Artificial Intelligence to generate live Algorithmic Dance Music

A new method for live audio

Bas Maat, 3026028

Professor: Marc Groenewegen

Music and Technology
University of the Arts Utrecht

June 7, 2021

Abstract

With the current state of AI music being impressive, but still not at its full capacity. Two factors are supposed culprits of this: audio quality and the speed of generating for live settings. The lack of stereo information and the struggle to retain phase information correctly are the culprits of this problem, the algorithms are computationally heavy and slow. This study aims to improve these problems and make the music more applicable so it can be used for a general audience. In this paper two options are proposed: A different interpretation to create datasets for GAN and using LSTM algorithms for generating data while playing the music live. By giving GAN algorithms both the complex and real domain during training for the entire stereo image the GAN converges way better with recreating this data. To test its applicability to a general audience, a group 53 participants were asked to identify 20 pieces of audio within 10 questions (so two samples per question). The group could give the audio either the label AI or Human. As a result, the group who did not have a musical background (playing or composing) and the ones who did not listen to instrumental music as often were not as able to recognize the difference between an AI or Human. The question is whether AI currently could replace human music. Concluding I would like to suggest a new genre as a result of this research: Algorithmic Dance Music. Since AI music is difficult to recreate as a human, due to the way frequencies are represented over time. ADM could be a new cultural movement where AI lives alongside human music.

Contents

1 Introduction

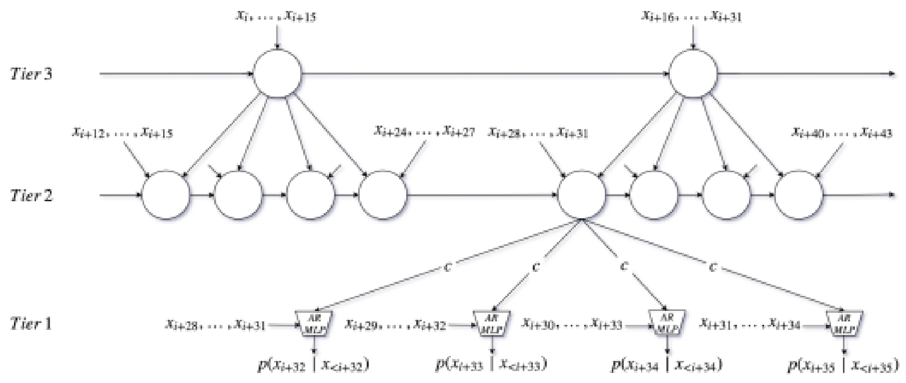
It almost feels like a fever dream when thinking about Artificial Intelligence (AI) taking over human labour. This phenomenon is something that can be seen with day-to-day tasks. Smart home automation is taking over the jobs of maids, self-driving cars the jobs of taxi and truck drivers. Rather not all jobs can be taken over by AI yet, even the aforementioned jobs are still years away from being fully automated. For the time being AI will be used as a supportive role next to humans. This way humans can fully focus on more complicated tasks. AI implementation in the creative field is something that is further away than other tasks, especially in Audio and Music. To focus on one field that is the furthest behind in this field it is live audio, this due to the time sensitive data that live music uses and the still slow processing of AI. In this thesis there will be a closer look at this question, whether AI will be suitable for live audio. Where not only the live audio is the goal, rather the audio must be music and applicable to an audience. This audience in this research are people who enjoy listening to electronic music, here common terms come to mind like: Electronic Dance Music (EDM) and Intelligent Dance Music (IDM). Since this research is focussed on algorithmicizing this music the collective name for this music will be referred to as Algorithmic Dance Music (ADM). The manmade music will be classified as EDM. Ever since we have been able to automate some aspects of music creation there is a particular trend going coming with this evolution in music creation, the number of performers on stage. Going from the years 1850 where classical music arguably was at its peak. With mega orchestras playing all around the western world, there was a shift happening, new technology was being introduced into the music scene. First the Phonograph cylinder, which introduced ways to record sound waves to later be played. This technology rapidly expanded which over time was able to amplify music in a live setting. We needed less people to amplify the sound of a big orchestra, the microphone and speaker combination was quite suited for this. No need for an entire choir or opera singing when a microphone can closely amplify one human voice singing without needing its full body as amplifier. Sooner than later the quantity of people on a live stage started to shrink. Think of Jazz bands being 20 people, later four people in a rock band, in the 80's famous duo's taking the stage or even singers singing over backing tracks (pre-recorded music) and in current day music the epiphany of it all: the DJ. The DJ who does not even sing anymore, the person responsible for stitching songs together over a couple hours to entertain big crowds. One could argue that these DJs are brining nothing new to the table rather than rewinding the old tapes, especially the amateur DJs playing in small clubs who do not create their only music. These live musicians who only play other people their music are stagnating culture and invention of new types and styles of music. I digress, the next step in this live musical evolution would be a very logical one: no more people on stage. By the rate automation is improving, especially with AI becoming better with the year, it would be able to replace the role of humans in a live setting like the club DJ all together. In this scenario the electronic dance artist will not only be sufficient with playing other people's music, rather the artist would be pushed to innovate and write music since the machine can perfectly do its job. The goal of this research is to explore methods to generate live dance music, which will form a new category in the EDM scene: Algorithmic Dance Music. This in a mission to further push humans to be original and create, rather than algorithmically repeat. Computers are good at the latter, rinse and repeat, humans are the ones who were meant to create.

2 An AI for live ADM

The goal of this research is to create a method to generate live music with an algorithm, which sounds and feels human. In this thesis the type of music will be limited to EDM. With the goal being to let an algorithm generate a stream of live audio data without human intervention. The best type of algorithm would be to use AI to self-sufficiently generate audio. The generated audio should be perceived by the listener as some type of music, where the ideal would be to generate EDM. If this is not the case, the outcome could be that this type of music would be part of a new genre: Algorithmic Dance Music (ADM).

3 State of the art

There have already been multiple attempts to generate live music/audio using AI algorithms. One famous example of this is the Relentless Doppelganger [1] by the duo Dadabots. The Relentless Doppelganger is an algorithm that is trained to generate unlimited Metal. The music is live streamed 24/7 on YouTube [1]. The focus of Dadabots is music like Metal, Rock glam Rock. With other examples to be found on their website [2]. To generate this music they make use of the algorithm SampleRNN [2], which is a variation of Recurrent Neural Networks (RNN). This algorithm is based on the ClockworkRNN [3], both of these algorithms make use of the sample basis: RNN. A RNN is trained on time sensitive data. This data, in the case of SampleRNN, are the individual samples that move over time. Where a regular RNN works on one layer the SampleRNN algorithm uses three layers that work with each other, one that takes care of the overarching structure, one middle layer and one that analyses every sample individually. The ad-



$x_i, \dots, x_{i+15}$
$x_{i+16}, \dots, x_{i+31}$
Tier 3
$x_{i+12}, \dots, x_{i+15}$
$x_{i+24}, \dots, x_{i+27}$
$x_{i+28}, \dots, x_{i+31}$
$x_{i+40}, \dots, x_{i+43}$
Tier 2
c
Tier 1
$x_{i+28}, \dots, x_{i+31}$
AR MLP
$x_{i+29}, \dots, x_{i+32}$
$x_{i+30}, \dots, x_{i+33}$
$x_{i+31}, \dots, x_{i+34}$
$p(x_{i+32} \mid x_{<i+32})$
$p(x_{i+33} \mid x_{<i+33})$
$p(x_{i+34} \mid x_{<i+34})$
$p(x_{i+35} \mid x_{<i+35})$

Figure 1: The SampleRNN structure [2]

vantage of this algorithm is the use of direct samples, or audio. This is why the SampleRNN algorithm would be one of the best algorithms for live music generation using AI. There is rather still two big problems that needs to be solved, the audio quality and the lack of stereo information. Due to all the samples being generated some of the samples might experience sudden jumps which will result in added noise to the signal. Enough of these jumps and the signal becomes very noisy. The second issue being that the entire signal is mono, stereo imaging is an important factor of music. This part of music is part of the foundation of current day recorded, but especially live music, that it is too important not to include in research.

[1] `https://www.youtube.com/watch?v=MwtVkPKx3RA&ab_channel=DADABOTS`

[2] `https://dadabots.com/`

Another example of direct sampling is WaveNet [4]. This architecture makes use of a combination between Convolutional Neural Networks (CNN) and RNN. This method uses a similar principle to SampleRNN, rather the audio samples are being analysed by a CNN. The result were quite accurate

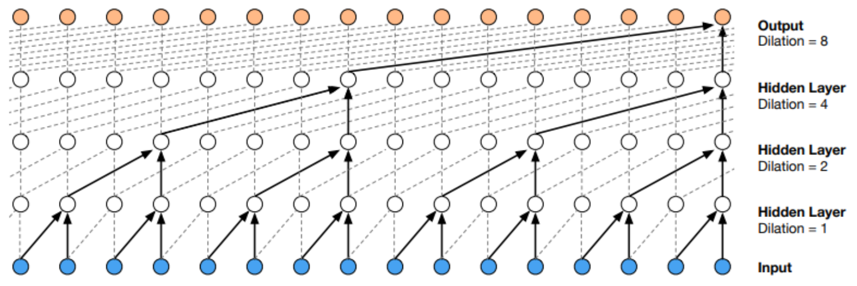


Figure 2: The WaveNet structure [4]

in which it is able to tackle problems of the SampleRNN. Rather this network is not being used much in music since the audio quality of longer musical pieces are still too difficult for the network to train. The result being too noisy. A great application of WaveNet is with Text-To-Speech (TTS). Here this algorithm really shines.

Direct sampling is not the only way of generating live audio. Two other algorithms spectral information to generate music: GanSynth [5] and MelGan [6]. These algorithms are based on the Generative Adversarial Network (GAN)[10]. This network is a well-tested method with images. The GAN is given multiple images of the same type and prompt to generate variations of the given images. With Convolution filters the network can find the most important aspects of an image. In basis images are nothing more than matrixes, with colour images being 3D matrixes.GAN are able to analyse matrixes with spectral information and regenerate them. GanSynth is an extended algorithm of this idea. What makes GanSynth special is that it uses Instantaneous Frequencies (IF)[7] to train the algorithm in combination with Short Time Fourier Transform (STFT) representations. With this data as an input and using a Progressive GAN [8] the results in music is quite impressive. Where GanSynth rather gets its power is with the ability to combine different instruments and seamlessly cross-fade between them, which on its own has different creative applications. One problem that is prevalent in this algorithm is the lack of stereo imaging.

Another approach to representing the spectral data has been done by MelGan, which uses a GAN trained on the Mel frequency scale.[9]. Mel frequencies are a close representation of the human ear, an even better representation than logarithmic frequencies. The use of these frequencies gives a slight improvement over using regular logarithmic frequencies.

A disadvantage to both algorithms is the fact that both are less direct in processing. While the first two algorithms use frequencies directly which makes both faster for live use, the latter two have a better performance audio quality. The problem occurs with speed since the audio must be converted to a spectral representation and after generating these images it has to be translated back to audio. A problem that occurs during this process is the loss of phase information. This results in slightly noisy audio. To circumvent this the use of phase recovery methods is recommended, rather this is another processing layer which in its turn makes the whole algorithm slower.

With these algorithms being introduced into the world the implementation in live audio is still falling short. An example of a live implementation of these algorithms was during the Eurovision Songfestival of 2020. Due to circumstances the festival was cancelled, it was supposed to take place in The Netherlands which is well known for its presentations in the field of AI. During this time the organizers of the Eurovision Songfestival created a contest where the contestants were asked to create music using AI algorithms. Where one of the contestants was Dadabots competing for Germany. Not all contestants were using live music, a lot of performance were AI supported, where they are performing themselves and an AI is helping the performance. With this being an example of semi-live or live AI music there are not many examples of true live AI music, where an AI is performing by itself.

4 An improved method

Previously used methods lack two major factors for achieving the goal at hand: stereo and phase. These two parts in audio are necessary for music to be applicable for a general audience to be interested. Besides these two factors most research lacks the aspect of true live music. The AI that do create live music do need some form of human support to actually be considered music. In this research the aforementioned problems will be answered. Since music can be generated using the General Adversarial Network (GAN) [10] algorithm together with STFT snapshots, there is no reason for this algorithm to use more parameters. Where traditional GAN systems are trained on images, these images are nothing more than a matrix with three dimensions: Width, Height, Depth. The data contained by the matrices can be any values.

Since the data can be unlimited in theory a GAN algorithm can be trained on matrices containing: Frequencies in height, Frequency onsets in the width, and in depth both the complex and real number. This last bit is used to train on not only the Magnitude spectrum, but also the phase data. If we then take the complex and real data of both the left and right channel the matrix will have a shape of: Frequencies, Frequency onsets, Right-Real Right-Complex Left-Real Left-Complex (h*w*4). Where the frequencies and onsets are variable *figure 3*. The data is given to the AI as Numpy matrices as explained above and stored as .npy files, for purpose of representation in this thesis the matrices are converted to magnitude spectrum.

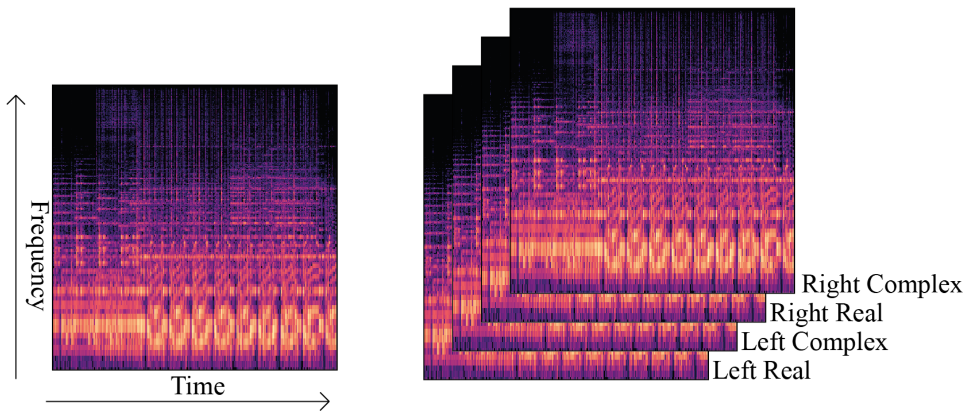


Figure 3: The layout individual of matrices.

Since the goal is to generate EDM, which is beat oriented music, the algorithm will chop up the music into separate beats. In this way the AI can focus on generating the most important part of a piece of music: a beat.

When generating music, the audio will be fully unorganized, this is a problem since we want to transform this audio into fully generated music.

We will let the GAN keep generating beats, or pre generate them, then to order these snippets over time there needs to be another system in place. This system will be an Long-Shot Term Memory (LSTM)[11] algorithm. An LSTM can interpret data over time. When prompted with new data an LSTM can predict what data will come after the fact.

By combining these two systems an AI should be able to generate live music.

5 Methodology

5.1 training

The training of the algorithm is split into multiple phases, these phases follow each other chronologically but are not meant to be followed to reproduce the research. These steps were taken to come to the final model. The broadly explained steps of the training can be found in the appendix of this document. All the steps are found to be in three phases: The small dataset, the bigger dataset, the exponential dataset. During these training phases the model goes through slight changes in its hyperparameter and composition [14].

5.2 The Dataset

The data is an audio stream with floating-point numbers between -1 and 1. This audio data is converted to STFT format, so the data is represented in a matrix format. Rather, before transforming the audio into an STFT format it is cut into individual beats. The beats are transformed into 256 frequency bins and 256 timesteps exactly. For the extraction of the spectrum Librosa version 0.8.0 [12] is used. After applying the STFT to both channels (the stereo image) both are concatenated using NumPy version 1.19.2 [13].

5.3 The small dataset

This dataset is the dataset consists of a 18 minute live set playing EDM music by various artists. [3] All the data has been trimmed and cleaned of any vocals, set to a BPM of 130. The audio is recorded at 24bit and 44.1kHz.

[3]mau5trap radio episode 89: `https://www.mixcloud.com/deadmau5/deadmau5-presents-mau5trap-radio-089-speaker-honey-takeover/`

5.4 The big dataset

This dataset is the dataset consists of a two-hour long live set playing EDM music by various artists. [4] [5] All the data has been trimmed and cleaned of any vocals, set to a BPM of 130. The audio is recorded at 24bit and 44.1kHz.

5.5 The exponential dataset

This dataset is identical to the big dataset, rather this data is processed slightly different to the previous two. To match human listening more closely the frequency bins will be transformed into a logarithmic curve. The first difference is the number of frequency bins, the STFT algorithm will calculate 8192 frequency bins, which will later become 256 frequency bins after using Dynamic Time Warping (DTW) [15], an algorithm that shifts and combines data from one bin to the other. The curve of the frequency bins is calculated by using *formula* (1), with as result the output *figure 4* spectrum has a representation closer to the human hearing. The lower frequencies have a bigger representation than the higher frequencies. The higher frequencies are more clustered together than the lower frequencies.

$$y = \sqrt[\beta]{nbins - \alpha x + 1}^{x} + \alpha x - 1 \qquad (1)$$

where:

$nbins$ = Number of input bins
β = Number of target bins
α = Slope
x = Input bin
y = Output bin

[4] mau5trap radio episode 89: `https://www.mixcloud.com/deadmau5/deadmau5-presents-mau5trap-radio-089-speaker-honey-takeover/`

[5] mau5trap radio episode 115: `https://www.mixcloud.com/deadmau5/deadmau5-presents-mau5trap-radio-115/`

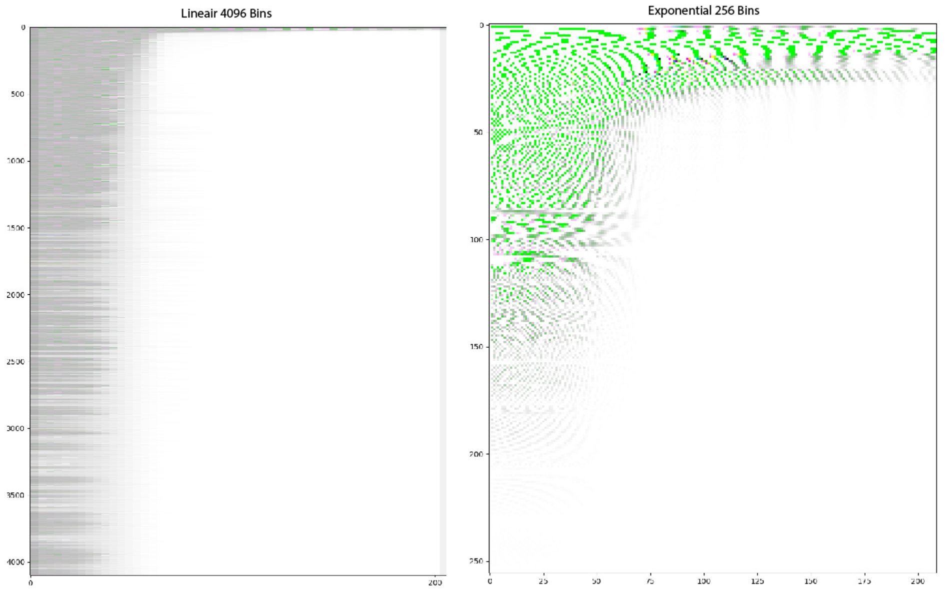


Figure 4: Left (input) the linear frequency bins, right (output) the exponential frequency bins

5.6 DCGAN

The DCGAN training loop is made of two different algorithms working together. For building the model TensorFlow [17] is used. In the training loop the generator is responsible for generating images. The discriminator will try to police the generator and try to estimate if the output of the generator is real or fake. During training, the discriminator will receive real samples to update its own weights so it can improve its guessing game. In *figure 5* the full training loop is walked through.

During this training loop we can zoom in both on the generator and discriminator *figure 16* The discriminator downsamples the image using convolution filters [20], the Conv2D layers. These filters are image filters that contain different kernels. These kernels can either sharpen, blur, edge detect etc there are too many filters to reside. The filters along the way downscale the image to a smaller version. This smaller version is then compressed to a single string of digests which then get compressed even more until the final output digit remains. This digit contains the most important value, this value is the predicted accuracy of the image either being 1 (real) or 0 (fake). These final couple steps make use of dense layers, which are a complicated web of chance calculations. [21]

The generator on the other hand uses the same principle as the discriminator but inverted. It upsamples the input string of numbers. The input string of numbers is a vector of random numbers, thus noise. This

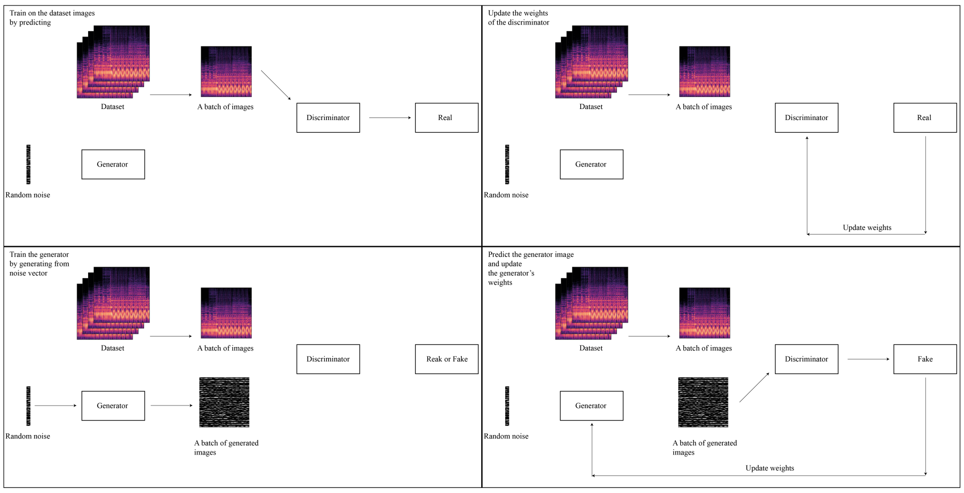


Figure 5: Complete training loop, from top left to top right to bottom right to bottom left

input vector is then squared by folding it into a square with repeating certain numbers. Then to upsample the image it uses the same kernels but use a reversed calculation of the kernels. This reverse calculation also increases the image size. The name of this process within TensorFlow is called Conv2DTranspose. In the final model the generator oversamples the image in the final two layers. The goal of the AI is to generate an image of 256*256*4, rather the upsampling goes up to 512*512*4. This upsampled image is then downscaled to 256*256*4. This technique uses the same principle as used during the production process of audio or the filming process of video: high quality recording which is later downsampled. When high resolution data gets downsampled it retains more of its important data and thus looks and feels sharper and clearer than data that is the same quality but that has never been "oversampled".

In *figure 16* there are a couple more settings which need to be addressed: Batch normalization [23], the process of normalizing the data in between steps in order to prevent the layers from overfitting. LeakyReLu activation [22], a process where output data of the layer is mapped between -0.2 and inf. Tanh activations [22], a process where the output data of the layer is mapped between -1 and 1. Dropout [24], a mathematical equation where certain values are forgotten on purpose to make it more challenging for the AI to train. Which is favourable since the AI can otherwise overgeneralize too easily.

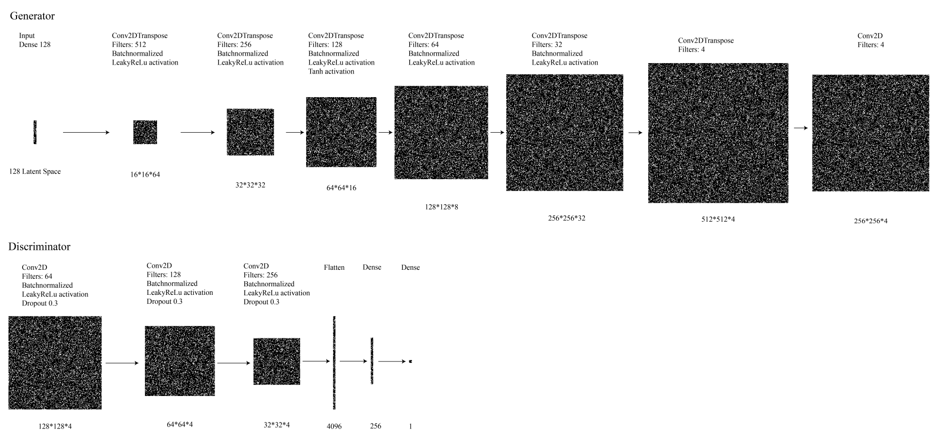


Figure 6: Diagram of the model for training the big and exponential dataset

5.7 The hyperparameter

The updating of weights happens according to different hyperparameters. These are mathematical functions that are used to calculate the prediction capabilities of the algorithm. The most important hyperparameters are the optimizer and the loss function. The optimizer [18] is a mathematical equation that guides and stores the pathway of learning. Optimizers and in parts the gradient descent are too complex to go over in a couple of sentences. On a considerably basic level the optimizer takes care of the learning rate, this is important for the algorithm does not overgeneralize too quickly. For this AI, the Adam optimizer [26, 27] is used, an optimizer that does not fall too quickly and is careful in search.

A second hyperparameter is the choice of loss function, which is responsible for calculation how good the output of the discriminator is compared to its real output. During training, the discriminator outputs one number, either it is 1 (real) or 0 (fake) or somewhere in between, by using a loss function we can calculate the difference and see its accuracy. So in case real image is given to the algorithm the loss function is given a one as the ground truth, the discriminator gives its output and gives the image a 0.6. The loss function will do its equation and find out its loss is for example 0.4 (the real output is way different since the equation is complicated). [19] The AI uses BinaryCrossEntropy as a loss function which in basic language means: It can only predict two types of images: True or False.

In between training phases these hyperparameters were tweaked with the help of a useful guide that was found online. [25]

5.8 LSTM

The LSTM algorithm is trained on tokenized audio data from the dataset. Each beat gets assigned a token, for this system implemented as an integer. Each token represents a point in a folder containing the musical data. The musical data is analysed on 7 different points of comparison using MIR (Music information retrieval) techniques:

- FFT data
- STFT data
- Fundamentals
- Spectral flatness
- Tuning
- Onsets

Each piece of audio gets compared to the previously analysed audio in the timeline. This comparison is done byte for byte in most cases in combination with averages of the matrices. The two pieces of audio can get two scores: 1 for similar or 0 for not similar. If the total score of all comparisons is larger than 3 there is a similarity between the audio files, and there must be a comparison in either the FFT or the STFT. When there are not enough comparisons to the previous audio fragments then it is unique and gets a new token, when there are enough comparisons this piece of audio is similar and gets the same token as this piece of audio. After this analysis, the LSTM is trained on this list of tokens over time. When prompted with a random number the LSTM can generate a number that it should follow it up with, based on what it knows from its input data.

5.9 The user test

To justify the research a survey will be held where users are requested to distinguish the difference between AI generated music and manmade music. The AI music is generated by the AI and the manmade music is created by Bas Maat. The human music will be downsampled and slightly deformed to create a level playing field since the AI is not perfect.

The survey has a couple sections where the participant's; background (place of residence and nationality); musical background and experience, will be taken into consideration. These factors are added to take into consideration whether musicians have an advantage over non musicians. The nationality is also an important factor since the algorithm is creating something that is typically western music. A person who is culturally not used to this music might have a different experience than a person who is used to this type of music.

6 Results

6.1 Training phases

6.2 The small dataset

In the first attempts the model *figure 14* used a dataset of roughly 2100 beats. This first model uses an Adam optimizer with a learning rate for the discriminator of 0.001 and for the generator of 0.001. After approximately 40 epochs the model collapses, it only produces the same result. This is mostly since that the dataset is too small, a GAN system needs a lot of data to keep creating enough data.[6]

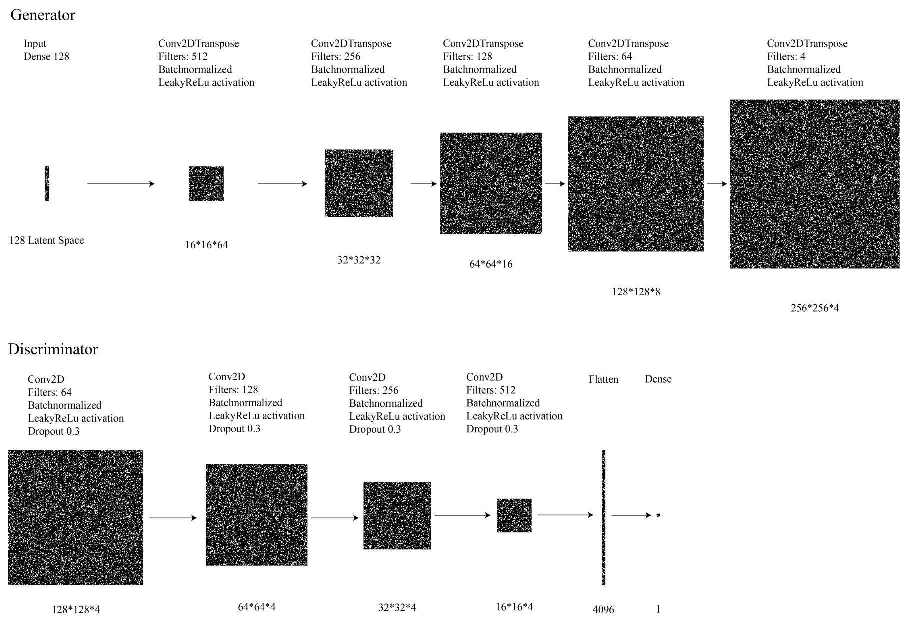


Figure 7: Diagram of the model for training with the small dataset

6.3 The big dataset

The second attempt the model *figure 16* uses a dataset of roughly 13000 beats, this number of samples was chosen after a couple of testing rounds. The limitation for data was hardware related, the training data is stored in a computer's memory during training. 13000 samples was the upper limit that could be reached on the used computer. This significantly improved training results. The model used the Adam optimizer with a learning rate

[6] audio 1, 2 and 3

of 0.0001 for the discriminator and 0.00012 for the generator. The model stabilized and did not suffer any model collapse, during training the models loss functions stabilized at 0.77 for the discriminator and 2.21 for the generator. This stabilizing of the loss shows that the model is at its optimal point of training and thus cannot improve more.[7]

6.4 The exponential dataset

The next step should be to create the spectral images in an exponential way. The previous models were trained on linear spectral images, where this model would be trained on exponential binned frequencies. After adding this possibility to the output, the model did not output its audio properly. This most likely is because the GAN is using the same architecture as the linear model. By tweaking the model accordingly, the results could drastically improve. The biggest problem with the output of the model is the still lower resolution in the output. A second way to improve the output results is to train on more frequency bins. If the dataset could become larger In the amount of represented frequencies then the model would be able to retrieve more information out the images. The currently trained exponential model were not able to deliver promising results.[8]

6.5 LSTM

6.6 First model, one level of comparison

The first model, which is trained on tokens of the data. The input data being a sample of Arctic Monkeys. [9] This sample was taken as an example since it has a very distinct flow between the different phases in music. The first way to measure similarities between samples after being chopped up is FFT. By comparing the pieces of audio based on their similar frequencies in that sample it is assigned its token. The similarity is based on a threshold of equality that it must overcome. This threshold has influence on the resulting training set. The optimal setting for this threshold is in this stage: 0.3. Here the difference between samples is measurable. This method was tested by hand each step of the way.

During training, the performance of the algorithm is strongly correlated to the number of time steps. This is Due to the amount of perception that the algorithm has, the higher the perception the better results overall. The following amount of time steps brought the highest accuracy: 40 time steps. The following time steps were tested:

[7]audio 4, 5 and 6

[8]audio 7, 8 and 9

[9]audio 10

timestep	accuracy
10	0.3 accuracy
15	0.36 accuracy
20	0.4 accuracy
25	0.5 accuracy
30	0.52 accuracy
35	0.6 accuracy
40	0.66 accuracy
45	0.5 accuracy

The optimal accuracy should lay around 0.6-0.7, this is due to the fact that we want to generate new patterns with the algorithm. A lower accuracy score will give a higher probability of more variation in the outcome, rather than perfect recreation. This randomness can be interpreted as creativity by a listener.

After training the resulting audio still has a way to come. It still has too little of a perception of flow of the music. The samples surely sound organized on a lower level, rather the song on a macro level is still too random.

6.7 Second model, two levels of comparison

The second model is trained in the same way as the first model. The same build-up, rather the tokenization has multiple levels of comparison. These levels are also tested according to a threshold, where the same threshold applies tested by trial and error on the human hearing: 0.3. similarly, to the first model this model uses more time steps, since there is less variation in the tokens the machine needs a bigger perception of time data. The number of time steps that is used is: 80. After training the audio result has a far better sense of depth and flow. The way it operates on a micro level it can generate audio in the same way as the first model, rather it is now able to generate its audio with more buildup overtime.

These two models compared against each other: [10]

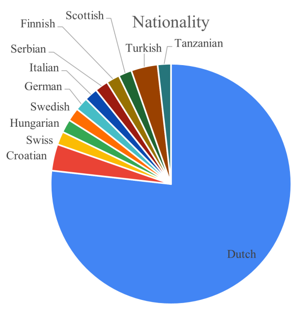
6.8 Survey results

The conducted survey gives a couple of interesting and noteworthy results. Instead of displaying all the data question per question, it becomes clearer and interesting when cross referencing the data. The total number of people who responded to the survey is 53.

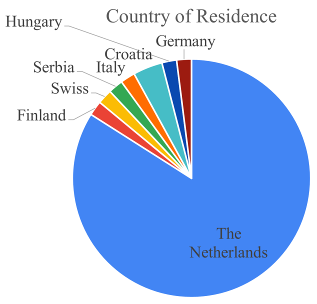
First off; some baseline data on who answered these questions. Most of population in this survey is from nationality Dutch see *figure 8a* and resides currently in The Netherlands see *figure 8b.*

[10]audio 11 and 12

Figure 8



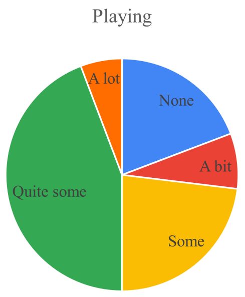
(a) Nationality of survey population



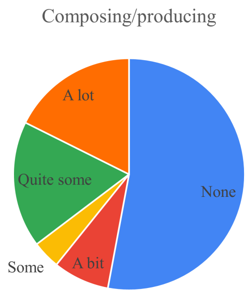
(b) Country of residency of survey population

Next, the group has been asked to grade their own skills in: listening, playing *figure 9a*, composing/producing *figure 9b*. These were separated into three questions.

Figure 9



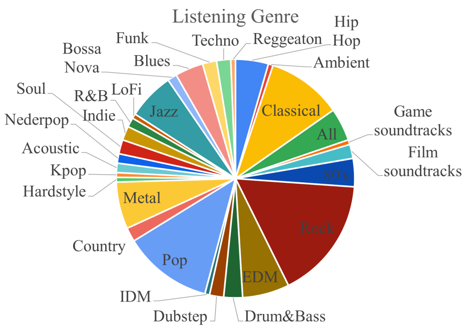
(a) Experience with playing music



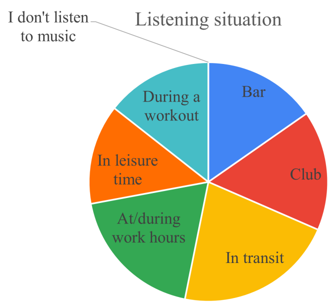
(b) Experience with composing/producing music

The questions about listening are split into two parts: preferred genre *figure 10a* and situation of listening *figure 10b*.

Figure 10



(a) Different genres respondents listen to



(b) Different situations people listen music in

Following, the participants were asked to listen to 20 different songs across 10 questions and rate each against each other *figure 11*. When combining these questions with the previous data the data becomes more insightful. In *figure 12* the scores are cross correlated with the experience in music; with *figure 12c* scoring the respondents per experience category, *figure 12d* scoring the respondents per experience category, and *figure 12a* scoring the participants per genre of music.

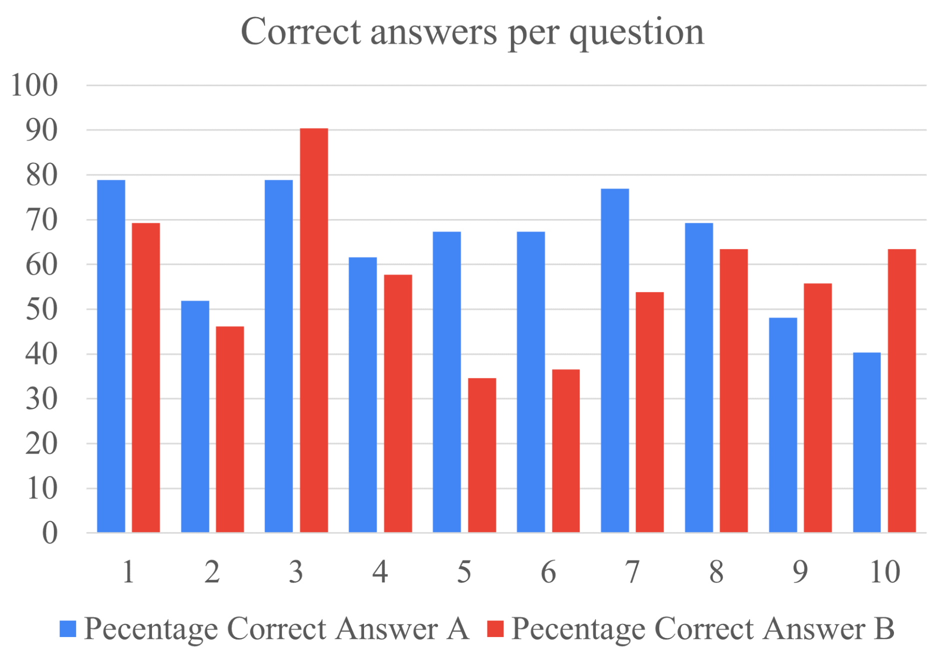
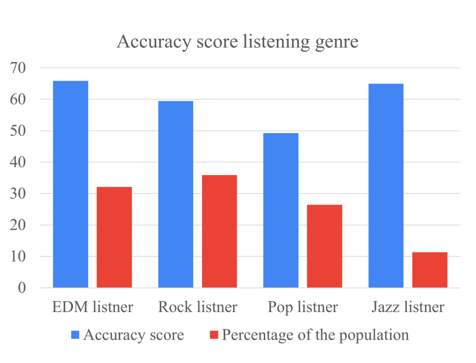
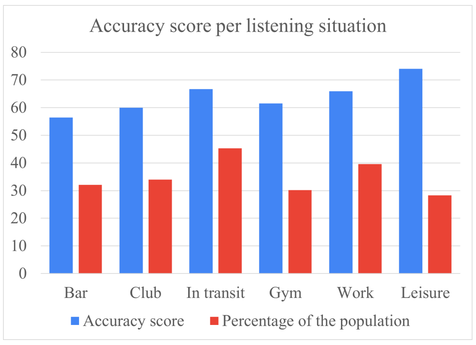


Figure 11: Scores of answers per question

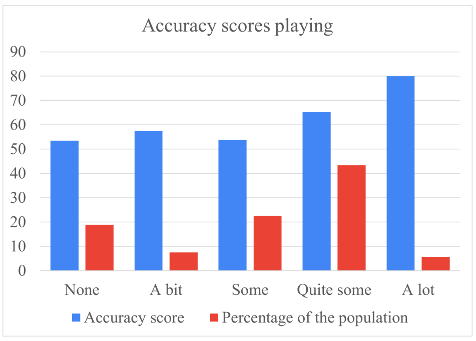
Figure 12



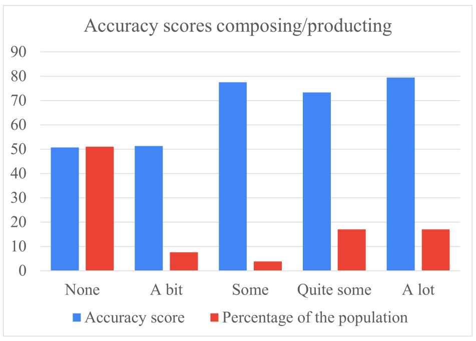
(a) Different genres respondents listen to



(b) Different situations people listen music in



(c) Different situations people listen music in



(d) Different situations people listen music in

7 Conclusion

7.1 About Training

There are a couple of lessons to be learned from the training process. First off, the process of training GAN systems is notoriously difficult. They are highly unstable and need proper care for it to become stable where both generator and discriminator train at an equal speed. By tweaking the hyperparameters, this goal can be reached with lots of care.

Secondly, when starting the research, the number of samples in the dataset was too few. GAN systems need lots of examples to start to understand generalized data. Where two hours of music, or 7000 beats, were enough to generate adequate sound which could be perceived as music. Besides the quantity of the dataset the quality should not be overlooked. High-quality audio where the frequency range of the spectral snapshots are high enough is a challenge on its own. With hardware limitations still being an issue, due to the massive amounts of RAM that these images take up during training, there is a balance to be struck between number of frequency bins and timesteps and the amount of RAM available during training. Also, the type of frequency bins matter, with research still to be conducted on whether exponential frequency bins are a viable solution the first results seem to go in the right direction. Improving the way of representing the data is still ways to come, but these first few steps seem promising enough to further research in the future.

Lastly, there are many ways to shape the model. The current model layout seems to be in its best form, where the models can be moulded according to the data. The most important step was to add the tanh activation in between the layers, this drastically improved the output of the model.

7.2 The LSTM

The LSTM is a more abstract version which came to mind later during the research. Due to the GAN algorithm needing to increase it became much slower with generating its musical data and the separate beast having no correlation to each other a LSTM was introduced into the mix to organize the output of the GAN. LSTM are not perse particularly good at analysing audio-rate samples, rather the tokenization of the data is a common solution. The latter was then as well the biggest hurdle to overcome, it required a method of comparison. With these methods the LSTM was able to converge in a good way. The output of the LSTM still leaves doors open for improvement, but this is a start. Next steps would include: a clockwork architecture, faster tokenization, the use of transfer-models.

The clockwork architecture should be able to analyse the smaller and bigger scope of the audio samples. With the bigger structure analysing whole four measure phrases, the second only single measures and the last layer only organizing the separate beats. Such an algorithm would be able to converge time placement of these tokens better than the current model.

A faster tokenization method should improve training speeds. A couple of ways could be to make the code work in a multiprocessing fashion or to pre-process more of the data. It could even be able to train an AI to tokenize the data, where how is still the biggest question.

Lastly, transformer models. Models like GPT-2 by openAI use a new and improved algorithm to LSTM. These so-called transformer models have a deeper understanding of time placement of time sensitive data. The reason for not using these models in the first place is the fact that these models are huge cans of worms which will take another year to at least understand to create which would deviate from the main subject of this research.

7.3 AI music in general

With all the training being done and a working algorithm in place that can generate audio with relative ease what is there to make of all this? Music of course! This algorithm is far from perfect, with perfection being human quality audio. AI music still has got big steps ahead of itself. First, the quality, even with the steps taken in this research by creating stereo phase related music, the quality is still far from ideal. The crunchy, robot feel might not be what a human creates it is for a matter-of-fact music created by an algorithm. This would call for the suggestion that this new wave of AI music to imitate humans, should not be the goal, rather an AI to create AI music should be the goal instead. Which is an idea that is an accident of the circumstances. The aim of this thesis to create music to replace the DJ could still be met with furthering this algorithm and eventually pushing humans forward into creating music, instead of replaying it in a club. Rather with the current state of the AI I would suggest that this music would not right away replace the DJ it would live alongside it and inspire humans of a new type of music: Algorithmic Dance Music. Where I see multiple moves into creating human like music with an AI, it by the end is the most human thing and thus creates new music and pushes culture forward. The new direction of music definitely is AI, which eventually will even push humans off the live stage. Still, we are far from this dystopia, I would rather let the AI do what it does best create ADM.

7.4 Public Perception

Finally, the most important part of any research: does it have any real-world application? In short, yes, the current state of the algorithm could have real-world applications. Rather, the real answer is more complicated. First off, the application of this conducted research has for 80% taken place in one European western country, The Netherlands *figure 8b*. Which in its turn consists for roughly 25% of culturally non-traditional Dutch people[11], this fact also roughly becomes clear from the survey results where 25% of people are of a different nationality *figure 8a*. The recipients who did come from a different nationality than Dutch were still solely from the western-world. [12] This information needs to be taken in consideration since these results will not prove whether the algorithm would work on a true international stage.

Another part that needs to be taken into consideration is the fact that most participants do play an instrument, but but do not necessarily compose/produce music themselves *figure 9*, which means that the knowledge in how to create music is readily available with the recipients, but they have not put this into practice yet by writing or producing music. This fact can also be traced back in *figure 12c*, here we can clearly see that the recipients who have more experience in playing music were better able to find the

[11]`https://www.cbs.nl/nl-nl/dossier/dossier-asiel-migratie-en-integratie/hoeveel-mensen-met-een-migratieachtergrond-wonen-in-nederland-`, graph bevolking, mei 2021

[12]`https://en.wikipedia.org/wiki/Western_world`

difference between the AI music and manmade music. This becomes even more evident when a participant also composes/produces music *figure 12d* Here, as soon as the participants have some experience with creating music they are better able to understand the difference between the two audio examples. The person their exposure to playing and composing music is of significant influence on their perception of the music, thus how human and realistic the recreation of the music sounds.

A third side to the story is the exposure a person has had to the type of music. If a person listens regularly to EDM *figure 12b* they are more likely to distinguish the difference between AI and manmade music EDM. This difference is most likely since they are more common with the music so they can nit-pick the parts in the music that sound closer to original EDM rather than the AI generated music. Another part that stands out in this part of the research are the people who listen to Jazz music. This group scored just as high as the EDM group. Between the two groups there is a commonality to be found: both listen to music which mostly contains no lyrics. While Rock and Pop both scored lower, their music is more focussed around the singer and their lyrics which could imply that the listeners are trained more on listening to the features of the singer rather than the the backing instruments. Jazz and EDM listeners are more common to listen to the instrumental only and thus developed a more defined hearing to hear abnormalities in this part of the music if needed. The algorithm was trained on non-lyrical music so this correlation would make sense.

Lastly, the situations where people listen often. What can be found is that in transit, during working hours and during leisure time scored higher than the other three situations. This mostly would be due to these situations are places and time where the listener can be more focussed on the music. In a gym, club or bar the music has an atmosphere filling purpose where music in the other situations have a main or second focus. Considering that the goal of the algorithm live music is this data shows that the current state of the music would already work as human music.

7.5 Final words

As shown in this research the current state of the algorithm is far from perfect. Music is something that depending on its listeners is widely perceived differently, which besides the current algorithm technical challenges is another challenge on its own. The public perception of this music is overall that it is like human music, but there still needs work to be done to iron out its wrinkles. Still there is ways to go to improve the quality of the audio itself and to improve the way the algorithm displays the music over time. After these two points improve an algorithm like this would be ready for primetime. With the current state of this type of music one could argue that music created by AI is a completely different type of music, Algorithmic

Dance Music. ADM could be something new in our western culture where an algorithm is inspired by human music and creates its own interpretation of it. ADM is something unique to manmade music, it is difficult to reproduce since the biases and the way the music is created is something we cannot do as easily by hand, something I had to experience first-hand. The way the algorithm places the frequencies overtime is very mechanical and there is beauty to be found in that. So, these algorithms do not necessarily need to reproduce manmade music there could form a movement where ADM is the goal of the algorithm not the perfect repetition of already existing music, we already got enough DJs in the world to do that job for us so lets make something unique with our algorithms.

References

[1] Carr, C.J. and Zukowski, Z., 2018. 'Generating Albums with SampleRNN to Imitate Metal, Rock, and Punk Bands', arXiv:1811.06633 [cs.SD]

[2] Mehri, S. et al., 2017. 'SampleRNN: An Unconditional End-to-End Neural Audio Generation Model', arXiv:1612.07837 [cs.SD]

[3] Koutník, J. et al., 2014. 'A Clockwork RNN', arXiv:1402.3511 [cs.NE]

[4] Oord, A. van den et al., 2016. 'WaveNet: A Generative Model for Raw Audio', arXiv:1609.03499 [cs.SD]

[5] Engel, J. et al., 2019, 'GANSynth: Adversarial Neural Audio Synthesis', arXiv:1902.08710 [cs.SD]

[6] Kumar, K. et al., 2019, 'MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis', arXiv:1910.06711 [eess.AS]

[7] Boashash, B., 1992, 'Estimating and Interpreting the Instantaneous Frequency of a Signal-Part 2: Algorithms and Applications', Proceedings of the IEEE 80(4):540 - 568

[8] Karras, T. et al., 2018, 'Progressive Growing of GANs for Improved Quality, Stability, and Variation', arXiv:1710.10196 [cs.NE]

[9] Logan, B., 2014, 'Mel Frequencies Cepstral Coefficients for Music Modeling', ismir2000

[10] Goodfellow, I.J. et al., 2014, 'General Adversarial Networks', arXiv:1406.2661 [stat.ML]

[11] Hochreiter, S. et al., 1997. 'Long Short-Term Memory', Neural Computation 9(8):17351780, 1997

[12] McFee, B. et al., 2015. librosa: Audio and music signal analysis in python. In Proceedings of the 14th python in science conference.

[13] Harris, C.R. et al., 2020. Array programming with NumPy. Nature, 585, pp.357–362.

[14] Yo, T. and Zhu, H., 2020. 'Hyper-Parameter Optimization: A Review of Algorithms and Applications', arXiv:2003.05689 [cs.LG]

[15] Senin, P., 2008. 'Dynamic Time Warping Algorithm Review', University of Hawaii at Manoa

[16] Radford, A. et al., 2016. 'Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks', arXiv:1511.06434 [cs.LG]

[17] Abadi, Martin;in et al., 2016. Tensorflow: A system for large-scale machine learning. In 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16). pp. 265–283.

[18] Doshi, S., 2019, 'Various Optimization Algorithms For Training Neural Network', `https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6`, 13 Jan 2019

[19] Godoy, D., 2018. 'Understanding binary cross-entropy / log loss: a visual explanation', `https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a`, 21 Nov 2019

[20] ormesher, I., 2020. 'Convolution Filters', `https://medium.com/@ianormy/convolution-filters-4971820e851f`, 9 may 2020

[21] Heidenreich, H., 2019. 'Understanding Keras — Dense Layers', `https://medium.com/@hunterheidenreich/understanding-keras-dense-layers-2abadff9b990`, 14 jan 2019

[22] Sharma V, A., 2017. 'Understanding Activation Functions in Neural Networks', `https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0`, 30 Mar 2017

[23] D. N, K., 2018. Understanding Batch Normalization, `https://medium.com/@krishna_84429/understanding-batch-normalization-1eaca8f2f63e`, 10 Aug 2018

[24] Budhiraja, A., 2016. 'Dropout in (Deep) Machine learning', `https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-ma` 15 Dec 2016

[25] Brownlee, J., 2019. 'Tips for Training Stable Generative Adversarial Networks', `https://machinelearningmastery.com/how-to-train-stable-generative-adversarial-networks/`, 12 Sep 2019

[26] Kingma, D. P. and Ba, J. L., 2017. 'Adam : A method for stochastic optimization.', Published as a conference paper at ICLR 2015, arXiv:1412.6980v9

[27] Brownlee, J., 2014. 'Gentle Introduction to the Adam Optimization Algorithm for Deep Learning', https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/, 13 Jan 2021

Appendices

A Training phases

A.1 Training 1

The first attempt at training the GAN it was trained on a dataset of approximately 1000 samples of beats. The Model was built up as follows:

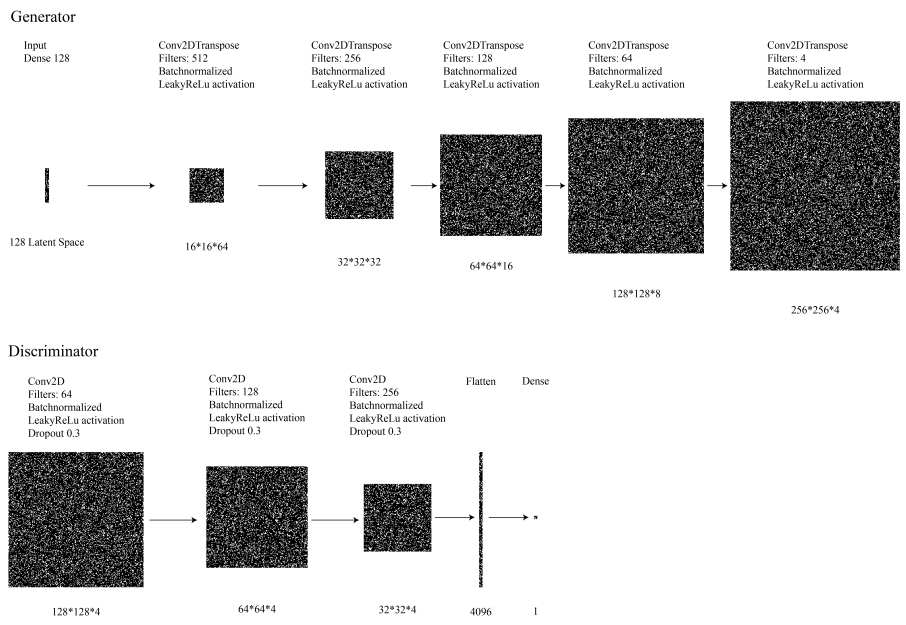


Figure 13: Diagram of the model for training 1

With an Adam optimizer of 0.001 for the Generator and 0.001 for the Discriminator. During training, the model kept collapsing around 40 epochs.

A.2 Training 2

The second attempt at training the GAN it was trained on a dataset of approximately 1000 samples of beats. The Model was built up as follows: With an Adam optimizer of 0.001 for the Generator and 0.001 for the Discriminator. During training, the model kept collapsing around 50-60 epochs, so there was some improvement to be found here.

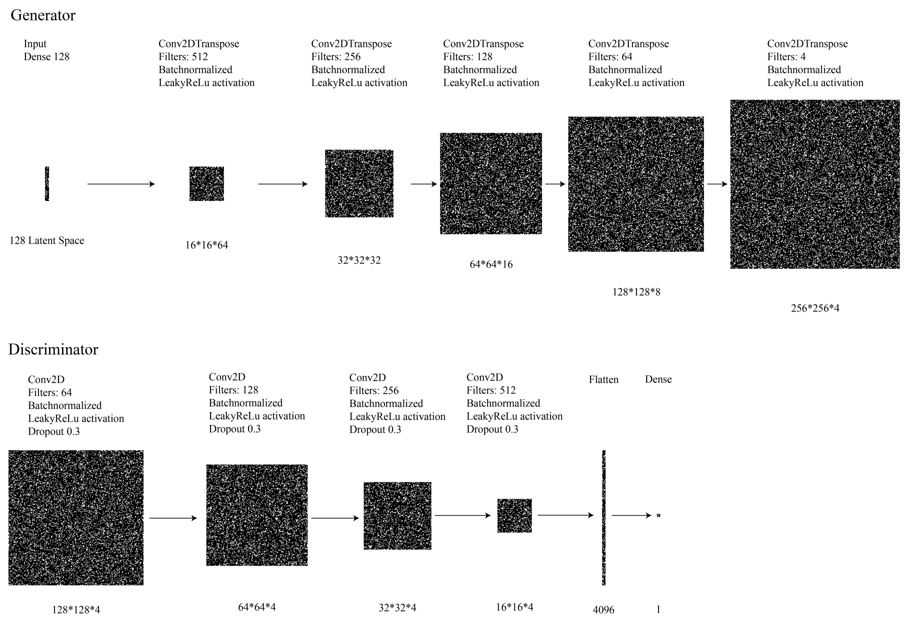


Figure 14: Diagram of the model for training 2

A.3 Training 3

For the next training model the dataset was approximately 7000 samples of beats. The build-up of the model stayed the same as the previous training. With an Adam optimizer of 0.001 for the Generator and 0.001 for the Discriminator. During training, the model kept collapsing around 40-50 epochs. Here the gap between both algorithms has shrunk significantly. The generator and discriminator now stabilize at a loss of 2.21 (G) and 0.77 (D). What was rather peculiar is the fact that the output in audio was still very flawed. It is too low in volume.

A.4 Training 4

For the next training model the dataset was approximately 7000 samples of beats. In the model the generator got extra activation layers in between with the tanh algorithm, this way the values which would become too big would stabilizes and normalize better so the audio would be clearer. The resulting audio was way clearer, and the training results stabilizes earlier than expected. With a slower learning rate this can be quite easily circumvented. By setting the learning rate to 0.0001 and 0.00012 the AI stabilizes not as quickly but seems to move towards a stable direction over time.

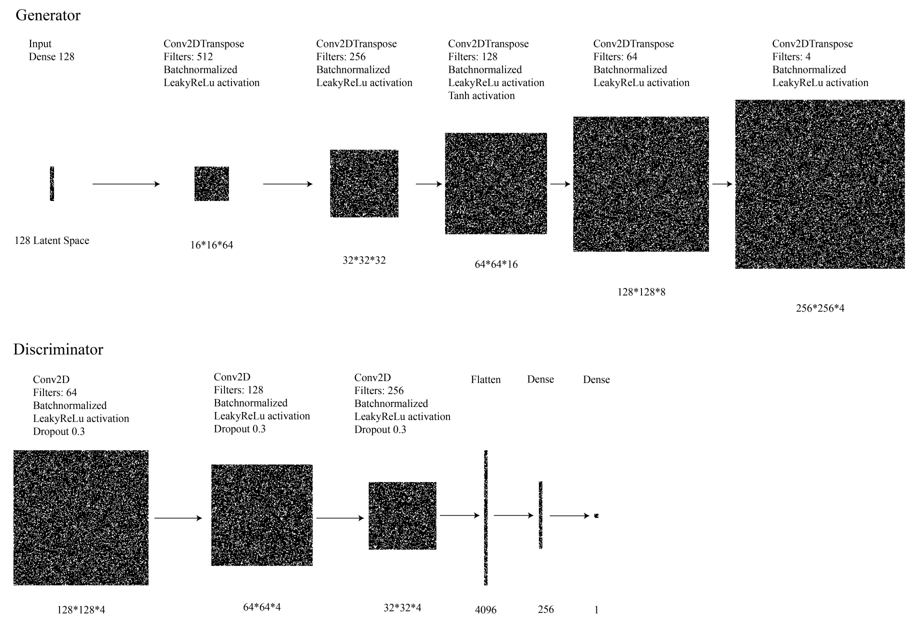


Figure 15: Diagram of the model for training 4

A.5 Training 5

A last way that training results were improved is by adding an extra scaling to the very last layer of the generator. The generator will in that case oversample the incoming matrix. Following the oversampling there will be a downsampler to go back to the right output dimensions. The theory behind this output is the same as a creator would render their audio on a higher samplerate to then downscale it. This is not anything like decryption or encryption, rather this is remarkably similar to creating on a high resolution. The results do confirm this matter, the algorithm sounds even clearer in the output.

A.6 Training 6

The next step should be to create the spectral images in an exponential way. The previous models were trained on linear spectral images, where this model would be trained on exponential binned frequencies. After adding this possibility to the output, the model did not output its audio properly. This most likely is because the GAN is using the same architecture as the linear model. By tweaking the model accordingly, the results could drastically improve. The biggest problem with the output of the model is the still lower resolution in the output. A second way to improve the output results

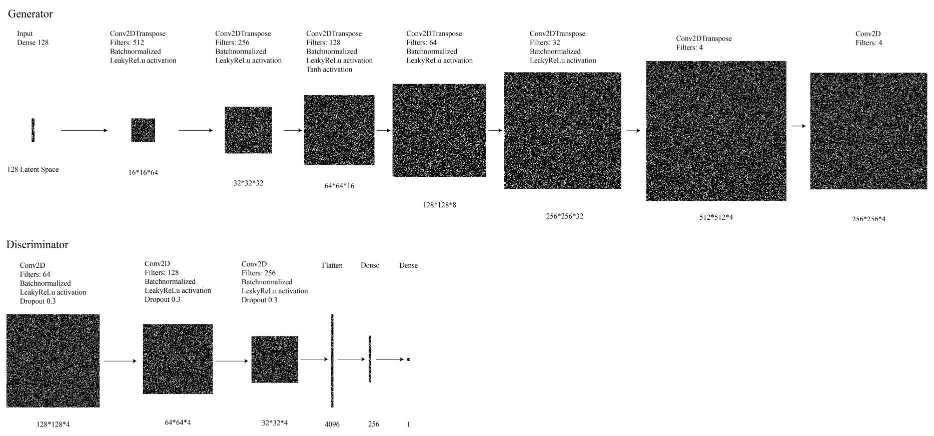


Figure 16: Diagram of the model for training 5

is to train on more frequency bins. If the dataset could become larger In the amount of represented frequencies then the model would be able to retrieve more information out the images.