Artificial Dance Music

Creating Electronic Dance Music with Artificial Intelligence

Lars Løberg Monstad

Master's Thesis
Department of Musicology
Faculty of Humanities
University of Oslo

Spring 2021

Artificial Dance Music

Artificial Dance Music

http://www.duo.uio.no/

Abstract

The development of artificial intelligence (AI) methods like machine learning (ML) and deep learning (DL) has created new opportunities and tools for many industries such as self-driving cars, voice recognition and image classification. But can this new and improved AI be used to create popular electronic music? And how?

This thesis explores the possibilities of using AI to make Electronic Dance Music (EDM). Artificial Dance Music (ADM) is music created by AI and in particular, by DL, which trains on EDM music to learn from it and then recreate a symbolic representation. In this study, a Turing test is used to evaluate the generated music, with participants listening to both algorithmically-generated music and the original samples. A statistical evaluation of the Turing test results proves that there is no significant difference that distinguishes machine and human-generated EDM music. The goal is to investigate to what extent a machine can create original music and if it can be used as a music-producing tool in the future.

The thesis focuses on the possibilities of using computer creativity, what this means and how to evaluate it. Similar approaches have been explored before but mostly in other genres such as jazz, classical, and contemporary music. A reason for this is that a genre like classical baroque music contains large collections of pieces written by the same composer, which makes it ideal for AI research. The hypothesis of this thesis is that an AI model can successfully create music from a small collection of songs if specific aspects are considered in the system implementation. The key contribution of this thesis is the method of using AI to create popular music, which offers a unique approach in the field of Algorithmic Music research.

Acknowledgments

I would like to thank my supervisors Kristian Nymoen and Stefano Fasciani for their crucial feedback, support, and patience throughout this process. Your insightful feedback pushed me to sharpen my thinking and brought my work to a higher level.

I want to thank all the participants in the Turing test, without you this research could never have been done.

Special thanks to Ole Kristian Bekkevold for advice, support, and beer.

Also, it is important to acknowledge my parents, Ingunn and Nils (Dj_Nilz) for all the support and feedback during the writing of the thesis.

Oslo, May 2021
Lars Løberg Monstad

Contents

1 Introduction

Algorithmic Music composition is a genre that mostly consists of music made with the aid of machines or some kind of code-like algorithms (Dean & Mclean, 2018, p)[1]. There are various methods used to create music this way. The music created by artificial intelligence (AI) is only a small part of this, but the technical development has given Algorithmic Music composition new opportunities. There are different kinds of AI and different methods, but the most common method, and the one used in this thesis, is the AI capability of learning. So, instead of programming software to create music, it is possible to make it learn from music to create it. In other words, this approach can make a machine go beyond what it has been programmed to do to solve a task. From a scientific point of view, music is subjective, and it is difficult to program software to create music when it is not known exactly how humans create music. How is it possible to translate creativity into an algorithm? Attempts to create music with AI have been around since the birth of AI (Boden, 2018, p. 7)[2], but recent developments in AI, like neural networks, deep learning (DL), and recurrent neural networks have upgraded and changed the ways it is possible to make music. This thesis goes into depth to show exactly how this new kind of AI can be used to create Electronic Dance Music (EDM). This research uses DL models to generate a symbolic representation of music, which is EDM music converted into MIDI (Musical Instrument Digital Interface) and interpreted by a computer. To do this, a system was created to convert the music into MIDI, which will be presented in Chapter 4.

The motivation to use EDM in particular is that this genre consists of elements that are well suited for AI research but is not often studied in the AI Music field. The aforementioned elements are electronic instrumentation, repetitive rhythms, instrumental songs, and basic repetitive harmonies, which will be elaborated on in Chapter 2.

1.1 Motivation

Four years ago, a YouTuber called Carykh published a video titled *"Computer evolves to generate baroque music!"*[3]. In this video, Carykh demonstrated how he was using AI to create

[1] Dean, R. T., & Mclean, A. (2018). *The Oxford Handbook of Algorithmic Music*. Oxford University Press. https://doi.org/10.1093/oxfordhb/9780190226992.001.0001
[2] Boden, M. A. (2018). *Artificial Intelligence A very short introduction*. Oxford University Press.
[3] carykh. (2017). *Computer evolves to generate baroque music!* https://www.youtube.com/watch?v=SacogDL_4JU

baroque music by using a recurrent neural network trained on MIDI files from Bach. Information from this video raised several questions and motivated this researcher to investigate the extent to which a machine can create original music. Is it possible for AI to be creative, and what are intelligence and creativity? After watching the video, research was gathered about other projects involving AI and music, and it was discovered that there are few projects involving commercial music like EDM. Also, there are no commercial pop songs that were created by AI. If AI can learn and create original classical songs, then why has it not been used to create hit songs that we can listen to on Spotify's top charts? Surely a big commercial music label like Sony or Warner would have seen this as an investment opportunity and maybe used AI instead of songwriters or producers. A human can only create a certain number of songs in a day, but if AI is making music, then the number of songs produced could be significantly increased. Further, this raises the question: Is the development of AI going to be a threat to music producers and composers, or can it be a tool for musicians to work faster and better?

This thesis explores possible answers to these questions by investigating the process of using AI to create EDM and evaluating it. The idea of creating an EDM sub-genre was the aim of using AI to create commercial music. The motivation for this thesis was to gain a deep understanding of exactly how AI can be used to create original EDM music and how to measure computer creativity. One problem with Deep Learning is that it usually needs a significant amount of data to successfully train and learn from (Chollet, 2018, p. 133)[4], which in this context means a large collection of songs. This can make it problematic to use for popular music genres such as EDM and created the need to devise a system to solve this problem. Also, it is important to obtain an understanding of the current state of AI in music and how AI can impact the music industry. In music technology, the term is used frequently as a brandmark for audio plugins and instrument applications (Shaw, 2019)[5], but never in the music product itself, at least in commercial music. This thesis also explores briefly why the music industry is not using AI as a trademark for selling products to the listener.

Music from the Canadian EDM producer Deadmau5 is used in the main research for this thesis, while music from the EDM artists Avicii and David Guetta are also featured in additional tests.

[4] Chollet, F. o. (2018). *Deep learning with python*. Manning Publications Co.
[5] Shaw, D. (2019). *8 AI tools that are at the cutting edge of music production*. happymag.tv. Retrieved September 14, 2020 from https://happymag.tv/8-tools-that-use-artificial-intelligence-to-produce-music/

As context to the research, some background is given about how Deadmau5 makes music, key elements of his music, and why his music is a good fit for AI.

1.2 Goals

The goal of this thesis is to demonstrate the possibilities of using AI in modern music composition. The hypothesis is that if an appropriate machine learning (ML) system is trained with the appropriate musical data, the results can be music of the same quality as music written by a human. In this context, quality is defined as music that is created by AI and is difficult to distinguish from the music it was trained on, while still being original. The goal is also to explore the possibilities of using AI as a tool that can benefit the human composer. If AI can develop modern harmonic and rhythmic ideas that are both original and of good quality, then a musician can use these ideas as a starting point to further create new songs. By using this approach, the creative process of writing a song can be both faster and generate more ideas. A human mind is limited by how fast it can create melodies or rhythmic patterns; AI, however, does not have the same limitations. Because an AI model is a piece of software that runs as fast as the GPU processing allows, it can run 24/7, unlike a human, who needs concentration and sleep. Yet AI still has limitations, it is as limited as the data being used. ML, for example, will never create anything very different than the data it was trained on. However, the idea is that, if used correctly, AI could be a new way of making music, with a musician and AI in a creative collaboration.

1.3 Research Question

The key research question in this thesis is:

Can artificial intelligence be used to create Electronic Dance Music?

To answer this question, an iterative process was used as a research method to create EDM with a DL model. DL is a method of AI that is a sub field of ML. In ML, the different methods are referred to as models or algorithms, and the definition of intelligent in this context is the machine's ability to learn from data to solve a task. This means that if music is used as data, then the DL model will train on it to learn from it and then use what it has learned to solve the task, which is to create music. The big challenge, however, is to convert the music into a system of symbolic representation that an ML algorithm can use. This also raises a sub-question: How important is *how* the music is presented to the AI?

1.4 Overview of Structure

This thesis consists of six chapters: Introduction, background, method, implementation, evaluation, and conclusion. The introduction, background and method chapters are the chapters that provides the necessary background information for the thesis. The implementation and evaluation present how the research occurred. The ADM system is presented in the implementation chapter and evaluated in the evaluation chapter. The last chapter, the conclusion, provides a summary of the research, discussion, reflections, and suggestions for future work.

2 Background

This chapter presents the background for the research. It commences with a short summary of EDM and why it was selected as the music genre for this project. The next section is an introduction to AI and the different categories of AI that are relevant for this thesis. The last section is about algorithmic composition and evaluation of computer creativity, which is a brief presentation about AI Music in both a historical context and in the current state, as well as how to evaluate AI music. The goal of the chapter is to provide the necessary information to understand the research of the thesis. Since this thesis is written for a master's degree of musicology, an understanding of Western music theory is assumed, while AI will be explained in detail.

2.1 Electronic Dance Music (EDM)

Electronic Dance Music is a collection of different kinds of rhythmical electronic music genres, also known as dance music, club music, or simply dance (Koskoff, 2004, p. 44)[6]. The music is usually performed by a DJ using turntables, a mixer, or a laptop, primarily at clubs, raves, or large festivals. EDM gained mainstream popularity in Europe around the 1980s to 1990s but was for a long time rejected by the American music industry until it had a breakthrough in 2010 with the *"Electric Daisy Carnival"* music festival (Reynolds, 2012)[7]. In 2011, it was described as the fastest-growing music genre in the world by *Music Trades Magazine* (Music Trades Magazine, 2014)[8] and in 2018, the EDM industry was worth 7.3 billion dollars (Smirke, 2018)[9].

Some of the most popular genres of EDM are house, techno, trance, drum and bass, trap music, future bass, and dubstep, but some of these subgenres existed long before the umbrella term EDM was coined (Music Trades Magazine, 2014)[10]. Armadamusic.com claims that the definition of EDM is *"all music produced electronically for the sole purpose of having people dance to it"* (Armadamusic.com, 2021)[11]. This thesis will focus on house music because it is the EDM style that is being used for the research. The house genre started out in the early 1980s underground club scenes in Chicago. When disco began to fade in popularity, DJ Larry Levan started to experiment with mixing regular disco, Italian disco, with his own bass lines and drum tracks (Butler, 2006, pp. 12-13)[12].The style followed a 4/4 drum bass pattern, with claps and snare at the 2nd and 4th backbeat using drum machines such as TR-808 (Bashill, 2002)[13]. This style was eventually known as *house* and can be described as more electronic and minimalistic than the disco genre. As DJ Larry Levan describes:

> *"By 1981 they declared that Disco was dead and there were no more up-tempo dance records. That's when I realized I had to start changing things to keep feeding my dance floor…"* (Snoman, 2009, p. 40)[14]

---

[6] Koskoff, E. (2004). *Music Cultures in the United States: an Introduction.* (Vol. 1 edition). Routledge.
[7] Reynolds, S. (2012). How rave music conquered America. *The Guardian*.
[8] Magazine, M. T. (2014). JUST HOW BIG IS EDM? *Music Trades Magazine*.
[9] Smirke, R. (2018). *IMS Biz Report: Global EDM Market Falls 2 Percent to $7.3 Billion*. Billboard.com. Retrieved Nov 10, 2020 from https://www.billboard.com/articles/business/8457536/ims-biz-report-2018-global-edm-market-electronic-dance
[10] Magazine, M. T. (2014). JUST HOW BIG IS EDM? *Music Trades Magazine*.
[11] Armadamusic. (2021). *THE COMPLETE GUIDE TO EDM (OR ELECTRONIC DANCE MUSIC)*. Retrieved April 1, 2021 from https://www.armadamusic.com/news/edm-electronic-dance-music
[12] Butler, M. (2006). *Unlocking the Groove: Rhythm, Meter, and Musical Design in Electronic Dance Music*. Indiana University Press
[13] Bashill, P. (2002). Six Machines That Changed the Music World. *Wired Magazine*, (10).
[14] Snoman, R. (2009). *DACNE MUSIC MANUAL*. Focal Press.

The genre eventually developed into acid house, which is a harder sounding and more synthesizer glitching subgenre of house music (Huxtable, 2014)[15]. This genre was heavily influenced by the inaccurate use of the TR-303 Bass Synthesizer, which made a *squelching* bass sound (Vitos, 2014)[16]. Another subgenre of house music is progressive house. Progressive house appeared after the first wave of house music in the 1990s and was typically more melodic and more upbeat than acid and Chicago-style house. The songs were longer, had heavy basslines, and built toward a crescendo climax (Huxtable, 2014)[17].

The reason house music was chosen as the data source for AI in this thesis is its rhythm and instrumentation, and its somewhat originality in AI research. To train a DL model on EDM music, the music needs to be both predictable and reliable without any big changes in the arrangement. Most house music has an arrangement of only bass, drums, synth lead, synth arpeggio, and a synth pad. Further, the drum section can be simplified to only a kick, snare, and hi-hat without losing much of the musical rhythm in the songs. A good example of this is the song *"Moar Ghosts 'N' Stuff"* (Deadmau5, 2009)[18]. The song uses the same minimalistic rhythm in 4/4 for the whole song with only the snare, kick, and hi-hat, which makes it very consistent and a good choice for DL. The song is also instrumental, which is common in both house music and Deadmau5 tracks. This is a requirement for the project since all of the layers of music are transcribed into MIDI and will then be replicated with a DAW using MIDI instruments. Vocals are an instrument that is challenging to recreate using MIDI and software instruments because it has additional layers like text and timbre, which are difficult to transcribe into MIDI, as will be elaborated on in section 2.4.1.

[15] Huxtable, S. (2014a). What is Progressive House? *Decoded Magazine*. https://www.decodedmagazine.com/what-is-progressive-house-2/
[16] Vitos, B. (2014). Along the Lines of the Roland TB-303: Three Perversions of Acid Techno. Dancecult: . *Dancecult: Journal of Electronic Dance Music Culture*
[17] Huxtable, S. (2014b). What is Progressive House? *Decoded Magazine*. https://www.decodedmagazine.com/what-is-progressive-house-2/
[18] Deadmau5. (2009b). Moar Ghosts 'n' Stuff On *For Lack Of A Better Name*.

2.1.1 The Sound of Deadmau5

Joel Thomas Zimmerman is a Canadian EDM artist/performer known as Deadmau5, born in 1981 in Nigeria Falls (Stanley, 2015)[19]. His music is in the genre of progressive and electric house music, and his own description of the music is a *“spacey, melodic brand of fist-pumping house”* (Stanley, 2015). Deadmau5 has won multiple Grammy awards for his music and has become one of the biggest artists in EDM (Stanley, 2015). His signature logo is a mouse head with dead eyes, which is a reference to his name *“dead mouse”*. Deadmau5 is known to use a lot of analog synthesizers in his tracks, which are a heavy influence on the sound. Since this thesis is only generating music thru MIDI as a symbolic representation, the thesis will focus more on the aspects of Deadmau5’s sound that are possible to recreate with a MIDI arrangement, and why his music is a good fit for AI. Also, all the songs used in the analysis examples are featured in the dataset.

Polymeter

An iconic sound feature of Deadmau5 that is important to address is called a *polymeter*. A polymeter is a rhythmical concept that means that two or more melodies/*meters* play at the same time but with different groupings of note values (Stitzel, 2021)[20]. For example, a piano in 9/8 over a drum pattern in 4/4. They start at the same time but will gradually desynchronize until eventually they will be on the beat again. This is often confused with a *polyrhythm*, which is two different rhythms playing at the same time but that are resolved in the same amount of time. A polymeter as a rhythmical pattern is a great fit for DL music creation because it is easy to analyze if the model is using the pattern.

[19] Stanley, L. (2015). *Deadmau5*. the canadian encyclopedia. Retrieved november 13, 2020 from https://www.thecanadianencyclopedia.ca/en/article/deadmau5
[20] Stitzel, R. (2021). What is a "Polymeter"? *DrumMantra*. https://www.drummantra.com/blog/what-is-a-polymeter

In the song *"Brazil"* (Deadmau5, 2014)[21], a polymeter is used in the chord pattern. As illustrated by Figure 1, the kick drum plays on every quarter note in 4/4, but the chord pattern is playing on dotted eight notes for two bars straight before it resets.

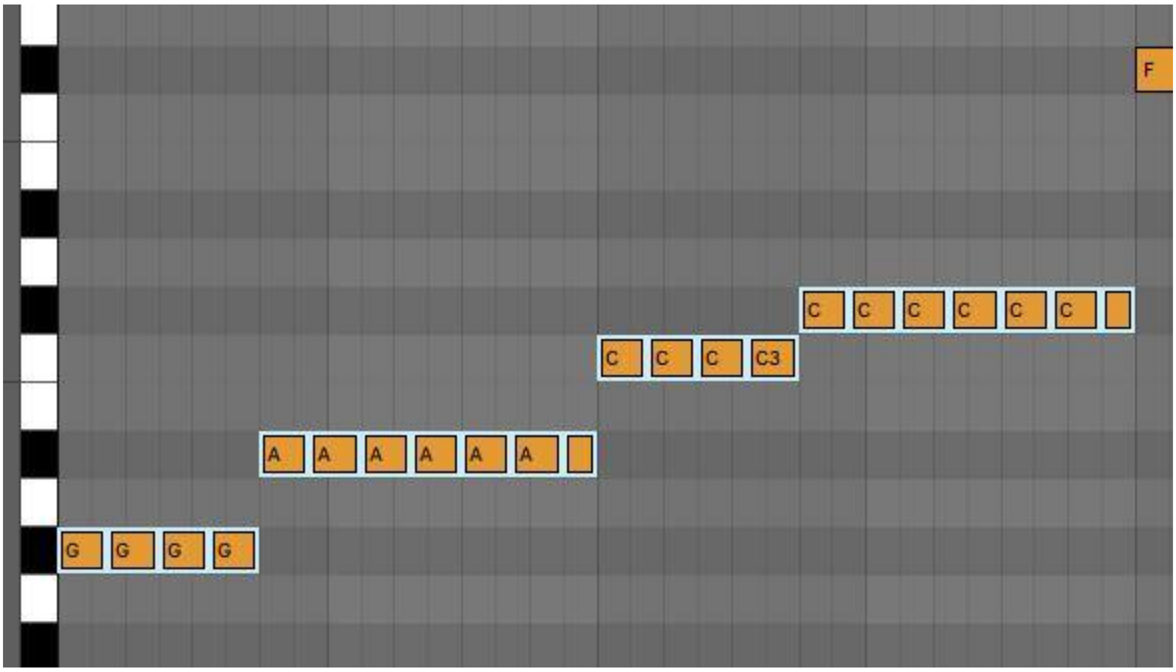


Figure 1. Illustrating the polymeter in MIDI with Ableton.

Figure 2 displays another example of this with the song *"Arguru"* (Deadmau5, 2009)[22]:

[21] Deadmau5. (2014). Brazil. On *5 years of Mau5*.
[22] Deadmau5. (2009a). Arguru. On *Random Album Title*.

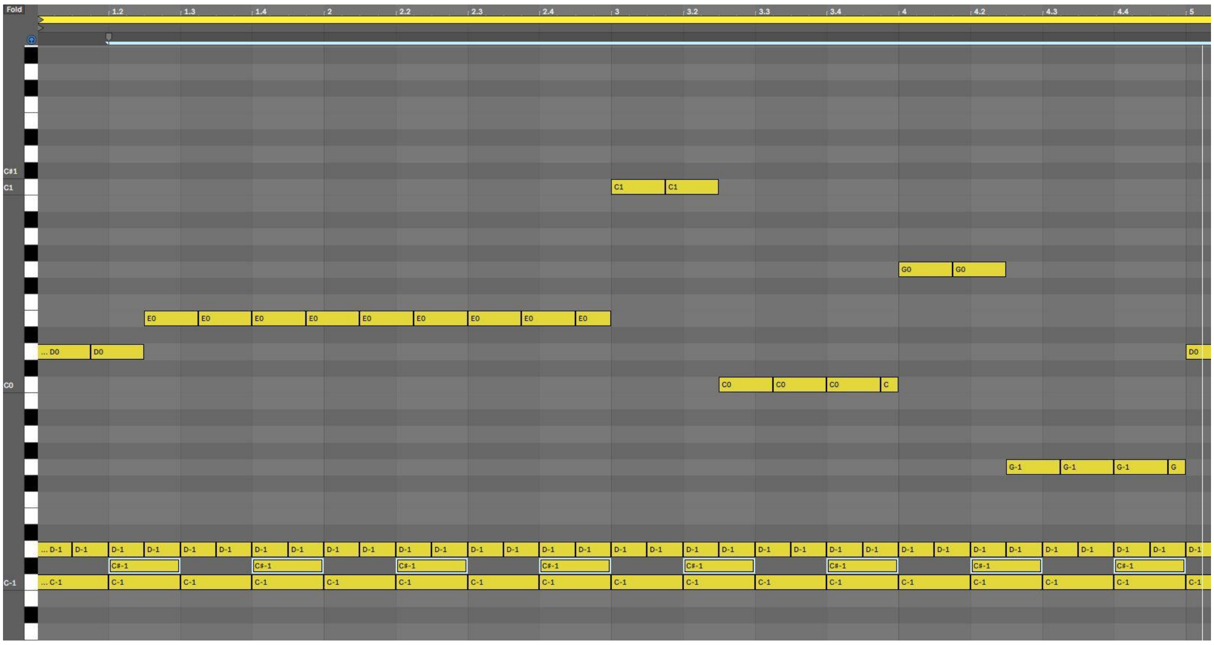


Figure 2. Illustrating a polymeter in MIDI with Ableton.

In this song, the bass synth plays in a dotted eight-note pattern for two bars and resets, then it plays the same pattern two more times but resets after one bar. The lower voices in this MIDI arrangement are the drums, which play a kick on every quarter note, a snare on every second quarter note, and a hi-hat on every eight notes. This is also the signature Deadmau5 drum rhythm that he uses in most of his songs. These two examples demonstrate how Deadmau5 utilizes the polymeter pattern and provides valuable information that was used during the analysis of the music created by AI in the Qualitative Observation test in Section 4.5.

The Arps

Another iconic sound feature of Deadmau5 is the arpeggio synthesizers. The word *"Arpeggio"* is a term in music theory that means broken chords (Djupdal, 2015, p. 150)[23]. The method means that instead of playing a chord harmonic, which means playing all the notes at the same time, the chord is played with one note after the other in a specific order. For example, if you play a C major chord on a piano, you can play C, E, and G after each other to play a C major Arpeggio chord. In synthesizers, this a very popular method, and analog iconic synthesizers

[23] Djupdal, K. (1995). Musikkteori. In (pp. 150-151). Ad Notam Gyldendal.

like the Juno 106 have an ARP function built-in (Rik Marston Official, 2015)[24]. This method is used in almost all of Deadmau5's songs and can be seen in Figures 3 and 4.

The Harmony

Deadmau5 has stated that he only knows basic music theory and that he makes all his songs by experimenting with the looping function in his DAW (Digital Audio Workstation) and Ableton (Masterclass.com Developing melodic structures, 2021)[25]. In this interactive process, he creates a loop and starts experimenting with the melody with one note as a starting point. In his masterclass, Deadmau5 demonstrates this by creating an arpeggio melody over a F# major chord with 3 notes repeated over 2 bars, as shown in Figure 3.

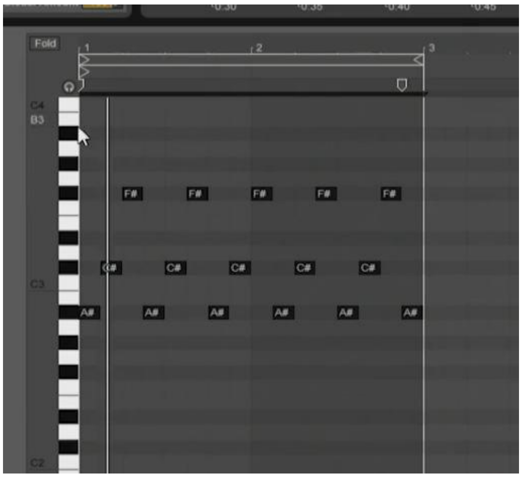


Figure 3. Illustrating MIDI harmony with Ableton.

He then copies all of his notes to the next two bars, where he changes the notes to whatever he thinks sounds good and then repeats this step twice more until he has an 8-bar progression with 4 different melodic chords, illustrated by Figure 4.

[24] Official, R. M. (2015). *Analog Monsters Roland Juno-106 does ARP SOLINA String Synthesizer!* https://www.youtube.com/watch?v=OvQHMaDoABs

[25] Deadmau5. *Developing Melodic Structures*, Masterclass.com. https://www.masterclass.com/classes/deadmau5-teaches-electronic-music-production

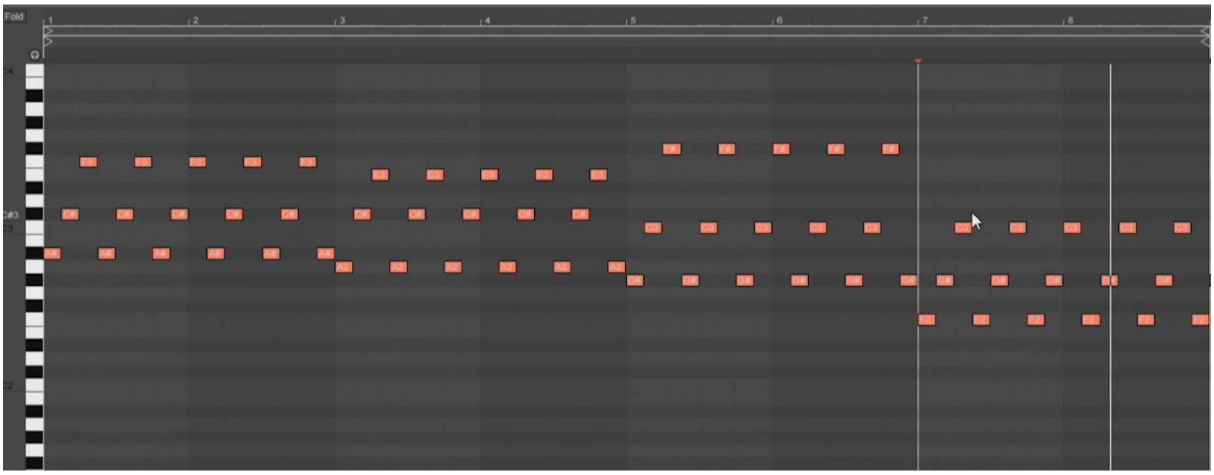


Figure 4. Illustrating harmonic pattern with MIDI notes in Ableton.

The next step is that the last melodic pattern should fit together and naturally lead to the first one. This is a method that Deadmau5 calls *"resolving"* (Deadmau5, Masterclass.com), which means that the last part of the melody is resolving into the first part. In this example, he changed the first chord from major to minor to fit with the last chord:

> *"I approach music on an experimentation level, like I don't.. I don't dream up things, im not… you know.. fucking… whats his name from the beatles?.. like "I had It in a dream and you know, I transcribed it".. I need to hear it. I need to be on it and be experimenting with it, right then and there"* (Deadamau5, Masterclass.com).

Another method that Deadmau5 uses to create harmonic structure is called *"drone note"*. In this process, Deadmau5 starts with one synth playing one note over all the bars in a loop and he then experiments with writing all the chords to fit with that synth. He states that, by doing this, it is impossible to get lost and write a harmony that does not fit, and that it helps him to be free and creative but still manage to write a harmonic structure that fits together (Deadmau5, Masterclass.com). When he is done with the arrangement, he decides if he wants to leave the drone note on the track or to just remove it. Figure 5 is an example of a drone note on F in the song *"Phantoms Can't Hang"*.

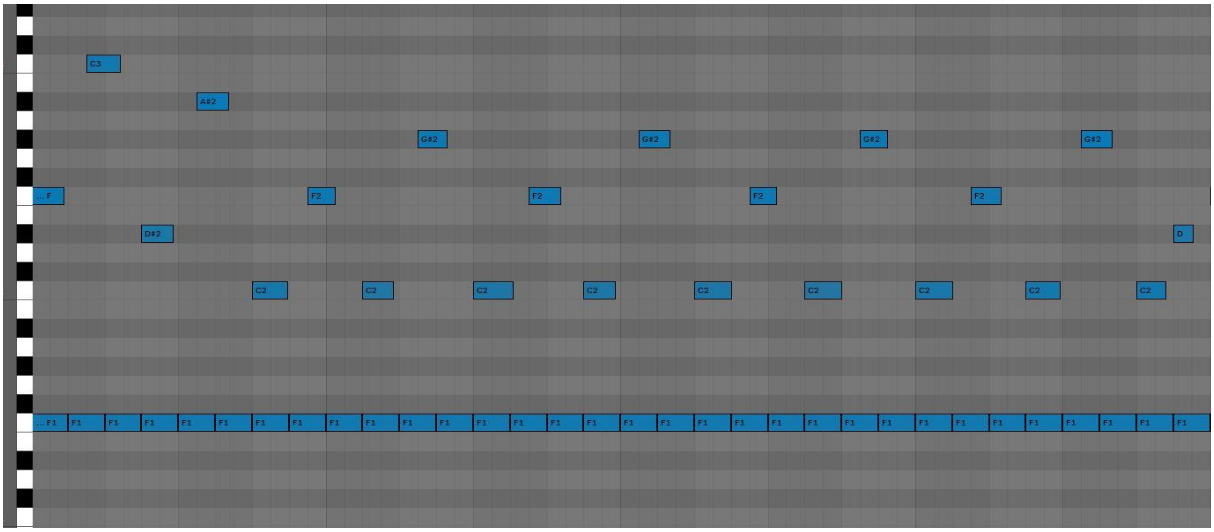


Figure 5. Illustrating Drone note harmony with MIDI notes in Ableton.

Deadmau5 uses this feature in songs like *"Phantoms Can't Hang," "Slip," "Brazil,"* and *"Aural Psynapse,"* which are all included in the training dataset for this study. The variation in methods Deadmau5 applies to create music can make it challenging to analyze its harmonic structure, but some harmonic features are common to his sound. An example of this is the song *"Some Chords,"* which has the chord progression:

G#m – E - F# - G#

This can be analyzed as a modal mixture type of chord progression. Modal mixture means that a chord is borrowed from the parallel key (anranga95)[26]; in this case, G sharp major is borrowed from the parallel key of G sharp minor. Deadmau5 has used this trick on other songs, and the most common version is to use a major chord variant of the tonic minor chord (anranga95).

In conclusion, this analysis of Deadmau5's music highlights some specific methods that were useful in the harmonic and rhythmic analysis of the music created by the AI, especially the concept of polymeter. Also, this in-depth explanation of how Deadmau5 creates music demonstrates that his music is a result of an iterative creative experimentation process.

[26] anranga95. *Harmony & Scales*. Retrieved November 2, 2020 from https://anranga95.wixsite.com/bookofmau5/blank-b4ggw

2.2 Artificial Intelligence

What is artificial intelligence?

Artificial intelligence (AI) can be described as making computers do things that human minds usually do. The idea is to make computers be able to use methods that we describe as intelligent, such as reasoning, learning, prediction, planning, and communication (Boden, 2018, p. 1).[27] However, it can also be about things that we do not categorize as intelligent, like computer vision, for example. It can be difficult to describe exactly what AI is because a good description of what human intelligence is does not exist (Boden, 2018, p. 1). In this context, the focus is on fundamental parts of intelligence such as learning and adapting, and making an AI model use these, which is also called symbolic processing (Marsland, 2014)[28]. There are many different types of AI and it is found everywhere (cars, phones, the internet of things, Siri, customer-segmentation and even the Google search engine) (Boden, 2018, p. 1). This thesis focuses on the different types of artificial neural networks, more precisely ML, unsupervised ML (also known as clustering), and DL.

"Intelligence is whatever machines haven't done yet." (Tesler's Theorem)

When did it start?

The idea of AI dates hundreds of years ago to Lady Ada Lovelace in 1840 (Boden, 2018, p. 6). Her aim was purely technological, and she believed that a machine could be able to compose music in any complexity and to express *"the great facts of the world"* (Boden, 2018, p. 6). Despite her effort to create an intelligent machine, the first AI system, the universal Turing machine, was created many years later by Alan Turing in England in 1936 (Boden, 2018, p. 7). The universal Turing machine was a mathematical system that could solve any mathematical problem by converting it to binary digits (0 and 1). In the 1950s, a handful of pioneers created a system called Symbolic AI, which is known as the first AI method (Chollet, 2018, p. 4)[29]. Symbolic AI, also known as GOFAI (Good Old-fashioned AI) used hardcoded rules created by programmers to solve a task and was only suitable to solve well-defined logical problems. Because Symbolic AI could not learn, and needed to have predefined rules, it was unable to

[27] Boden, M. A. (2018). *Artificial Intelligence A very short introduction*. Oxford University Press.
[28] Marsland, S. (2014). *Machine learning: An Algorithmic Perspective*. Chapman and Hall/CRC.
[29] Chollet, F. o. (2018). *Deep learning with python*. Manning Publications Co.

solve complex and abstract problems, like image classification (Boden, 2018)[30]. Symbolic AI was the most popular form of AI from the 1950s to 1980s until ML was invented and took its place (Boden, 2018, p).

AI can be split into two aims: technological and scientific

The technological aim of AI is about making AI do useful tasks. The scientific aim is about using AI to research human beings and other living things (Boden, 2018, p. 2). It is common for most scientists to specialize in only one of these fields, but many consider both (Boden, 2018, p. 2). While the development of AI has provided a lot of technological opportunities, like self-driving cars and virtual assistants such as Siri, it has also contributed to research in the fields of psychology and neuroscience. One example of this is that AI research has contributed to the understanding that the human mind is more complex than researchers previously thought (Boden, 2018, p. 2). This thesis will focus on both the technological and the scientific and try to connect them in the conclusion chapter. On the one hand, using AI to create music with unsupervised ML trained on EDM is indeed very technological. However, on the other hand, music as an object and the creative process is a scientific mystery, as will be elaborated on in Section 2.4.1 and discussed in Section 6.1. The approach to the psychological side is that, if AI can be used to create music, then maybe it can obtain some answers about how the human mind works in the creative process. In other words, how does the human mind make music, and what is creativity? This also raises the question of what is the purpose of music, and what is the connection between intelligence and music composition or intelligence and creativity?

As such, the goal of this thesis was never to create a kind of AI application that creates music without the aid of a human but rather to research how AI can be used in a creative collaboration with a human, and what we can learn from it. That is, not just to learn about the creative process itself but also to learn about how AI can be used in creative tasks, which can benefit AI science, neuropsychology, the music industry, and musicology to a very small extent. The big question that remains is: What is intelligence anyway, and is this kind of AI really intelligent? These questions about intelligence can be more philosophical than scientific (Boden, 2018, p. 106). Further, to evaluate a computer's intelligence, there must first be a basic definition or idea about what intelligence really is. Unfortunately, this is a subject goes beyond the scope of this thesis.

[30] Boden, M. A. (2018). *Artificial Intelligence A very short introduction*. Oxford University Press.

The goal of AI is to achieve general intelligence, known as Artificial General Intelligence (AGI) (Boden, 2018, p. 18). In AGI, the concept is about making AI use all of the different methods of intelligence, such as reasoning, vision, and communication, and combining them when needed (Boden, 2018, p. 18). In other words, to create an AI system that can function more like a human mind. Even though the current state of AI is impressive, compared to AGI, today's AI is still far away from achieving its goal. Modern supercomputers have proved to be beneficial for the development of AI, but more computer power is not enough. AGI needs new methods to solve problems (Boden, 2018, p. 19). The problem with having expectations about what we can achieve with AI that are too high is that this has previously led to *the "two winters of AI"* (Chollet, 2018, p. 12). These occurred when the expectations were too high over the short-term and technology failed to deliver what was promised. The first AI winter was in the 1970s, when experts believed that general intelligence was just around the corner, and as the technology failed to deliver, government and researchers' funds were diverted from the field (Chollet, 2018, p. 12)[31]. The second AI winter was in the 1980s, when expert systems were declared the next big thing. This was a knowledge-based system that used symbolic representation of human knowledge to try to resemble human reasoning. This technology became very popular among industries and from 1985 to 1990, several organizations invested around $1 billion each year to develop expert systems in their businesses. However, expert systems require a lot of human expertise, they are expensive to maintain and are very limited in scope. So, by the start of 1990, interest in expert systems died after years of spending. This led to the second AI winter (Chollet, 2018, p. 12).

This leads us to the third AI winter, which may be happening right now. As history has shown, predicting the future possibilities of AI is both difficult and often too optimistic. On the one hand, AI tools like self-driving cars and Siri both indicate promising advancements for the future of AI. On the other hand, the current state of AI is still far away from achieving AGI, and there is still no proof that it will ever achieve it. A common myth about AI is that it is an artificial representation of the human mind, and that it can think like a human to a small extent. The truth, however, is that we have no idea how a mind works, and there is no proof that there is a similarity between how human minds work and the different types of AI (Chollet, 2018, p. 8). Neural networks are a good example of this. The neural term is from the field of neurobiology, and the whole concept of a neural network is based on an old theory from that

[31] Chollet, F. o. (2018). *Deep learning with python*. Manning Publications Co.

field about how our mind works and learns. However, as stated before, there is no proof of this, and there are now more updated theory that indicate that the human mind is far more complex and works differently than a simple neural network (Chollet, 2018, p. 8).

Narrow intelligence

The current state of AI is Artificial Narrow Intelligence, which means AI is being used to solve a specific task that is narrow; that is, the opposite of Artificial General Intelligence (Dickson, 2020)[32]. So, to successfully use an AI method to solve a task, the researcher needs to find a way to narrow the task to make it possible for AI to complete it. In this thesis, this is done through the implementation of the MIDI system in Chapter 4 and the transcribing of the songs.

[32] Dickson, B. (2020). *What is artificial narrow intelligence (Narrow AI)?* Retrieved April from https://bdtechtalks.com/2020/04/09/what-is-narrow-artificial-intelligence-ani/

2.2.1 Machine Learning

Machine learning (ML) is a subcategory of AI that uses patterns and interference instead of instructions when performing a task. The concept of ML is that an application can learn from data to program itself instead of needing to be programmed manually (Chollet, 2018, p. 28)[33]. ML is inspired by ideas from biology, statistics, mathematics, physics, and neuroscience, and became recognized as an AI category during the 2010s (Marsland, 2014)[34]. The principle of ML arose from this question posed by Alan Turing: Could a computer go beyond what we know how to order it to perform and learn on its own how to perform a specified task? And could a computer surprise us? Rather than programmers crafting data-processing rules by hand, could a computer automatically learn these rules by looking at data? (Chollet, 2018, p. 5)[35] With these questions, Turing is reflecting on the core fundamental of ML, which is to make a computer learn from data to solve a task. The most important aspect about this is the question*: "Could a computer go beyond what we know how to order it to perform?"* This means that ML is not just about making a program to execute a task without the need of a programmer, but rather that ML could create a solution to a problem that programmers have not thought of yet. In other words, ML gives AI the ability to create tools that could solve tasks, which was impossible to do in classical programming. An example of this is image classification.

In this image classification example, we want an application to distinguish pictures of cats from pictures of both cats and dogs. To do this with supervised ML, we need three elements:

Input data (samples)

Since the task is image classification, then the data, in this case, would be pictures.

Examples of expected output (answers)

If the task is to classify if the input image is a cat or not a dog, then we need a lot of pictures of cats.

A measuring algorithm (learning)

[33] Chollet, F. o. (2018). *Deep learning with python*. Manning Publications Co.
[34] Marsland, S. (2014). *Machine learning: An Algorithmic Perspective*. Chapman and Hall/CRC.
[35] Chollet, F. o. (2018). *Deep learning with python*. Manning Publications Co.

This is to measure the distance between the algorithm's current output and the expected output. This step is the learning part and is necessary to adjust the algorithm so that the machine can learn to recognize cats.

The reason it is better to choose ML to solve this task instead of classical programming is that, in classical programming, there is a need to specify all of the rules to make the software distinguish pictures of cats from dogs and tell the software exactly what to look for in each pixel. This can also be very time-consuming and expensive since a programmer can only work at a certain speed, and if the program code itself is 300,000 lines, it can take years. Basic supervised ML can easily do this task with just a computer, a bit of code in python, appropriate training data, and some hours of training. Figure 6 is what the process looks like in an ML model, where the data are pictures of cats and dogs, the answers are pictures of cats, and the rules are the program code that defines how the program should solve the task.

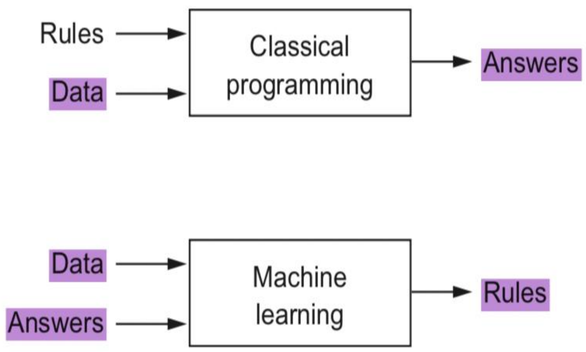


Figure 6. The difference between classical programming vs ML (Chollet, 2018, p. 5).

Even though there probably are some algorithms and code libraries that can speed up the process of classical programming, if the task changes to other types of data, there may be a need to start all over. ML is superior for this type of challenge because the same model of ML that can work with image classification can also be applied to other tasks like music and writing. In other words, the same type of ML model can be used to solve different types of task since the machine is responsible for making the rules; the programmer only needs to make the input data and answers fit with the model. By explaining it like that, it can look like ML is doing all the heavy lifting, and in some cases, that is probably true. However, in others, presenting data for the ML model can be a very difficult task. This is especially true when it comes to using ML to create music. In this thesis, for example, the difficulty is not how to use the ML model or application but how to translate the music into a symbolic representation data that the ML application will understand and can learn from.

ML can be divided into 4 main categories (Marsland, 2018)[36]:

Supervised ML:

The learning algorithm in supervised ML is trained on data that is labeled so that the algorithm can measure its accuracy against examples that are correct (Marsland, 2018), like in the cat image classification example above. This the most common type of ML and has some similarities with unsupervised ML.

Unsupervised ML:

The learning algorithm in unsupervised ML uses data that is unstructured and unlabeled. The algorithm must analyze and identify patterns in the data and try to replicate them by using predictions based on the patterns (Marsland, 2018). This method is also known as clustering and is the method of AI used in the thesis, as will be elaborated on in Section 2.2.2.

Reinforcement learning:

Reinforcement learning is a combination of supervised and unsupervised learning where the learning algorithm gets told whenever the output is wrong but not how to correct it. This is also called learning with a critic (Marsland, 2018).

Evolutionary learning:

This is an ML algorithm that is based on the evolutionary concept, like survival of the fittest. It models organisms that improve their survival rates in their environment (Marsland, 2018).

[36] Marsland, S. (2014). *Machine learning: An Algorithmic Perspective*. Chapman and Hall/CRC.

Figure 7 is an information chart of the different types of ML, with illustration of methods that are similar.

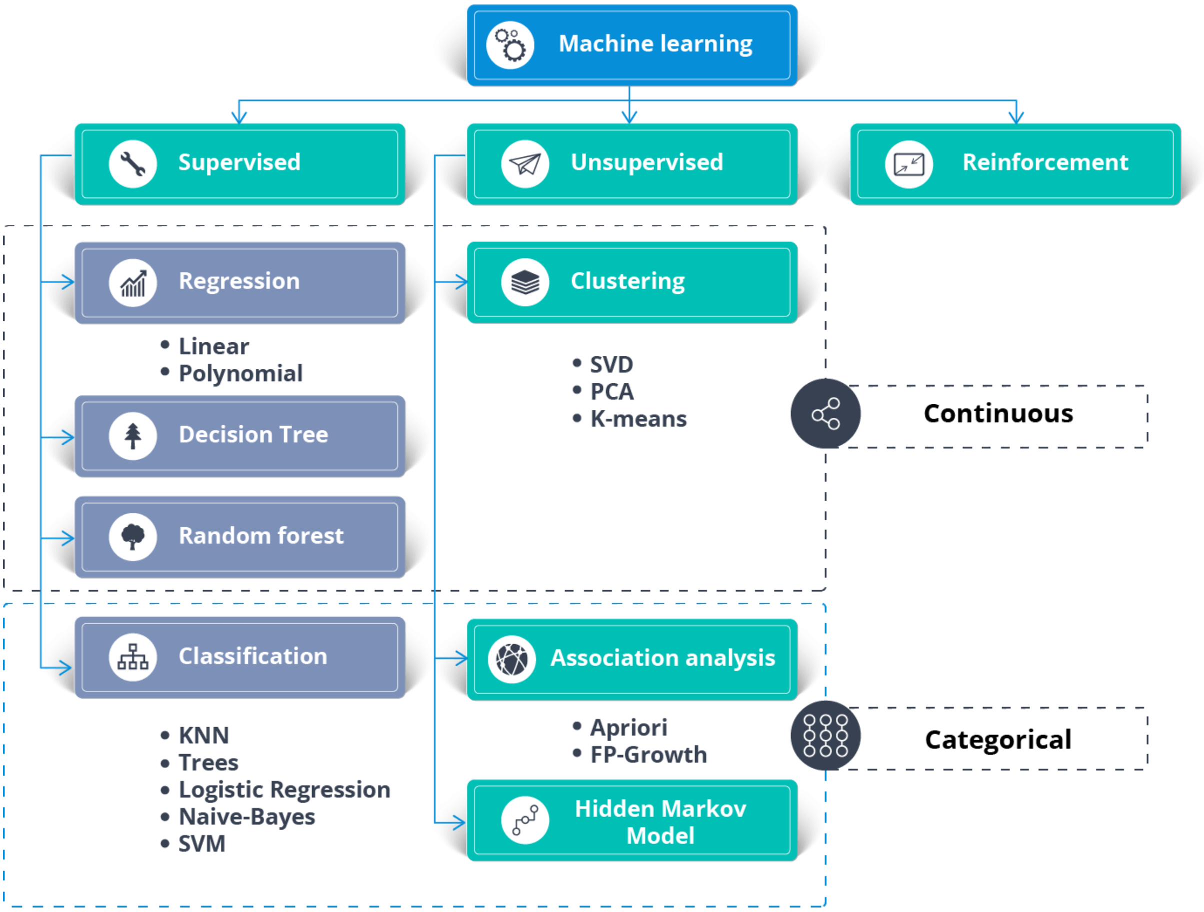


Figure 7. An information chart displaying different types of ML (Marsland, 2014).

Terminology

Before we explore DL, it is important to define some basic ML terminology that will help to understand how ML works. This is also the terminology that is used in the research and discussion part of this thesis.

Overfitting:

A common problem in ML, it means that the ML model is learning the data too well and starts to learn noise and random notes as fundamental concepts (Chollet, 2018, p. 104)[37]. In this

[37] Chollet, F. o. (2018). *Deep learning with python*. Manning Publications Co.

context, it means that instead of making original music, the algorithm is copying some of the musical ideas that it was trained on. The reason this was a problem in this thesis is that the ML algorithm was intended to create music that is as close to the original music as possible, but still original. The reason this happens is that the ML algorithm has only been exposed to the music that was used as training data, so it has no other option than to try to recreate it.

Underfitting:

This is the point where the ML algorithm has inappropriate data to produce accurate results (Chollet, 2018, p. 104). In other words, the ML algorithm fails to learn because the data size is too small, or the data is too unstructured for the algorithm to learn a pattern. An example of data that are too unstructured in this case would be if the songs had a lot of variation in the arrangements and too many layers, like different instruments. This would make the task to find a common pattern too challenging, as in the Avicii example in Section 4.4.1.

Law of diminished returns:

The law of diminishing returns is a common terminology in economics that means that if the production input increases while all other factors are constant, the output will eventually decrease (Saumelson & Nordhaus, 2001, p. 108-109)[38]. In this context, this means that a DL model can continue to train but will eventually stop increasing in learning.

Batch.Size and Epoch

The epoch is a value that contains the number of times the whole dataset is processed in an ML model. However, in most cases, the data size is very large and must be divided into different groups, and then multiple iterations are used to process the whole data size (Brownlee, 2019)[39]. The amount of data per group is called the Batch.Size and the number of the Batch.Size will decide how many iterations of data is needed for each epoch (Brownlee, 2019). For example, if the data size is 100 images, and the Batch.Size is 2, then the model will split the data into 2 groups of 50 images and use two iterations per epoch. Increasing or decreasing the Batch.Size can affect how fast the model learns, but in this context, changing the epoch amount will achieve the same results. This will be explained further in Section 4.3.2

[38] Samuelson, P. A., & Nordhaus, W. D. (2001). Micoreconomics. In (pp. 454). McGraw-Hill Education
[39] Brownlee, J. (2019). *Difference Between a Batch and an Epoch in a Neural Network*. Retrieved April 10, 2021 from https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/

Temperature:

Temperature is a value that changes the way an ML model generates data after the training process. A high temperature means that the MIDI files generated will be more experimental, but in this context, it generally means more noise and less house music. Low temperature will make the model generate music that is closer to the original but can lead to overfitting (Karpathy 2015)[40].

[40] Karpathy, A. (2015). *The Unreasonable Effectiveness of Recurrent Neural Networks*. Andrej Karpathy blog. Retrieved November 20, 2020 from http://karpathy.github.io/2015/05/21/rnn-effectiveness/

2.2.2 Deep Learning

Like ML is a subcategory of AI, Deep Learning (DL) is a subcategory of ML. The *term "Deep Learning"* is not about gaining a deeper understanding but using more layers in the training model. The number of many layers that contribute to a training model is called depth, which explains the name *"Deep"* Learning (Chollet, 2018, p. 8)[41]. While classical ML tends to focus on only using one or two layers, modern DL models usually use between ten to hundreds of layers (Chollet, 2018, p. 8). Figure 8 illustrates that DL is a subcategory of ML, which again is a subcategory of AI. Neural networks are used in both DL and ML, but in different ways.

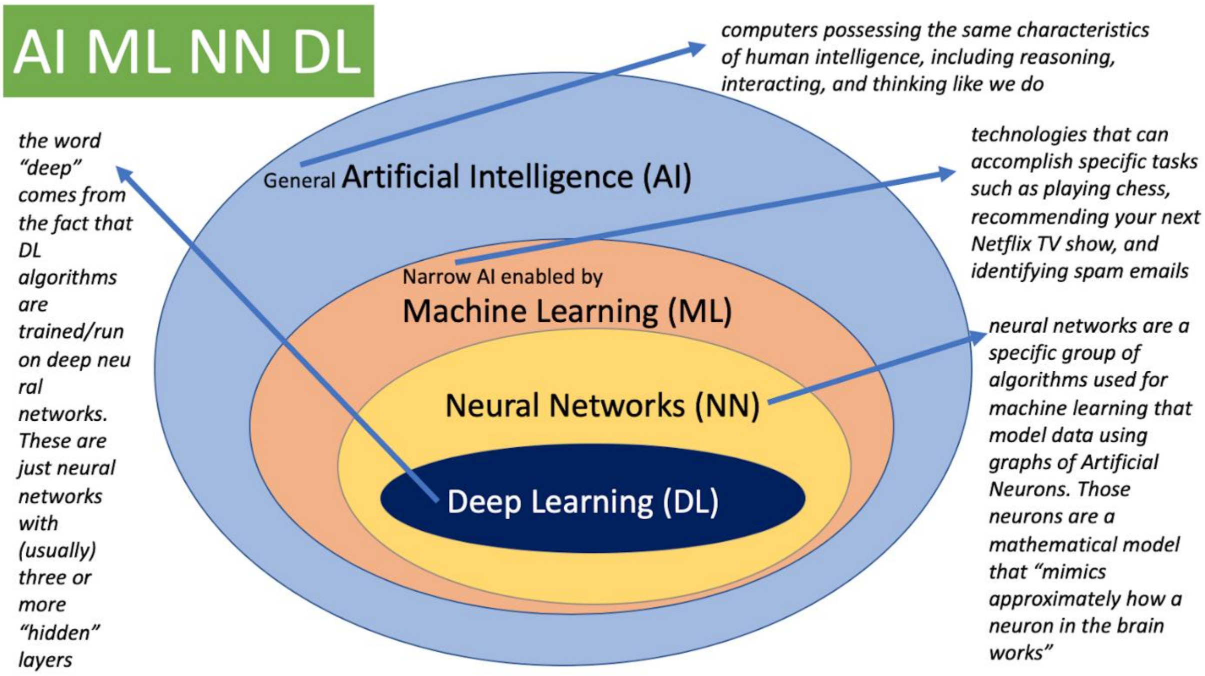


Figure 8. Overview of the different AI models[42].

This can be a bit confusing since neural networks are often referred to alone as a main category of AI. To explain it simply: A neural network is a training model that both ML and DL uses; the difference is that ML uses it in a simpler way, while DL use it in a more complex way. In this context, more complex means that it is often used with more layers and alongside additional methods such as Long Short-Term Memory (LSTM) and backprop. Another way to describe this is that DL is a further development of ML and that they are used for different purposes.

---

[41] Chollet, F. o. (2018). *Deep learning with python*. Manning Publications Co.
[42] https://1.bp.blogspot.com/-gezxES34R-M/XS0eEXAOySI/AAAAAAAACFg/tWdqscm6ci0Zv5dqap76lgL5YSsUeDHOQCLcBGAs/s1600/Screen%2BShot%2B2019-07-15%2Bat%2B11.14.11%2BAM.png

The fundamental science behind DL, like backpropagation, is that neural networks and LSTM were already understood and developed in 1997, but it was not until 2012 that DL become a popular field of AI (Chollet, 2018, p. 20)[43]. The reason for this is the development of hardware, dataset, and algorithmic advances (Chollet, 2018, p. 20). This made it possible for scientists to run small DL models on their laptops using a GPU, which was unthinkable in the 90s. There are 3 key features that explain why DL has become the new revolution of AI, as follows (Chollet, 2018, p. 23).

Simplicity

In contradiction to early approaches in ML, the DL model can be used without having to do anything with the model. Everyone can simply use a DL model on a computer using programming language like python combined with a DL library like Keras. With a simple code in python, it is possible to train your own DL algorithm right away; the challenge is more about getting enough data and the right kind of data rather than using a DL model.

Scalability

DL can take full advantage of processing power, like GPU parallelization. This makes the only bottleneck the amount of GPU processing power available. Refer to Section 4.3 for more on that issue.

Versatility and reusability

Since DL can continue training without having to be restarted from scratch this makes DL capable of continuous learning. As mentioned in the cat image classification example, a DL model can be used to solve a lot of different tasks. Another example of this is the YouTube video by Carykh referenced in the introduction to this thesis. In that video, Carykh used a DL model called Char-RNN that was intended to learn from text (books, novels, everything written) and write original text based on the training (Karpathy, 2015)[44]. However, Carykh used this model to create baroque music by converting music to text and training the model on it, without doing anything to the model.

[43] Chollet, F. o. (2018). *Deep learning with python*. Manning Publications Co.
[44] Karpathy, A. (2015). *The Unreasonable Effectiveness of Recurrent Neural Networks*. Andrej Karpathy blog. Retrieved November 20, 2020 from http://karpathy.github.io/2015/05/21/rnn-effectiveness/

2.2.3 Deep Learning Frameworks

Since the field of DL is changing rapidly, it can be difficult to get an overview of the current state, all the contributors, and the different models in AI. However, the different frameworks for DL are significant to the current state of AI and relevant to this thesis. At the time of writing, Pytorch, Keras, and TensorFlow are the three most popular DL frameworks among both data scientists and beginners in the field of AI (Deb, 2020)[45].

Keras

Keras is a high-level DL library for python that is one of the most used and leading contributors in the field of ML. It is created to be easy to use, so you do not have to be a computer scientist to use ML (http://keras.io/about)[46]. Combined with TensorFlow, this is the model that is being used in the research in this thesis and will be explained in-depth in Chapter 4. Keras is built on an easy architecture, which makes debugging less frequent and easy (Deb, 2020)[47].

TensorFlow

TensorFlow is an end-to-end open-source platform for ML. It offers a lot of the same features as Keras, but TensorFlow is a low-level library that can be much harder to use (Yegulalp, 2019)[48]. While the goal of Keras is to be easy to use, TensorFlow's focus is more on maximizing hardware processing power (Chandra, 2019)[49]. It is common to use Keras as a wrapper for TensorFlow, meaning that you can access TensorFlow features from the Keras interface. However, *"one size fits all"* does not apply to ML; for example, if you want to do low-level changes to a DL model, then you might want to use a low-level API like TensorFlow instead of Keras (Chandra, 2019).

Pytorch

[45] Deb, S. (2020, April). Keras vs TensorFlow vs PyTorch : Comparison of the Deep Learning Frameworks. *Artificial Intelligence*. https://www.edureka.co/blog/keras-vs-tensorflow-vs-pytorch/
[46] *About Keras*. (2021). Retrieved February 10, 2021 from https://keras.io/about/
[47] Deb, S. (2020, April). Keras vs TensorFlow vs PyTorch : Comparison of the Deep Learning Frameworks. *Artificial Intelligence*. https://www.edureka.co/blog/keras-vs-tensorflow-vs-pytorch/
[48] Yegulalp, S. (2019). *What is TensorFlow? The machine learning library explained*. https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html
[49] Chandra, R. (APR 4, 2019). *The What's What of Keras and TensorFlow*. https://www.upgrad.com/blog/the-whats-what-of-keras-and-tensorflow/

Pytorch is an open-source DL framework developed by Facebook for the purpose of computer vision and natural language processing (Deb, 2020)[50]. Pytorch also uses python as a programming language and is faster than Keras, but it is a low-level API, which is more difficult to use (Deb, 2020). Pytorch has better debugging capabilities than the other two and is best suited for short-duration training projects.

[50] Deb, S. (2020, April). Keras vs TensorFlow vs PyTorch : Comparison of the Deep Learning Frameworks. *Artificial Intelligence*. https://www.edureka.co/blog/keras-vs-tensorflow-vs-pytorch/

2.3 Algorithmic Music

As mentioned in the introduction chapter, Algorithmic Music can be described as using computers in the creative process of music making. More precisely, it is not just about making music with technical tools, like creating a song in Logic Pro on a MacBook, but it is also about involving an objective entity that influences or decides how the music is going to sound (Dean & Mclean, 2018)[51]. In traditional music creation, the musician decides how the music should sound, but with Algorithmic Music, the algorithm takes some or all of the artistic choices. The word *algorithm* can be defined as a set of operations or rules, but this definition is not always right when it comes to Algorithmic Music (Dean & Mclean, 2018). Such a definition is very broad and would mean that basically every computer program used to make music should be defined as Algorithmic Music. Because Algorithmic Music is more about how to use the different algorithmic approaches to make music and not what tools are used to make music (Dean & Mclean, 2018).

2.3.1 Music and AI

Examples of AI in music can be divided into the following categories: compositional AI, improvisational AI, and AI for musical performance (Mantaras & Acros, 2002)[52]. The history of musical AI has its roots in Algorithmic Music composition. Therefore, the examples of compositional AI have the longest history. There have been numerous projects using AI to create music, so in order to limit the examples, only projects with similar AI methods are mentioned here.

Illiac Suite is the first composition made with the aid of a computer in the creative process, also known as computer-assisted composition (Hiller & Isaacson, 1958)[53]. It was a string quartet that used several methods of random processes such as probability distribution (Sandred et al., 2009, p. 1)[54].

[51] Dean, R. T., & Mclean, A. (2018). *The Oxford Handbook of Algorithmic Music*. Oxford University Press. https://doi.org/10.1093/oxfordhb/9780190226992.001.0001
[52] Arcos., R. L. D. M. a. J. L. (2002). AI and music: From composition to expressive performance. *AI magazine*
[53] Hiller, J., L. A., & Isaacson, L. M. (1958). Musical Composition with a High-Speed Digital Computer. *Journal of the Audio Engineering Society*, *6*, 154-160.
[54] Sandred, Ö., Kuuskankare, M., & Laurson, M. (2009). Revisiting the Illiac Suite - A rule-based approach to stochastic processes. *Sonic Ideas/Ideas Sonicas*, *2*, 42-46.

FolkRNN is a research project that uses AI to create Celtic Folk Music by converting sheet music into a symbolic system of ASCI characters (ABC) (Sturm & Ben-Tal, 2017)[55]. The AI model is a Recurrent Neural Network that is trained on more than 23,000 ABC transcriptions of folk music. The method is to predict the ABC character based on the training data.

BachBot is an automated composition system that creates polyphonic arrangements in the style of Bach by using a DL model with LSTM (Liang, 2016)[56]. The aim of the project was to create an automatic stylistic composition model that is capable of imitating Bach's composition style by using generative probabilistic sequence models trained on Bach Chorales (Liang, 2016, p. 2).

DeepBach is another DL project that specializes in creating music in the style of Bach (Hadjeres et al., 2017)[57]. This model uses a pseudo-Gibbs sampling trained on an adaptive representation of musical data and is more flexible than BachBot because BachBot can only generate music in the key of C and only the soprano voice can be fixed (Hadjeres et al., 2017, p. 2).

2.3.2 Music and Machine Learning in Composition

In Section 2.2.1, it was explained that ML is a method of AI that has the ability to learn in order to solve a task and what features this contributes to technology. But what features does this offer to music composition? In the section *"Machine learning as a creative Tool"* in the *Oxford Handbook of Algorithmic Music* (Dean & Mclean, 2018)[58], ML is defined as a creative tool when it replaces the role of a creative agent in a music composition collaboration. The section uses works created by AI such as Assayag (2006) and investigates the role of AI. However, it is not just about replacing the role of a human in a creative collaboration but instead the desire for the stylistic injection the AI adds to the creative work. In this context, the stylistic injection is defined as the AI method of collaborating with creative ideas in its unique fashion that is different from that of a human artist. So, the goal is not to make the AI create music like a

[55] Sturm, B., & Ben-Tal, O. (2017). *'Machine folk' music composed by AI shows technology's creative side*. Retrieved March from https://theconversation.com/machine-folk-music-composed-by-ai-shows-technologys-creative-side-74708
[56] Liang, F. (2016). *BachBot: Automatic composition in the style of Bach chorales* University of Cambridge]. Churchill College.
[57] Hadjeres, G., Pachet, F., & Nielsen, F. (2017). DeepBach: a Steerable Model for Bach Chorales Generation.
[58] Dean, R. T., & Mclean, A. (2018). *The Oxford Handbook of Algorithmic Music*. Oxford University Press. https://doi.org/10.1093/oxfordhb/9780190226992.001.0001

human but rather to use its features like an external memory that can redefine musical ideas from the past with an innovative reconstruction.

2.3.3 BachProp

BachProp is a neural composer algorithm that is designed to create new music scores in any style, devised by Florian Clombo and Wulfram Gerstner in 2018[59]. Its developers noticed that previous attempts to create music with neural networks were mainly designed to create music in a specific style with limitations (Colombo & Gerstner, 2018, p. 1). For example, FolkRNN can only create monophonic melodies and Bachbot was designed to create Bach chorale harmonies, which require preprocessing of the MIDI data. BachProp, however, requires no preprocessing of the data, and Colombo and Gerstner claim that it can be used to create music in any style without having to do additional modification to the MIDI (Colombo & Gerstner, 2018, p. 1). In the context of this thesis, however, some preprocessing was done to the MIDI data using a custom MIDI system to make up for the lack of data size, as explained in Chapter 4.

BachProp also has a MIDI normalization method that removes unnecessary information, like note on and note off, and replaces it with a duration code. In this way, it processes the MIDI notes into a rhythm normalization that maps the note duration and timing into a system of note values (Colombo & Gerstner, 2018, p. 2). This is a feature that makes the algorithm a great choice of method for the thesis, because Deadmau5 music is mostly based around rhythmical melodic themes, like the polymeter, and the Arp melodies mentioned in Section 2.1.1.

Another important feature of BachProp that is important to address is its ability to learn transposition and tonality. In music theory, one song can be played in 12 different keys but still be the exact the same song. For example, if the children's song *"Mary had a Little Lamb"* was played in the key of C, the notes on the first two bars would be: E D C D E E E. But if it was played in the key of D, the notes would be: F# E D E F# F# F#. Even though the notes are different, people will still recognize that it is the exact same melody. A DL algorithm, however, will not recognize that it is the same melody, and can therefore treat the different keys as two

[59] Colombo, F., & Gerstner, W. (2018). BachProp: Learning to Compose Music in Multiple Styles.

different languages because it needs to learn what tonality is. BachProp addresses this issue by transposing each song in all the different tonality variations in order to teach the algorithm about tonality and transposition (Colombo & Gerstner, 2018, p. 4):

> *"In order for BachProp to learn tonality and transposition invariance of music, we randomly transpose each song at the beginning of every training epoch within the available bounds of the pitch set. In other words, for each song we compute the possible shifts of semitones (basic musical unit separating each pitch) and sample one that is then applied as an offset to all pitches in the song. Because a single MIDI sequence will be transposed with up to 50 offsets, this augmentation method allows BachProp to learn the temporal structure of music on more examples"* (Colombo & Gerstner, 2018, p. 4).

2.4 Evaluating Computer Creativity

To explore the different evaluation methods of computer creativity, there was a need to gather information about how other research projects mentioned in this thesis have evaluated the music created by AI. Folk RNN used feedback from folk music enthusiasts on thesession.org to evaluate the music (Sturm Ben-Tal)[60]. Wallace used cross-validation and based her thesis on qualitative interviews (Wallace, 2018, pp. 6-7)[61]. BachBot implemented a Turing test with a statistic to measure how many participants successfully discriminated the AI music from the human music (Liang, 2016, p. 3)[62]. DeepBach uses the same evaluation, with a Turing test and the discrimination statistical method (Hadjeres et al., 2017, pp. 6-7)[63]. BachProp used a similar approach with an online survey and a cross-validation test (Colombo & Gerstner, 2018, p. 6)[64]. The main method that is being used in this thesis is the same one that DeepBach uses, which is a Turing test with a discrimination statistical analysis, which is presented in Chapter 3. Another evaluation method is to analyze the AI music in the context of the harmonic signature of the artist, which is presented in 2.1.1 and analyzed in 4.5.

---

[60] Sturm, B., & Ben-Tal, O. (2017). *'Machine folk' music composed by AI shows technology's creative side*. Retrieved March from https://theconversation.com/machine-folk-music-composed-by-ai-shows-technologys-creative-side-74708
[61] Wallace, B. (2018). *Predictive songwriting with concatenative accompaniment* University of Oslo]. University of Oslo.
[62] Liang, F. (2016). *BachBot: Automatic composition in the style of Bach chorales* University of Cambridge]. Churchill College.
[63] Hadjeres, G., Pachet, F., & Nielsen, F. (2017). DeepBach: a Steerable Model for Bach Chorales Generation.
[64] Colombo, F., & Gerstner, W. (2018). BachProp: Learning to Compose Music in Multiple Styles.

2.4.1 Music as Data and Difficulties With Creative Prediction

This section aims to explain why music can be challenging to create with machines. Notably, the most significant reason for this is the complexity of music itself and not the technology. The definition of music is difficult, complex, and depends on which kind of music we are talking about (Ruud, 2016, p. 33)[65]. In this context, music is narrowed to Western popular music songs only, and not all kinds of music. The way a DL model works in this context is that the model trains on the dataset to learn from it by finding patterns and trying to recreate them using prediction. However, to do that, it needs to have data that contains a sufficient representation of what it is supposed to learn. ML is not magic, and it cannot be fed with just anything and be expected to learn. This is a crucial problem with music as opposed to other cases of ML like image classification. Perhaps when the development of ML becomes more advanced, this could be easier, but right now the hypothesis is that the problem is more about music itself than the limits of AI, due to the following aspects:

- **Sound**

An AI model can work with music as an input source in two ways: symbolic representation and sound. Symbolic representation as data input is most common in ML and is the approach used in the thesis. Using ML with sound as an input is still under development, with most using voice recognition applications like Amazon (Geitgey, 2016)[66]. For the sound to be used as an input, it needs to be analyzed, but sound analysis is still very undeveloped, and no analysis methods exist for using sound as a direct input to the extent of training on full songs (Clarke & Cook, 2004, p. 195)[67]. There are, however, methods to use AI to extract polyphonic pitch from audio files, such as *"Polyphonic Pitch Tracking with Deep Layered Learning"* by Karl Elowsson[68]. Yet methods like this are not sufficient to extract the musical information from EDM songs, like the songs used in this thesis. Also, this method is only using the AI to extract the pitch and is not training to predict pitch or create music. Technically, both inputs are eventually the same

[65] Ruud, E. (2016). Musikkvitenskap. In (pp. 33). Universitetsforl.
[66] Geitgey, A. (2016, February 2, 2021). Machine Learning is Fun Part 6: How to do Speech Recognition with Deep Learning. *Adam Geitgey*. https://medium.com/@ageitgey/machine-learning-is-fun-part-6-how-to-do-speech-recognition-with-deep-learning-28293c162f7a
[67] Clarke, E., & Cook, N. (2004). Empirical Musicology: Aims, Methods, Prospects. In (pp. 157-197). Oxford University Press, Inc.
[68] Elowsson, K. A. (2002). Polyphonic pitch tracking with deep layered learning. *Journal of the Acoustical Society of America*, 446-468.

data since the sound will at one point be converted into 0 and 1, which is commonly called digital signal processing (DSP).

- **Symbolic representation**

Another problem is that the music used as data will need be presented to an AI while still having all of the necessary elements of the music. The intelligence is not the problem here but rather the translation of music to a symbolic representation that is sufficient for AI. The most common and developed notation system for music are sheet music, tablature, chord charts, and MIDI. For the purposes of AI, sheet music is way too advanced, has many variations, and needs human interpretation (Dahlstedt, 2018, p. 22)[69]. Also, sheet music is a good representation of classical piano music, but it cannot explain modern sound design. For example, if dubstep music is transcribed with sheet music, the sheet music does not contain any information about how the music sounds, only what notes are played. This is because in some genres like dubstep, the sound can be more important than the harmony. The same goes for chords, tabs, and MIDI, and any other notation system for music.

- **Layers**

In the genre of electronic house music, a layer is a definition of all of the different elements of music, such as instruments, song structure, sound effects, and harmony. For example, one song could contain a bass synthesizer, drums, pad synth, and melodic synth, all playing different harmonic parts of the song, while the song also contains a structure of different parts like the intro, build, drop/chorus, outro, and bridge. The sum of all layers combined makes this very complex and maybe impossible for a DL model, even with a lot of data. That is also a challenge since most songs have variations and do not follow the same recipe.

- **Harmonic Glitches/noise**

As mentioned before in the qualitative observation in Chapter 2, when using ML to create something, it is common to get an unfiltered output. For example, if a DL model is trained on a lot of books to get it to output good results, the output would still probably still contain some words that sound like jibberish. Even though this is just a small percentage of the output, it would still need a person to supervise it and filter out the glitches. This is an example that

[69] Dahlstedt, P. (2018). *Big Data and Creativity (preprint)*. https://doi.org/10.13140/RG.2.2.23657.60007

suggests DL needs supervision and cannot yet be used as a standalone application to create a creative product such as a song. However, this is only in the case of using DL to create the whole product or a significant part of it. Regardless, the conclusion here is that DL is not yet able to create a complex creative product on its own.

- **Genre differences**

As mentioned briefly before, using DL in classical music is a totally different task than for house music because the differences between the genres are significant. This means that if a model is developed to create classical music, like DeepBach (Hadjeres et al., 2017)[70], it would be unable to create pop music. However, this is specific to music and not literature, for example. Because crime novels use the exact same written language as poetry, a DL model can be trained on different data without having to adjust to the genre. The same example goes for image classification. BachProp's developers claim that it can create music in any style (Colombo & Gerstner, 2018, p. 1)[71], but this only if the style of music has a symbolic representation. As mentioned before, there exists no symbolic representation to presented EDM like dubstep in the way that sheet music can represent classical piano music.

- **Vocal**

As mentioned above, sound as a direct input is still a challenging field, and even though there are some methods, the technology is simply not adequate yet (Clarke & Cook, 2004, p. 195)[72]. Also, there is no symbolic system to represent vocals because vocals as an instrument itself consists of many elements like timbre, pitch, diction, and vocal techniques. An instrument like piano, for example, is mechanical and developed by music technology, hence it can be converted to a symbolic representation with sheet music or MIDI without the loss of musical interpretation as long as the transcribing is exact. Some will argue that the MIDI system only covers a velocity scale of 128 and that an interpterion of human playing requires a bigger scale to be a perfect representation. Even though that can be true, it is not seen as a challenge when training ML models on classical music.

[70] Hadjeres, G., Pachet, F., & Nielsen, F. (2017). DeepBach: a Steerable Model for Bach Chorales Generation.
[71] Colombo, F., & Gerstner, W. (2018). BachProp: Learning to Compose Music in Multiple Styles.
[72] Clarke, E., & Cook, N. (2004). Empirical Musicology: Aims, Methods, Prospects. In (pp. 157-197). Oxford University Press, Inc.

- **Trends and sociologically esthetics**

Billie Eilish fans will listen to her music, not only because of the esthetics of the music but also because they like Billie Eilish. How can a machine create the same relationship? This is a broad question about music identity vs. AI that will not be elaborated on to limit the scope of this thesis. The point here is that identity is important in commercial music and this could affect the motivation to not use AI. To summarize, people want icons not machines.

- **Small or unstructured data**

This is a common problem in the field of ML and especially DL since this method requires a large amount of appropriate data. This thesis proposes a solution to this, as presented in Chapter 4. However, this solution only applies to using AI to create ideas for instrumental house music, so a small or unstructured dataset will still be a problem. Also, this approach has limits and disadvantages, such as the effort of building a dataset.

- **Scientific and philosophical understanding of music**

What is music? There is no short answer to this question, so no attempt will be made to answer it in this thesis. However, it is important to mention that music as an object can be challenging to define and it would be problematic to do a scientific explanation of how music can be made by computers when it is challenging to explain how humans make music. This thesis and similar research projects mentioned propose a solution to this, by using ML to create music. Yet since the current state of AI is Artificial Narrow Intelligence, and a song consists of different layers such as melody, harmony, and text, this can make the task too advanced for a narrow method in the context of creating full-length songs.

3 Method

This thesis uses an iterative research process, a Turing test, and a statistical evaluation of the Turing test as research methods. There is also a small qualitative observation method to check the quality of the music before initiating the Turing test. The research for this thesis can be divided into two main parts: *creating the music* and *evaluating the music*. The first part is where the iterative process is used, which is about how to use AI to create music. The part starts with assembling a list of which songs to use, transcribing them into MIDI using a custom-made system, and training the music on a DL model using BachProp with Keras. As explained in Section 2.3.3, Colombo and Gerstner have demonstrated the possibilities of creating music with BachProp by using examples of classical music, but this raises some challenges when used with EDM, because of the lack of consistent data, harmonic structure, and instrumentation. In classical music, the songs can be a collection of hundreds of songs that follow the same arrangement made by the same composer, like the chorale works of Bach. The large collection of songs and the consistent harmonic arrangement used in them makes classical music ideal for ML. Nothing like this is common in mainstream EDM music, and so the system in this thesis is designed to solve this challenge by manipulating the way MIDI data is presented to the neural network, which is shown in Chapter 4. Also, the songs are carefully selected to be best suited for the task and to make sure that the collection of songs (data) was large enough to achieve the best results when trained on DL. The next step was to do a qualitative observation test to check if the quality of the music is suitable to run a Turing test. In this early stage of self-evaluation of the music, there are 5 steps to evaluate if the quality is met, which will be explained in the quality observation test Section 3.1.4. The reason for this is to avoid the Turing test immediately failing and to ensure the quality of the artificial music is as close to the original music as possible while still being an objective and representative artificially-made product. The concept is that if the music fails to pass the qualitative observation, then the process will start over again and repeat all of the tasks until the qualitative test is passed. When the music passed the threshold of the qualitative observation test, part two of the thesis began, which is the evaluation of the music. In part two, three types of online Turing tests were executed, where the participants listen to the music and distinguish which songs were made by AI from the collection of samples. Finally, the statistical evaluation from the results of the Turing test is then used to support the conclusion in the discussion part of the thesis. In Section 2.3.1., we saw that research like this has been done before; however, the combination of DL, a small dataset, and EDM is still, to the researcher's knowledge, very original. A challenge when considering similar research for inspiration is that most of the other projects use AI to recreate music alike to the original training data. One example of this is Nuanáin's doctoral thesis (Nuanáin, 2018)[73], in which he uses AI to recreate loops with a sound recreating method. This allows him to use quantitative analyzing methods, which makes it much easier to evaluate. In this thesis, the goal is to use AI to create original music, not just replicate the music. This makes it necessary for different research methods to be used. Another example of this is Wallace's master thesis, *"Predictive Songwriting with concatenative accompaniment"* (Wallace 2018)[74], in which an ML algorithm is used to create an accompaniment to melodies in real-time while improvising. Since the task of the algorithm is to predict the harmony to the melody, there exists an objective way of measuring how accurately the algorithm is so. Because of this, Wallace uses quantitative research methods like cross-validation to evaluate how much the models have learned. In this thesis, the task of the algorithm is to create music that is original, and it is supposed to create all of the music by itself, except for the arrangement (melody, harmony, and rhythm). The goal was not to train the algorithm to create specific parts like a harmony to a melody or similar. Therefore, this thesis does not adopt quantitative research methods such as cross-validation like Wallace did. The only quantitative method that was used in this project is a statistical evaluation of the results from the Turing test. Another research method that was considered is a qualitative interview method. This method is very common in the field of musicology because, as stated before, music can be very abstract and hard to evaluate. There is no right or wrong in music because of the aesthetics of art. However, there are tools to evaluate it, and qualitative research is a popular choice. Yet this researcher chose a Turing test instead, mainly because it can use statistics to evaluate the outcome. Also, the Turing test was created with the purpose of evaluating computer intelligence, which is a part of this thesis since creating music can be defined as an intelligent task.

3.1.1 Iterative Process

The iterative process is a common research method in informatics, mostly used in the design of a computer application. In this thesis, the process was used to constantly check the trained

[73] Nuanáin, C. r. O. (2018). *Connecting Time and Timbre: Computational Methods for Generative Rhythmic Loops in Symbolic and Signal Domains* [Phd Thesis, Universitat Pompeu Fabra].
[74] Wallace, B. (2018). *Predictive songwriting with concatenative accompaniment* University of Oslo]. University of Oslo.

results of the sample data in the research. This was done by trying out different music, different parts of the songs, changing the data size, and changing the different instruments.

The fundamental concept behind the iterative process is to develop a software system gradually, step by step, to make it possible for the developer to take advantage of information from earlier development steps. The developer can learn from both the development and use of the system. In the iterative process, it is important to start with a basic implementation of the software requirements and gradually enhance the software until it reaches the whole system is implemented. The developer makes a design modification at each iteration including adding a new functional capability, as illustrated by Figure 9 (Larman & Basili, 2003, p. 5)[75].

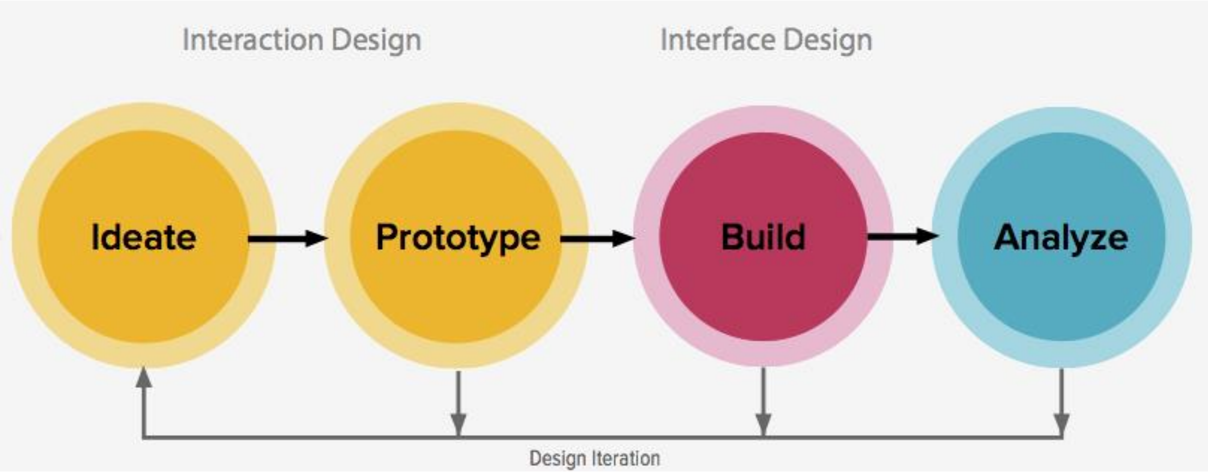


Figure 9. Showing the design iteration in an iterative process[76].

One of the benefits of using an iterative process is that it gives more control to the researcher to evaluate how the ML model is working with the data. Instead of just transcribing all of the music until all songs are finished, the songs were constantly tested by training it with BachProp and changes were made to the transcription method until the result was satisfying. This allowed the researcher to test different strategies of transcribing and presenting MIDI data to a neural network, a crucial aspect of creating music with ML when using a small data size. For example, what would happen to the results if it changed to using songs with only two instruments, instead of 5 instruments? Since it is not known what the best strategy is to use when creating music with ML, this iterative process made it possible to find the best-suited method for this thesis. If the quality of the music is not high enough to pass the qualitative observation test, then it will

[75] Larman, C., & Basili, V. (2003). Iterative and incremental developments. a brief history. *Computer*, *36*, 47-56.
[76] https://candidosalesg.files.wordpress.com/2013/02/zurb-process-design-2.png

start the evaluating process to find out why the training has failed and then make the necessary changes to the data.

Another method that is common in the field of informatics is the waterfall method. While the iterative process is constantly checking and repeating steps, the waterfall method focuses on completing each step before moving on to the next (Larman & Basili, 2003, p. 3)[77]. In this case, that would mean that all songs are transcribed into MIDI and modified, then moved on to the next stage before making any more changes to the training data. While the waterfall method can be useful in other scenarios, in this thesis, the iterative model was superior because it was unknown exactly how the music should be presented to the DL model to make it learn enough, so an iterative process was required to figure that out. A common problem with the waterfall method is that if a bug or error in the early stage is undiscovered, it could be very time-consuming or too late to fix it. In the iterative model, all the steps are constantly tested, which makes it easier to both discover and fix bugs (Kruchten, 2014, p. 4)[78].

[77] Larman, C., & Basili, V. (2003). Iterative and incremental developments. a brief history. *Computer*, *36*, 47-56.
[78] Kruchten, P. (2014). From Waterfall to Iterative Development-A tough transition for project managers. *The Rational Edge.*

The flowchart in Figure 10 shows how the thesis is divided into two parts, where one part is the iterative research and the other is evaluation and conclusion. It also displays how the qualitative observation was used as a failsafe to ensure the quality of the music before initiating a Turing test.

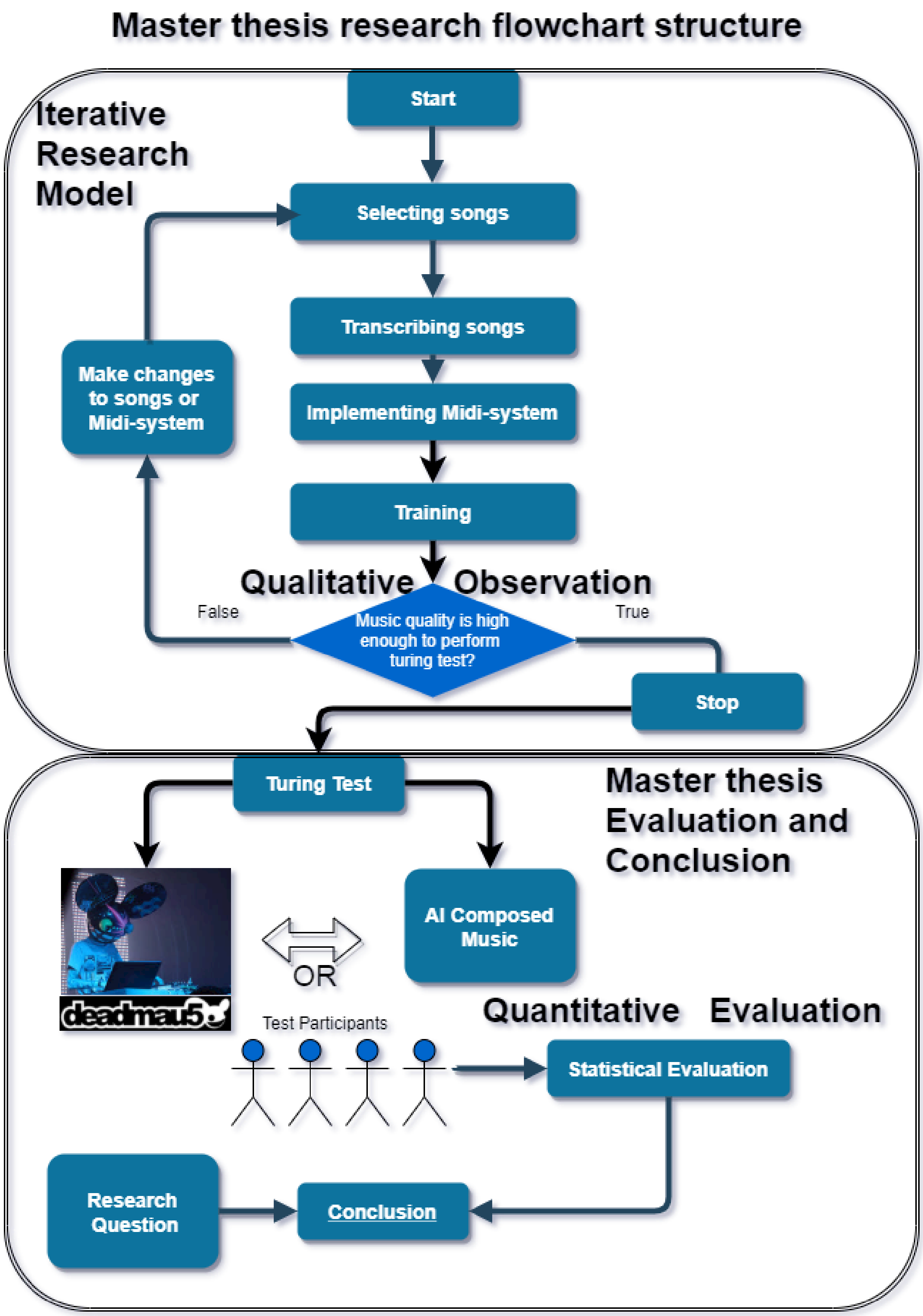


Figure 10. Flowchart showing how the iterative process works in the first stage of the thesis.

This flowchart is a simplification of the process since additional evaluation methods such as the Avicii test and harmonic and rhythmical analysis were also used. However, the Turing test is adopted as the main evaluation method since the other tests do not evaluate the aesthetic quality of the music, only that the model is learning, and that the MIDI system is required. Also, a pilot Turing test was also arranged as part of the iterative method since the feedback was used in the design of the ADM.

3.1.2 The Turing Test

In 1950 Alan Turing published an article in the philosophical journal *MIND* presenting the Turing test (Turing, 1950, p. 433)[79]. The purpose of the test was to check if a computer was able to think by making participants chat with a person and a computer without knowing which was which. The concept was a blind test that went on for up to 5 minutes, and if the participant was unable to distinguish between computer and human, the Turing test was passed and *"There`d be no reason to deny that a computer could really think"* (Boden, 2018, p. 107).[80] The Turing test is illustrated by Figure 11.

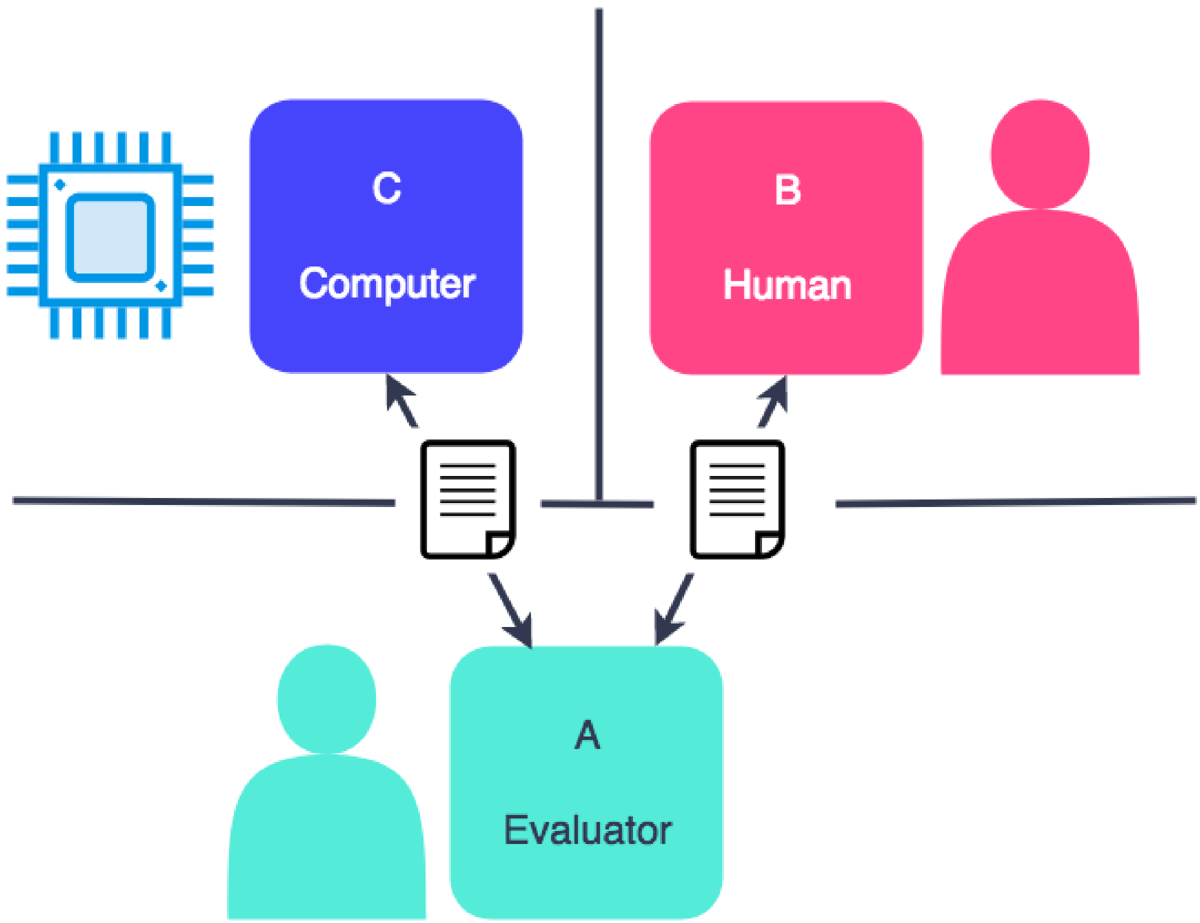


Figure 11. Illustrating the process of distinguishing computer from human in the Turing test[81].

Even though the Turing test is relevant in the philosophical field of AI, most researchers today are more focused on using AI to create useful tools (Boden, 2018, p. 108)[82]. An example of this

[79] Turing, A. M. (1950). Computing Machinery and Intelligence. *MIND, New Series*, *59*(236), 433-460.
[80] Boden, M. A. (2018). *Artificial Intelligence A very short introduction*. Oxford University Press.
[81] https://botsociety.io/blog/wp-content/uploads/2018/03/TuringTestScheme-3.png
[82] Boden, M. A. (2018). *Artificial Intelligence A very short introduction*. Oxford University Press.

is AI applications like customer segmentation in marketing that uses unsupervised ML to divide different customers with something in common into a group using unstructured data (Tripathi et al., 2018, p. 803)[83]. This is done without having the essential methods to pass a Turing test because it is an AI tool designed to solve a specific marketing task and without the need for communicating, it is unable to pass a Turing test.

The most common critique of the test is that it did not prove intelligence but only that a computer could achieve or observe behavior like a human, and that this proves nothing about consciousness (Boden, 2018, p. 107)[84]. However, this critique did inspire a debate about what consciousness is and how we define intelligence, which is beyond the scope of this.

The initial Turing test used conversations with text as data, which is perfect for ML because text is easily translated and represented as binary. This leads on to the core challenges in this thesis, which is how to convert music into binary. The other key challenge is that the ML model that is creating music in my thesis is not interactive, but even if it was, music is not a communication language like ordinary language. Sure, there are some fundamentals in music improvisation, but the whole point is to explore the possibilities of using AI to create popular music and not to create an AI jazz improvisation tool. In order to do the former, a Turing test is needed to evaluate the music that the AI has created. The approach to the Turing test is:

- 20 samples of music which lasts 7.5 seconds each. The samples are from the *drop* part of the song, which in EDM terminology is the chorus/main part.
- 10 samples are original samples that were used to train the model.
- 10 samples are created from the DL model.
- Since the Turing test was online, each participant could only hear each sample once.

The Turing test consists of 3 different tests: one beta closed test, one main test, and one experimental test. The results from the tests are used in the statistical evaluation in Chapter 5 and discussed in Chapter 6.

[83] Tripathi, S., Bhardwaj, A., & Eswaran, P. (2018). Approaches to Clustering in Customer Segmentation. *International Journal of Engineering & Technology*, *7*, 802-807.
[84] Boden, M. A. (2018). *Artificial Intelligence A very short introduction*. Oxford University Press.

The Test Design

The samples: The 4-bar, 7.5 seconds length was chosen because of the genre and the limits of the ADM system, which are explained in Chapter 4. Electronic house music is often very minimalistic and repetitive, and it can be claimed that there is no need for more than 7 seconds to get an idea of the song depending on the song and what part the sample is from. As stated earlier, it was never the intention to use AI to create finished products, and in this context, the goal is to explore the abilities of using AI to create small parts of a song, like the intro, main theme, or the drop. The reason for this is that focusing only on one specific small part of a song, like the drop/chorus part, limits the task for the DL algorithm. If a model is trained on full length songs, then it is basically trained to figure out music forms; for example, 10 bars intro then 8 bars buildup followed with 16 bars drop with transitions. This would take the focus away from learning more necessary patterns like the harmonic and rhythmical patterns. The hypothesis here is that limiting the AI to only focus on one specific part of the song will enhance its ability to learn patterns correctly. However, this raises a new challenge, because this will also mean that only the drop part of the songs is used in the training data and that will decrease the training dataset a lot, from using 4-5 minutes per song to only 10-40 seconds. This can be justified by using songs that are very similar in the context of harmonic and rhythmical patterns, as will be explained more in Chapter 4. The reason for using only 20 samples is because the hypothesis is that it is the smallest amount necessary to gain sufficient results for the Turing test. The smallest amount was needed because increasing the number of samples would significantly increase the amount of research time spent on transcribing and training the model. The small amount of samples and short length also allowed for the test to be as short as possible, which is a positive feature since long online surveys can deter participants or prevent them from finishing. In this case, fewer samples and their short length resulted in a low dropout percentage.

The two tests: Using a Turing test can raise a lot of challenges including insufficient number of participants, spurious responses, website errors, dropouts, or something unexpected. Therefore, a pilot/beta test was arranged that was only sent to a selected group of people, which returned feedback about the test. This made it possible for a small evaluation and some changes before the next test. Test two was the main test, and this test was run for a longer period and shared with more participants. It was challenging to set a goal for the number of participants, but based on the first test, the goal was 100 participants for the statistical evaluation to be considered significant. The last test was called an experimental test and was sent out on a variety of different platforms such as Reddit, Twitter, and a range of forums. Since this test is an experimental test, the test result will not be used in this thesis and the test will start after the work is finished. This also gives the opportunity to continue the work and add performance feedback for the participants. The goal of this last test is to gain some attention and to make it go viral. However, the danger with gaining popularity is also that it can attract people with bad intentions, like hackers or trolls that could discover a flaw in the website coding and decide to take advantage of it to ruin the results. Since the first and second test were only shared with people from the researcher's personal social network and the University of Oslo, it is much more likely that these types of threats will only happen in the last test, which is why it is experimental only. The steps that were taken to avoid threats was focusing on making a secure web survey framework and monitoring the IP addresses from the test. The IP monitoring was only to see countries and to check for suspicious patterns, like a bot network or DDOS. An advantage of having multiple tests is the ability to cross-check the results from all the tests to see if there is a similarity or if there is a significant difference. The duration for the tests was: three days for the pilot days, seven days for the main test, and infinite for the experimental test.

The two questions: There were only two questions in the Turing test which were not samples: *"Are you a musician?"* and *"Are you familiar with Deadmau5's music?"* The main reason there were only two questions is to make the evaluation task simpler and to get straight to the point. Having no questions at all was considered, but the decision was to have just two since it still gives the ability to not use them in the evaluation. More questions could make the test take longer for the participants and there are also legal issues when collecting information about people. The goal was to test if people can distinguish AI created music from human created music, so there was no need for more questions to achieve that. The question *"Are you a musician?"* was added out of curiosity to check if there are any significant differences between musician and non-musician participants' test results. Also, there is a personal reflection that a lot of musicians are very passionate about AI creating music, which will be elaborated on in the reflection section 6.1. The other question *"Are you familiar with Deadmau5 music?"* was crucial, because if a participant is familiar with the songs that were used as a sample, then they would know it is not created by an AI. Due to the COVID-19 pandemic, an online test was even more necessary, although it was always planned to be conducted online as an online test can gain more participants than a physical, offline test. Also, electronic house music and AI is a very international topic and not specific to Norway or Norwegians, so it made sense to publish the test online to get participants from around the world. This could also help gain attention to the research.

3.1.3 Statistical Evaluation

The goal of the evaluation is to obtain statistically significant results where the result has confidence and cannot be due to chance. To establish confidence, the number of participants needs to be at a minimum threshold. There are different methods to determine the number of participants for probability-based surveys and polling surveys, such as the sample size calculator of the Australian Bureau of Statistics[85]. In this context, the population are consumers of EDM, which makes both the population size and the sample size large. A smaller sample size means that the statistical significance would not carry much weight and that the margin would be error is larger (Field, 2013, p. 42)[86]. However, to limit the thesis, the statistical evaluation has a threshold of 100 participants, which is smaller than the recommended sample size, but not too small to get a statistically significant result. Also, the statistical evaluation is not the only evaluation method used in the conclusion of the research question; the qualitative observation method and the Avicii test in Section 4.4.1 are also used in combination with the statistics in the conclusion.

The Turing test is a discrimination test, where the participants will listen to one sample at a time in a randomly selected order and must discriminate between Human or AI for each. The participants can only hear the sample once and will be evaluated after in terms of how many correct discriminations they get for each answer. The goal of the statistical evaluation is to check how the points are distributed and how many participants will pass the test to get a statistically significant result to reject or support the hypothesis. High test scores mean that the ADM has failed the test while low test scores mean the opposite. Measuring of the test scores is done using percentage of correct answer as a value. In this evaluation, there is not a fixed threshold to pass the test, but if the average score of participants is over 10% higher than 50%, then it means that the ADM project is performing poorly. Since each question has a 50% chance of guessing right, the percentage of guessing is 50%.

---

[85] statistics, A. B. o. (2021). *Sample Size Calculator*. Retrieved April 10, 2020 from https://www.abs.gov.au/websitedbs/d3310114.nsf/home/sample+size+calculator

[86] Field, A. (2013). *Discovering statistics using IBM SPSS statistics: and sex and drugs and rock 'n' roll, 4th Edition*. London.

Good test results: < 60%

Bad test results: > 60%

The statistics are displayed with a basic bar chart using the percentage of successful discrimination on the Y-axis to measure performance, as shown in Section 5.1. Standard Error was used as error bars in the measurement of participant groups while standard deviation was used for the measurement of each sample performance. The reason for this is that standard deviation is used to illustrate how the data is distributed on one sample, while the standard error is used to show how the data is distributed across all of the samples (Field, 2013, p. 54). Since the standard deviation is used to measure how the data is spread, a low standard deviation means that the data is close to the means and statistically significant, while a high deviation means that the results are an expected variation (Field, 2013, p. 54).

3.1.4 Qualitative Observation Test

The qualitative observation test is the evaluation process that will ensure that the quality of the AI-created music is high enough for the Turing test. The test is the last step in the iterative process and if the test is passed, the thesis proceeds to the next stage, which is the evaluation and Turing test. In this context, the quality of the music means that the music created by the AI sounds like the original music, without plagiarizing. Also, it is to make sure that there is enough AI created music and that the music has variety. It is called a qualitative observation test because it is based on the researcher's own subjective opinion while listening to the music according to a list of qualities.

The 5 steps of the qualitative observation test are:

- *Does it sound like the original music?*
  If yes: move on to the next step.

- *Does it contain any glitches that will expose that is not created by a human?*
  If no: move on to the next step.

- *Is the loop long enough?*
  If yes: move on to the next question.

- *Is it overfitting?*
  If no: move on to the next step, but if in doubt, do an overfitting analysis before moving on.

- *Are there enough samples to proceed to a Turing test? Minimum 20*
  If yes: then the iterative model is finished and move on to the next stage of the thesis.

The first question is basic because the music must sound like the original music to defend the hypothesis of the thesis, which is that AI can be used to create EDM. The second question about glitches is a common concern when using DL to create music. Glitches, in this context, are defined as random MIDI notes that have no harmonic or rhythmic structure. For example, if the MIDI notes are playing a simple c major chord over 4 bars, and suddenly a rapidly unrhythmical fast f sharp note in the highest octave possible is played once, there is no way of describing this as anything other than a glitch. If an AI-generated sample consists of 98% notes playing correctly, and 2% glitches, then it would be a success, especially if the MIDI sounds good. The problem is that a participant in the Turing test would easily distinguish it as created by AI, because no human would ever create such glitches on purpose. This provided a handicap when searching for potential samples in the generated AI music experiment. To deal with this situation, a rule was made that a glitch could be removed if it was a very small part of the notes, like 5%. Fortunately, this was rarely needed, and the samples used in the research are 95-98% unfiltered. The small sample length of 4 bars made this easier, while still relevant for the hypothesis, which is to only create small ideas and not full-length songs. The third question about length is also basic; since the loops in the Turing test have a duration of 7.5 seconds, all the samples created by AI should have the same length. The fourth question about overfitting is to make sure that the AI is not just copying the music it is training on. The overfitting analysis is not complex; it is just a simple project containing all the training data + the suspected sample, where fast comparison was done by using visual, harmonic, and listening comparison. Because of the way the MIDIs are structured, this provided a visual clear overhead. The fifth and last question was to secure enough samples before the thesis proceed to the Turing test. Since the requirement is 10, the goal was set on 20 so that there were 10 as a backup.

4 Implementation

This chapter will demonstrate how the music was implemented into an ML model and how the model used was trained on the music to create original music. Both the system and the specific methods that were used to modify the MIDI are explained. The training log describes the iterative process in detail while the samples that are included in the appendix show the end result. The chapter ends with a description and analysis of the results and a test to verify the system's integrity.

4.2 Dataset

As mentioned in the EDM section, Deadmau5 was chosen because of all the instrumental house songs in his discography. Using just one artist was preferred because the hypothesis is that Deadmau5 has his own pattern of making music, like the way he uses polymeters, as mentioned in Section 2.1.1 It is common for a music artist to follow their own recipe when creating music, and the whole purpose of ML/clustering is to find a pattern in unstructured data. If different artists were mixed in the dataset, it could be more challenging to find a pattern, even though it would provide more data. Further, a larger dataset is not necessarily better; it is the quality of the data and the way it is represented that is most crucial. Other artists that were considered included Daft Punk and Avicii because their music has similar qualities to Deadmau5's music, such as repetitive rhythms and instrumental songs, which are a good fit for ML. Also, Deadmau5 has produced a masterclass and livestream where he talks about his music and the process of making it. This is useful information in the evaluation of the music and how to modify the dataset, which made Deadmau5 the superior choice.

The list illustrated by Figure 12 consists of 24 songs, which were carefully selected and then transcribed by ear with MIDI. Fortuitously, the researcher found MIDI arrangements for some of the songs, which saved time, but since the music is very repetitive, the transcribing part was not too time-consuming.

Artificial Dance Music

TITLE	ARTIST	ALBUM
4ware	deadmau5	W:/2016ALBUM/
Arguru	deadmau5	Random Album Title
Aural Psynapse	deadmau5	5 years of mau5
Avaritia	deadmau5	while(1<2)
Brazil - 2nd Edit	deadmau5	5 years of mau5
Fn Pig	deadmau5	> album title goes here <
Moar Ghosts 'n' Stuff	deadmau5	For Lack Of A Better N...
Imaginary Friends	deadmau5	W:/2016ALBUM/
Let Go	deadmau5, Grabbitz	W:/2016ALBUM/
Maths	deadmau5	> album title goes here <
Monophobia	deadmau5, Rob Swire	mau5ville: Level 1
Not Exactly	deadmau5	5 years of mau5
Phantoms Can't Hang	deadmau5	Phantoms Can't Hang
Polaris	deadmau5	We Are Friends, Vol. 6
SATRN	deadmau5	SATRN
Slip	deadmau5	Random Album Title
So There I Was	deadmau5	Random Album Title
Some Chords	deadmau5	5 years of mau5
Strobe	deadmau5	For Lack Of A Better N...
There Might Be Coffee	deadmau5	> album title goes here <

Figure 12. Playlist of songs to be used in the training.

4.2.1 The MIDI System to Represent the Music

Each song was transcribed into one MIDI file, which consists of an arrangement of 3 parts:

1. **Lead Synthesizer.**

   MIDI octave range: **C6 – C8**

2. **Arp Synthesizer**

   MIDI octave range: **C2 – C5**

3. **Bass Synthesizer**

   MIDI octave range: **C0 – C1**

Each part is divided into different octaves, a method which allows the whole song to be represented in one MIDI file. The benefits of this is that it makes it easier to structure the training with one MIDI file per song instead of three, and it makes it possible to analyze all of the songs together on one MIDI track to get an overview of the structure. This was very helpful to locate errors in the arrangement of some songs. After the training was done, the MIDI files could be split into four instruments or into two, rhythm and harmony. This makes for quick analysis of the output for the self-evaluation test.

The MIDI was divided into three parts because the training data must be as simple and repetitive as possible, to simplify and limit the data that the DL model is supposed to train from. In the BachProp project, they used between 135 to 1,035 songs for each training (Colombo & Gerstner, 2018, p. 5)[87]. How is it possible to achieve the same with only 24 songs? The answer could be that the music is so simplified and repetitive that it makes it possible to see a pattern. For example, the drum pattern in all of the songs is almost the same rhythm and the core of the drums is three instruments: Kick, snare/clap, and hi-hat. The approach here is to remove the drum section completely from the MIDI dataset. Deadmau5 stated himself in the masterclass

[87] Colombo, F., & Gerstner, W. (2018). BachProp: Learning to Compose Music in Multiple Styles.

that he does not focus on it and that he would have liked to make music without it, but the genre forces him to use it (Deadmau5, Masterclass.com)[88]. Another reason is that the model struggles with the rhythmical pattern in the training process. If the rhythmical pattern for the drums is mostly the same, then it feels it is unnecessary to train the model to learn it. The drum part does not need to be original; on the contrary, it is the only part that needs to be a copy of the original. Only the harmonic part of the music should be original but sound like Deadmau5, and it seems like an unnecessary task to train a model to both copy and not copy the same training data. Finally, the MIDI grid is based on time, so it is easy to fit Deadmau5's signature drums into the MIDI files that is created by the model. Thus, if the music is very complex and diverse, a lot of songs are needed for the DL model to find a pattern, but if the music is very minimalistic and repetitive, then a small collection of songs could be enough if it is done in a particular way, as in the method presented in this chapter.

[88] Deadmau5. *Developing Melodic Structures*, Masterclass.com. https://www.masterclass.com/classes/deadmau5-teaches-electronic-music-production

4.2.2 MIDI Transcribing System

Figure 13 shows how the MIDI transcribing looks in Ableton. This visualization gives an overview of the dataset by making it easy to spot an error in the MIDI. This demonstrates the benefit of arranging everything into one MIDI file, divided by octaves. The transcribing part was mainly done by ear, but some MIDI arrangements were found online. In every song, the parts were carefully selected using the MIDI system. To remain objective, every song that raised a challenge or doubt when transcribing with the system was skipped to avoid creating an arrangement based on a subjective opinion. This was done to ensure that the integrity of the research is as objective and technical as possible.

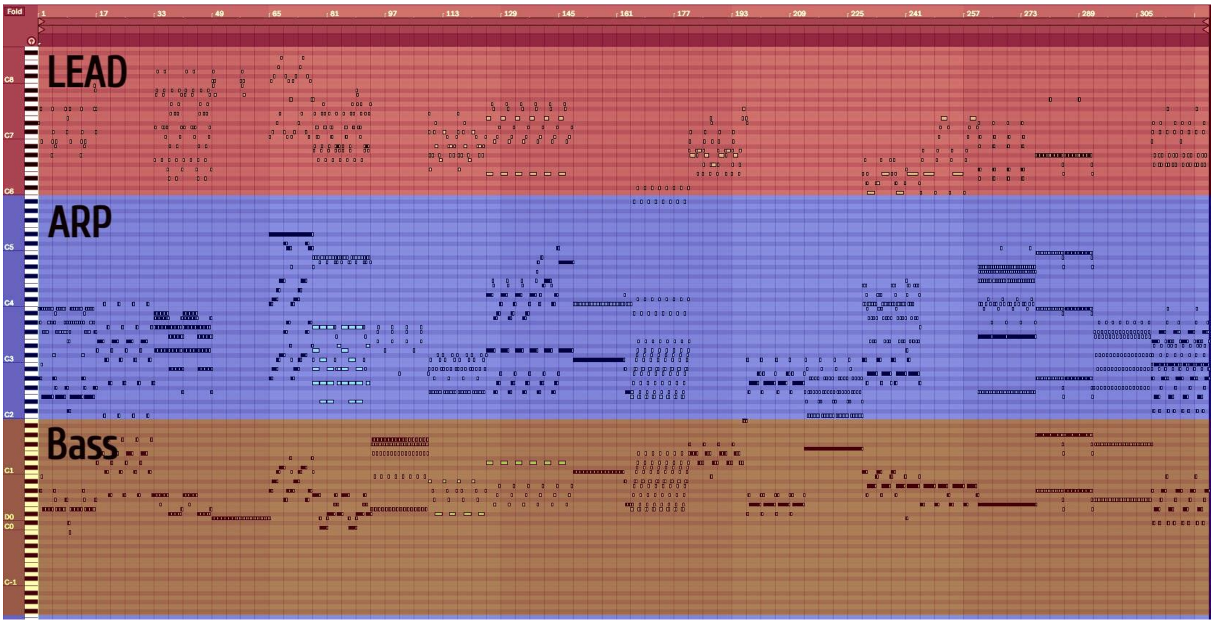


Figure 13. Visualization of all the MIDI used in the training in Ableton.

4.3 Training

GPU vs CPU vs TPU

Training with a DL model can be very time-consuming, and the choice of hardware can make a significant difference in how long it takes to run a model. A CPU is a central processing unit, which exists in every computer to execute every instruction by a computer program. The problem with neural networks is that the CPU is designed to execute the instructions one by one, which can lead to a bottleneck compared to a GPU, which can execute parallel calculations when training a neural network (Naushad 2020)[89]. The GPU is a graphics processing unit, which was originally designed to compute graphics but has become very popular in the use of neural networks because of the abilities mentioned above. A TPU is a Tensor Processing Unit, which was specifically designed to execute parallel matrix multiplication in neural networks. This makes it superior to both CPU and GPU, but it is still much more expensive (Naushad, 2020).

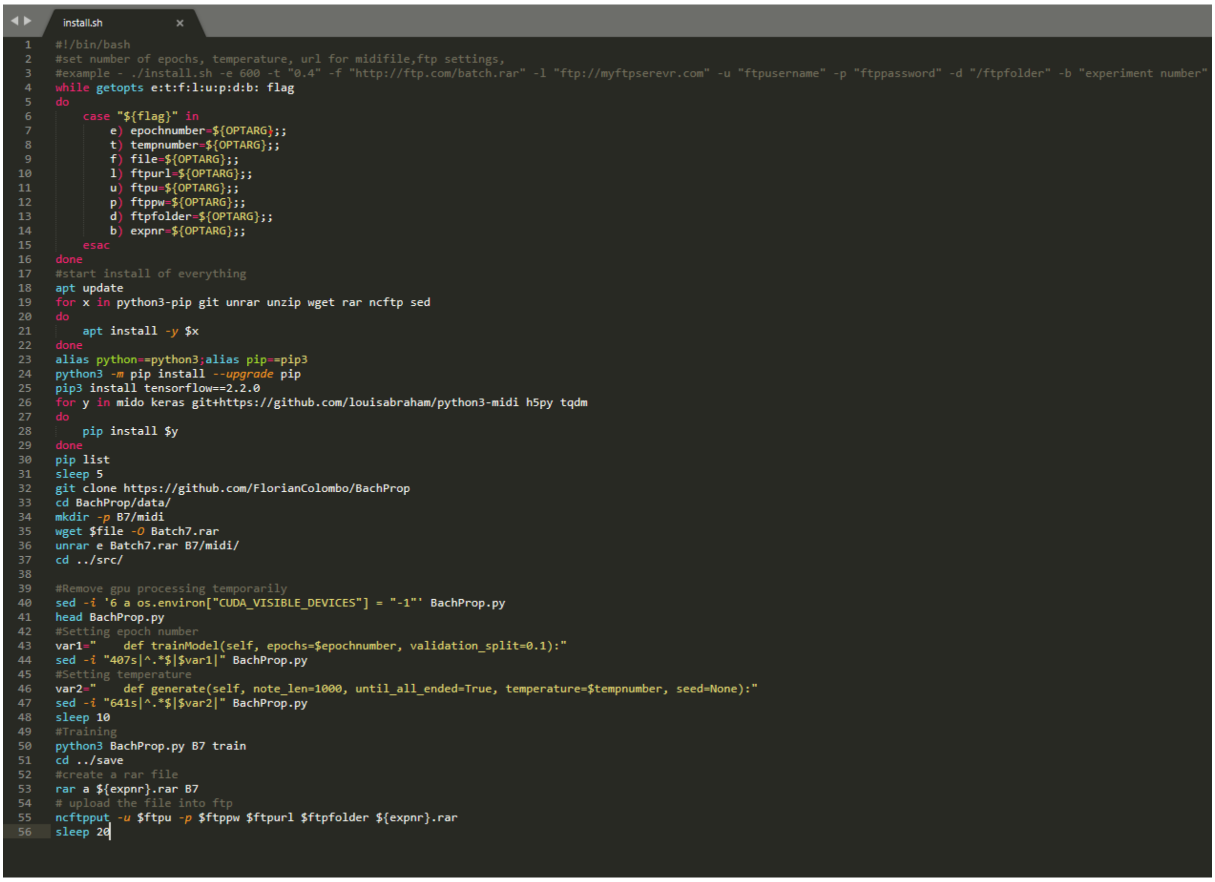
It is very popular among musicians to buy an expensive MacBook pro, which has a good CPU, but the GPU is not DL-friendly at all. Since it can be expensive to buy a GPU, the only other solution is to rent one online. This thesis uses VASt.ai, which has a cloud GPU rental system for an inexpensive price per hour. Although the data size is small, the training time on the researcher's MacBook is still around 4-7 hours; hence, a cloud GPU rental system is the best alternative for this thesis. The biggest advantage is this allows the possibility to investigate every MIDI modifying approach without spending weeks training. For example, what would happen if the model is trained using only a lot of copies of two songs? Or what if all of the rhythmical elements are removed? Or how will the harmony be impacted if only songs in major keys were used? Adjustments like this will likely impact the output of the training, and to use the rental time for the GPU cloud most efficiently, a list was prepared with all kinds of potential modifications to the MIDI.

The best way to describe the training system and the iterative process in the training is through a training log, which consists of highlights of the training and self-evaluation process.

[89] Naushad, R. (2020). *CPU, GPU and TPU — Machine Learning*. https://medium.com/dataseries/cpu-gpu-and-tpu-machine-learning-5033e25b8a0f

4.3.1 Implementation Script

After countless hours were spent to make the model work on a Linux environment, an installation script was created that allows the implementing of the model to perform faster every time it is training on a new cloud host, as displayed in Figure 14. The script is available on GitHub at https://github.com/lrs11/adm/blob/main/install.sh



```bash
#!/bin/bash
#set number of epochs, temperature, url for midifile,ftp settings,
#example - ./install.sh -e 600 -t "0.4" -f "http://ftp.com/batch.rar" -l "ftp://myftpserevr.com" -u "ftpusername" -p "ftppassword" -d "/ftpfolder" -b "experiment number"
while getopts e:t:f:l:u:p:d:b: flag
do
    case "${flag}" in
        e) epochnumber=${OPTARG};;
        t) tempnumber=${OPTARG};;
        f) file=${OPTARG};;
        l) ftpurl=${OPTARG};;
        u) ftpu=${OPTARG};;
        p) ftppw=${OPTARG};;
        d) ftpfolder=${OPTARG};;
        b) expnr=${OPTARG};;
    esac
done
#start install of everything
apt update
for x in python3-pip git unrar unzip wget rar ncftp sed
do
    apt install -y $x
done
alias python==python3;alias pip==pip3
python3 -m pip install --upgrade pip
pip3 install tensorflow==2.2.0
for y in mido keras git+https://github.com/louisabraham/python3-midi h5py tqdm
do
    pip install $y
done
pip list
sleep 5
git clone https://github.com/FlorianColombo/BachProp
cd BachProp/data/
mkdir -p B7/midi
wget $file -O Batch7.rar
unrar e Batch7.rar B7/midi/
cd ../src/

#Remove gpu processing temporarily
sed -i '6 a os.environ["CUDA_VISIBLE_DEVICES"] = "-1"' BachProp.py
head BachProp.py
#Setting epoch number
var1="    def trainModel(self, epochs=$epochnumber, validation_split=0.1):"
sed -i "407s|^.*$|$var1|" BachProp.py
#Setting temperature
var2="    def generate(self, note_len=1000, until_all_ended=True, temperature=$tempnumber, seed=None):"
sed -i "641s|^.*$|$var2|" BachProp.py
sleep 10
#Training
python3 BachProp.py B7 train
cd ../save
#create a rar file
rar a ${expnr}.rar B7
# upload the file into ftp
ncftpput -u $ftpu -p $ftppw $ftpurl $ftpfolder ${expnr}.rar
sleep 20
```

Figure 14. The implementation bash script in sublime.

With this script, the implementation of the model and training is done automatically. The only requirement is to change the experiment number for each training and upload new MIDI data to an FTP server. To keep track of progress and changes, each training is called an experiment (exp) with a number. To start the training, contact is initiated with the cloud hosting server through SSH and the script is executed with flags to obtain the right settings.

Each flag is linked to a variable:

- -e = epoch number. Range "100-1000". Default is 500

  Sets the number of epochs to train for.

- -t = temperature. Range “0.1-1”. Default is 0.5
  Sets the temperature for the MIDI generation. As explained in Chapter 2, High temperature will make the model experiment, while Low temperature (under 0.5) will make the model generate music that is closer to the original and can lead to overfitting.

- -f = file
  The url file of the collection of MIDI data. In this case it is set to use a rar compressed file, but it can easily be changed to zip.

- -l = ftpurl
  Sets the url address of the ftp server to upload the generated MIDI files.

- -u = ftp username
  Username for the ftp host to upload files

- -p = ftp password
  Password for the ftp host to upload files

- -d = ftp folder
  Location of the folder to upload the files

- -b = Experiment number,
  which will be the name of the .rar file uploaded when the training is finished.

Example of install:

apt install -y curl

-curl https://raw.githubusercontent.com/lrs11/adm/main/install.sh -o start.sh

chmod +x start.sh

./start.sh -e 600 -t "0.4" -f "http://ftp.com/exp.rar" -l "ftp://myftpserevr.com" -u "ftpusername" -p "ftppassword" -d "/ftpfolder" -b "Experiment number"

Alternatively:

apt install -y curl; -curl https://raw.githubusercontent.com/lrs11/adm/main/install.sh -o start.sh; chmod +x start.sh;./start.sh -e “600” -t "0.4" -f "http://ftp.com/exp.rar" -l "ftp://myftpserevr.com" -u "ftpusername" -p "ftppassword" -d "/ftpfolder" -b "Experiment number"

The goal with the script was to make it faster to implement the model, to automate the process, and to create a structure of the training process. For the first three experiments, all of these commands had to be typed manually, which took a lot of time, especially the first time when there was a need to figure out what commands to enter. The reason flags were implemented for the ftp upload is that the training process can take up to 20+ hours, so instead of manually uploading the result when it is finished, in this case it was much easier to automate the process by making the script upload the files automatically when the training is done. This approach can also make it easier for other people to access the necessary tools to create music with AI by saving them substantial time in figuring out all of the python packages. The combination of this study’s installation script and the BachProp repository on GitHub will automatically create an environment on a Linux computer with everything ready. Another important feature of the script is that it modifies the temperature value and epoch size in the BachProp main script. So instead of scrolling in the main script to line 407 and 642 to edit the numbers, the script automates this process and changes the values to variables that easily change every time a new dataset is trained. This is an important function that will be elaborated on in Section 4.3.2. Since the script is a bash script that runs on Linux OS, it is very easy to edit and customize and can be used with other models. Another approach is to take all of the source code and compile it with the script into a piece of software, which would make the model significantly easier to use, especially if it is compatible with both OSX and Windows. Yet this is very time-consuming, and would require the rights for the source code, which this researcher does not have. Also, it removes the combability for the script to work even if there are changes or updates to the BachProp source code on GitHub. If there are any updates to the BachProp on GitHub, the script will still work since it implements the model by cloning from the GitHub address. This is true unless the changes are adjustments to parameters that are necessary for the code; for example, changing the code line by adding or removing a line before the script edits the epoch and temperature, as shown in Figure 15.

```
47  #Setting epoch number
48  var1="    def trainModel(self, epochs=$epochnumber, validation_split=0.1):"
49  sed -i "407s|^.*$|$var1|" BachProp.py
50  #Setting temperature
51  var2="    def generate(self, note_len=1000, until_all_ended=True, temperature=$tempnumber, seed=None):"
52  sed -i "641s|^.*$|$var2|" BachProp.py
53  sleep 10
```

Figure 15. Illustrating how the bash script is changing codes in the python script.

In line 49, the sed commands replaces the code line 407 in the BachProp script with a new line that has the updated *"epochnumber"* that is stored in a variable from the flag input by the user. But if changes are made in the *"BachProp.py"* source code by adding a line in the beginning, and it is uploaded to GitHub, the script will fail to run because it will replace the epoch number at the wrong line in the code, since the new correct line will be 408 and not 407. A possible solution to this could be to make the script search for the line where the epoch number is set, but this still will not work if the line itself is changed. So, there is no easy solution to this, but since the script is made for use by people with basic knowledge about Linux, GitHub, and AI, it will still be a helpful way of implementing a model. Further, since the training requires significant CPU or GPU processing, it is not possible to make web-based software that runs in a browser or an app that runs on Android/iOS.

Minimum Requirements for training with the model using the script:

- OS: Debian Linux
- Processor: Intel Pentium 4 2.0GHz / AMD Athlon XP 2000+
- System Memory: 4 GB RAM
- Storage: 5 Gb Hard drive space
- Cudnn: 10.1

4.3.2 Data + Epoch + Temperature = Music

Creating music with AI involves a lot of scripts and python packages, and DL models with a lot of features. But as part of the hypothesis, there are three features-Data, Epoch, and Temperature that are significant in the outcome of the training. While BachProp model does a lot of work in normalizing and transposing the MIDI, and setting up the network architecture, make it work with a small dataset, these are the only three features that need tweaking. In other words, finding the right combination of epoch size, temperature value, and data size, was a major part of the success factor for this project.

As stated before, the Epoch value is the number of times all of the data has run through the neural network once. In context, this means that increasing the epoch value forces the ML model to train longer on the dataset. However, this can lead to overfitting. So, to make sure the DL was not overfitting, it was necessary to monitor the validation data and stop the training before the validation of the data begins to degrade. Also, this process should be automated, like another DL model, RNN, as mentioned in 6.3. The temperature setting is only for the generation of the MIDI files and can be changed after the training is done. In this case, 0.3 was the most successful ratio for the last experiments, which is a very low temperature.

4.3.3 Overview of the Training Framework

Figure 16 is another flowchart that demonstrates the training framework in detail from listening to Deadmau5 to training with a DL model. The goal of this architecture is to make a fast-adaptive system that can be used with different MIDI data without changing the data. It also shows how multiple hosts are used to train the data simultaneously.

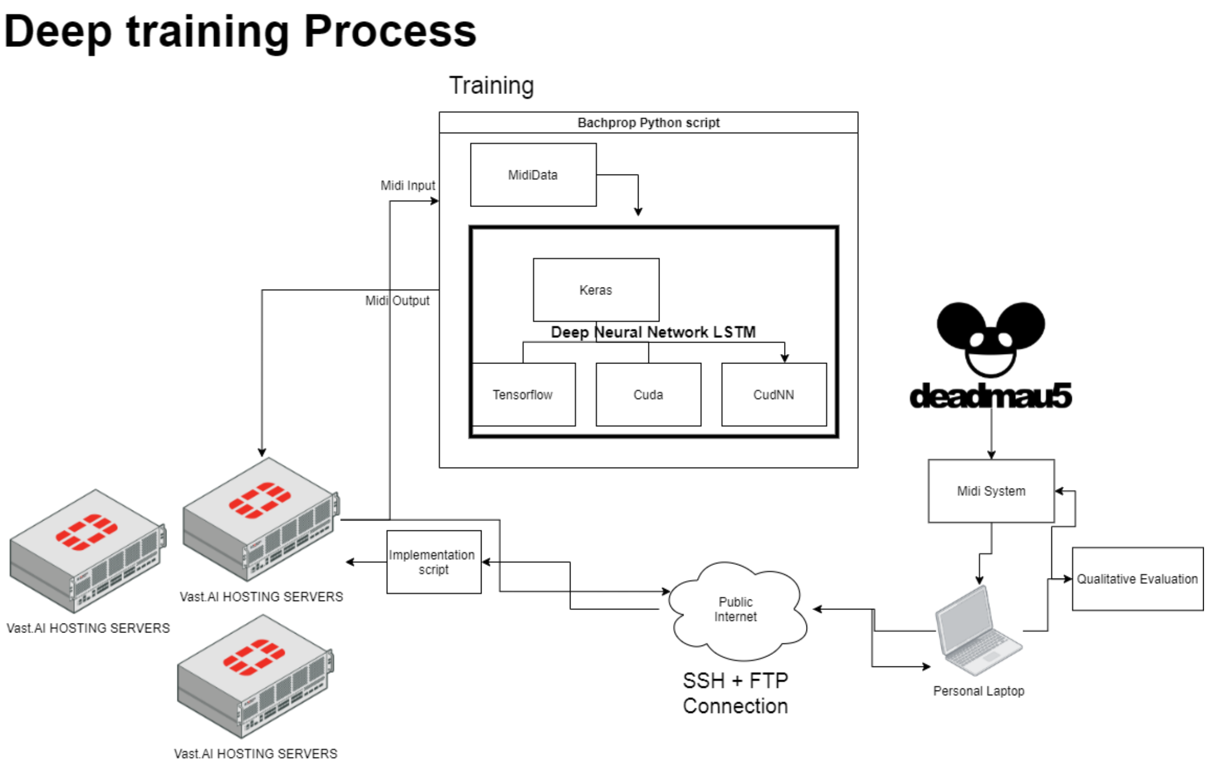


Figure 16. The training process.

4.3.4 Training Log

Table 1. Summary training Log:

Data	Changes	Notes	Music output	Evaluation
Exp 1 6 songs		The Bachprop script will crash if the number of MIDI files is under 10, and it took many hours to figure this out. Doubled the songs from 6 to 12 and it works.	Sounds nothing like the music it was trained on, but some rhythm and harmony parts are there. I did not expect to succeed on the first attempt anyway, so I guess it is a good start.	Not even close.
Exp 2 10 Songs	Changes from full length to chorus/drop part only		Progress stored 1 sample that actually sounded ok. Since it was the only one, it could just be a happy accident, but it is still made by AI.	Still not close.
Exp 3 12 Songs	Changes to MIDI system: changed the MIDI to arrangement from 5 parts to 4.	Removed the pad instrument, since a lot of songs do not have it.	The music is finally starting to sound like the original music. The rhythm part is a bit off, but it is an improvement from Exp 2. Made 4 samples from the Exp, but I think that I can achieve more. The new system clearly shows progress.	Not Passed, but stored some samples

Exp 4 15 songs		No changes expect more songs.	Sounded worse than the last Exp. A lot of noise and stutter.	Not passed.
Exp 5 18 songs	Changed MIDI system to 3 instruments.	Removed drum instrument from the MIDI files.	Some parts sound like the original, but the parts only last seconds.	Not passed but removing the drum part significantly improved the results.
Exp 6 20 songs		No changes except more songs.	Sounds like the last Exp.	Not passed.
Exp 7 22 songs	Epoch changed from 500 to 850.	Epoch.		Not passed.
Exp 8 24 songs	Epoch changed from 850 to 1000.	Epoch and more songs.	Diminished returns reached; the model is overfitting.	Not passed.
Exp 9 24 songs	Epoch 700 and temperature 0.6.	Epoch and temperature.	Increasing the temperature was not a good idea.	Not passed.
Exp 10 24 songs	Epoch 650 and temperature 0.4.	Epoch and temperature.	Good short ideas that sound like the original. Lacking chord progressions and melody.	Passed, samples stored.
Exp 11 24 songs	Epoch 650 and temperature 0.3.	Epoch and temperature.		Passed, more samples.

Table 1. Illustrating the iterative process in the training with a training log.

4.3.5 Observations From the Training

Design choices

The process of transcribing the songs was time-consuming, so the training process was performed simultaneously as the songs were transcribed. As a result of that, the first experiments with under 10 songs was heavily underfitting, which meant that the music was random noise. After the first experiment, the decision was made to limit the songs to only the drop part of the song. Even though the early stages of the model were underfitting, improvements could already be observed. After reaching 12 songs, the model started to perform better, but there was still a lot of noise in the drum instrument part. The decision to remove the drum instrument and the MIDI system with three instruments significantly improved the model's performance. The MIDI system started originally on five instruments, so three was the furthest possible limiting without losing the integrity of the research. Also, the hypothesis is that the new MIDI system made it possible for the model to perform with significant results on a small dataset. The small dataset is not only defined by the low number of songs, but also that the songs only contain the drop part of the songs, which means that each song is only 8 or 16 bars in length, approximately 30-50 seconds. However, the hypothesis is that the MIDI system and the music create an environment of patterns that are repetitive and simple, as illustrated by Figure 13. This makes the model successfully perform on a significantly small dataset. The difference in the performance after adding more songs was insignificant after reaching 20, so there is a possibility that the point of diminishing returns was met. This means that increasing the number of songs would not make the model perform better. In this research, the quality of data is more important than the quantity of data.

In this context, the performance of the model is defined as to what extent the system generates human-composed-like EDM.

The model started to overfit when the epoch was higher than 800. When trained on high epochs, the model produced music that could be just one note looping in a rhythmical pattern for 30 minutes. For the current dataset, the epoch at 650 was the setting that worked best. As stated, a low temperature like 0.3 increased the model's performance more than a higher temperature like the default at 0.5. This was a bit surprising, as overfitting was expected when such low temperatures are used.

4.4 Results and Generating the MIDI Files

After Experiment 11, the self-evaluation process was passed and the iterative process of this thesis was performed. The samples that were generated sounded like the original, but there was a lack of harmonization in the samples. The ARP rhythms and harmony were there, but a chord progression or melody that sustained over time was missing. Regardless, the quality of the samples generated was more than good enough for the research. In this context, good is defined by the similarities in the notes generated by AI and the original. The result was 20 samples that were ready to go, and the thesis proceeded to the next step, which is the Turing test. Before the Turing test could be initiated, 10 samples from the AI music and 10 from the original music needed to be selected and processed with the exact same sound design. The length of the samples were 4 bars and the tempo 128 BPM, which is approximal 7.5 seconds.

For the Turing test to make sense, it is crucial that the sound design is the same. In this approach of using AI to create music, the focus was only on the harmonic and rhythmical aspects and not the sound design. Therefore, to make the sound element objective, there was a need to process all the music with the same sound design. In detail, this means that the exact same kick, snare, and hat sample was used for all of the rhythmical samples; the same goes for the synth. The synthesizer sound that was chosen for this test was a Deadmau5 signature pluck synth that was created from a tutorial on YouTube (SadowickProductions, 2012)[90]. The reason for this is that there is no chord instrument that plays pads, only rhythmical ARP figures, bass rhythms, or lead melodies. Also, a benefit of this is that it enables the use of only one synth with enough polyphony to play all of the parts. This also makes sense since there is a different arrangement on both the original and AI samples. For example, some of the deadmau5 songs like *"Argugua"* only had a bass synth and drums, while others like Strobe had all three different instruments. Still, both could be played with only one synthesizer without losing any major musical aspects, because the synthesizer was polyphonic and can play bass, arp, and lead. To process all samples objectively, the same cutoff filter settings were used and there was no automation on the samples.

[90] SadowickProduction. (2012). *Make The Pluck From Deadmau5 - The Veldt With Sylenth1*. https://www.youtube.com/watch?v=lNhdpOToeaw

4.4.1 The Avicii Test

The Avicii test was created to challenge the results of the MIDI system presented in this chapter. The hypothesis is that the system is necessary to enable BachProp to successfully train on a small dataset of EDM music. To challenge that, experiments were done with unprocessed EDM MIDI files found online by the artist Avicii to see if BachProp could train on a small dataset without any processing. The MIDI files were collected from a free MIDI site at www.nonstop2k.com and the dataset consisted of 30 songs. After five hours of training with default settings, the result was noise and nothing close to the original music, as shown in Figure 17

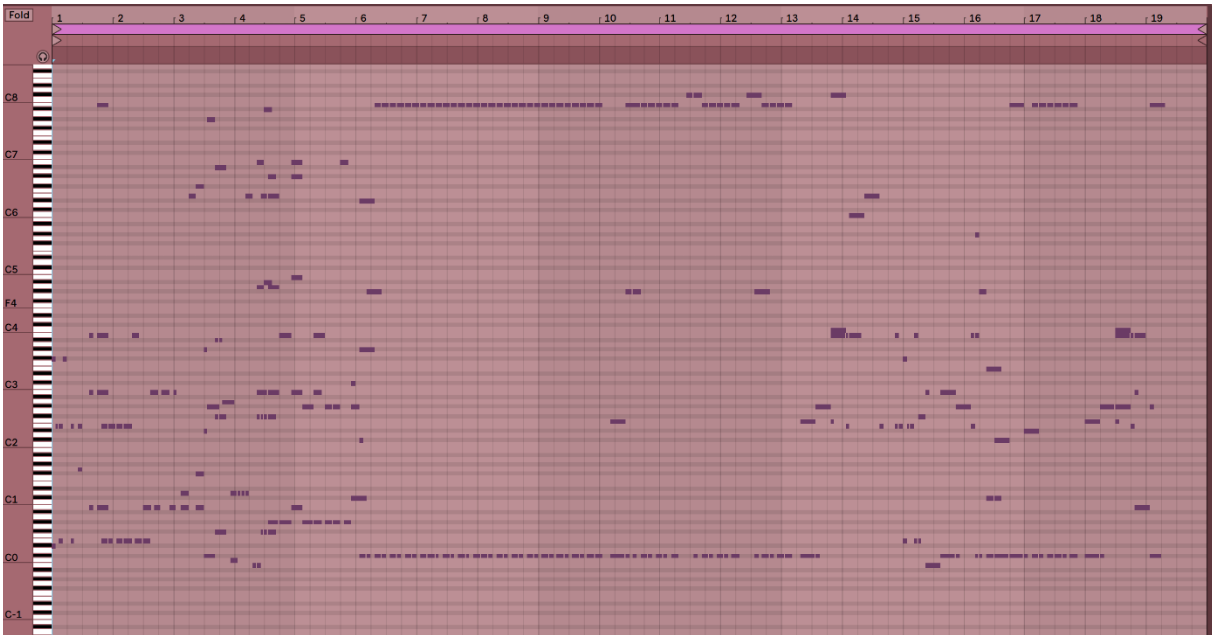


Figure 17. The noise generated by the Avicii test.

To investigate further, another experiment was performed with the same settings, but with a different artist, David Guettea. The dataset was collected from the same site and had the same size. Figure 18 shows that the training result was the same as the Avicii test. These two results do not prove anything, but they do support the hypothesis that the system created in this thesis works.

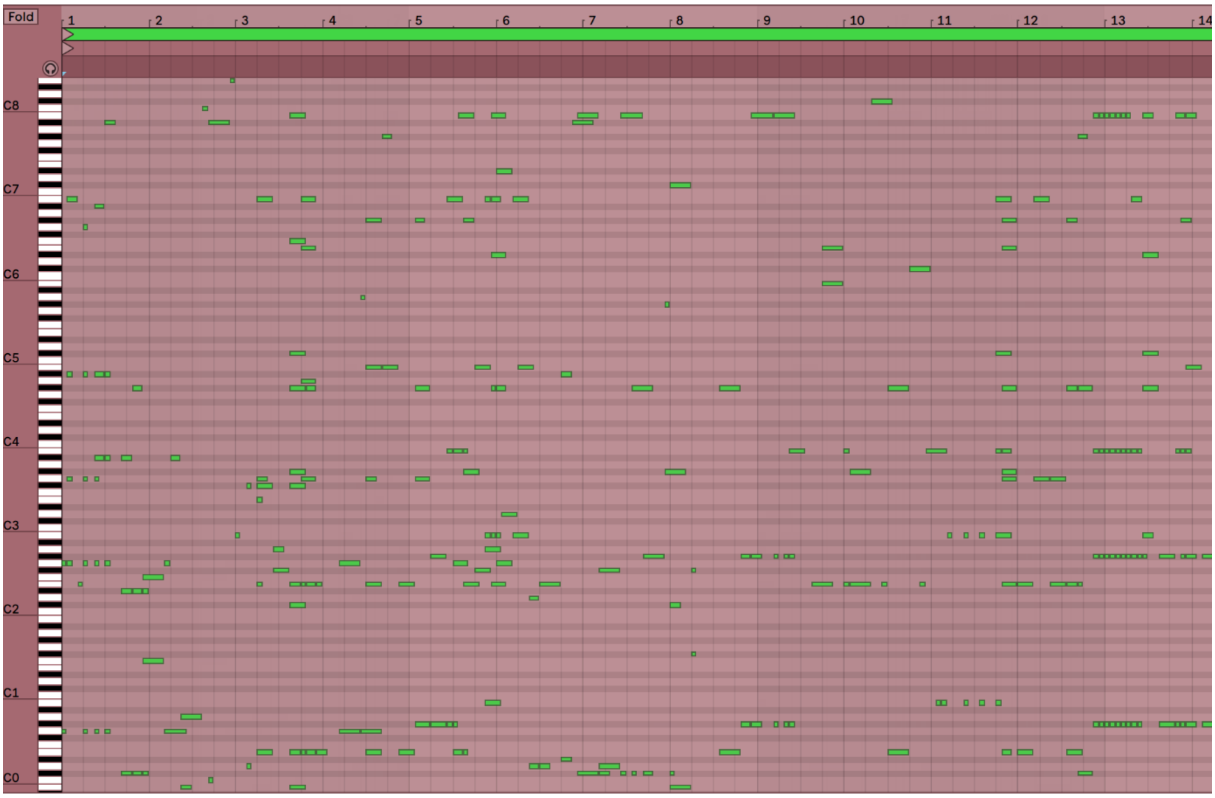


Figure 18. Another example illustrating the noise generated by the Avicii test with David Guetta as data.

4.5 Harmonic and Rhymical Analyzation

Section 2.1.1 *"The sound of Deadmau5"* demonstrated the different harmonic and rhymical methods that Deadmau5 uses in his music. One of the methods mentioned was the polymeter effect, which is two instruments playing at a different time and eventually simultaneously.

In the music generated by AI, several samples contained the polymeter effect, which is evidence that the AI is learning from the music. The first example is *"AI Sample 4,"* illustrated by Figure 19.

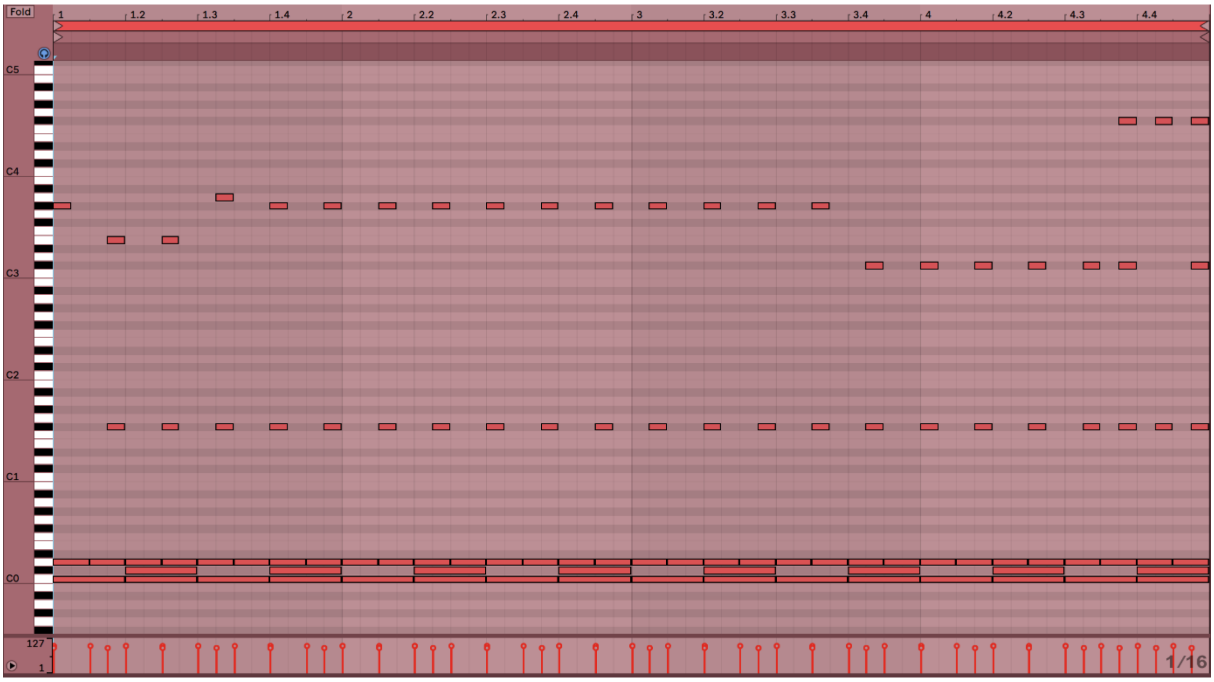


Figure 19. Evidence of polymeter effect sample 4. Octave C1-C4 is ARP, while octave c0 is drums.

In this example, the arpeggio rhythm in octave c1-c4 is playing in dotted eight notes and continuing with this rhythm until it plays in time with the drums on bar four. Another example of this is Sample 8, illustrated by Figure 20.

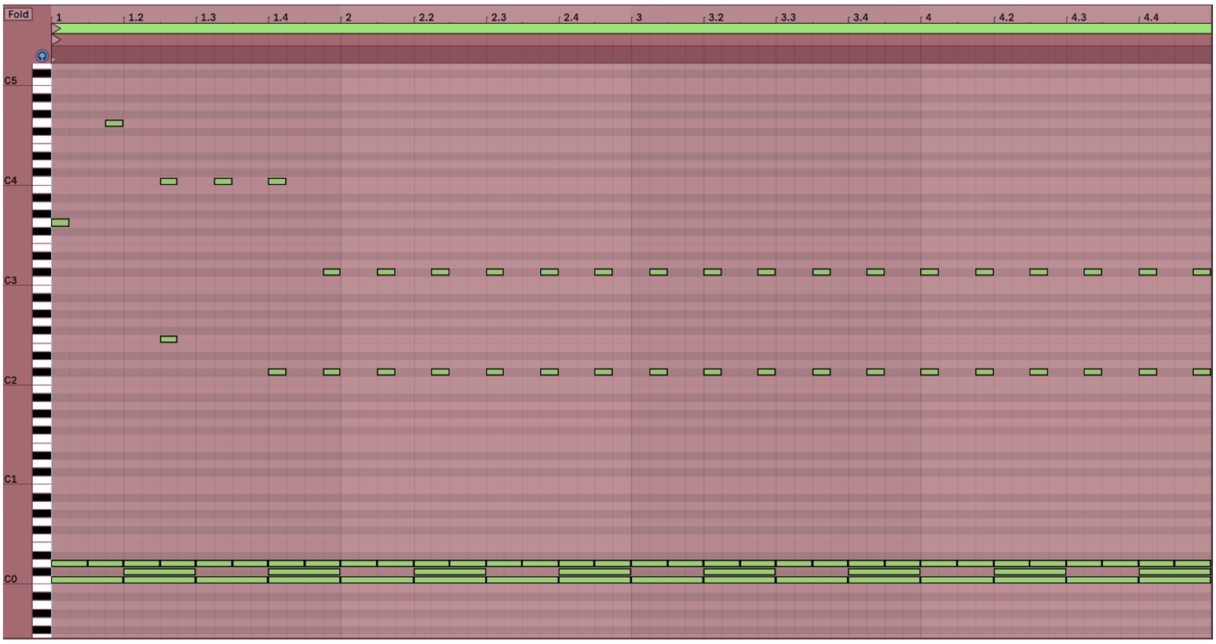


Figure 20, Evidence of polymeter effect in AI 8 sample. Octave C2-C6 is ARP, while octave c0 is drums.

These two examples prove that the AI is learning correctly from the data and support the hypothesis that AI can be used to train on a small dataset. Both examples were featured in the Turing test, and can be heard in Appendix A. There are also other examples of ARP methods being used, which can be heard in the Appendix samples 2 and 3. However, the AI-generated music contained no chord progressions and lead harmonies, which resulted in short rhythmical ideas over 4 bars as samples for the Turing test. The conclusion is that the model was successful in replicating the style of Deadmau5 to a significant extent.

5 Evaluation

This chapter presents the evaluation process of the Turing test results. It provides both a statistical evaluation of each test and a comparison of the different participant groups. The goal of the chapter is to measure the performance of ADM using the Turing test as a large-scale subjective analysis. The chapter ends with an evaluation of the potential weaknesses of the test.

5.1 Turing Test Framework

The Turing test was hosted on the website of the thesis (http://artificialdancemusic.com), as shown in Figure 21. Participants were recruited from both social media and email links. The total number of participants was 140, which was 40 higher than the goal of 100. http://QuestionPro.com was used as the survey software because of its inclusion of important functions, like the evaluation features and autoplay function on sound samples. The reason to use online survey software instead of coding a website from scratch was time efficiency. The survey site code from BachBot research was considered[91], but QuestionPro had better built-in functions.

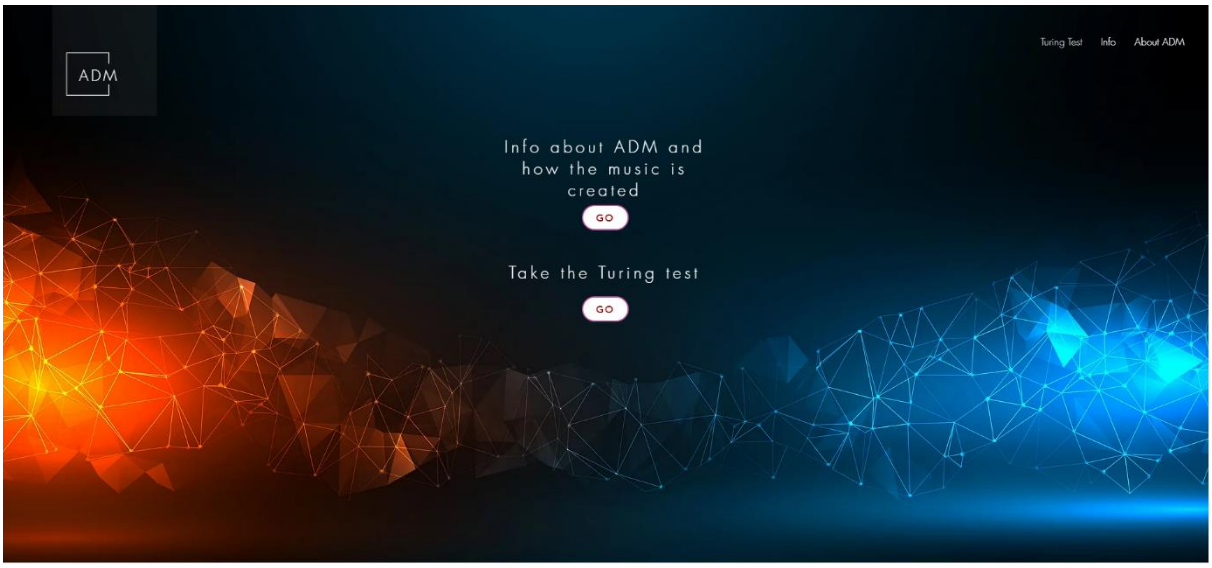


Figure 21. The Turing test webpage at http://artificialdancemusic.com.

[91] https://github.com/feynmanliang/subjective-evaluation-server

5.1.1 Turing Test 1—Small Group Beta-Test

The first Turing pilot test was to get feedback to check if the test worked, such as the absence of technical issues and making sure the statistical data is working. The test consisted of 20 samples; 10 AI and 10 Original. The AI samples was selected from the last training and are 98% AI with only a drums layer. The 10 original samples were randomly selected from the original samples that were used for training. The question order was also random, but the same for every participant, as illustrated in Table 2. The test had 20 participants and lasted for three days and revealed important feedback about technical issues in some browsers and the quality of the AI samples. The result was as expected: the participants who were not musicians could not distinguish the AI samples from the original samples to a significant extent, while the participants that stated they were musicians could distinguish close to a significant extent. The feedback from some of them was that they actively listened to music harmony, like the chord progression, and that made it possible to distinguish the AI samples from the original.

Sample	Original song
1. Sample H 1	Aural Psynapse
2. Sample AI 1	AI
3. Sample AI 2	AI
4. Sample AI 3	AI
5. Sample AI 4	AI
6. Sample H 2	Move For Me
7. Sample H 3	Let Go
8. Sample AI 5	AI
9. Sample H 4	SATRN
10. Sample H 5	Everything You Are
11. Sample AI 6	AI
12. Sample H 6	Not Exactly
13. Sample AI 7	AI
14. Sample H 7	4ware
15. Sample H 8	Phantoms can't hang
16. Sample H 9	Fn Pig
17. Sample AI 8	AI
18. Sample AI 9	AI
19. Sample AI 10	AI
20. Sample H 10	October

Table 2. The order of the samples and the original song in the Turing test.

5.1.2 Turing Test 2—Main Test

Since the model passed the first Turing test with the non-musician's participants but was close to failing with the musicians, feedback was used from the participants to continue with the training. After three experiments with different epoch and temperature values, along with loss numbers, new AI samples was created. The new samples had some more chord progressions and were added to the Turing test along with the original human samples and in the same order. In other words, in the next test, only two AI samples were replaced and everything else was the same. All of the samples can be heard in appendix A and the test is still online at www.artificialdancemusic.com.

5.1 Results

Figure 22 shows the distribution of participants in the Turing test. The response distribution shows that participants from a total of 11 different countries took the test, a somewhat surprising result. A lot of participants requested a score of their test results and this was considered but not added to the test. The reason for this is that a score could attract participants to take the test multiple times, which would affect the statistics. Also, the participants scored lower than expected, which made the idea of hiding their poor test scores a good option!

Figure 22. The response distribution of the test participants.

5.1.1 Statistical Evaluation

Figure 23 shows how well the ADM project performed in the Turing test.

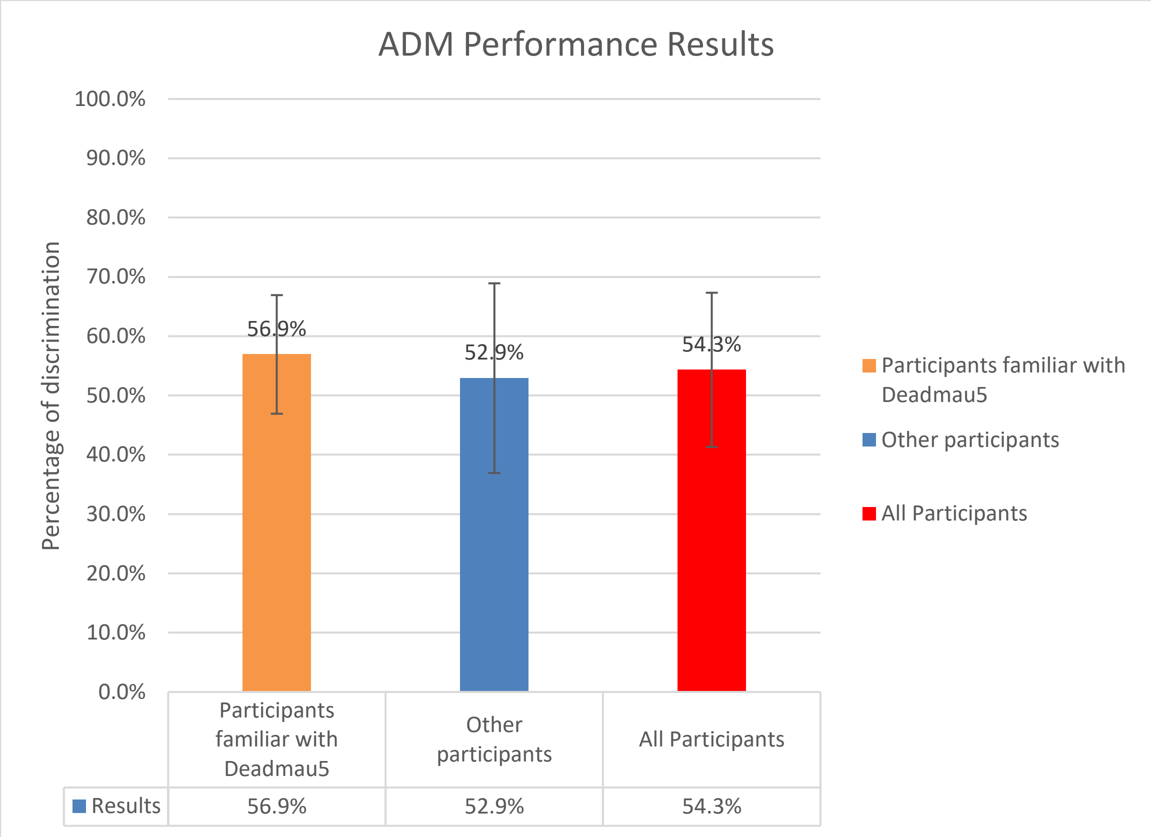


Figure 23. A bar chart showing the percentage of correct answers of the participants.

The result from all participants at 54.3% is only 4.3% higher than random guessing, which is a significant result that supports that ADM passed the Turing test and performed well. Not surprisingly, it also shows that participants who were familiar with Deadmau5 music scored a higher result, but 56.9% is still a good score. This is also a good score when compared to other studies using similar Turing tests; DeepBach with 50% (Hadjeres et al., 2017, p. 6)[92] and BachBot with 59% (Liang, 2016, p. 54)[93]. However, both of those studies had a much larger pool of participants; DeepBach, 1,272 (Hadjeres et al., 2017, p. 6)[94] and BachBot, 759 (Liang, 2016, p. 54)[95].

---

[92] Hadjeres, G., Pachet, F., & Nielsen, F. (2017). DeepBach: a Steerable Model for Bach Chorales Generation.
[93] Liang, F. (2016). *BachBot: Automatic composition in the style of Bach chorales* University of Cambridge]. Churchill College.
[94] Hadjeres, G., Pachet, F., & Nielsen, F. (2017). DeepBach: a Steerable Model for Bach Chorales Generation.
[95] Liang, F. (2016). *BachBot: Automatic composition in the*

Figure 24 shows how the performance result was different for musicians compared to other participants.

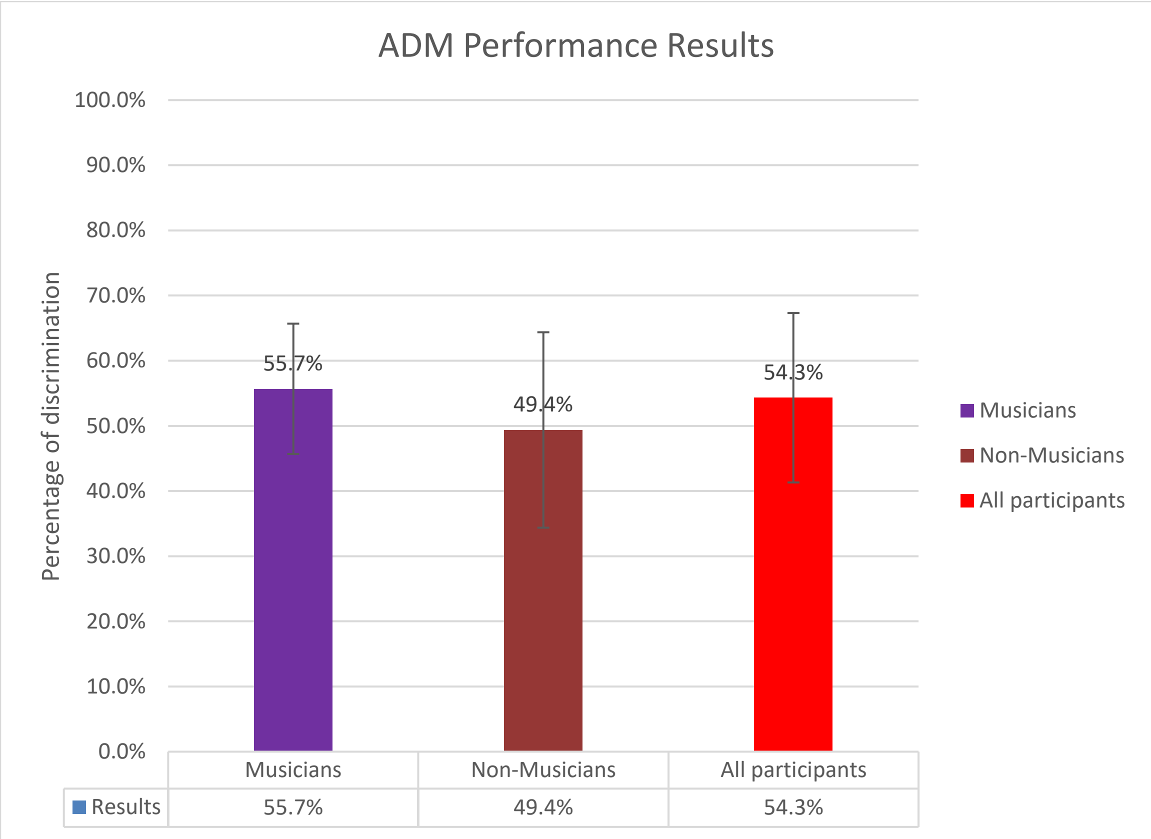


Figure 24. The percentage of correct answers of the participants.

This result indicates that musicians had a 6.3% higher chance of correctly distinguishing AI from human samples, which was as expected. 49.4% for the non-musicians is lower than 50%, which means that the result is approximative the same as guessing. Both results show that participants familiar with Deadmau5 music and musicians had a higher score, but overall, the score was low for the participants and high for the performance of ADM. The error bars also indicate that there is a spread in the results as expected, but not to a significant enough extent to reject the results.

Figure 25 displays the distribution of test results by each question. This model gives a visualization of each sample performance; a high percentage means that the sample was correctly distinguished by the participants. Sample 9 had the lowest value of 27%, while sample

style of Bach chorales University of Cambridge]. Churchill College.

10 had the highest of 81%. In context, sample 9 was a sample from the Deadmau5 song SATRN, which 73% of the participants thought was made by AI.

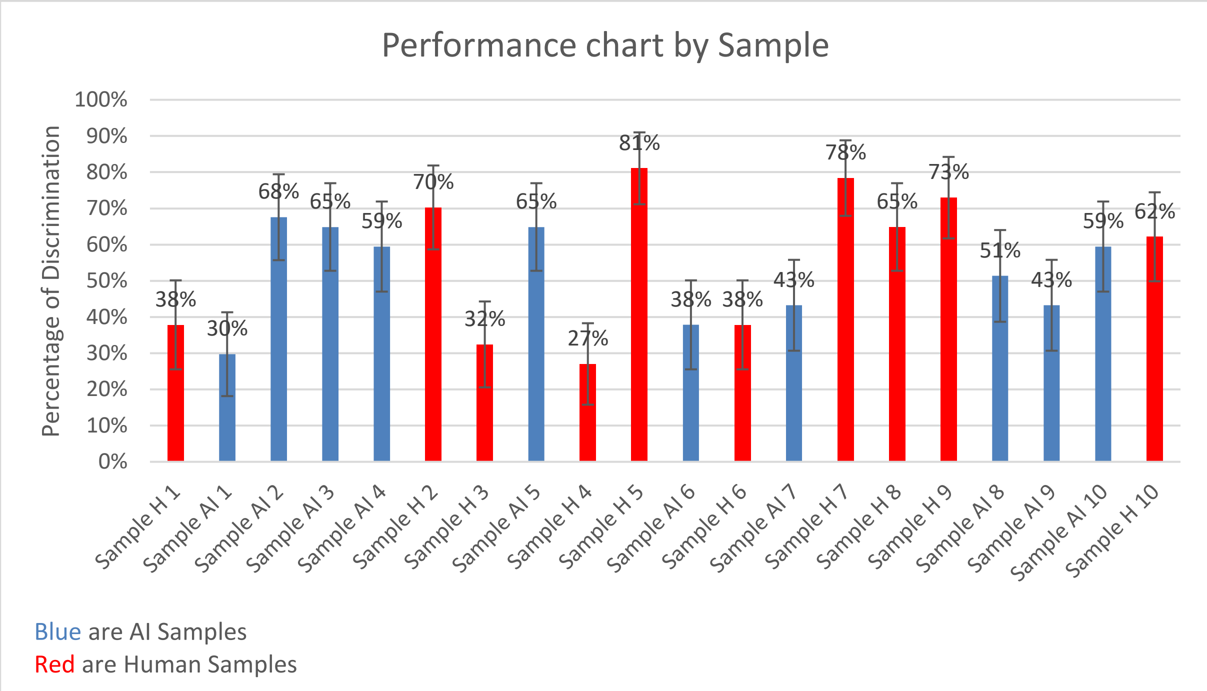


Figure 25. The performance of correct discrimination from each question.

Figure 26 presents the performance of all of the AI-generated samples. The samples that were most challenging to distinguish from the human samples were samples 1, 6, 7, and 9.

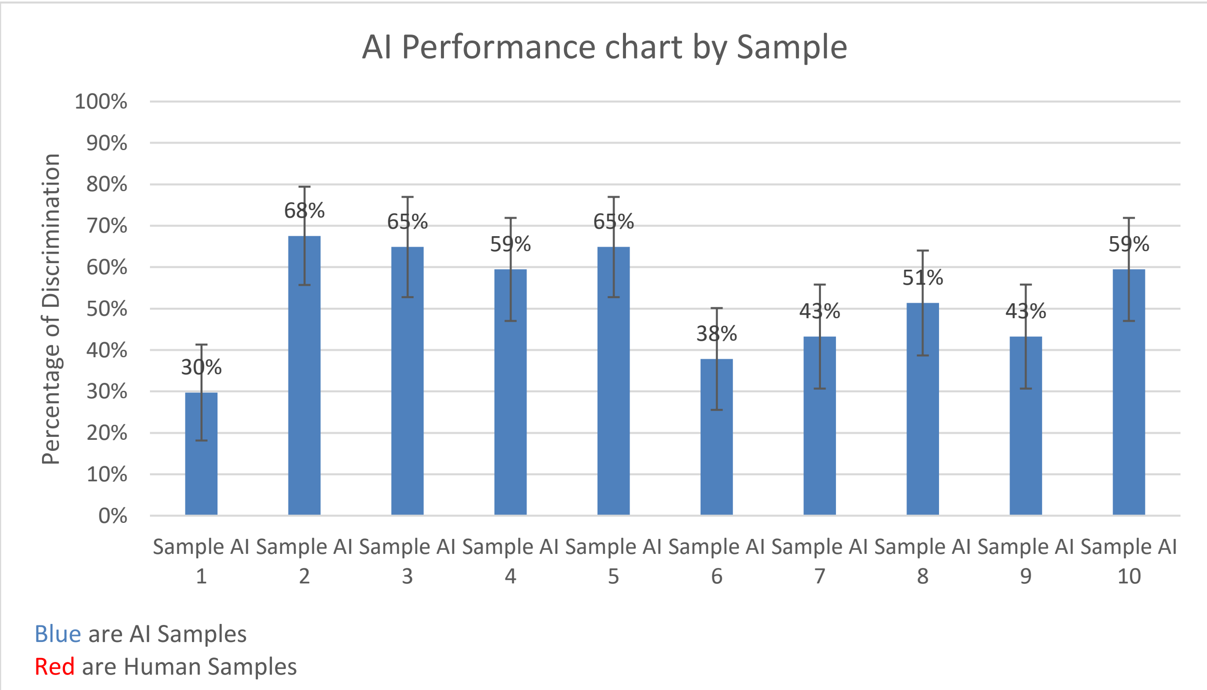


Figure 26. The performance of correct discrimination by each AI sample.

5.1.2 Statistical Conclusion

The statistical evaluation is that there is statistically significantly results, which does not prove the hypothesis, but it does support it. However, significantly good test results are not always a good thing, as they could also mean that the test is not working as it should be, such as if the sample size is too small. The only surprising thing about the tests was the low score on some of the Deadmau5 original tracks. This could mean that some of the original Deadmau5 songs that were used, from the perspective of the participants, sound like something an AI could have made. Also, there are some weaknesses in the test, which will be explained in the next section.

5.1.3 Limitations of the Turing Test and Evaluation

The first limitation of the Turing tests and the evaluation is the sample size. This must not be confused with the samples used in the research, as in the audio files, but instead refers to the number of participants from a population, as pointed out in Section 3.1.3. As mentioned before, other studies conducting similar research like DeepBach, BachBot, and BachProp used much larger samples sizes in their evaluation. However, in this thesis, other evaluation methods were also used that provided the test results with some more weight. Another problem was the sample order. For some reason, the first sample scored under 40%, as shown by Figure 25. This means that over 60% of the participants thought that it was made by AI, which was quite a surprising result. This could mean that the participants did not know what to expect and halfway through the test got a better understanding of what to expect from the samples. In other words, if this sample was moved to the last question, it would probably have a higher score. This could also have been fixed by using a randomly generated question order for each test, instead of the randomly fixed order implemented.

Another weakness of the test is that it does not evaluate how well the AI is making music, only how well it is mimicking the original music. The participants are listening to try to distinguish the AI from the Human samples and are not listening to evaluate their aesthetic quality. One example here is that if a participant likes the sound of an AI sample, but gets the feeling that this is not something Deadmau5 has made, then the sample will get a negative score. In this statistical evaluation, this means that it performed poorly and rejects the hypothesis that AI could create EDM music in this way. However, one could claim that the aesthetic quality of the music is indirectly evaluated if the music is indistinguishable from the original music. Also, asking participants about subjective opinions about the music could raise other challenges such as personal taste and understanding the aim of the task.

The last weakness that must be addressed is that the participants know that the test consists of 20 samples where 50% are made of AI. Knowledge about how the samples are distributed could affect their decision. Also, some of the participants reported that they felt a motivation to pass the test and distinguish the AI samples. Their test motivation was about passing the challenge rather than curiosity. For example, if a participant has confidently voted human on 50% of the samples before the rest were done, then statistically the rest of the samples are likely to be AI. But this could also go the other way; Figure 26 shows that 4 of the 5 first AI samples in the test got the weakest score. This could have been fixed by adding a randomly generated order and hiding the distribution of samples from the participants.

6 Conclusion

This conclusion chapter includes three parts: reflection, discussion, and future work. The goal of the chapter is to summarize the significant contributions of the thesis, defend the hypothesis, answer the research question, and give suggestions to future research, along with an important discussion about AI and music.

The first chapters presented the goals for the thesis, which were to use traditional AI methods to create EDM music using the methods presented in Chapter 4. They also gave the necessary background information about AI and Algorithmic music composition. In Section 2.4.1, it was claimed that one of the challenges with AI and music is the conversion of music into symbolic representation that can be easily interpreted by an AI algorithm. The important elements of music, such as layers, sound, timbre, and vocals, get lost in the symbolic musical translations. The music created by AI scored a statistically significant performance in the Turing test, which concluded that participants could only distinguish AI from Humans at a rate of 4.3% better than random guessing. Also, there was only a 6.3% difference between musicians and non-musicians in the test. However, further investigations discovered some limitations with the evaluation including the sample size, question order, and that the participants did not directly evaluate the quality of the music. The Avicii test in section 4.4.1 concluded that unprocessed EDM MIDI files could not be used with the BachProp script and the analysis in section 4.5 reported that there was evidence of Deadmau5's signature harmonization techniques in the AI.

6.1 Reflections

The researcher's personal reflection from building the system is that the research and developments in Algorithmic Music composition have come quite far. The BachProp provided methods that were crucial for this thesis such as MIDI normalization (Colombo & Gerstner, 2018, p. 2)[96]. The original method was to use Andrej Karpathy char-rnn[97] and create a custom MIDI processing script, which would have been extremely time-consuming. The most surprising aspect about this work was the quality of the music created, which was better than expected. Some of the samples were inspiring to listen to and gave me motivation and ideas to create music. This was very satisfying since the goal of the thesis was to investigate the extent to which ML can create human-like EDM music that can be used in a creative collaboration with an artist. The researcher's personal favorites were AI sample 7 and 10, the latter of which unfortunately was not among the samples that performed best in the Turing test.

Another surprising experience was the strong opinions that other musicians had about this subject when the researcher discussed this project. Either they think it is worrying because it threatens their job or they confidently opined that it is simply impossible for a machine to create music, without any other explanation than *"It's art."* Since this is the researcher's personal experience with mostly young musicians, this is in no way representative of all musicians, but this phenomena did lead to the investigation of any differences between musician and non-musicians in the Turing test. Could this skepticism against AI be the reason that it is not used in commercial music? Or is it that the technology is not mature and user-friendly enough? An interesting view about skepticism of AI-generated music is the focus on AI automating the process of creating songs or replacing the artist. But is this method different from traditional music composition? This researcher's perception is that a composer listens to their favorite music (dataset) and creates their own music based on that information. In this work, the researcher had a significant part in shaping the outcome of the music generated, so in this context, was only shifting the paradigm from actual composition to creating a dataset and tune in the parameters. As stated before, using AI to create music like this cannot be done without the aid of a human, and creating music in this way is more work than traditional composing.

[96] Colombo, F., & Gerstner, W. (2018). BachProp: Learning to Compose Music in Multiple Styles.
[97] https://github.com/karpathy/char-rnn

Another experience with AI and musicians, is that there is a common sci-fi misconception toward AI. Many speak of AI like it is an artificial version of the human *"mind"* and thus they consider the possibilities of AI both fascinating and scary. Worryingly, misunderstandings like this can generate a hype for AI that creates expectations that are too high. AI needs to be explained as it really is, avoiding the sci-fi image that fosters unrealistic expectations.

According to Thomas (2019)[98], the development of AI is going to change the world more than anything in the history of humankind. Based on this information, it would be naive to believe that AI will not impact music in the future. Conversely, the two AI winters have shown that it is typical to exaggerate the expectations of what AI can do.

The disappointing aspect of the ADM's performance in this study was the lack of both chord progressions and melody, which were featured in the dataset. There are melodies featured in the AI samples, but it is not a dedicated layer like a lead melody an octave over the chord progression. The harmonic arrangements were mostly rhythmical ideas over the same chord progression. Chord progressions was featured in the dataset, such as the chord progression described in section 2.1.1. When reflecting on the training process, the goal of achieving chord progression was too ambitious when training this way, because the chord progressions were not presented to the DL model using a sufficient method. One solution to this could be to transcribe Deadmau5 music into a symbolic representation of chord progressions (like 2-4-6-1) and only having the melody in MIDI. This would probably work, but then the training process would be simplified and limited by the data, which would also limit the output. This researcher's opinion is that research like that would be too narrow and lacks the excitement and the unpredictability that DL provides.

6.2 Discussion

This thesis has presented a method to enable an ML model to successful train on a small dataset of EDM using the MIDI system presented in Chapter 4. The hypothesis from this is that training an ML model on a small dataset can be successful if it is done in a particular way:

[98] Thomas, M. (2021). *THE FUTURE OF AI: HOW ARTIFICIAL INTELLIGENCE WILL CHANGE THE WORLD*. builtin.com. Retrieved Februrary 13, 2021 from https://builtin.com/artificial-intelligence/artificial-intelligence-future

Hypothesis for successfully training on a small dataset:

- **Make the right choice of music**
  Choosing house music and instrumental songs was crucial for the project to succeed, as explained in Chapter 2.

- **Include only one artist**
  More artists were considered, which would have given more data, but as stated in Section 2.1, more data is not necessarily better.

- **Remove drums**
  As explained in the research log in Section 4.3.3, removing the drum instrument from MIDI significantly improved the AI-generated music. The removal of drums was justified by the lack of variation in the drum patterns in the songs.

- **Include only one part of the song, in this case, the drop**
  In the beginning, whole songs were processed with the ML, but changing to only one part significantly improved the results. Also, the goal was to create small music ideas/samples, not full-length songs.

- **Organize the different instruments into three different layers and spread it out to a specific octave in each MIDI file**
  As mentioned in Chapter 4, this method increased the performance of the ADM and made it possible for a visual analysis.

- **Reduce the sample length to four bars**
  The AI-generated music consisted of long MIDI files with short ideas. The reduction of the sample length used in the Turing test was necessary for the test to be passed. This is also justified by the goal, which is to use AI to create short EDM ideas.

- **Change the Epoch and Temperature value until the desired output quality is met**
  This was one of the success factors for this thesis, as described in section 4.3.5.

- **Create an implementation script to automate the process and use an iterative process to make the necessary changes**
  Both the implementation script and the iterative process provided the necessary

methods to succeed with the training. The implementation script made it possible to initiate faster training on multiple cloud hosts while the iterative process created a research environment that helped design the system.

Evaluation of the Hypothesis

After the research was done, the Turing test was a success, but the test results only support the hypothesis and do not prove it. However, 4% more than random guessing is strong evidence indicating that the ADM project passed the Turing test with a strong performance. This means that there was not a significant difference in discriminating the AI from Human samples. This was also a good result when compared to similar research. However, there are some weaknesses in the test, such as small sample size, order of questions, and the lack of aesthetic evaluation. Those weaknesses reduce the validity of the test results and therefore, other evaluation methods were also needed to support the hypothesis. The qualitative observation method described in Chapter 4 consisted of a harmonic and rhythmical analysis (Section 4.5) of the AI-generated music using Deadmau5's composing methods, as described in Section 2.1.1. This method concluded that there was evidence of the Deadmau5 signature harmonizing method, *"polymeter,"* featured in the AI-generated samples. This evidence is proof that the AI is learning from Deadmau5 to create music and that the ADM MIDI system is working. The last piece of evidence is the Avicii test in Section 4.4.1. This test reveals nothing about the aesthetic quality of ADM music, but it tests the importance of the MIDI system in the implementation part of the work. The test concluded that without the necessary preprocessing described in the implementation chapter, the BachProp model fails to generate EDM music from a small dataset. The conclusion is that AI can be used to create EDM music to the extent of creating harmonic and rhythmic ideas trained on a small dataset to be used in a collaborative creative process with an artist. A further conclusion is that AI used this way cannot create whole songs without the aid of a human, only short ideas. The problem, however, is that the current method used in this thesis is too complicated and time-consuming for that to be an effective approach. Of course, this is only the current state, and development of the research could change this in the future. The science and theoretical framework are available, but the methods are not, which will be elaborated on in the next section. The research question also raised the sub-question, *"How important is the way in which the music is presented to the AI?"*. The hypothesis for this was that an DL model could successfully train on a small data size if it was done in a particular way, which was described in the implementation chapter and evaluated with the Avicii test, qualitative observation, and statistical evaluation.

6.3 Future Work

Modify BachProp

- Create a method to save the model after a certain amount of epoch with the validation log to make it easier to find the right epoch for generating music. Char-RNN has a feature like this (Karpathy, 2014)[99].

- Make it more user-friendly to implement data into DL models for other musicians to experiment. It should not be needed to be a computer scientist or a Keras expert with a black belt in Linux to experiment with AI and music. The GitHub implementation script developed in this thesis is a strong contender as the right way to approach this. The researcher also made a website with the Turing test and information about how the music was made including links to the appropriate articles and GitHub pages, which is available at http://artificialdancemusic.com

- Experiment with changing the weights inside the model after it is trained to alter the result of the music. This could be done to make the model go beyond the limits of the dataset.

Research on how MIDI information can be limited in order to enable successful training on small datasets

- There should also be a guide on how to preprocess MIDI according to genre to make it ready for AI processing. Bachprop claims that the MIDI data needs no preprocessing. In traditional cases like large collections of baroque music, this is true, but when

[99] Karpathy, A. (2015). *The Unreasonable Effectiveness of Recurrent Neural Networks*. Andrej Karpathy blog. Retrieved November 20, 2020 from http://karpathy.github.io/2015/05/21/rnn-effectiveness/

working on significantly small data collections in commercial music, the current research proves that the MIDI needs some preprocessing. The unprocessed Avicii MIDI data failed in Section 4.4.1 and by removing the drums layer, and splitting instruments into different octaves in the proposed MIDI system, the output of the music from noise to samples that passed the Turing test significantly increased.

Perhaps this thesis will gain some attention to the topic of AI and music. The average musician does not know how easy it can be to do research with AI and hopefully small projects like http://artificaldancemusic can advertise it. After all, it was a simple YouTube video that initially led this researcher to this topic.

Appendix A - Turing Test Sample List

Sample	Link
1. Sample H 1	https://github.com/lrs11/adm/raw/main/Samplelist/HUMAN_1.wav
2. Sample AI 1	https://github.com/lrs11/adm/raw/main/Samplelist/AI_1.wav
3. Sample AI 2	https://github.com/lrs11/adm/raw/main/Samplelist/AI_2.wav
4. Sample AI 3	https://github.com/lrs11/adm/raw/main/Samplelist/AI_3.wav
5. Sample AI 4	https://github.com/lrs11/adm/raw/main/Samplelist/AI_4.wav
6. Sample H 2	https://github.com/lrs11/adm/raw/main/Samplelist/HUMAN_2.wav
7. Sample H 3	https://github.com/lrs11/adm/raw/main/Samplelist/HUMAN_3.wav
8. Sample AI 5	https://github.com/lrs11/adm/raw/main/Samplelist/AI_5.wav
9. Sample H 4	https://github.com/lrs11/adm/raw/main/Samplelist/HUMAN_4.wav
10. Sample H 5	https://github.com/lrs11/adm/raw/main/Samplelist/HUMAN_5.wav
11. Sample AI 6	https://github.com/lrs11/adm/raw/main/Samplelist/AI_6.wav
12. Sample H 6	https://github.com/lrs11/adm/raw/main/Samplelist/HUMAN_6.wav
13. Sample AI 7	https://github.com/lrs11/adm/raw/main/Samplelist/AI_7.wav
14. Sample H 7	https://github.com/lrs11/adm/raw/main/Samplelist/HUMAN_7.wav
15. Sample H 8	https://github.com/lrs11/adm/raw/main/Samplelist/HUMAN_8.wav
16. Sample H 9	https://github.com/lrs11/adm/raw/main/Samplelist/HUMAN_9.wav

17. Sample AI 8	https://github.com/lrs11/adm/raw/main/Samplelist/AI_8.wav
18. Sample AI 9	https://github.com/lrs11/adm/raw/main/Samplelist/AI_9.wav
19. Sample AI 10	https://github.com/lrs11/adm/raw/main/Samplelist/AI_10.wav
20. Sample H 10	https://github.com/lrs11/adm/raw/main/Samplelist/HUMAN_10.wav

Bibliography

About Keras. (2021). Retrieved February 10, 2021 from https://keras.io/about/

anranga95. *Harmony & Scales*. Retrieved November 2, 2020 from https://anranga95.wixsite.com/bookofmau5/blank-b4ggw

Arcos., R. L. D. M. a. J. L. (2002). AI and music: From composition to expressive performance. *AI magazine*

Armadamusic. (2021). *THE COMPLETE GUIDE TO EDM (OR ELECTRONIC DANCE MUSIC)*. Retrieved April 1, 2021 from https://www.armadamusic.com/news/edm-electronic-dance-music

Bashill, P. (2002). Six Machines That Changed the Music World. *Wired Magazine*, (10).

Boden, M. A. (2018). *Artificial Intelligence A very short introduction*. Oxford University Press.

Brownlee, J. (2019). *Difference Between a Batch and an Epoch in a Neural Network*. Retrieved April 10, 2021 from https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/

Butler, M. (2006). *Unlocking the Groove: Rhythm, Meter, and Musical Design in Electronic Dance Music*. Indiana University Press

carykh. (2017). *Computer evolves to generate baroque music!* https://www.youtube.com/watch?v=SacogDL_4JU

Chandra, R. (APR 4, 2019). *The What's What of Keras and TensorFlow*. https://www.upgrad.com/blog/the-whats-what-of-keras-and-tensorflow/

Chollet, F. o. (2018). *Deep learning with python*. Manning Publications Co.

Clarke, E., & Cook, N. (2004). Empirical Musicology: Aims, Methods, Prospects. In (pp. 157-197). Oxford University Press, Inc.

Colombo, F., & Gerstner, W. (2018). BachProp: Learning to Compose Music in Multiple Styles.

Dahlstedt, P. (2018). *Big Data and Creativity (preprint)*. https://doi.org/10.13140/RG.2.2.23657.60007

Deadmau5. *Developing Melodic Structures*, Masterclass.com. https://www.masterclass.com/classes/deadmau5-teaches-electronic-music-production

Deadmau5. (2009a). Arguru. On *Random Album Title*.

Deadmau5. (2009b). Moar Ghosts 'n' Stuff On *For Lack Of A Better Name*.

Deadmau5. (2014). Brazil. On *5 years of Mau5*.

Dean, R. T., & Mclean, A. (2018). *The Oxford Handbook of Algorithmic Music*. Oxford University Press. https://doi.org/10.1093/oxfordhb/9780190226992.001.0001

Deb, S. (2020, April). Keras vs TensorFlow vs PyTorch : Comparison of the Deep Learning Frameworks. *Artificial Intelligence*. https://www.edureka.co/blog/keras-vs-tensorflow-vs-pytorch/

Dickson, B. (2020). *What is artificial narrow intelligence (Narrow AI)?* Retrieved April from https://bdtechtalks.com/2020/04/09/what-is-narrow-artificial-intelligence-ani/

Djupdal, K. (1995). Musikkteori. In (pp. 150-151). Ad Notam Gyldendal.

Elowsson, K. A. (2002). Polyphonic pitch tracking with deep layered learning. *Journal of the Acoustical Society of America*, 446-468.

Field, A. (2013). *Discovering statistics using IBM SPSS statistics: and sex and drugs and rock 'n' roll, 4th Edition*. London.

Geitgey, A. (2016, February 2, 2021). Machine Learning is Fun Part 6: How to do Speech Recognition with Deep Learning. *Adam Geitgey*. https://medium.com/@ageitgey/machine-learning-is-fun-part-6-how-to-do-speech-recognition-with-deep-learning-28293c162f7a

Hadjeres, G., Pachet, F., & Nielsen, F. (2017). DeepBach: a Steerable Model for Bach Chorales Generation.

Hiller, J., L. A., & Isaacson, L. M. (1958). Musical Composition with a High-Speed Digital Computer. *Journal of the Audio Engineering Society*, *6*, 154-160.

Huxtable, S. (2014a). What is Progressive House? *Decoded Magazine*. https://www.decodedmagazine.com/what-is-progressive-house-2/

Huxtable, S. (2014b). What is Progressive House? *Decoded Magazine*. https://www.decodedmagazine.com/what-is-progressive-house-2/

Karpathy, A. (2015). *The Unreasonable Effectiveness of Recurrent Neural Networks*. Andrej Karpathy blog. Retrieved November 20, 2020 from http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Koskoff, E. (2004). *Music Cultures in the United States: an Introduction.* (Vol. 1 edition). Routledge.

Kruchten, P. (2014). From Waterfall to Iterative Development-A tough transition for project managers. *The Rational Edge.*

Larman, C., & Basili, V. (2003). Iterative and incremental developments. a brief history. *Computer*, *36*, 47-56.

Liang, F. (2016). *BachBot: Automatic composition in the*

style of Bach chorales University of Cambridge]. Churchill College.

Magazine, M. T. (2014). JUST HOW BIG IS EDM? *Music Trades Magazine*.

Marsland, S. (2014). *Machine learning: An Algorithmic Perspective*. Chapman and Hall/CRC.

Naushad, R. (2020). *CPU, GPU and TPU — Machine Learning*. https://medium.com/dataseries/cpu-gpu-and-tpu-machine-learning-5033e25b8a0f

Nuanáin, C. r. O. (2018). *Connecting Time and Timbre:*

Computational Methods for Generative Rhythmic Loops in Symbolic and Signal Domains [Phd Thesis, Universitat Pompeu Fabra].

Official, R. M. (2015). *Analog Monsters Roland Juno-106 does ARP SOLINA String Synthesizer!* https://www.youtube.com/watch?v=OvQHMaDoABs

Reynolds, S. (2012). How rave music conquered America. *The Guardian*.

Ruud, E. (2016). Musikkvitenskap. In (pp. 33). Universitetsforl.

SadowickProduction. (2012). *Make The Pluck From Deadmau5 - The Veldt With Sylenth1*. https://www.youtube.com/watch?v=lNhdpOToeaw

Samuelson, P. A., & Nordhaus, W. D. (2001). Micoreconomics. In (pp. 454). McGraw-Hill Education

Sandred, Ö., Kuuskankare, M., & Laurson, M. (2009). Revisiting the Illiac Suite - A rule-based approach to stochastic processes. *Sonic Ideas/Ideas Sonicas*, *2*, 42-46.

Shaw, D. (2019). *8 AI tools that are at the cutting edge of music production*. happymag.tv. Retrieved September 14, 2020 from https://happymag.tv/8-tools-that-use-artificial-intelligence-to-produce-music/

Smirke, R. (2018). *IMS Biz Report: Global EDM Market Falls 2 Percent to $7.3 Billion*. Billboard.com. Retrieved Nov 10, 2020 from https://www.billboard.com/articles/business/8457536/ims-biz-report-2018-global-edm-market-electronic-dance

Snoman, R. (2009). *DACNE MUSIC MANUAL*. Focal Press.

Stanley, L. (2015). *Deadmau5*. the canadian encyclopedia. Retrieved november 13, 2020 from https://www.thecanadianencyclopedia.ca/en/article/deadmau5 statistics, A. B. o. (2021). *Sample Size Calculator*. Retrieved April 10, 2020 from https://www.abs.gov.au/websitedbs/d3310114.nsf/home/sample+size+calculator

Stitzel, R. (2021). What is a "Polymeter"? *DrumMantra*. https://www.drummantra.com/blog/what-is-a-polymeter

Sturm, B., & Ben-Tal, O. (2017). *'Machine folk' music composed by AI shows technology's creative side*. Retrieved March from https://theconversation.com/machine-folk-music-composed-by-ai-shows-technologys-creative-side-74708

Thomas, M. (2021). *THE FUTURE OF AI: HOW ARTIFICIAL INTELLIGENCE WILL CHANGE THE WORLD*. builtin.com. Retrieved Februrary 13, 2021 from https://builtin.com/artificial-intelligence/artificial-intelligence-future

Tripathi, S., Bhardwaj, A., & Eswaran, P. (2018). Approaches to Clustering in Customer Segmentation. *International Journal of Engineering & Technology*, *7*, 802-807.

Turing, A. M. (1950). Computing Machinery and Intelligence. *MIND, New Series*, *59*(236), 433-460.

Vitos, B. (2014). Along the Lines of the Roland TB-303: Three Perversions of Acid Techno. Dancecult: . *Dancecult: Journal of Electronic Dance Music Culture*

Wallace, B. (2018). *Predictive songwriting with concatenative accompaniment* University of Oslo]. University of Oslo.

Yegulalp, S. (2019). *What is TensorFlow? The machine learning library explained*. https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html