# Music Genre Classification Using a MFCC-Fed Neural Network

Román Via-Dufresne Saus
Mentored by Barbara Hajdarević

July 2019

**Abstract**

The aim of this project is to develop a music genre classifying algorithm and to optimize it to obtain the most accurate classification possible. To do so, MFCC features have been extracted from 30s audio files and fed into a neural network. Different feature extraction methods and artificial neural network architectures are presented and evaluated in this paper.

Feature extraction method which yields the best results consists in extracting 13 MFCC using a 20 Hz to 8000 Hz Mel filterbank from full 30s clips and using mean and covariance matrix to feed it into the neural network.

It has also been found that the best configuration for our neural network consists in a simple network with one hidden layer of 10 neurons.

By using the mentioned feature extraction method and artificial neural configuration, an accuracy of 87.6% was obtained.

## 1 Introduction

Music genre classification using machine or deep learning is present in many popular apps and platforms such as Spotify and Pandora. There are different methods and algorithms that can be used to achieve so. In this paper, an artificial neural network has been used.

Artificial neural networks are machine learning algorithms that mimic human brain neurons architecture. Each neuron and each connection between neurons are represented with a numerical value. Applying each value to the input data results in the input data. Neural networks can be designed by changing the number of layers of neurons and the number of neurons in each layer. Once a neural network is built, it is trained by changing the value of each neuron and connection to make the output as accurate as possible. More information about artificial neural networks is exposed in section 2.3.

However, neural network cannot be directly fed with an audio signal. Audio signals contain a lot of data that carries little significant information for our purpose. It would be extremely inefficient and impossible due to the huge amount of computational capacity required for it.

Therefore, some smaller data that represents the audio characteristics is needed. In this project, Mel Frequency Cepstral Coefficients (MFCC) have been used. These are based on frequency properties of the signal. However, they differentiate from standard frequency spectral coefficients in that they are obtained with a method that relies on the Mel scale. The Mel scale is a scale that takes into account human sensitivity of frequency changes.

## 2 Methods

### 2.1 Database

The GZTAN database has been used for this project. GZTAN is a free database that contains 1000 songs of 10 different genres (100 songs per genre). GZTAN songs are 30 seconds long ".wav" files with a 22050 Hz sample rate[1]. However, only 5 genres have been used in this project: jazz, classical, pop, rock and reggae.

As 100 samples are relatively few for a neural network, splitting the songs into clips of different lengths (5 and 3.5 seconds) to increase the number of samples has also been tried. The comparison between these and non-split songs can be found in the results section (section 3).

## 2.2 MFCC extraction

As MFCC are based in how an audio signal changes and its frequency, they are not directly calculated for the whole song. The first step is to split the audio signal into short frames. In this project, they have been divided into 23 ms frames. Each frame starts 10 ms ahead of the starting point of the frame behind, so they overlap. From here on, all steps are applied to each of these frames.

The next step consists in applying a windowing function to the frames to smooth them and avoid irregularities when doing the Fourier transform. Hamming functions have been used in this project.

Then, the Fourier transform is applied and a spectrogram obtained. After, a periodogram is obtained by applying the following formula:

$$P(x) = \frac{(F(x))^2}{N} \tag{1}$$

, where $P(x)$ is the periodogram, $F(x)$ is the Fourier transform, and $N$ equals the length of $F(x)$ in samples.

Here, the Mel filterbank is applied. It consists in a set of overlapping triangular filters that are calculated according to the Mel scale. In speech recognition, the filterbank normally extends from 300 Hz to 8000 Hz. However, both this standard configuration and reducing the minimum to 20 Hz have been tried in this paper. The reason of this is that although the lowest standard frequency used in speech recongition 300 Hz, we can hear sounds from 20 Hz[2].

To find the points that form the filters, the bottom and top frequencies are converted into the Mel scale. To do that, equation 2 is used.

$$M(f) = 1125ln(1 + \frac{f}{700}) \tag{2}$$

, where $f$ is a frequency and $M(f)$ is its Mel conversion.

After, a scale of equally separated points is calculated from the bottom to the top Mel frequencies. 26 filters have been calculated. 27 points are required. Once this points are obtained, the inverse of equation 2, is applied:

$$M^{-1}(F) = 700(e^{(\frac{F}{1125})} - 1) \tag{3}$$

In equation 3, $F$ is a Mel frequency and $M^{-1}(F)$ is its normal frequency. As it can be observed, it is an exponential function; and so are the points for the filterbank.

With this points, each filter starts at the top point of the preceding filter, reaches one (the top) in the next point, and goes back to zero in the next one; as can be observed in figure 1

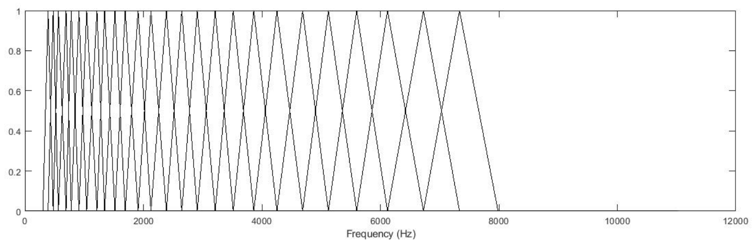


Figure 1: 26-filter Mel filterbank that has been obtained.

Each MFCC coefficient equals the sum of the values of one filter applied to the periodogram. Therefore, 26 filters have been obtained. Normally, only the 13 first coefficients are used. [3] indicated that just the first 5 coefficients worked better for their genre classification. Both have been tested and discussed in the results section (3).

These features have been fed into the neural network in two different ways. The first one is simply concatenating the MFCC of each frame, which lead to a lot of input data. The second is by calculating the mean and the covariance matrix of the MFCC values represented by a matrix with one row per frame. In the second case, values from the mean vector and the covariance matrix (avoiding repetitions, as it is a symmetrical matrix) are concatenated.

## 2.3 Neural Network

The classification algorithms used in this project are artificial neural networks. Neural networks are machine learning algorithms that are inspired by the biological neural networks. They are formed by interconnected layers of neurons. Each of this neurons and neural connections has a different effect in data. Data is input in the input layer, processed by the hidden layers and a result is obtained in the output layer.

To build the neural networks used to evaluate MFCC extraction, MATLAB's `nprtool` has been chosen due to its simplicity. To compare different methods' performance, a neural network with one hidden layer of 100 neurons has been used.

After, a better neural network has been tried to be found. In this case `patternnet` (also MATLAB's), has been used. This tool allows to tweak the number of layers and the number of neurons for each layer. Three experiments have been carried to find a good architecture for our neural network:

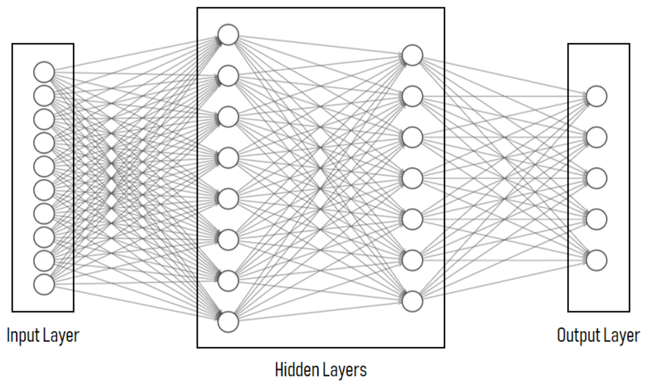


Figure 2: Neural network diagram with two hidden layers.

1. Using one hidden layer and changing the number of neurons in it from 20 to 1000 in steps of 20.
2. Using two hidden layers with the same amount of neurons each and changing this amount from 20 to 1000 in steps of 20.
3. Changing the number of hidden layers from 1 to 15 (in steps of 1), using the result of experiment 1 (see section 3.2) as the number of neurons per layer.

In all cases, data has been distributed as follows:

- 70% for train set.
- 15% for validation set.
- 15% for test set.

## 3 Results & Discussion

### 3.1 Feature Extraction

As explained in section 2.2, data was fed into the neural network in two different ways. However, concatenation had to be discarded after the first attempts as the obtained accuracy was near the statistical one of random classifying. It showed no capacity of proper classification. Therefore, all the results given below have been obtained with the mean and covariance matrix method.

Similarly to the case above, using just the first 5 MFCC instead of 13 was rapidly discarded as it gave too low accuracies.

In both cases, result accuracies are not even displayed due to the fact that they varied a lot each time the model was retrained and the network was totally unable of classifying the songs correctly.

All following tweaks were done according to these first two premises. However, as in all cases the accuracy was not exactly the same each time the neural network was retrained, these results are based on the mean and standard deviation obtained after retraining a model several times.

Moreover, all experiments carried before tweaking the neural network, have used a `nprtool` (see section 2.3) network with one hidden layer formed by 100 neurons.

In section 2.1, it is explained how splitting the audio files in clips of a) 5 seconds and b) 3.5 seconds has been tried to increase the number of samples in the database. However, results have shown that the neural network works better with the full 30 second songs, although the smaller number of samples.

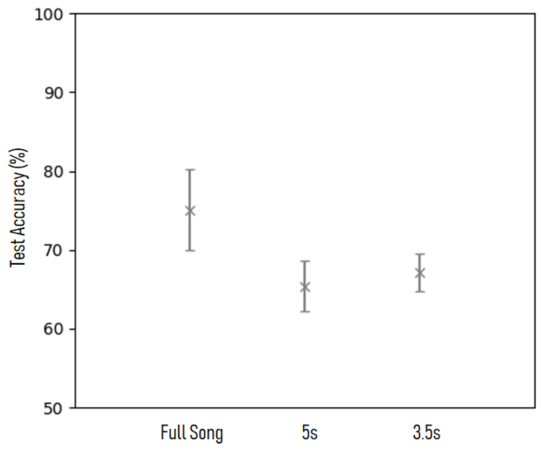


Figure 3: Mean (represented by $\times$) and standard deviation (error bars) of the accuracy values obtained by the neural network after retraining it 10 times each for non-splitted data, data splitted in 5-second clips and data splitted in 3.5 seconds.

As it can be observed in figure 3, splitting the samples into smaller ones, reduces the standard deviation of the accuracy as increasing the data makes the model more stable. However, their accuracy is much lower than using the full 30-second audio samples and it use has been discarded.

Using 20 Hz as the baseline for the Mel filterbank instead of 300 Hz (standard) has also been tried (section 2.2). As expected, 20 Hz has shown to be better than 300 Hz for musical genre classification. The mean accuracy obtained in the first case is 78.2 against the 75.1 of the second one. Also, the standard deviation obtained in both cases is nearly 5.4.

## 3.2 Neural Network

Once the results above have been obtained and the best input data within our options have been specified (mean and covariance, 13 MFCC, 30-second non-splitted clips, 20 Hz - 8000 Hz filterbanks), some different neural network architectures have been tried. Description of the three carried experiments can be found in section 2.3.

To gain certainty, the accuracy values have been obtained by doing the mean of the accuracy values obtained after retraining the neural network in each single case 20 times.

It is important to show that in this section accuracy experienced an improvement, even when using the same architecture as in previous experiments. This fact could be attributed to the change of neural network tool (`patternnet` for this section) and the differences in the algorithms they use.

For one-layer networks, results show that their performance does not vary much in function of the number of neurons in it, at least for the range between 20 and 1000 neurons. However, some irregularities can be observed as the number of neurons increases (above 500), see figure 4.

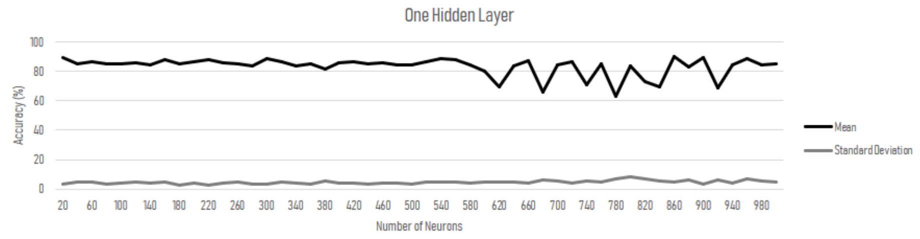


Figure 4: Mean (black) and standard deviation (grey) of the accuracy values obtained by a one-layer neural network for different number of neurons.

Therefore, it is better to use a small number of neurons as it makes the retraining of the neural network faster and it has been shown to be less irregular. After this results, a network with one 10-neurons layer has been also tried and the same results have been obtained.

To vary the number of hidden layers, the results of varying the number of neurons of a single-layer network have been taking into account. Accuracy has been evaluated for networks of 1 to 14 layers of 10 neurons each.

Figure 5 shows a clear decrease in accuracy as more layers are used. Moreover, standard deviation increases with the number of layers. This shows that using more layers makes the model less accurate and steady. This is probably because using too many layers leads to overfitting the model to the train set.

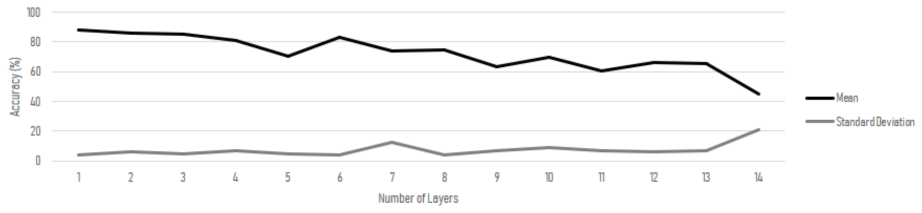
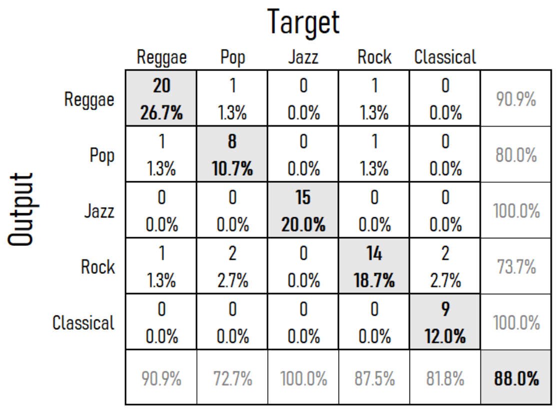


Figure 5: Mean (black) and standard deviation (grey) of the accuracy values obtained by neural network for different number of hidden layers of 10 neurons each.

## 3.3 Best Result

After analyzing all the results above, the best configuration has been determined (mean and covariance, 13 MFCC, 30-second non-splitted clips, 20 Hz - 8000 Hz filterbanks, one-layer network with 10 neurons) and executed. Its confusion matrix is displayed below:



| Output \ Target | Reggae | Pop | Jazz | Rock | Classical | |
|---|---|---|---|---|---|---|
| Reggae | **20** **26.7%** | 1 1.3% | 0 0.0% | 1 1.3% | 0 0.0% | 90.9% |
| Pop | 1 1.3% | **8** **10.7%** | 0 0.0% | 1 1.3% | 0 0.0% | 80.0% |
| Jazz | 0 0.0% | 0 0.0% | **15** **20.0%** | 0 0.0% | 0 0.0% | 100.0% |
| Rock | 1 1.3% | 2 2.7% | 0 0.0% | **14** **18.7%** | 2 2.7% | 73.7% |
| Classical | 0 0.0% | 0 0.0% | 0 0.0% | 0 0.0% | **9** **12.0%** | 100.0% |
| | 90.9% | 72.7% | 100.0% | 87.5% | 81.8% | **88.0%** |

Figure 6: Confusion matrix resulting of using mean and covariance, 13 MFCC, 30-second non-splitted clips, 20 Hz - 8000 Hz filterbanks and one-layer network with 10 neurons

By retraining this model 1000 times it has been found that the mean accuracy obtained is 87.6% ±3.7%. Therefore, figure 6 shows a confusion matrix of a very close result to the mean accuracy.

## 4 Future Work

Some more research on this topic is required to make an even better model. Some things that could be done are exposed below.

The same model could be tested with a different number of music genres. More accuracy with less genres is expected than when using more genres.

A different database could be used. Some database with clips with different properties - length, file format, sampling rate... - and with a different number of samples. It is expected that a database with a larger number of samples would give better results because the neural network could be better trained (75 training samples per genre are relatively few).

Using a different method to build the neural networks could be also tried. The method used, MATLAB's `patternet`, is limited and has a lot of predetermined configuration. Some further tweakings to the ones performed could be tried to find a neural network that fits better our purposes.

To further evaluate our model's accuracy, some tests with songs that are not contained in the database used to train the model could be used.

Also, some other audio features than the MFCC could be tested. However, these have shown to perform well and are recommended in most musical genre classification papers.

## 5 Conclusion

As explained above (section 3.3), the best way to obtain the MFCC values among the different options that have been tested in this paper is to use the full 30-second clips of the database and to extract the first 13 MFCC from a 26-filter filterbank that extends from 20 Hz to 8000 Hz.

To feed the artificial neural network, the mean and covariance method has performed well; it is explained at the end of section 2.2.

The best architecture for the MATLAB's `petternet` artificial neural network has been shown to consist in a single hidden layer of 10 neurons.

By using this, an accuracy of 87.6% $\pm$3.7% has been achieved. This represents a good performance that can be classified as over average compared to some similar papers.

However, it is believed that this accuracy could be improved by developing some of the further research explained in section 4.

## References

[1] *Datasets - Marsyas*. URL: `http://marsyas.info/downloads/datasets.html`.

[2] Stuart Rosen and Peter Howell. *Signals and systems for speech and hearing*. Vol. 29. Brill, 2011.

[3] George Tzanetakis and Perry Cook. "Musical genre classification of audio signals". In: *IEEE Transactions on speech and audio processing* 10.5 (2002), pp. 293–302.