Elena Georgieva

Professor Zili Liu

Psych 186B: Neural Networks

18 March 2016

Final Paper: Music Genre Classification

The music industry is constantly changing, and in the 21st century the majority of music is created, purchased, and transferred electronically. Now that music collections exceed thousands of songs, organization and classification of these collections is more important than ever. Song genre tends to be an attribute that can help in selecting songs to play, especially for online streaming services such as Spotify, Pandora, and Apple Music. However, music genre classification has been a challenge for the industry as music genres are hard to systematically and consistently describe due to their inherent subjective nature. In this paper, we try to address the problem of genre classification using only the audio content of a music file and neural networks techniques, all coded in MATLAB.

In this paper, the general process for genre classification involved using Mel-frequency Cestrum Coefficients to decompose an audio signal into one matrix and using a back-propagation neural network with hidden layers to classify the music. The dataset selected was the GTZAN Genre Collection. This is a dataset of 1,000 songs in .wav format, each of length 30 seconds. These 1,000 songs represent ten genres, but only four were selected for the purpose of this study. The genres Classical, Blues, Pop, and Metal were selected, and of the 100 songs that represent each genre, 70 were used for training the neural network, and 30 were used for testing. Mel-frequency Cepstral Coefficients (MFCCs) are time-spectral decompositions of an audio signal that convey the general frequency characteristics important to human hearing. MFCCs are commonly used in the field of speech recognition, and some research shows that MFCCs can capture useful sound characteristics of music files as well (Logan, 2000). MFCCs are computed over short intervals of a song, so they do not carry information about the rhythm and tempo of a song. Similarly, they are much more focused on the low frequencies of a song than the high frequencies, as humans are more sensitive towards these lower tones.

To compute MFCC features, a sound file is first subdivided into small frames of 20 ms. Next, each frame is multiplied by a hamming window, this smoothens the edges and prepares the sound wave for the Fourier transform. A Fast Fourier Transform transforms a signal from its original domain to a representation in the frequency domain. Next, these frequencies are mapped to the Mel scale, which models human perception of pitch. This map is approximately linear for the lower half of audible lower frequencies (20Hz-1kHz) and logarithmic for higher frequencies (1kHz-2kHz), and ends up grouping the frequencies into 20 bins. Next the Discrete Cosine Transform is taken to de-correlate the frequency components. The 5 bins with the highest frequencies are discarded, as the higher frequencies are the details that make less of a difference to human perception contain less information about a song. The MFCCs are the amplitudes of the resulting spectrum. These are 39 values that represent a song (Haggblade et all, 2011).

Neural networks are generally successful in many machine-learning problems, and we chose a back-propagation model. Before feeding data into our model, we pre-processed the data by combining MFCC characteristic vectors into one matrix. The input to the network was this matrix. Next there were 5 hidden layers of size 30, 20, 10, 5, and 2, and the output was a 1 or 0. First, each input was matched with a desired output: 1 for one genre, and 0 for the other. Next, six matrices were created for the weights that connect the input to the hidden unit, the hidden unit to the next, etc., until the final hidden unit is connected to the output unit. These matrices were filled with uniform random numbers between -0.5 and 0.5, generated by the rand() function. A loop was created to perform the back-propagation. This loop continues to iterate until the sum of squared error between the observed and desired output is less than .01 and/or the number of epochs is greater than 1,000. For each input pattern, it is multiplied by the weight connecting it to the next layer. This value is then passed into an input activation function that takes in a matrix of activation function and returns a sigmoid on those values.

A sigmoid is the non-linear unit used in back-propagation. It is a monotonically increasing function, usually bounded between -1 and +1. In this case, the sigmoid function selected was the hyperbolic tangent function. This was selected through a trial-and-error process through all the commonly used sigmoid functions. Hyperbolic tangent gave by far the strongest results. Next, this hidden activation is passed from the hidden units to the output units. Output activity is determined by passing the input to the output units through the same activation function. Next, the output error is computed by subtracting the output activation from the desired output. The weight changes are determined then applied to each layer of connection. The weight changes depend on a learning rate. In the study, it was found that a smaller learning rate gives more accurate results and a learning rate of 0.001 was used. Once these steps are done for every input, the sum of squared errors is computed over all input patterns.

Next is the testing phase in which the 30 songs not yet presented to the matrix are used for testing. These vectors go through a similar process to arrive at an output activation value matrix. This is a matrix of zeros and ones, where the first 30 digits represent one genre and the second 30 represent the other. In a perfect result, the first 30 digits would be ones and the remaining 30 would be zeros. This project was run six different times, pairing up each two of the four genres together. The accuracy of each was scored out of 30: how many of the first 30 values were ones as predicted, and how many of the second 30 values were zeros as predicted. Each of the six trials was executed twice, and results were identical or within 2/30 = 7% of each other. The first trials were recorded, and these results can be seen in Figure 1. Experimenters were surprised by the strength of the results, 94% of the songs were categorized into their correct genre! That is an average of 28.2 out of the 30 test songs. For some category pairs, the results were actually 100% accurate, for example Blues-Pop and Pop-Metal.

It is notable that on around 15% of trials the model did not converge, and the results were all zeros. These trials were discarded, and the experiment was ran a second time, when it was likely to converge. The results of the music genre categorization experiment were very strong, but it is important to take note of all the experimentation and trial-and-error that went into the process. Before settling on the back-propagation neural network, several other techniques were discussed and attempted including the k-nearest neighbor classifier, and the multi-class support vector machine. These are both machine-learning algorithms, and it became necessary to use a neural network model due to the nature of the course. Furthermore, this network was modified heavily before settling on the final version. The network has five hidden layers, when the experiment was attempted with three hidden layers the results were very inconsistent. There was also a lot of experimentation done with the sigmoid function before the hyperbolic tangent was selected, and with the learning constant before 0.001 was selected. With other sigmoid functions and learning constants, the model did not converge—rather it oscillated or approached a local minimum. In a few trials, 85 songs were used to train and 15 songs were used to test. This resulted in a ceiling effect, where almost all of the songs were characterized correctly every time.

This project makes a great approach on the music genre classification problem, but it could be extended in several ways. First, it would be interesting to create a model that differentiates between four genres, instead of only two. Results would likely be lower, but more significant and applicable to real-world scenarios that music streaming services might use. Another direction could be to attempt finer classifications between genres that are more similar, such as alternative rock, and punk rock. An obvious way to improve the classification even further would be to get a larger dataset. The database used featured 100 songs per genre. If the network could be trained on a few hundred songs per genre, the results would be very strong. Ultimately, the project in its current state gives strong results that can be used to classify music exclusively based on its genre.

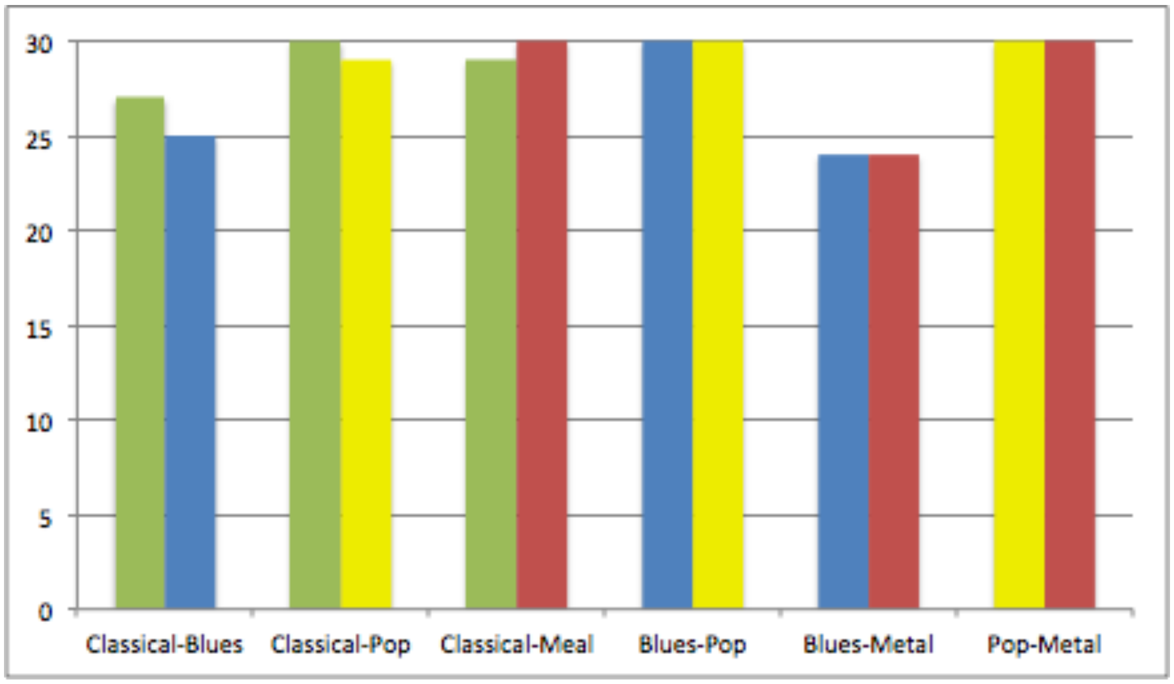

Figure 1: Results (x axis: pairs of two genres, y axis: number of songs correctly categorized)

References

Haggblade, M., Hong, Y., Kao, K (2011). Music Genre Classification, http://cs229.stanford.edu/proj2011/HaggbladeHongKao-MusicGenreClassification.pdf

Logan, M (2000). Mel Frequency Cepstral Coefficients for Music Modeling, http://ismir2000.ismir.net/papers/ logan_paper.pdf