Automatic music genre classification of MIDI files

Politecnico di Milano – Numerical Analysis for Machine Learning

INTRODUCTION

This report discusses the techniques employed in the practical project for the Numerical Analysis for Machine Learning course, which aimed to classify the musical genre of a MIDI file using the approach described in [1]. The process involves segmenting the file into sections representing different musical sequences, followed by the extraction of six features from each section to be input into a neural network (NN).

However, the average and standard deviation of pitch and intensity, combined with vocal range and tempo, may not be sufficient indicators of musical genre, despite the paper's claim of achieving a 90.1% accuracy. Additionally, the paper lacks clarity on other techniques and architectures used to develop the NN model. Although it mentions the use of a BIGRU network in the conclusions, there are no further details provided.

It is also perplexing that the analysis of the experimental results refers to high-order moments for music classification of '.wav' files, even though the paper's objective was to build a model for classifying MIDI files.

This project tries to incorporate all the ideas presented in the article, adhering to the methodology as closely as possible, even in instances where alternative choices may have led to better results.

REFERENCE PAPER

A. Summary

In the paper, Qi He outlines two primary steps: segmentation and feature extraction [1].

Segmentation is performed using Foote's method [2]. First, a piano roll is generated from the MIDI file, merging all the instruments together. The result is a 128xM matrix, where M represents the number of samples collected at a chosen frequency. Each row corresponds to a specific pitch, and the values indicate the note velocities. Next, a self-similarity matrix (MxM) is computed using the Euclidean distance to measure similarity between frames. This matrix is symmetric, with zeros along the diagonal, and higher values correspond to lower similarity:

$$D(i,j) = \sqrt{\sum_{k=1}^{128} (x_{ik} - x_{jk})^2}$$

The next step involves generating a novelty function using a convolution kernel formed by a Kronecker product:

$$K = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

The first matrix can be expressed as:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

The first term accounts for the self-similarity on either side of the centre point, while the second one measures the cross-similarity between the two regions. The difference between the two values estimates the novelty of the signal at the centre point. Typically, kernels are smoothed (e.g., with a Gaussian function) to minimize edge effects, assigning higher values at the center and reducing values near the borders. The Kronecker product allows this approach to be applied to larger kernels. A larger kernel corresponds to a wider time window

for detecting novelty. For instance, a 10x10 kernel used with a self-similarity matrix sampled at 10 Hz detects novelties within a one-second window, which may not be optimal for modern music.

After constructing the kernel, it is convolved with the self-similarity matrix, though only the diagonal values are retained. Each diagonal element represents the novelty function of a time frame. The authors also noted that they zero-padded the matrix before performing the convolution, enabling the calculation of novelty values for time frames near the edges. Finally, the peaks of the novelty function are identified to determine the beginning of new segments. The second step, instead, is feature extraction. The authors have identified six meaningful characteristics that are supposed to account for rhythm, tunes and strength:

1. Average pitch
2. Standard deviation of the pitch
3. Vocal range
4. Playing speed (tempo)
5. Average velocity
6. Standard deviation of velocity

In the conclusion it is also mentioned that they used this sequential data to feed a BIGRU neural network.

B. Criticism

The first thing that comes to mind reading the paper is the extraordinarily high accuracy of 90,1% that the authors claimed to have achieved with such a few basic features. Paragraph four is not very clear about what experiments they are describing, but they mention a paper that deals with Mel-frequency cepstral coefficients and probably they somehow incorporated these into their model too, even though it is not explained how. However, this is quite strange considering that MFCCS can be derived only from a '.wav' file or similar formats. What would be the point of converting a MIDI into a wave file, if the whole point of the method is to try to build a model that works only with event-based files like MIDIs?

Moreover, there is also some uncertainty about the six features they identified:

1. How is vocal range calculated if there is no instrument corresponding to voice in MIDIs? The vocal melody is usually represented by the instrument that fits best with the others, but there is no unequivocal way to identify which one plays the vocal part.
2. Velocities in MIDI files are not always accurate and most of the times flat. They can range from 0 to 127, but many files are not high-quality and do not take into account intensity. How can two out of six features be based on such unreliable values?
3. A new version of a song that has been shifted up significantly in pitch, even by as much as seven semitones, remains within the same genre as the original. So, why is average pitch being used as a feature? In fact, pitch-shifting is one of the first techniques that comes to mind when augmenting a dataset. This raises the question of whether average pitch is a reliable indicator of genre, given that significant changes in pitch don't necessarily alter the genre classification.

The hypothesis that the set of features they described is incomplete is confirmed by the fact that they did not use any information about instruments, which play an important role in determining the genre of a song. For example, metal pieces are characterized by electric guitars and heavy drums, while disco/techno music comprises more synthesizers and pianos.

DATASET

The dataset mentioned by the paper has not been disclosed, even after requesting it. Hence, the results have been very different, especially because no clean dataset has been found on the internet. The one used in this project has been built putting together several repositories. It comprises five genres:

1. Classical music, taken from the *MAESTRO* dataset [3] (513 pieces)
2. Country music, taken from [4] (738 pieces)

3. Jazz music, taken from [5] (750 pieces)
4. Electronic music, taken from the *LAKH* dataset [6] (638 pieces)
5. Metal music, taken from the *LAKH* dataset [6] (486 pieces)

The choices of the categories have not been casual: the goal was to heuristically try to make genres as different as possible: classical is easily recognizable by its length, lack of drums, long sections and strings presence; country by its guitars and simple structure, jazz by its complex harmony, brasses sounds and intricate rhythm patterns; electronic by its synthesizers, heavy basses and punchy drums; metal by electric guitars, fast-paced drums and high intensity. However, it is important to point out that these characteristics are not as easy to identify in a MIDI file compared to a recording. This is due to the fact that playing style is a key factor in distinguishing genres. For example, jazz syncopation is hardly detectable in an event file as well as the harsh vocals that tell apart metal from rock, and the thick accent that makes a song sound like country instead of pop. This shows how MIDI music classification has strong limitations, especially considering that the label that we want to determine needs to be consistent with the 'recording version' of the song and not the event transcription, which can sound significantly different.

A. Exploration

The classical music pieces appear to very similar to each other: they are lengthy piano recordings which have then been translated into MIDI files. On the other hand, spot-checking of the labels for the other genres' labels has been performed. Unfortunately, there appears to be many wrong or doubtful classifications. For example:

1. *you_shouldnt_kiss_me_like_this_wr_kar.mid* is labelled as country, but it sounds more like jazzy pop (probably due to the MIDI/recording difference)
2. *windmills_of_your_mind_sn.mid* is played by strings that resemble more classical music, even though it is a jazz piece
3. *803b424aa1cef2da475a789fe2a651df.mid* is a metal piece whose sound is far from it; this is probably due to a misclassification
4. *187a1d0b21552c514390de8bf29a88fe.mid* is an electronic song that doesn't have any synth in it. It surely sounds more like a pop piece

These are just some of the classifications that make this dataset very noisy and difficult to learn. However, as shown later classical music is easy to identify because the *MAESTRO* dataset is much cleaner and has quantifiable characteristics that make its pieces different from the others like length and the fact that only a piano is playing.

An important note needs to be mentioned about electronic and metal files: they both come the *LAKH* dataset, whose songs have been automatically matched to the *one million* song dataset and labelled accordingly. This approach has obviously resulted in several misclassifications.

B. Cleaning

The first step to cleaning was to remove all files that weren't readable by the python library *mido*, which has been used for feature extraction. Then all only-drums songs were removed because they can neither be segmented (since the pitch is all flat) nor provide enough useful information to be classified.

While additional cleaning could have been performed, it wasn't deemed worthwhile due to the excessive noise in the dataset from the start.

MODEL

A. Features

Aside from the six features presented in the paper, three additional attributes have been extracted for each song segment.:

1. Duration
2. Instruments, a one-hot encoded array that tells whether a group of instruments is present in the segment. For example, if the first value is one it means that pianos are playing. (The categories are fifteen and have been constructed arbitrarily, based on the classification purposes of this project)

3. The average and standard deviation of the first five MFFCS, extracted from the *.wav* file obtained by converting the MIDI with *librosa*.

As far as the last features are concerned, it is undoubtedly intricate, time-consuming and ultimately wrong to extract them and use them for MIDI classification, but the goal was to stick to the paper as much as possible within reasonable limits.

To summarize, the features of a song are a list (variable length based on the segments number) of numpy arrays of length 33:

- (0) Average pitch
- (1) Pitch deviation
- (2) Vocal range
- (3) Tempo
- (4) Average intensity
- (5) Intensity deviation
- (6) Duration
- (7-16) MFCCS
- (17-32) Instruments groups

To make the shape of the features suitable for a NN, the sequences have been zero-padded to make sure that they are of the same length.

B. Neural network

The neural network is composed of four layers:

1. BIGRU layer of 20 units (preceded by masking layer to ensure that zero-padded segments are ignored)
2. Dense layer of 13 units with activation function *tanh* and l2 regularizer set to 0.01
3. Dropout layer set to 0.3
4. Output layer of 5 units with activation function *softmax*

Other parameters include:

- Learning rate: 0.002
- Batch size: 50
- Optimizer: Adam
- Maximum number of epochs: 50
- Early stopping: 5 epochs patience – 0.02 delta on validation loss
- Maximum epochs: 50
- Validation split: 0.2

The layers sizes, learning rate and batch size have been adjusted manually to try to reduce the number of parameters and improve performance. The dropout layer and the regularizer, instead have been inserted to prevent the model from overfitting, especially on such a noisy dataset.

RESULTS

The following results compare the performance of the model with different features sets, to try to understand whether the ones described in the paper are effective or whether they can be improved.

A. Full experiment

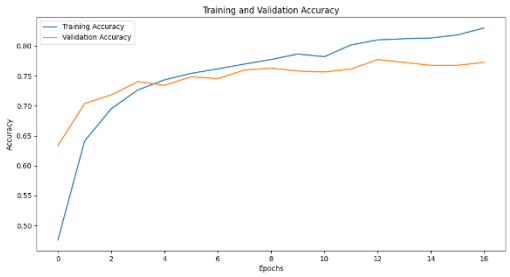
The final validation accuracy of the experiment with all the features is around 0.77 and the confusion matrix obtained is the following:

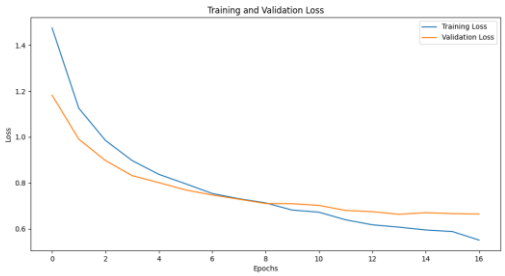
	Classical	Country	Electronic	Jazz	Metal
Classical	100	0	0	0	0
Country	0	74.31	4.86	16.67	4.17

Electronic	0	6.98	68.99	13.18	10.85
Jazz	1.29	14.84	5.16	77.42	1.29
Metal	0	11.11	12.12	6.06	70.71

The first thing that stands out is the high accuracy on classical music. As stated above, this is probably due to the distinctive characteristics of this subset of the dataset. On the other hand, the model mainly struggles to tell apart country from jazz and electronic from metal. As far as the first couple is concerned, they are tricky to distinguish because they sound more pop from a MIDI file, since a key characteristic of these genres is playing style. The second couple, instead, is probably similar in tempo, bass sounds and punchy drums.

However, considering the low-quality dataset available, the results are arguably better than expected.





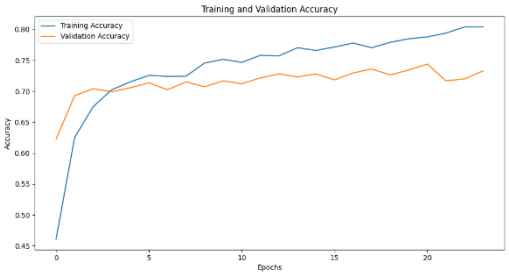
B. MFCCS experiment

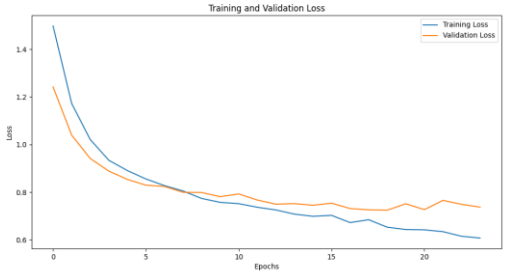
Given the doubts about the role of MFCCS, another experiment has been conducted, where these coefficients were left out from the features. The new confusion matrix is:

	Classical	Country	Electronic	Jazz	Metal
Classical	98.98	0	0	1.02	0
Country	0	75.69	1.39	9.72	13.19
Electronic	0.78	13.95	61.24	9.3	14.73
Jazz	3.23	19.35	3.23	68.39	5.81
Metal	0	12.12	9.09	10.1	68.69

The accuracy is approximately 0.72 which is slightly lower than the full experiment. Given the fact that the increase is only 0.05 and that the extraction of these coefficients is extremely

time-consuming, it is probably best to leave them out.



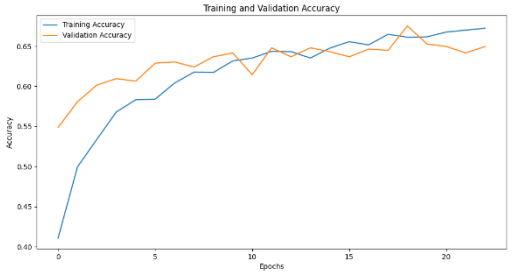


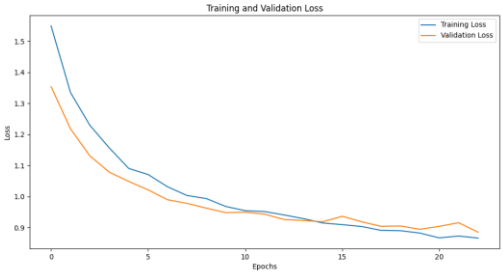
C. Basic experiment

Just to have a reference, another experiment was carried out with only the first six features mentioned in the paper. The results are not disappointing but definitely not as stellar as the authors claim. The accuracy was just 0.64 and the confusion matrix shows how the model struggles to identify especially jazz songs.

	Classical	Country	Electronic	Jazz	Metal
Classical	95.92	0	1.02	3.06	0
Country	0	69.44	8.33	13.19	9.03
Electronic	1.55	24.03	56.69	5.43	9.3
Jazz	0.65	36.77	10.32	48.39	3.87
Metal	0	20.2	20.2	5.05	54.55

Nonetheless, given that the features are few and very basic, it is undeniable that the performance of this model is not bad.





FUTURE WORK

There are a lot of points of improvement for this project, starting with finding a cleaner dataset. Firstly, it should be hand-labeled to make sure that not only the genre of the song is correct, but that is also distinguishable from sound of the MIDI file.

As far as features are concerned 33 is definitely a high number, so selection should be the next step, considering that the results of the current set indicate that it is effective, even if redundant. However, it lacks information about rhythm and harmony, which can be crucial in classifying the genre. For example, jazz has very complex chords sequences, as well as strong syncopation, while country doesn't. Hence, these factors could improve the model significantly.

Another area where more experimenting could be done is the NN architecture. This project

only took into consideration a BIGRU architecture with one subsequent dense layer, but no other option has been explored.

CONCLUSION

The method described in Qe He's paper [1] proved to be effective, but not as satisfactory as expected: the accuracy achieved with the six features highlighted in the article was only 64%, contrary to the 90.1% claimed by the author. A key factor that determined this result was the dataset, which was different from the original, and probably much noisier. However, this likely wasn't the only cause of such a different performance: the set of features mentioned is probably incomplete and could be enriched by information about rhythm and harmony. A further step would be to compare the features and select the ones that lead to better performance or find combinations that can shrink the dimension of the input.

REFERENCES

[1] He, Q. (2022). A music genre classification method based on deep learning. *Mathematical Problems in Engineering*, *2022*, 1–9. https://doi.org/10.1155/2022/9668018

[2] Foote, J. (2002). Automatic Audio Segmentation Using a Measure of Audio Novelty. https://doi.org/10.1109/icme.2000.869637

[3] Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck. "Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset."

[4] Bob, B. W. (n.d.-b). MidKar Country Music MIDIs. https://midkar.com/Country/Country_A_to_Z.html

[5] Bob, B. W. (n.d.). *Jazz MIDI Index page*. https://midkar.com/Jazz/Jazz.html

[6] Colin Raffel. "Learning-Based Methods for Comparing Sequences, with Applications to Audio-to-MIDI Alignment and Matching". *PhD Thesis*, 2016.