consists of the .wav audio files as well as a .csv containing numerous different features of the dataset such as tempo, rolloff, spectral centroid, among others. The spectral centroid is the average frequency contained in the spectrogram. The rolloff is the low bass and high treble limits in a frequency response curve and numerous other continuous variables can be obtained from this. In total, there are 59 features in the dataset, with continuous and discrete values. Shown below is an example of a few lines from our dataset.

filename	temp	beats	chroma_stft	rmse	spectral_ centroid	spectral_ bandwidth	rolloff
blues.00081.au	103.359375	50	0.38026021	0.24826229	2116.94296	1956.61106	4196.10796
blues.00022.au	95.703125	44	0.30645087	0.11347541	1156.0705	1497.66818	2170.05354
blues.00031.au	151.999081	75	0.2534871	0.15157077	1331.07397	1973.64344	2900.17413

Table 1: First three entries and first eight columns of the data set

4 Models

4.1 K-Nearest Neighbours

For the KNN approach, the features were extracted from the 30-second .wav files for each of the songs. As the audio signals in the .wav files are constantly changing, the files were divided into frames of about 20 milliseconds long. For each frame, Mel Frequency Cepstral Coefficients were extracted.

The Mel Frequency Capstrum (MFC) is a way of transforming the linearly spaced frequency bands on the normal Mel scale to a different scale which is a much better approximation for how human beings perceive the frequencies using their ears. This not allows for a better representation of the music, it also allows for the selection of frequencies that can offer the most information when it comes to genre classification.

The Mel Frequency Cepstral Coefficients (MFCC) make up an MFC for a song. The means and covariances of these coefficients were then used as features for the KNN classification.

The labels of the genres were mapped as follows:

Genre	Pop	Metal	Disco	Blues	Reggae	Classical	Rock	Hiphop	Country	Jazz
Class	1	2	3	4	5	6	7	8	9	10

Table 2: Genre to class mappings

After extracting this data, it was then randomly split into a training set and a testing set with the training set being two-thirds of the full data set and the testing set being the other third.

To determine the optimal number of neighbours that worked best for the KNN model, 10 models were fit with the k values ranging from 1 to 10. These models were then tested against the test data. A plot of the number of neighbours vs the accuracy on the testing set is shown below.

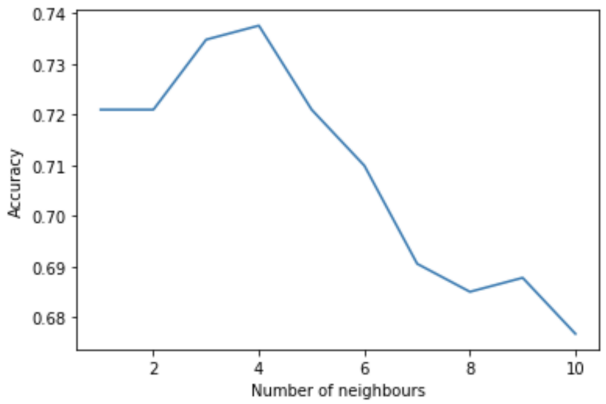


Figure 1: Accuracy vs k value of the KNN model

From the plot above, the KNN model performs best when using four neighbours to predict the class of a particular song. Using the model with four neighbours yields an accuracy of about 73.76% on the test data. The confusion matrix, along with a colour coded visualisation of the matrix is given below.

```
[[27.  0.  3.  0.  1.  0.  0.  1.  0.  0.]
 [ 0. 27.  2.  2.  0.  0.  0.  2.  1.  0.]
 [ 1.  1. 25.  1.  2.  0.  2.  2.  0.  0.]
 [ 0.  0.  0. 22.  0.  0.  0.  2.  3.  0.]
 [ 0.  1.  2.  0. 25.  0.  1.  5.  0.  0.]
 [ 0.  0.  1.  0.  2. 33.  0.  0.  0.  1.]
 [ 1.  1.  9.  3.  4.  0. 20.  1.  4.  0.]
 [ 3.  2.  2.  0.  3.  0.  3. 32.  0.  0.]
 [ 0.  0.  1.  1.  2.  0.  4.  1. 24.  1.]
 [ 0.  0.  2.  0.  1.  5.  0.  0.  2. 32.]]
```

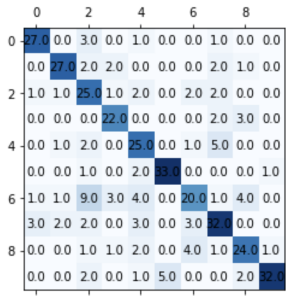


Figure 2: Raw confusion matrix and colour coded confusion matrix

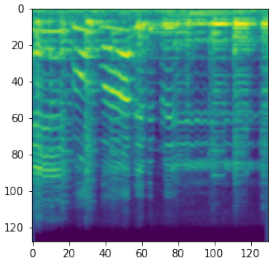
From the plot above, the KNN model most often misclassifies rock music as disco music, which is quite interesting. Perhaps the similarities in instrumentation such as the heavy use of bass drums are what causes the high rate of misclassification of the genres.

Hip-hop and reggae and classical and jazz are the next two pairs of music genres that get misclassified for each other the most. This is less surprising than rock and disco. Hip-hop and reggae are both genres that make extensive use of turntables and rely on making music from sounds that might not originate from actual instruments. Classical music and jazz also have a lot in common. Both genres are primarily instrumental with little to no lyrical content and both employ the use of western classical music.

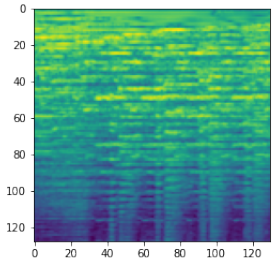
Rock also happens to be the genre that gets misclassified the most, while jazz gets the highest accuracy among the genres.

4.2 Convolutional Neural Networks

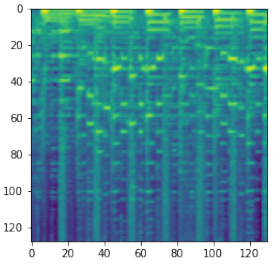
The following pictures are spectrograms that were generated by the librosa library in python on three second audio snippets. Each of these is a representation of the range of sound frequencies that are contained in a certain .wav audio file.



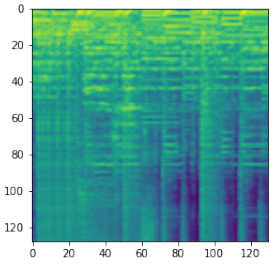
. Blues



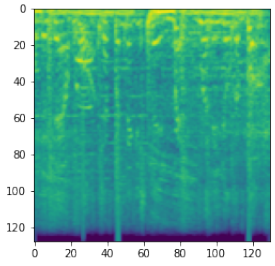
Classical



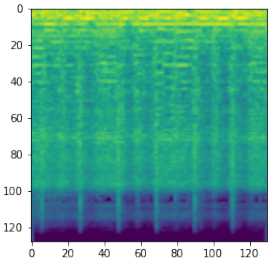
Country



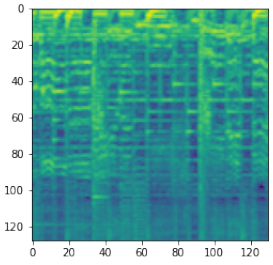
. Disco



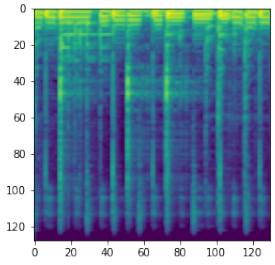
Hip Hop



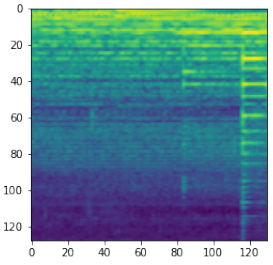
Metal



. Pop



Reggae



Rock

In order to understand the mel spectrogram, it first requires a basic knowledge of sound engineering. Audio signals are measured in variations of air pressure over time. By capturing this, we can generate a waveform of signals of a given piece of audio. In order to generate a spectrum, we must apply a Fourier transform

to the waveform to decompose a signal into it's individual frequencies and the frequency's amplitude. It changes the waveform representation's domain from time to frequency, creating one for each frequency. The spectrogram is the culmination of these Fourier transforms stacked on top of one another so that we may see the range of frequencies and their prevalence in a piece of audio.

As we can see already, certain genres seem similar to each other based on their sample image representations of their frequencies. For example, the disco and pop spectrograms and the reggae and blues seem very similar to each other. Distinctive genres such as metal and hip hop don't appear to be very similar to any others. Based on this, we can predict that our model will be better able to capture those genres of music.

The genre mapping that we used is the same as that used in the KNN implementation.

After generating each of these spectrograms on 3 second snippets of the dataset, we randomly split up the samples and reserved 20% of them for validation data. We then used keras and tensorflow to create our CNN classifier. The classifier consists of four 2-dimensional convolution layers using Relu as our activation function. A Dense layer was then applied over the model with the activation function being softmax so that we could obtain probability values for each genre for each test sample.

After running through 25 epochs of training on the data, we were able to obtain an accuracy of 0.9977 on the test set and 0.7426 on our validation set. The graph of the validation and testing accuracy is shown below:

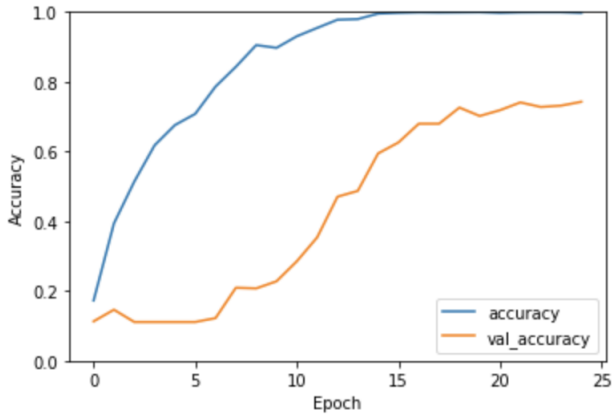


Figure 3: Plot of Accuracy over Epochs during training

4.3 K-Means++

By using 500 random samples from GTZAN's features_30_sec.csv and ultilizing the means of these music features: chroma STFT, RMS (Root Mean Square), rolloff, zero-crossing rate, harmony, and perceptron for K-means and K-means++ learning models, it results in scatter graphs (for x-axis: chroma STFT and y-axis: zero-crossing rate) below:

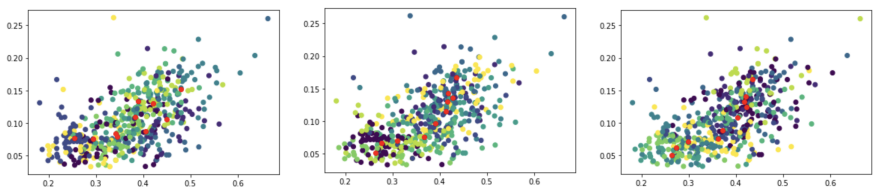


Figure 4: True labels, K-means predicted labels, K-means++ predicted labels

Note that these graphs are displayed in two dimensions with two features for visualization purposes, but the K-means algorithm was trained on 6 features.

5 Results

The accuracy for our models on the test data is as follows:

Model	Accuracy
KNN	73.76%
CNN	74.26%

Table 3: Model accuracy data on testing sets

After obtaining our testing set accuracy we ran the following songs through each of our models to see what each classifier would predict: Sweet Child O' Mine by Guns N' Roses for rock, Fur Elise by Beethoven for classical, Three Little Birds by Bob Marley for reggae, N.Y. State of Mind by Nas for hiphop, Stayin Alive by the Bee Gees for disco, Mercy, Mercy, Mercy by Cannonball Adderley for jazz, 22 by Taylor Swift for pop, Enter Sandman by Metallica for metal, Take Me Home, Country Roads by John Denver for country, and Billy's Blues by Billy Stewart for blues.

We thought that each of these songs has a distinct and recognizable genre to the human ear and wanted to see if the classifiers were able to capture that. The results for each mode are given below.

Song	Artist	KNN	CNN	Actual Genre
Sweet Child O'Mine	Guns N' Roses	Pop	Rock	Rock
Fur Elise	Beethoven	Country	Blues	Classical
Three Little Birds	Bob Marley	Pop	Blues	Reggae
N.Y. State of Mind	Nas	Hiphop	Hiphop	Hiphop
Stayin' Alive	Bee Gees	Pop	Hiphop	Disco
Mercy, Mercy, Mercy	Cannonball Adderley	Reggae	Blues	Jazz
22	Taylor Swift	Pop	Disco	Pop
Enter Sandman	Metallica	Pop	Rock	Metal
Take Me Home, Country Roads	John Denver	Pop	Reggae	Country
Billy's Blues	Billy Stewart	Hiphop	Blues	Blues

Table 4: Individual song predictions for each model

For unsupervised tasks such as K-means, it is usually unnecessary and not straightforward to validate the output, thus, a classical accuracy method cannot apply. On the other hand, by observing each dataset and the obtained centroids (centered around true/predicted homogeneous labels), each sample are assigned to a cluster based on the learning prediction and the centroid to which it is set by K-means and K-means++. Also due to the fact that our K-means implementation was trained on the .csv values such as rolloff, chroma_stft, rmse, among others, we were not able to generate predictions for our sample songs as we did not have access to this information.

6 Discussion

The KNN model performed rather poorly during the individual song tests. Interestingly, it predicted most of the songs as pop songs. This might simply be due to the nature of the GTZAN data set. Most of the songs pop songs in the data set are from the 1980s, a time when popular music was incredibly synthesizer as well as drum heavy, making it sound quite similar to rock and disco music. Metal and rock music were also quite popular during the aforementioned decade, further blurring the line between what actually counts

as ‘pop’ music and if the genre description can stay the same over time. Although easy to implement, the KNN model can output some questionable results sometimes.

The CNN model performed pretty well during testing. By trimming the audio recordings to three seconds each, this gave us a wide variety of data for training. The most popular genres that the CNN would classify songs as were blues and hip hop. We can infer from this that rhythmic, heavy drums that you would typically find in a hip hop song were commonly captured in the other pieces of music that we tested. Blues music is also fairly diverse and it makes sense that genres such as classical, jazz, and reggae would be misclassified as such. Interestingly enough, very few songs were classified as pop songs. This may be due to the slightly outdated nature of our data set as well as the subjective nature of what constitutes "pop" music. Because our songs were trimmed into three second snippets, there were likely many samples from a song that did not accurately represent the frequency content of the rest of song. In the future, it could be more effective to use some sort of "smart" trimming that analyzes a song's spectrogram and pulls the snippets that are more indicative of the sounds that one might here in the genre.

Because the K-means' nature is an unsupervised learning technique, which produce an output data that lacks of consistency with the input (or training) data for accuracy calculation, therefore it is often deemed as less accurate or trustworthy than a supervised algorithm. On the flip side, supposedly the dataset is too huge that the knowledge on all music genres is not concrete and monitoring the accuracy is not needed, K-means is a simple and flexible method to implement as it helps find unknown patterns in data by combining specific features and doesn't require any sort of supervision, which is an exhaustive direction to take for an enormous tableset.

In the future, we want to use this information in order to build a more interpretive, user friendly music classifier as opposed to the "black box" classifier that many streaming services utilize. For the KNN model, we need to be aware of the classifiers tendency to overfit based on input data and feed it training samples that are indicative of the genres themselves. The K-means implementation can be used to help find hidden patterns in the data based on technical sound engineering features. And finally, the CNN classifier can also have a tendency to overfit but by using a better trimming system and the softmax distribution of probabilities, it has an increased capability to capture genre crossover and give better recommendations to users.

7 Code Repository

Github

References

[1] Li, Tom L., and Antoni B. Chan. “Genre Classification and the Invariance of MFCC Features to Key and Tempo.” Lecture Notes in Computer Science, 2011, pp. 317–327., https://doi.org/10.1007/978-3-642-17832-0_30.

[2] Lam Hoang. 2018. Literature Review about Music Genre Classification. In Woodstock ’18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/ 1122445.1122456

[3] Choi, K., Fazekas, G., Sandler, M., Cho, K.: Convolutional recurrent neural networks for music classification. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2392–2396. IEEE (2017)

[4] Kostrzewa, D., Kaminski, P., Brzeski, R. (2021). Music Genre Classification: Looking for the Perfect Network. In: Paszynski, M., Kranzlmüller, D., Krzhizhanovskaya, V.V., Dongarra, J.J., Sloot, P.M.A. (eds) Computational Science – ICCS 2021. ICCS 2021. Lecture Notes in Computer Science(), vol 12742. Springer, Cham. https://doi.org/10.1007/978-3-030-77961-0_6