Feature extraction and non-binary bass line classification in a drumbeat generator application

Jasmin Curtz

Department of Computer Science, HAW Hamburg

Berliner Tor 7, 20099 Hamburg

March 7, 2022

Abstract

AI art is becoming more and more popular and has paved the way for both new forms of artistic expression and new business concepts. In the music industry, so-called AI music generators have already been commercialized to create royalty-free music for content creators. But instead of just generating static music, this work examines how human and machine can interact to play music together just like musicians in a jam session and which technical barriers need to be broken down in order to achieve this goal.

Keywords: AI Music, Feature Extraction, Classification

1 Introduction

1.1 AI and music

Art has been a fundamental aspect of human society since the early days of mankind. From stone-age cave paintings over Roman statues we've evolved to enter a new era of art: AI art. It does not come as a surprise that artists and scientists alike are interested

in using machine learning technology and artificial intelligence as an art medium. For visual arts, this has already been accomplished and popularized by platforms like Artbreeder [2], an online tool that allows users from all over the world to create AI-generated images reaching from landscapes over abstract art to portraits and covering styles from rough painting to photo-realism. Tools like these rely on generative adversarial networks (GANs) which are trained to create fake images by learning from large data sets of real images. When it comes to music, projects like Sounddraw [4] or Amper [1] have already succeeded in commercializing AI-generated tracks by promising unlimited royalty free and individual music for content creators on social media platforms like YouTube. However, these companies probably do not fully rely on AI to generate their music (this must be framed as an assumption only due to lack of technical insight on the companies' websites), instead they most likely utilize libraries of pre-made musical phrases[1].
Everytime we try and teach AI how to make music, we face two problems: The first is that before we can build machine learning algorithms and neural networks, we must understand the mathematical concepts of music. Depending on the musician given this task, this will either be considered a challenge or a ridiculously easy task. When learning music by only listening and playing, it can be hard to break music down to just mathematical concepts, but someone who has learned music by theory will know that much of popular music can be described very simply only by numbers. If we assume we only play music in $\frac{4}{4}$ time signature and we know both the tempo of a song in bpm and the song's length, we can easily calculate that a 1 minute song played in 60 bpm consists of 15 equally long bars. Each bar can be broken down into 4 quarter notes that each have a length of 1 second. These again can be divided into eights with the length of $\frac{1}{2}$ second and so on. But now, things get more complicated as we dive deeper into the rhythmical variety that music has to offer. A quarter note could also be divided into three, making it a triplet of notes with the length of $\frac{1}{3}$ second as well as a sextuplet ($\frac{1}{6}$ second) or even a quintuplet ($\frac{1}{5}$ second). This however is still not enough to comprehend the mathematical concepts of music because there are not only whole, half, quarter and eighth notes, but there are also dotted half notes that equal three quarter notes or dotted quarter notes that equal three eighths or even quadruplets (a note divided into four) on dotted quarter notes. The possibilities are endless and drummers like Benny Greb have understood very well how to utilize all different kinds of polyrhythms [2]. Moving away from rhythm, there are also underlying mathematical concepts to melody and harmony. Western music uses a range of 12 notes (C to B), 15 major keys and 15 minor keys as well

[1] A phrase is a small set of notes forming a section of a melody or rhythm
[2] Overlapping different measures like 3 against 4 or 5 against 7

as different modes (such as dorian or phrygian) and scales in these keys. Chords build up in defined intervals, chord progressions follow schemata like I-IV-I-I-IV-IV-I-I-V-IV-I-V (a 12-bar blues) or I-V-VI-IV, which is commonly used in western pop music.
But even though all this leads to a large variety of concepts, they can still be described by simple and deterministic rules. It would not even require machine learning to implement an application that can create music by following these rules. Just as musicians work together, the user could simply tell the application to play a 12-bar blues in E major and would get a satisfying result. What instead makes computer generated music all the more complicated is the concept of musicality. Great musicians do not just repeat and play all the lines they have memorized, but they improvise, they invent, they create. And this requires not just knowledge of music theory, but also creativity, artistic sense, intuition and feel - all of which computer science struggles to reflect in AI. So there comes the leading question of this project, which is: How can a computer learn to play along with real musicians?

1.2 AI drum machine

As a bass player myself, I stand between the rhythmic and the harmonic section of my band. The best friend of every bass player is probably the drummer - not only because they share the experience of being the most overseen members of the band while at the same time keeping everyone on time, but also because their instruments complement each other so well. For this reason, I wanted to create an AI drum machine that I could practice with just like I would with a real drummer. What I would do practicing with my band's drummer is to first agree on a tempo and a time signature, then I would play a few bars of my bass line to provide an understanding of the style I am aiming for before the drummer finally joins in and plays a beat that he considers suitable (leaving out the times he just wants to show off the new polyrhythms he has just learned). For the drummer, this requires the following skills and experience:

1. Having memorized a large amount of different drum beats.
2. Categorizing the bass line that I am playing.
3. Being able to create something new based on the memorized beats and the reference given by the bass.
4. Predicting how the music might continue.

Beside this, an important feature of a good drum groove is that it accentuates the rhythm of the bass line or guitar riff. This can only be achieved by the drummer closely listening to what the rest of the band are playing and placing every accent exactly where is can unfold the greatest impact. Also, fills[3] make up a huge part of drum lines, which are another challenge to artfully place and play.

1.2.1 Benefits of an AI drum machine

Even in times of social media it can be hard for young musicians and especially beginners to find a group to practice with. But playing with other musicians is essential for improving on your instrument, learning how to coordinate within an ensemble and developing a general understanding of rhythm, precision, harmony and overall musicality. Also it helps provide both inspiration and motivation which can be hard to find when only playing on your own. Browsing through YouTube, one can already find a large variety of backing tracks to serve the purpose of substituting a band - but using AI has the potential of taking this idea to yet another level. Instead of being bound to a backing track, the musicians themselves can lead the jam session, knowing they can rely on the drum machine to follow them whatever they play.

1.3 Technical concept

To simulate what a human drummer needs to accomplish, the applications must

1. be capable of recognizing and categorizing (classifying) a bass line,
2. know different grooves and
3. create or choose a groove given the bass line as a reference.

This requires a neural network that can perform the classification of bass lines, taking in consideration their rhythm, melody, tempo, maybe even tone and general style. For generating the drum track, there are two options already hinted to in the introduction:

1. A generative adversarial network (GAN)
2. A library of drumtracks and phrases

[3] A passage to fill a break in the melody or rhythm, usually played differently from the underlying groove

1.3.1 GANs

Generative adversarial networks consist of two networks working against each other, one being the generator and one being the discriminator. While the discriminator trains to recognize fakes, the generator trains to create convincing fakes. Together, GANs can be trained to generate fakes so realistic even humans cannot recognize them anymore. The project MuseGAN [7] by the Academia Sinica uses the GAN concept to create midi music via two different methods they call „Generation from scratch“ and „Track-conditional generation“. The second method fits best what the AI drum machine aims to achieve: A human user specifies a sequence of music which the network uses as a foundation to generate the music. But while the MuseGAN project has achieved some impressive results, it cannot nearly compete with commercial tools like Sounddraw and once again showcases how complicated it is to train AI to generate anything that does not only follow the rules of music, but also sounds pleasant and musical.

1.3.2 Libaries and phrases

Going back to comparing the technical concepts to human musicians, a purely GAN based application would resemble a child not knowing anything about music theory or playing an instrument being put on a drum stool, given some drum sticks and then being told to play something. The child randomly hits the drums, perceives the reaction of the listeners and might slowly start to learn how to play something that does not make the people in the room covering their ears. If you wanted to train this child to actually become a musician, they would have to practice with more concept: First learning simple beats, then fills, then rudiments, then polyrhythms and so on. Thereby, the child starts to build up a kind of library in their head and someday they will be able to play something musical on their own. For a drum machine application, the use of libraries and phrases will also help to create better and more musical tracks. This concept is not only very reliable in that it will always result in an actual groove, but it also requires much less training and therefore much less resources and data.

1.3.3 The problem with real time

A drum machine that can successfully replace a human drummer in a jam session must be able to perform both the classification and the drum track generation in real time.

However, this comes with two major challenges, one being processing speed and the other being synchronisation. Just as a human drummer, the application must first listen to the bass player, then it needs to process the information and decide on what to play, and lastly it needs to join in exactly on the correct beat. Data processing always takes some time, which means the drum machine would have to make up for that delay by purposely coming in too early. What must be considered in the same context is the used audio driver. If the latency is too high, this makes the synchronisation even more complicated. Beside that, it must also considered how to separate input and output. Imagine the drum machine using a microphone for recording the bass and speakers to play the groove. Chances are the microphone will start picking up the signal from the speakers, distracting the signal coming from the bass. Also, other background noise could be recorded and must be filtered out. The best way to avoid this is by using an audio interface, which would also solve the problem of latency.

1.3.4 Data format: midi or wave?

When creating AI generated music, that leads to the question of which data format to preferably use. Two different approaches are being used by different projects, both offering different advantages and disadvantages. The project „Jukebox“ [6] from 2020 creates music with lyrics and convincing vocals. Therefore, it needs to produce wave audio signals that imitate instrumental backing tracks as well as a human voice. While this approach offers music that can be listened to immediately, it requires the implementation of a decoder and encoder as well as a complex network structure. Also, this method is prone to glitches, unreal sounds and eerie side effects.
Another common approach for projects like MuseGAN is that using midi. Midi is short for Musical Instrument Digital Interface and has been around since the rise of electronic music in the 1980s. Until today, it remains the standard data format for storing musical information and can be easily converted into sheet music or rendered into audio using samplers. Modern music production heavily relies on midi as it is easy to work with and the most versatile form of recording music. The only limitation when it comes to creating AI music in midi format is that it cannot be played back immediately by a standard media player, but instead needs to be interpreted and rendered by a sampler first.

1.3.5 Specification

In order to implement the project within the given time frame and with limited resources, not all of the above aspects can be incorporated and some simplifications must be made:

- The application does not perform in real time.
- The input is provided as wave data and should be free of background noise.
- Only tracks in 4/4 time signature are allowed.
- The applications only takes 8 bars of a bass line as input and only creates 8 bars of drum track as an output.
- A consistent tempo is required and must be declared by the user.
- The application uses a library of drum tracks to create its output.
- The output format is midi.

Following these requirements, the applications will perform the following steps:

1. The user provides a bass line as a wave file and declares the tempo in bpm.
2. A neural network classifies the style of the track.
3. A set of pre-built midi drum tracks in the corresponding tempo and style is created inside an output folder.

2 Implementation

2.1 Frameworks

For audio import and feature extraction, the Librosa library is used, which provides a framework of different functions for audio and music processing. For exporting midi, the Mido library is used. The neural network will be implemented using Keras from Tensorflow.

2.2 Data

2.2.1 Selection and preparation

To train the neural network, a data set of both bass lines and drum tracks in different styles and tempos is required. The bass lines need to be converted to wave audio while the drum tracks will be stored as midi files. Using the rock song midi library from midiworld.com [3], a set of 230 tracks in total was selected and prepared following these steps:

1. Isolating 8 bars of a song
2. Rendering and exporting the bass line as a wave file
3. Exporting the drum track as a midi file

The bass lines will be used as training and test data whereas the drum tracks will be used to create libraries for each class.

2.2.2 Classes

Each wave file is assigned to one of three tempo categories and one of 19 style classes. The tempo categories are:

- M (Moderate): from 67 bpm to 120 bpm (inclusive)
- S (Slow): Up to 66 bpm (inclusive)
- F (Fast): Above 120 bpm

The classes are: Alternative, Blues, Blues Rock, Disco, Folk Rock, Funk, Funk Metal, Grunge, Hard Rock, Hip Hop, Industrial, Nu Metal, Pop, Power Ballad, Progressive, Rock'n Roll, Slow Rock, Standard Rock, Thrash Metal.
Table 1 shows how many files were used to train each class. The lack of usable data for some classes will probably lead to an unbalanced classification with strong tendencies towards classes like Blues Rock, Hard Rock, Grunge or Thrash Metal, but as the data preparation took a lot of work, this could not be improved without losing too much time for other aspects of the project.

Alternative	6
Blues	5
Blues Rock	30
Disco	7
Folk Rock	4
Funk	3
Funk Metal	22
Grunge	27
Hard Rock	28
Hip Hop	4
Industrial	4
Nu Metal	11
Pop	18
Power Ballad	5
Progressive	15
Rock n Roll	6
Slow Rock	4
Standard Rock	8
Thrash Metal	23

Table 1: The number of wave files per class used to train the network

2.3 Feature extraction

The uncompressed wave data format is the standard used when working with digital audio. However, due to lack of compression, files with this format tend to be very large. The most common sample rate when recording or rendering audio is 44.1 kHz, which is two times the highest frequency the human ear can hear. Reducing this rate to reduce the file size can and probably will cause aliasing and is therefore not even implemented in common DAWs. What can be considered to reduce data size is reducing the bit depth of the recording. While this leads to loss of the quality and dynamics of a recording, it can be very useful for processing the data. If 10 seconds of audio are recorded using 44.1kHz sample rate and 16-bit sampling depth, this will result in $44,100\text{Hz} \cdot 10\text{s} = 441,100$ data points of 16-bit float values. Using a vector this large as input for a neural network requires many neurons and therefore slows down the training process. For image data, various methods of downsampling as well as feature extraction are being used to solve this problem. Feature extraction is the process of reducing data to only the information required for a specific problem. In the case of analyzing bass lines, the important information or features are the rhythm and the melody - which is

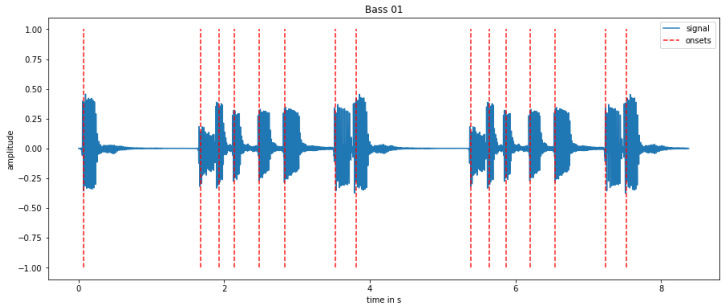

Figure 1: Wave signal and onsets of Dreadlock Holiday by 10cc

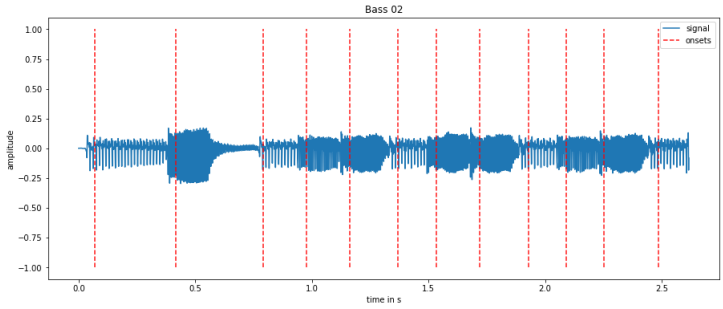

Figure 2: Wave signal and onsets of Forty Six & 2 by Tool

exactly what sheet music, midi data or tabs aim to display.

2.3.1 Onset detection

Onset detection is the process of detecting amplitude peaks in signals such as audio files. In a musical context, this can be used to detect the drumbeat in a song and calculate the tempo in bpm. For analyzing recordings played by a single instrument instead of a whole band, it might be useful to extract the rhythm of a musical line. To evaluate the benefits of using onset detection for feature extraction, tests have been performed with different audio files using the Librosa library.

As seen in figures 1 and 3, the onset detection works best on bass lines that are only moderately fast and very accentuated. Bass lines that instead rely heavily on hammer-ons and ringing notes like the one seen in figure 3 make it harder to detect onsets. The detection becomes even less viable when the signal is distorted like the rhythm guitar line seen in figure 4.

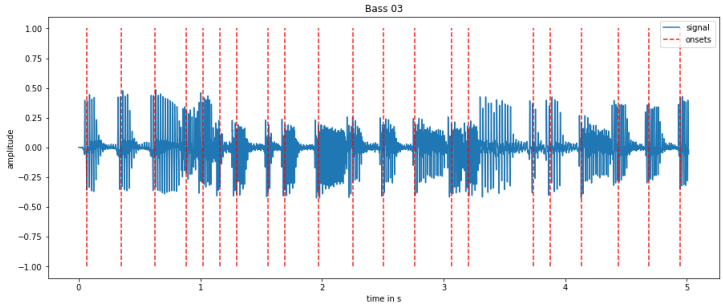

Figure 3: Wave signal and onsets of Hush by Deep Purple

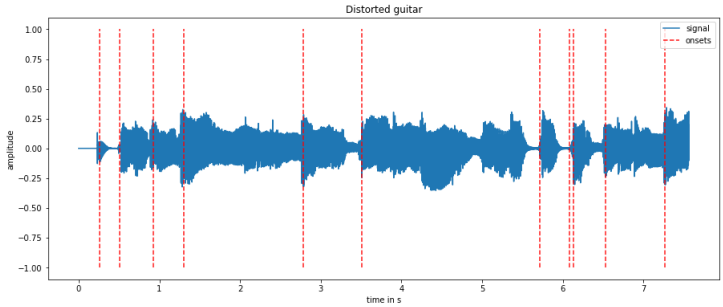

Figure 4: Wave signal and onsets of a distorted rhythm guitar

In conclusion, while the onset detection can be a useful feature for clear signals, it tends to fail when short or ringing notes are being played or effects are used. This could also result in problems in a use case scenario where bass lines are not recorded with good hardware and therefore contain noise. For these reasons, the system cannot rely on onset detection too heavily and other methods of feature extraction must be evaluated.

2.3.2 Q transform and chroma

The Q transform [5] maps frequencies from audio data to a western musical scale reaching from C2 to C8 in relation to time. That makes it capable of outlining the melodic sequence of a piece of music the same way a piano roll does. A chroma vector as provided by the Librosa framework makes use of this transform and again maps it to a spectrum of one octave (C to B), making up 12 so-called pitch classes. This reduces both the noise created by overtone frequencies in the Q transform and the size of the vector. In figures 5 and 6, both the wave signals and the chromagrams of three different bass lines are shown for comparison. The red highlights in the chromagram are the dominant pitch classes

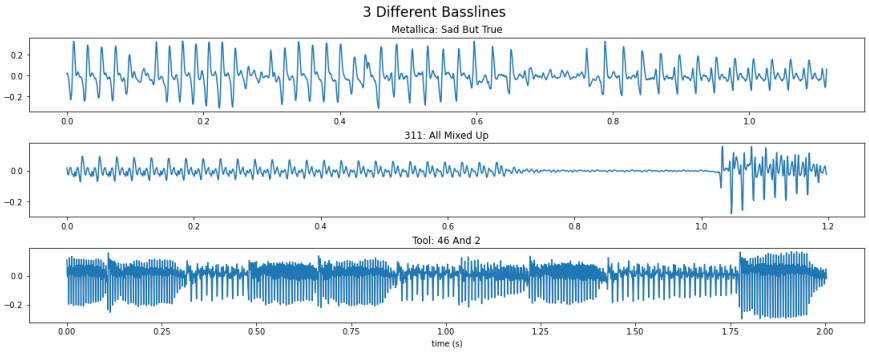

Figure 5: Waves signals of three bass lines used as examples for the chroma feature extraction

Size (x = wave)	Size (x = chromagram)	Data size reduction
Metallica: Sad But True 498176	11688	0.977%
311: All Mixed Up 505600	11856	0.977%
Tool: 46 n 2 576768	13524	0.977%
Limp Bizkit: Rearranged 467968	10980	0.977%

Table 2: Data size reduction achieved by using chromagramms

for each time frame. The different bass lines are very well distinguishable even for the human eye.

In order to test how sensitive the chromagram is to different sounds, another bass line was rendered using five different bass sounds: Fingered electric bass, fingered acoustic bass, picked electric bass, slapped electric bass and synth bass. The results in figure 7 show that the chromagram remains almost the same even if the sound changes, which makes it a suitable tool for extracting melody without considering the tone. Also, chromagrams achieve a huge reduction of data size, as shown in Table 2, which will allow for a much smaller network structure compared to using raw wave data.

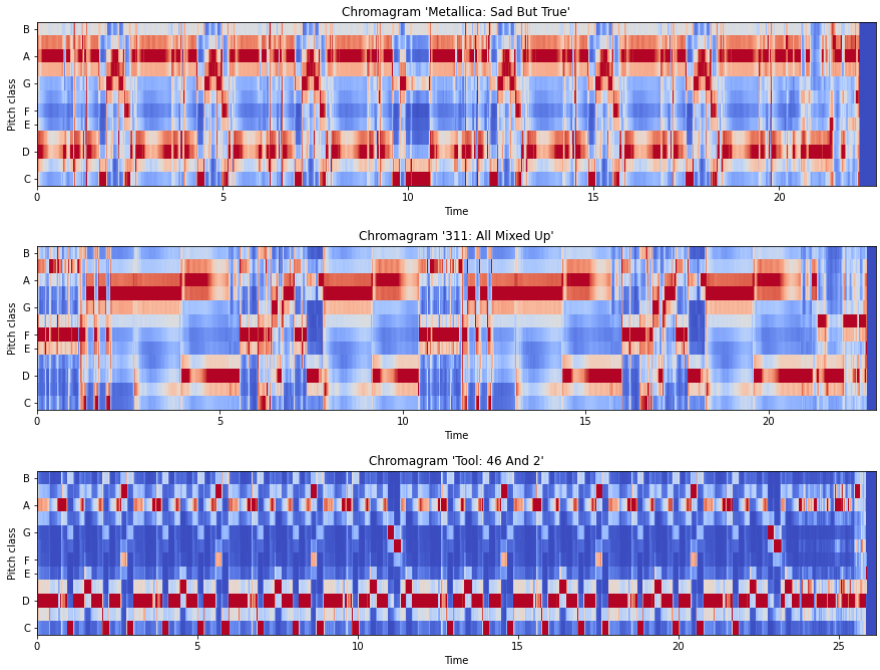

Figure 6: Chromagrams of the bass lines, computed using librosa.feature.chroma_stft

2.4 Network design

Using chromagrams as the tool for feature extraction, wave files can be standardized to matrices of the shape (12, 100) with 12 being the pitch classes and 100 being the time steps. The unified timescale is achieved by compressing the matrix using cv2.resize. While the absolute time is lost in this process, the relative time information required for recognizing the rhythm should remain undamaged. This (12, 100)-matrix is used as the first input of the neural network. The second input required in order to make up for the loss of time information is the tempo class (S, M or F) which has the shape (1,). Because the two inputs have different dimensions and carry different types of information, a multiple input neural network is required. This can be achieved by using the Keras functional API. As shown in figure 8, the network consists of two branches, one for the tempo class and one for the chroma matrix. While testing different approaches on the network design, a 3-layer deep feed forward network for the chroma matrix achieved the highest accuracy of up to 90 %. In the end, both branches are concatenated into one and a last dense layer with 19 neurons (one per class) finally produces the output vector. The complete structure of the network can be seen in figure 8.

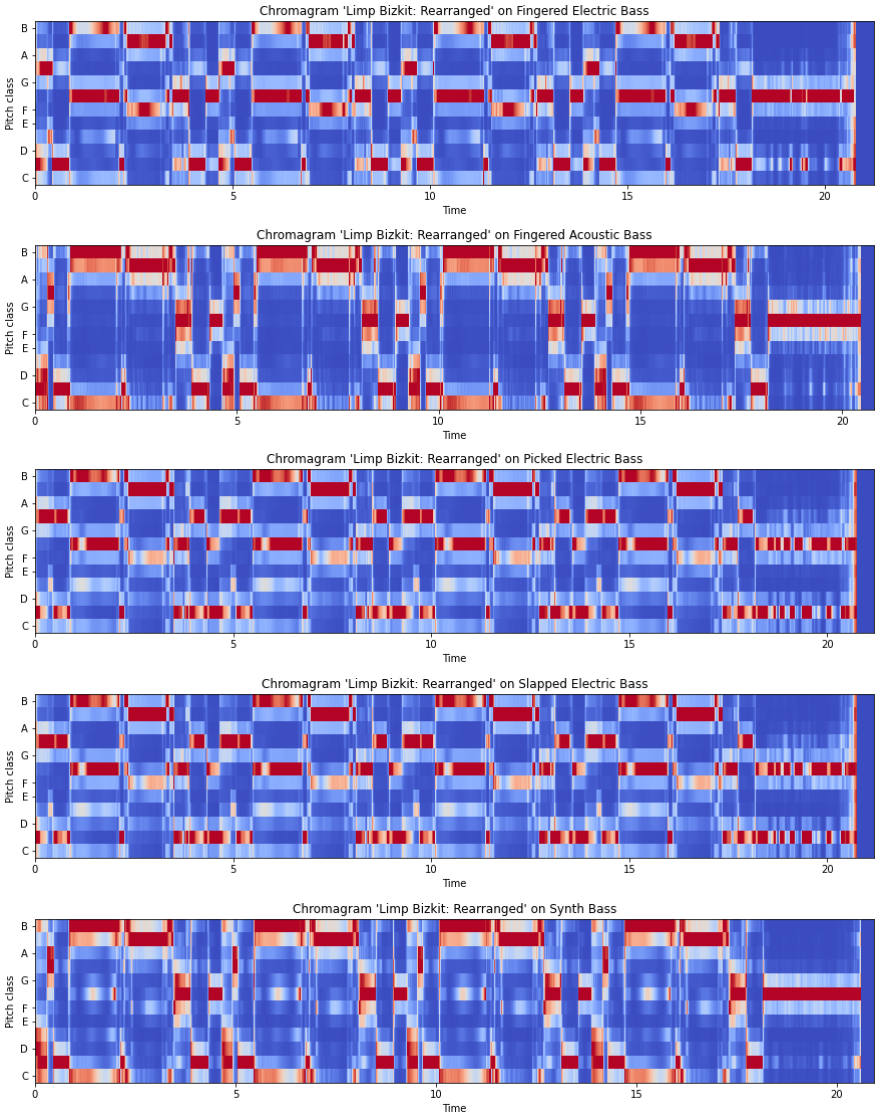

Figure 7: Chromagrams of the same bass line rendered with different bass sounds

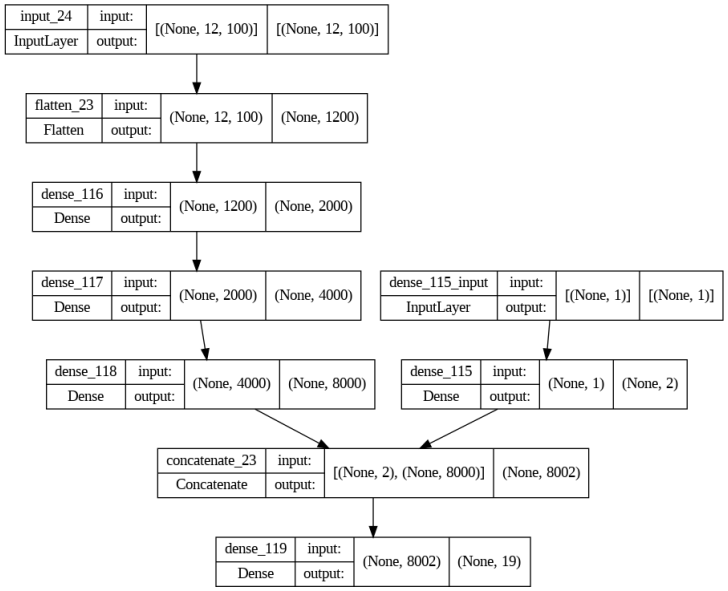

Figure 8: Network model

2.5 Evaluation

The network was evaluated using the keras function evaluate(x, y). This functions returns the accuracy of the network's predictions, which was used as a rough benchmark of how well the network performs. But as the classification problem can barely be broken down to right and wrong answers, further evaluation had to be done by testing and rating the results from a musical perspective.

3 Conclusion and future work

3.1 Tests

The model was trained using a training set of 200 and a test set of 30 files. Using the test data set and the Keras function model.evaluate, the model has achieved accuracies of up to 90 %. After that, the actual performance of the network was tested using a set of 20 different bass lines recorded by me. Some were bass lines that the network had already been trained on (or stupid tests as I liked to call them), but most were lines the network had never seen before. I had my own expectations of what the lines should be classified as, and while the network sometimes shared my opinion, it totally surprised

me at other times and even made me as a musician reconsider the way I perceive music. Some of these interesting results were:

- Stargazer by Rainbow: Originally Hard Rock, classified as Progressive. The iconic band Rainbow, founded by Deep Purple guitarist Ritchie Blackmore, was just like Deep Purple known for the progressive elements in their music, so this classification is not inherently wrong.
- Sunshine Of Your Love by Cream: Classified as Disco. The original song by the iconic rock band Cream contains elements of hard rock, psychedelic and pop. With this mixture, it is hard to put it into only one category, but the classification as Disco still confused me at first. My guess is that the network might have found similarities to some David Bowie bass lines that it had trained on, which were declared as Pop and Disco.
- Bad Guy by Billie Eilish: Originally Alternative Pop, classified as Hip Hop. With her song Bad Guy Billie Eilish has brought a new style of pop music to the world's attention. Unlike the uplifiting and almost kitchy pop music that characterized the beginning of the 21st century, she presented a kind of music that is a lot darker, less melodic, but more rhythmic and probably took inspiration from modern hip hop and rap music.

As already suggested due to the imbalanced training data, the network showed an overall strong tendency towards Grunge and Funk Metal especially.

3.2 The flaws of the data selection

The more surprising results I got from the classification, the more I started questioning my approach on the data selection and preparation. I realized how hard it actually is to precisely define the style of a bass riff by just hearing 8 bars without any further accompaniment. Especially as different genres influence each other, the borders become more and more blurry. Beside that, I also figured it is much more than just the rhythm or the melody of a phrase that defines the style: It is the tone, the attack and the dynamics. A metal bass line would most often be played with a pick, some slight distortion and very heavy and metallic sounding strings and pickups, whereas pop music relies on a much lighter tone as well as fingerstyle playing. Funk on the other hand is dominated

by slap and sometimes even wah-wah-effects. All these aspects should be taken into consideration when trying to classify a bass line.

3.3 The issues with using chromagrams

While the CENS is really good at reducing the data size, it has two flaws. The first one is that it only leaves behind an idea of the melody and rhythm and not of the tone and playing style as mentioned above. The second one is that performing a Q transform on a wave signal (using the Librosa library) is very slow. It takes about an estimated second until 8 bars of music have been loaded, converted to a chroma matrix and then compressed into a (12, 100) shape. That makes this feature extraction useless for a real-time application.

3.4 Possible improvements and further development

Taking in consideration what could be achieved in this project and what still remains to be examined, the following improvements could be made in the future:

1. Turning the drum machine into a real-time application. This would require a completely new concept for both the feature extraction and the network design.

2. Considering tone and style. Therefore new training and test data needs to be created, also the feature extraction and network design must be adapted.

3. Defining new classes. For now, the classification is based on genre. Now that it has become clear how difficult that classification can be even for a human musician, it might be reasonable to focus more on the rhythmic aspect and to define the data classes by different kinds of drum grooves. But still, the classification will always (to some extent) be a matter of interpretation.

References

[1] : *Amper*. https://www.ampermusic.com/

[2] : *Artbreeder*. https://www.artbreeder.com/

[3] : *Midiworld*. https://www.midiworld.com/basics/

[4] : *Sounddraw*. https://soundraw.io/

[5] Brown, Judith C.: Calculation of a constant Q spectral transform. (1988). – URL http://academics.wellesley.edu/Physics/brown/pubs/cq1stPaper.pdf

[6] Dhariwal, Prafulla ; Jun, Heewoo ; Christine Payne, Jong Wook K. ; Radford, Alec ; Sutskever, Ilya: Jukebox: A Generative Model for Music. (2020). – URL https://arxiv.org/abs/2005.00341v1

[7] Dong, Hao-Wen ; Hsiao, Wen-Yi ; Yang, Li-Chia ; Yang, Yi-Hsuan: MuseGAN: Multi-Track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment. (2018). – URL https://salu133445.github.io/musegan/pdf/musegan-aaai2018-paper.pdf