UTRECHT UNIVERSITY

Department of Information and Computing Science

Proposal - Master Thesis - Artificial Intelligence

Rhythm and Reason: Adding fine-grained control to deep learning music generation models.

First examiner:
Anja Volk

Candidate:
Efraim Dahl

Second examiner:
Peter van Kranenburg

February 26, 2025

Abstract

Music is essential in video games, enhancing immersion and engagement, but players may disengage due to excessive repetition or personal preferences. This is particularly problematic in games like LastMinuteGig, a Musical Attention Control Training (MACT) application for Parkinson's patients. To maintain engagement and the effect of the intervention, a diverse set of controlled, adaptive music is needed. Advances in generative music offer a scalable solution. This research explores efficient methods of adding control to pre-trained models, to steer towards rhythmically adaptive music suitable for MACT. Insights from small-scale experiments will inform the application of rhythmic control in MusicLang, a transformer based music generator, which will be used to build musical assets for LastMinuteGig.

Contents

1. Introduction

Music is a cornerstone in modern video games. Music evokes emotional responses, triggers memories, can direct attention and improve immersion and the overall experience of a game. Yet, it is not uncommon for players to become disengaged from the game music, often due to excessive repetition or simply different personal preferences.**Rogers_Weber_2019** This can be a problem in serious games for therapeutic use such as "Last Minute Gig" a gamified Musical Attention Control Training (MACT) application, aimed to improve attention control in patients with Parkinson's disease **Chalkiadakis_2022**. Ideally MACT is a personalized experience with dynamic adjustments taking into account a patient's abilities and preferences. In the context of gamified self-directed therapy,this is crucial to keep patients engaged and progressing.[1] In order to accommodate a wide array of patient's abilities and preferences in a MACT-game, a considerable amount of musical material, specifically music that fits the constraints of MACT is required. One potential avenue to provide this is through controlled music generation.

Over the last years, music generation has grown from a mostly academic endeavor to a billion-dollar industry. Large technology companies are exploring music generation with foundation models such as Meta's MusicGen**copet2023simple** or Google's MusicLM **Agostinelli_Denk_Borsos_Er** The commercial startup Suno is valued at 500 million dollars, while chart-topping songs are being generated with Udio **Ferdinand_Meyen_2024Stassen_2024**. This breakthrough is powered mostly by advances in language modelling, bringing about Large Language Models such as GPT-4 or LLAMA3 applied to music.

In the context of games, generative music is promising to cater to different user preferences and enable a richer, more varied musical experience than conventional aproaches to adaptive audio. Video games can include hundreds of hours of game-play and branching story lines. When additional complexity is added to adapt the music to individual preferences, it quickly is no longer feasible to manually compose unique music to fit all scenarios. Especially in more resource constrained applications such as games for therapeutic use, generated music can offer an avenue to provide a large variety of music that can improve user engagement, and ultimately the effect of the intervention. One of the key components of a successful generative model is control. In music generation control could be a text-description, information about genre, style or instrumentation. Control can also include fine-grained musical parame-

[1] This type of therapy is not aimed at replacing traditional guided therapy. Ideally it is used as supplement, and may provide relief in situations where traditional therapy is not immediatly available

ters such as a melody, chord progression or musical structure. Generative models can often be controlled with human-composed music, where the provided music is continued, interpolated between or inpainted (generating accompaniments, melodies, or additional musical lines). In LastMinuteGig**Chalkiadakis_2022** the music periodically provides stimuli, a noticeable change in the music, based on which the user changes their playing, specifically their rhythm. One way this stimulus could be achieved in generated music is by adding control that introduces a shift in rhythmic patterns at semi regular intervals.

There are a variety of ways to achieve control in a music generator. For rule based systems, control is integrated explicitly. In statistical systems (including deep learning) control is typically applied through architectural constraints and choice of training data. Certain deep learning architectures lend themselves to certain types of control. I.e an auto-regressive transformer models a sequence based on prior parts of the sequence, it natively generates continuations of an input. In deep learning, other musical parameters (i.e chord progression or melody) can be controlled for by joint conditioning of a model, which means that the controlled musical parameters are made explicit while training. This method of adding control is quite cost-intensive, requiring as much, or even more training material as the baseline model without controls.

A more economical method of adding control to a deep learning generative model is through fine-tuning and post-hoc conditioning. In fine-tuning (sometimes referred to as transfer learning) a pre-trained model is trained on additional data. There are various methods of fine-tuning, involving different configurations of the model, the model parameter and the integration of control input, but typically fine-tuning uses considerably fewer resources and data than training a model from scratch. In post-hoc conditioning a generative model is adjusted while it is running to increase the likelihood of the output to fit the desired constraints. The goal of this thesis is to steer a pretrained generative model to generate music fit to an MACT context, without training a large model from scratch, but using efficient fine tuning to introduce new control mechanisms to a pretrained model. For this I will first explore and evaluate different methods of adding control using small generative music model. I will then apply the most successful techniques of adding learned from this first step to a larger generative music model, MusicLang to steer the output towards something usable in the context of MACT.

In a final step we will generate a large amount of music to extend the LastMinuteGig video game for MACT in patients with Parkinson's. LastMinutGig is a mobile application built with Unity3D. In its current iteration it contains a single button with which a player taps along to the music, controlling a set of guitar chords. When a musical change occurs the player is instructed to change their tapping. It uses a simple music generation process based on recombination, which I aim to replace with a set of generated music.

2. Literature

The following section contains an extensive (though not exhaustive) categorical literature study. The first part provides contextual information, including motivations, and ethical concerns as well as a comprehensive overview of techniques used in music generation. The second half is more technical, with a discussion on music representation, tokenization, control and fine tuning. As a starting point, I used the most recent ISMIR (International Society for Music Information Retrieval) papers on music generation and their referenced sources, alongside other papers suggested by my advisor, Anja Volk. In addition, I performed systematic searches using the following keywords: Deep Learning Music Generation, Diffusion Music Generation, Transformer Music Generation, Symbolic Diffusion Music Generation, and Controlled Music Generation.

2.1 Why Generate Music? - Motivation

2.1.1 Composition Co-Pilots

The earliest music generators (including the 18th-century musical dice games**Nierhaus_2009**) were justified as methods to inspire composers and music makers, including enabling novice composers to write music. Composer David Cope**Cope_1989** states that he turned to music generation to overcome writer's block. Commercial enterprises like Suno claim to be "building a future where everybody can make great music" **Suno_AI**. Many current and past efforts highlight music generation as a process that assists composers. In DeepBach**Hadjeres_Pachet_Nielsen_2017**, the authors go to great lengths to make the system flexible and usable in real-life composition scenarios. Initially, they developed a MuseScore integration and later tied DeepBach into the web app NONOTO[1] to support musical inpainting in Ableton Live scores. Similarly, with Composer's Assistant**Malandro_2023**, the authors explicitly enable musical inpainting and continuation in the REAPER music production software. AI has also been explored as a co-improviser for live music-making, such as in Pachet's Continuator**Pachet_2003** and Ben-Tal's musical dialog system**Kite-Powell_2023**.

[1] https://github.com/SonyCSLParis/music-inpainting-ts

2.1.2 Music in Games

Beyond co-creating music, AI-generated music serves functional purposes, such as background music in videos and therapy-assisting games. In gaming, AI-generated or AI-assisted music is particularly relevant due to the scale and interactivity of games.

Video games often facilitate hundreds of hours of gameplay, featuring branching storylines and complex player interactions. However, game soundtracks typically cover only a fraction of that time**Plut_Pasquier_2020**, **Worrall_Collins_2024**. Through different adaptive techniques, relatively short snippets of original material can be stretched into hours of unique audio, often relying on the recombination of different elements. However, rule-based recombination has its limitations. A recent study of player behavior **Rogers_Weber_2019** finds that many players eventually turn off game music. They cite various reasons, such as preferring their own music over the game soundtrack (46.7%) or finding the in-game music repetitive (29.6%).

Procedural generation of 3D assets, levels, and enemy behavior is commonplace, but music generation remains underutilized. Composing and adapting music for every possible scenario would be tedious. AI-assisted music composition could enable adaptive audio on a large scale, either by creating numerous musical assets anticipating player choices or generating new variations of the game score in real time to enhance player immersion. However, several challenges make AI music generation difficult in games. Performance issues arise due to the resource-intensive nature of AI generators. Additionally, AI-generated music is difficult to control, and there is little guarantee that the generated tracks will be appropriate**Plut_Pasquier_2020**. Furthermore, AI music generators currently lack proper integration into video game environments and engines**Worrall_Collins_2024**.

2.1.3 Music in Serious Games

Aside from games for general audiences, generative music has potential applications in serious games. Serious games are designed for purposes beyond entertainment, such as education or therapeutic use, including music therapy**Djaouti2011**. In music therapy, music can be used for emotion regulation, motivation, adherence, motor coordination, rhythmic entrainment, and facilitating social interactions**musicwellbeing_agres_2021**. Musical attention control training has been shown to help individuals with Parkinson's**Park_Kim_2021**, ADHD**Martin-Moratinos_Bella-FernÃ**, autism**Pasiali_LaGasse_Penn_2014**, and psychosis**van_Alphen_Stams_Hakvoort_2019** improve their mental capabilities for selective and switching attention. Serious games have the potential to supplement music therapy. "Last Minute Gig"**Chalkiadakis_2022** implements clinical music therapy protocols as a serious game to improve attention control in Parkinson's patients. However, users reported boredom and a lack of feedback, while more musically experienced users felt less challenged. Schlette**Schlette_2022** attempted to address these issues by introduc-

ing dynamic difficulty adjustment through a feedback system alongside more complex music generation. This thesis aims to develop a controlled music generation model to improve player engagement through a richer music system.

2.2 Why Not Generate Music? - Ethical Concerns

2.2.1 Introduction

There are several concerns related to AI-based music generation. First, there are legal concerns regarding copyright and licensing. Generative models may produce outputs that are identical or highly similar to copyrighted training data. Furthermore, the question remains whether models should be allowed to train on copyrighted data in the first place. Broader concerns include the devaluation of human labor and creativity, the oversaturation of cultural spaces with low-quality generated content, and the environmental impact of large generative models, which require substantial energy, water, and rare materials to operate.

2.2.2 Data Leakage and Copyright

Generative AI companies are achieving record-breaking valuations and this includes music generators with Suno at a valuation of about 500 million dollars after a 125 million dollar fund-raiser leading the pack. **Stassen_2024 Tencer_2024**. However, AI companies are facing back-lash from artists and record labels with an organization of record labels including the "big three" Sony, Warner Music, and UMG suing Suno and Udio for $150.000 dollars per infringed work**Kaba_River_Perry_2024**. Generative AI runs a substantial risk of parroting or leaking training data. In language models, the leakage problem is of concern when training on data that contains sensitive information, which may be revealed either through accidental leakage or through membership inference attacks **Duan_Suri_Mireshghallah_Min_Shi_Zettlemoyer_Tsvetkov_Choi_E** While leakage may raise privacy concerns in other generative models such as speech and image-generators **Carlini_Hayes_Nasr_Jagielski_Sehwag_TramÃĺr_Balle_Ippolito_Wallace_2023** for music, the risk of training data leakage is mostly an issue of copyright. Ed Newton Rex shows some examples of how Suno can be influenced **Newton-Rex_2024** to leak training data. This ability to create disconcertingly close reproduction of copyrighted work is also cited in the court documents. Suno has since started to prevent prompting with artist names (i.e. in the style of Eminem) and including known song lyrics.

2.2.3 Training and Copyright

Besides leaking training data, there is the more general question of whether AI models should be allowed to train on unlicensed work. Echoing the court case between OpenAI and the New

York Times **Reed_2024**, both Suno and Udio cite fair use in response to accusations of copyright infringement. In US copyright law the fair-use clause limits exclusive rights to a work, with four factors to consider: 1) purpose and character of the work in use, 2) nature of copyrighted work, 3) amount of the copyrighted work used, and 4) the effect on the potential market or value of copyrighted work.**copyrightlaw** Fair use is often granted to derivative works such as parodies and covers and works used in educational settings. AI's learning of structures has also been likened to the human learning process, humans learn based on copyrighted music they listen to, without giving credit or compensation to their influences. Newton-Rex **Newton-Rex_2024** rejects this comparison. In learning music, humans contribute to the musical ecosystem, they take lessons, go to performances, or at the very least generate some streaming revenue for artists, none of these are true for machine learning models learning from scraped data.

2.2.4 Devaluing Music

While few are following director Ram Gopal Varma's announcement to only use AI-generated music in his future films **Singh_2024**, AI-generated music is becoming increasingly difficult to differentiate from human production and already receiving considerable amounts of streams and it is not unlikely that music in film, video and game projects may be replaced or at least supplemented with AI-generated music. The online music market is saturated, with more than 100,000 songs uploaded to music streaming giant Spotify every day **Stassen_2023**. Generative AI may just further exacerbate this problem, resulting in a race to the bottom for creatives and musicians.

2.2.5 Environmental Impact

Digital infrastructure, traditional data centers, crypto-currency mining, and AI-centered data infrastructure account for about 2% of the world's energy consumption **Marechal_2024**. The performance of current large language models often scales with simultaneous increases in model size, training data, and computation time.**Kaplan_McCandlish_Henighan_Brown_Chess_Child_Gray** Each of these three factors requires considerable resources. Music generation is no exception. In a recent ISMIR publication **Holzapfel_Kaila_JÃd'Ãd'skelÃd'inen_2024** the authors make estimates on energy consumption of different projects related to music generation and computation-intensive MIR, finding an average energy consumption of 224.8kWh for model training (the energy consumption of an average western person over 2 months). The energy consumption is divided highly unevenly, with the median being at merely 18 kwh (3 days of an average westerner's energy consumption). Music generation models associated with large technology companies are responsible for about 89% of the estimated energy use. This is only for training, models that are deployed publicly, continue to use substantial energy for infer-

ence. Beyond just the carbon footprint of generative AI, the local impacts of resource use such as rare minerals and water are important to keep in mind.

2.2.6 How I plan to address these concerns

During this thesis process, I'm planning to address the outlined ethical concerns in the following manner. First, I will only train, and use open-source models trained on licensed data providing attributions where possible. I will also make my trained models publicly available with substantial documentation to contribute to the open-source ecosystem, research, and music. Second I am explicitly designing the generative process to be cooperative, this is facilitated on the one hand through the introduction of new control modalities, and on the other the choice of symbolic music over audio, which eases editing and integration into composition software. Finally, I am not attempting to train custom large foundation models, rather I'm trying to find ways to extend existing models to introduce new control mechanisms, requiring training on only a fraction of the model parameters, with substantially less need for data, computation, and energy.

2.3 Overview of music generation

From antique wind-chimes to classical period musical dice games to the aleatoric music of the 20th century - humans have used algorithmic, probabilistic, and statistical methods to create music. As early as the late 1940s, computers have played a role in composition as sound generators, instruments **France-Presse_2016** and providing musical material themselves, such as in the 1957 Illiac suite **Hiller_Isaacson_1959**. The following 50 years are characterized by disparate, academic experiments in music generation, mostly in the symbolic music domain. They utilize a variety of contemporary AI technologies, from expert systems and ontologies **Hiller_Isaacson_1959EbcioÄ§lu_1994** to evolutionary algorithms **Polito_Daida_Bersano-Begey_1997** to feed-forward **Todd_1989**, recursive **Mozer_1994**, and convolutional neural networks **coconet**. Some composers in the classical tradition, such as Iannis Xenakis **Xenakis_1992** and David Cope**Cope_1989** use computer algorithms in their creative work. In the 2010s a small ecosystem of commercial generative music startups such as Jukedeck, PopGun, and Ampermusic **Featherstone_2017** starts to emerge, alongside an increasing number of publications applying deep learning, particularly Generative Adversarial Networks (GANs), and Recursive Neural Networks (RNN) to music including MIDInet **midinet**, DeepBach **Hadjeres_Pachet_Nielsen_2017** and FolkRNN **Sturm_Ben-Tal_2016**. With the development of the transformer architecture in 2017 **Vaswani_Shazeer_Parmar_Uszkoreit_Jones_Gomez_Kaiser_Polosukhin_2017**, large technology companies start experimenting with music generators including OpenAI's Jukebox **Dhariwal_Jun_Payne_Kim_Radford_Sutskever_2020** and Musenet**Christine_2019**, Meta's MusicGen**copet20**

and Google's MusicLM **Agostinelli_Denk_Borsos_Engel_Verzetti_Caillon_Huang_Jansen_Roberts_Taglias** and the preceding Magenta project, with fully generative models capable of producing sequences of high-quality music modeled from raw audio. At the time of writing, commercial music generators such as Suno and Udio are raising millions of dollars in investment funds**Stassen_2024**, while generated music is being widely streamed and actively used in TV and video productions.

2.4 Non-neural music generation

2.4.1 Why look beyond deep learning

Neural, specifically deep learning (DL) systems, currently receive considerable attention. However, given their substantial drawbacks relating to explainability, transparency, computational efficiency, copyright and licensing issues, and their enormous need for data (more in section **??**), it is worthwhile considering alternative approaches to music generation. A recent study **Yin_Reuben_Stepney_Collins_2023** performs a comprehensive listening survey comparing neural net and non-neural net systems. The top-performing systems a Markov Model - MAIA Markov **Collins_Laney_2017** and a deep learning system - MusicTransformer **Huang_Vaswani_Uszkoreit_Sl** perform similarly well in the listening study. The choice of one of the earliest transformer-based music from 2018 **Huang_Vaswani_Uszkoreit_Shazeer_Simon_Hawthorne_Dai_Hoffman_Dinculescu_Eck_** and the restriction to symbolic music, raises questions, whether their conclusion of similar performance still holds. Considering the study was published in 2023, these are important limitations. However, their criticism that many DL-based music generation projects do not look beyond DL and compare their systems based on technical metrics - with no obvious impact on how human listeners perceive the output remains solid. Finally, many hybrid approaches successfully combine traditional rule-based or statistical methods with deep learning. Those methods may help researchers and developers maintain a more comprehensive toolkit of techniques and paradigms.

2.4.2 Rule-based music generation

Non-neural net systems can classified as either rule-based or statistical. Both types have been part of some of the earliest attempts at music generation. Centuries of style-defining musicological writing from ancient Mesopotamian tuning charts **Mirelman_2013** to Fux's Gradus ad Parnassum **Fux_1725** and Arnold Schönberg's 12-tone music have crystallized sets of rules that approximate various styles of music. Many approaches to music generation take advantage of this knowledge and codify it into a computer program, creating expert systems for music generation. In Hiller & Isaacson's Illiac Suite, a series of experimental, computational com-

positions, the first and second movements are generated following the rules of first species counterpoint **Fux_1725**, approximating Palestrina's contrapuntal technique. Some rules aim to contain the melody, such as limiting the range to an octave, enforcing the identical start and end notes, and avoiding consecutive melodic jumps. Other rules aim to constrain harmony, such as forbidding parallel octave, fifth, and fourth motion and enforcing consonant harmonies. Hiller and Issacson's approach is relatively simple, using only a handful of conditions (see figure **??**). Rule-based generation can be highly complex, such as CHORAL **EbcioÄ§lu_1994** (for which the developer also built a custom programming language), which encodes over 300 rules to realize bach-style chorales from a given melody.

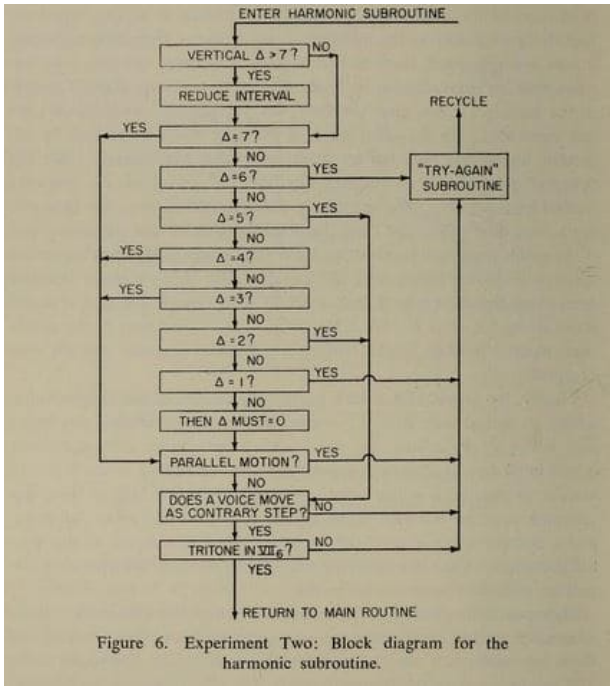


Figure 6. Experiment Two: Block diagram for the harmonic subroutine.

Figure 2.1: Rule-Based - Block Diagram from Hiller and Issacson's book - explaining movement two of the 1957 Illiac Suite.

2.4.3 Markov Model Music Generation

Markov models remain a popular method of generating music to this day. At its simplest, a Markov music generator could work off of a transition matrix for pitch classes, such as Richard Pinkerton's 1956 "Banal Melody Generator" **Pinkerton_1956**. Markov chains can be nested or constrained for more complex interactions **Collins_Laney_2017**. In Hiller and Isaacson's **Hiller_Isaacson_1959** fourth movement of the Illiac suite, they use Markov chains with a table of possible intervals stretching from unison to octave. In the latter sections of the movement, they introduce additional restrictions to add memory to the method through higher-order Markov chains that reference previously generated music. Transition matrices can be built from very little data, such as short improvisations **Pachet_2003**, but training over a whole corpus is also viable. Other systems configure Markov chains to take additional inputs into

account, such as Allan, **Allan_2002** who generates harmonies to given melodies in the style of Bach. This makes Markov model-based systems very flexible and relatively lightweight.

	O	C	D	E	F	G	A	B
O	0.38	0.17	0.10	0.10	0.06	0.13	0.03	0.02
C	0.36	0.23	0.13	0.07	0.02	0.10	0.03	0.07
D	0.26	0.20	0.21	0.19	0.03	0.06	0.01	0.05
E	0.22	0.15	0.18	0.16	0.16	0.12	0.01	0.00
F	0.15	0.00	0.14	0.35	0.14	0.20	0.01	0.01
G	0.29	0.14	0.00	0.16	0.06	0.26	0.08	0.00
A	0.17	0.05	0.07	0.00	0.02	0.36	0.15	0.17
B	0.18	0.30	0.12	0.01	0.01	0.08	0.21	0.08

Figure 2.2: Transition matrix from Pinkerton's 1956 "Banal Music Generator". Probability of a pitch (row) following on a pitch (column), likely pairs are marked in yellow. The probabilities are calculated from a set of 39 nursery rhymes.

2.4.4 Other statistical approaches

Music generation has also been attempted with other means, such as metaheuristic search for harmony or melody **Altay_Alatas_2018**. In Morpheus, **Herremans_Chew_Morpheus_2019** the developers use variable neighborhood search (VNS) to generate polyphonic pieces following a tension profile for long-term structure. Closely related to this approach are evolutionary and genetic algorithms such as Politio et al's **Polito_Daida_Bersano-Begey_1997** model of 16th-century counterpoint as a multi-population problem. Here, three separate populations of agents generate instructions focusing on different musical aspects, such as harmony or imitation, and are evaluated based on individual performance and symbiotic performance together with the other agents over multiple generations.

2.4.5 Early Neural Net based systems

Music generators based on neural nets were introduced as early as 1989. Todd et al. **Todd_1989** generate melodies using a fixed window for a conventional feed-forward neural network but also introduce a feedback loop feeding the network's previous state to the next iteration. Future work based on RNNs uses the latter principle, such as CONCERT,**Mozer_1994** a 1994 RNN trained to generate melodies based on datasets of Bach chorales, waltzes, and European folk songs. There are also hybrid systems such as HARMONET **Hild_Feulner_Menzel_1991**, an RNN-based music generator for re-harmonizing Bach chorales. It merges the RNN with a symbolic rule-checking algorithm. More recent RNN-based music generators, such as FolkRNN

Sturm_Ben-Tal_2016, a melody generator trained on Irish folk songs, use long short-term memory (LSTMs) or gated recurrent units (GRUs). Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs) remain popular architectures for music generation. **Civit_Civit-Masot_C**

2.5 Deep learning for music generation

State-of-the-art music generation, including many commercial applications, leverages the advances in language and image generation over the last five years. Two distinct approaches, namely autoregressive, transformer-based, and diffusion-based approaches dominate.

2.5.1 Transformers - sequence modeling without recurrence

Autoregressive music generation draws from successful natural language modeling tasks powered by the transformer model. Transformers were originally developed for the task of language translation.**Vaswani_Shazeer_Parmar_Uszkoreit_Jones_Gomez_Kaiser_Polosukhin_2017** They keep the self-attention mechanism already deployed in LSTMs for seq2seq tasks **Sutskever_Vinyals_Le_** but replace the recurrent connection with positional embeddings and masked attention. This allows the model to train on all tokens in parallel instead of one token at a time, which enables much larger and more capable models trained at a fraction of the time required to train similarly large RNNs. The transformer comes in several different configurations. The original transformer contains encoder and decoder layers - see figure **??**. Often, tasks relating to sequence understanding, such as music classification, use an encoder-only architecture. The BERT series of language models **Devlin_Chang_Lee_ToutanovaBERT_2019** and music understanding models such as MusicBERT**Zeng_Tan_Wang_MUSICBERT_2021** are examples. On the other hand, sequence generation tasks often employ a decoder-only architecture, this includes the GPT-series **Radford_Wu_Child_Luan_gpt2_2019** and many music generators such as MusicGen**copet2023simple**. The transformer architecture is the baseline for all current large language models. Transformers are used in modalities beyond text, including images, audio, or DNA sequences.

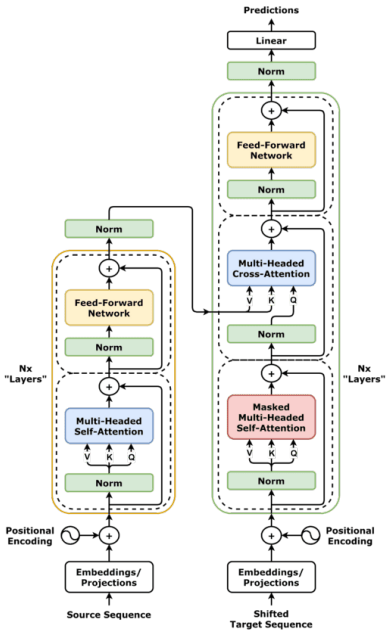


Figure 2.3: Schema of the full transformer encoder-decoder architecture
[a=] By dvgodoy - CC BY 4.0, https://commons.wikimedia.org/w/index.php?curid=151216016

2.5.2 Diffusion models - spectrograms and piano-rolls

Diffusion models are widely used for image and audio generation. Diffusion models learn to remove noise from a distribution (i.e. an image). Random noise is added to an image, and the model learns to undo this addition. During inference, the model starts with random noise (often accompanied by a guiding text prompt) and undoes it until it arrives at a clear image. AudioLDM **Liu_Chen_Yuan_Mei_Liu_Mandic_Wang_Plumbley_2023**, and StableAudio **Evans_Parker_Carr_Zukowski_Taylor_Pons_2024** are diffusion models that operate based on continuous audio encoding by a variational autoencoder. Diffusion models have also been used to generate symbolic music. Polyffusion **Min_Jiang_Xia_Zhao_polyffusion_2023** uses image representations of piano rolls and adapts their diffusion model for various tasks, inpainting, accompaniment, and melody generation and generation based on a given chord sequence or texture.

In this thesis, we will be targeting transformer models. Transformer models are tremendously popular and behind many state-of-the-art results in music generation. There is a tremendous amount of interest in model fine-tuning and control. In addition, many efforts aim to efficiently adjust models to related tasks without going through the relatively costly training procedure of large models. There is a large selection of open-source models that generate music and excellent infrastructure to support the training and evaluation of those models.

2.6 Representation and Format

The choice of representation of data is a crucial aspect of music generation. First, the choice of audio or symbolic music has consequences for data availability, dataset size, context of the output, and what features can be controlled. Second, tokenization, the method in which symbolic or audio data are chunked and ingested into the model is an important aspect, with tradeoffs to consider for different generative tasks and goals.

2.6.1 Symbolic Music vs Audio

Music can be represented digitally in two ways, either as an audio rendition or symbolically as a set of instructions. Working with different representations comes with various drawbacks and benefits for music generation. There are different types of symbolic representations of music, but the most common consist of discrete sequences of musical elements such as pitch or duration. Working with audio theoretically gives access to all audible qualities of music, including detailed information on instrumental timbre or acoustic settings. In symbolic music, this is restricted to pitch, duration, and instrumentation, which sometimes is extended to include formatting information such as bar lines, etc. Symbolic data is also far less available than audio data. Many symbolic music datasets are created by compiling hand-transcribed music. High-quality automatic transcription remains an unsolved issue.**Ji_Yang_Luo_survey_symbolic_2024Chen_Smith_** However, symbolic music gives more direct access to many higher-level musical features such as chord progressions, melodies, and instrumentation. When working with audio, these features have to be extracted first, requiring additional processing steps that are prone to inaccuracies. Another consideration to take into account is size. Raw audio is significantly larger than a corresponding digital score. In addition, rendered audio is difficult to edit once generated.

2.6.2 Tokenisation

Sequences are typically transformed into tokens, a numerical representation of data, to be handled by a machine learning algorithm. Audio-based music generation uses tokenization to condense audio while retaining its semantic content. Jukebox **Dhariwal_Jun_Payne_Kim_Radford_Sutskever_20** uses a variational autoencoder**Kingma_Welling_2014** with a discretizing bottleneck (VQ-VAE) to create tokens from audio. Musicgen **copet2023simple** tokenizes audio using the previously developed Encodec model for audio compression which similarly to VQ-VAE, learns a highly condensed discrete representation of audio **DÃl'fossez_2023_encodec**. These condensed encodings are crucial for generative modeling on audio.

2.6.3 Symbolic Tokenisation

Due to its discrete representation, symbolic music is typically not condensed further (with some notable exceptions, such as condensed piano roll representation for symbolic music diffusion **Min_Jiang_Xia_Zhao_polyffusion_2023Zhu_Liu_Jiang_Zheng_texture_2024**). However, the choice of tokenization influences the model's performance on different music generation tasks. **Fradet_Briot_Chhel_2021**. Common ways of tokenizing symbolic music align closely with the musical instrument digital interface (MIDI) standard, with individual tokens encoding different midi-events such as note-on, note-off, and velocity. The REMI+ **Huang_Yang_remi_pop_transformer_2020** tokenization expands on MIDI-based tokenization with tokens for bar and position, designed to help capture recurring musical patterns. The PerTok tokenizer designed by Lemonaid[2] encodes micro timings and offsets, designed to capture the full spectrum of rhythm in musical performances.

These extensions come at a cost: The resulting sequences can become very long, which adversely affects the model**Ji_Yang_Luo_survey_symbolic_2024**. Ongoing attempts are made to condense token sequences, such as compound words or nested tokens.**Ryu_Dong_nested_2024**. Dong et al.**Dong_Chen_MMT_Kirkpatrick_2023** combine six different MIDI-like events (type, beat, position, pitch, duration, instrument) into single tokens. Hsiao et al.**compound_word_Hsiao_Liu_Yeh_Y** differentiate between token types and group neighboring tokens into compound words, resulting in significantly shorter sequences (about 50% compared to individual tokens).

This thesis uses a third approach: Byte Pair Encoding (BPE) **Sennrich_Haddow_Birch_BPE_2016**. BPE is used widely in language modeling, including the GPT series of models**Radford_Wu_Child_Luan_gpt2_** and has been successfully applied to symbolic music generation.**Fradet_Gutowski_Chhel_Briot_2023** The approach is simple: the most common token-pairs of the dataset are repeatidly combined into new combined tokens until the total amount of unique tokens reaches a preset vocabulary size. This approach works independently of token types and semantic content of the tokens and allows for very flexible scaling of the vocabulary. As seen in figure **??**, this drastically shortens the sequence length by about a third ($mean_{individual} = 47976, mean_{bpe} = 14519$), which in turn improves both the quality and efficiency of the model.

[2]https://www.lemonaide.ai/

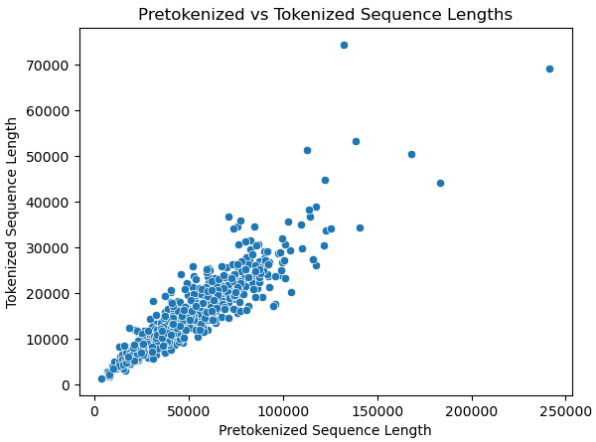


Figure 2.4: Scatterplot of individual vs bpe-tokenization sequence lengths

See table **??** in the appendix for a more elaborate display of representations used in symbolic music generators

Herremans_Chew_Morpheus_2019

2.7 Control

Control is an essential aspect of any generative model. Without control, even the best generative models producing beautiful music would be of very limited real-world use. Control allows generative AI tools to become proper collaborative systems, and generate for a wide array of scenarios. In music generation control covers essentially all musical parameters. Parameters vary by representation, symbolic music for instance leaves very little room for any type of timbre control (aside from instrument selection). "Raw" audio models such as Stable Audio (Evans et al., 2024) can for instance be controlled for acoustic settings (i.e jazz music playing in a busy restaurant, in a *large cathedral*, or *through an intercom*), something that is simply not represented in symbolic music. For musical parameters represented in symbolic music, there are different approaches to classifying them. We can differentiate between global and local features **Van_Kranenburg_Volk_Wiering_2013** or deep vs surface-level features **Blacking_1971**.

Global features in music generation encompass high-level descriptors such as genre, function, and instrumentation/orchestration. Global features also incorporate summarizing features, that are calculated from the music itself. As an example, McKay's **McKay_2004** general-purpose symbolic music classification system defines over 104 global features, 37 of them used for melody descriptors such as probability distributions of note durations, intervals, and pitch classes. These global features can then be used to infer other high-level descriptors such as genre if it's unknown, they can also support music retrieval systems **Van_Kranenburg_Volk_Wiering_2013**. **Local features** describe individual sequences such as melodic, rhythmic, or harmonic sections.

They are more information-rich, which may explain their superior performance in music retrieval **Van_Kranenburg_Volk_Wiering_2013**.

Also relevant in the music generation context are abstract vs concrete features. Music generation can take abstract concepts such as genre **Rouard_Adi_Copet_Roebel_DÃl'fossez_musicgenstyle_2024**, **Lu_Xu_Kang_Yu_Xing_Tan_Bian_MuseCoco_2023** or emotion **Tan_Herremans_2020 Lu_Xu_Kang_Yu_Xing** into account, which in turn effects more complex dynamics of concrete features. To illustrate: In Music FaderNets **Tan_Herremans_2020** the authors use a variational autoencoder to disentangle the abstract concept of arousal, into several concrete features including rhythmic density, note-density, tempo and dynamic, key. This allows them to change the abstract variable arousal, which cascades into a change of underlying features. This type of disentanglement can aid in situations where labled musical data with the abstract feature is less available, it can also aid in creating more accessible and interpretable generative models.

In the context of this thesis we differentiate between global features, and fine-grained or time-varying features **RÃijtte_figaro_2023**. What is time-varying or global is highly context dependent, a piece may have one time-signature and tempo as is assumed in **Lu_Xu_Kang_Yu_Xing_Tan_Bian_** or it may vary over time as is suggested in **RÃijtte_figaro_2023** or **Huang_Yang_remi_pop_transformer_2020**. Other time varying controls could be chords **RÃijtte_figaro_2023Wu_Donahue_musicontrolnet_2023Lan_Hs** melody **copet2023simpleMin_Jiang_Xia_Zhao_polyffusion_2023** or texture **Min_Jiang_Xia_Zhao_polyffusi** For the target application in an MACT - game, time-varying controls are necessary to provide a change in music that triggers a change in the patients improvisation. For a more complete list of the types of controls in many of the common symbolic music generators refer to table **??** in the appendix.

2.7.1 Rhythmic control

The types of control exercised over rhythmic components varies by representation as discussed in **??**. In CocoMulla **Lin_cocomulla_2024** generated audio is conditioned with drum tracks and a piano-roll. Similarly in JASCO**Tal_jasco** drums are used for conditioning. In MusicConGen.**Lan_Hsiao_Cheng_Yang_musicongen_2024**control for rhythm is added through tracking beats and downbeat. MusicControlNet**Wu_Donahue_musicontrolnet_2023** adds beat and downbeat conditioning to an audio diffusion model. For symbolic systems control of tempo and meter is relatively common **RÃijtte_figaro_2023**, **Huang_Yang_remi_pop_transformer_2020**, **Lu_Xu_Kang_Yu_Xing_Tan_Bian_MuseCoco_2023**. More fine grained control over rhythm is sometimes deployed through note-density (both vertical and horizontal)**RÃijtte_figaro_2023**,**Huang_rule_dif** Another option is control over texture, which merges harmonic and rhythmic elements. In Polyffusion**Min_Jiang_Xia_Zhao_polyffusion_2023**, texture is encoded by a pretrained variational auto-encoder**Wang_vae_chord_rhythm_2020**. Another approach **Zhu_Liu_Jiang_Zheng_texture_2024**

involves passing the piano-roll as factor to guide the diffusion process.

Specifically this thesis investigates adding a certain type of control of rhythm. While generative models feature control of drums **Lan_Hsiao_Cheng_Yang_musicongen_2024**, tempo, note-density or meter **RÃijtte_figaro_2023**, **Huang_Yang_remi_pop_transformer_2020**, control using inner metric weight, or any complex symbolic characterization of rhythm is novel. However given the success of this characterisation in classification and music-identification tasks it is a promising control feature. The closest approximation is that of texture **Min_Jiang_Xia_Zhao_polyf** which has been deployed to some success in diffusion model using VAE disentanglement of harmony and melody.

2.7.2 Target Features

In order to extend LastMinuteGig with generated music containing appropriate cues we are targeting control of rhythmic patterns. The most simple way of targeting rhythmic patterns is through note density, implemented as a bar-level instruction of how many notes the following bar should contain. A second simple target feature is note variability (unique notes per bar over number of notes per bar). Both note-density and note-variability are simple time-varying features that are easily extracted, tokenized and introduced to the algorithm.

A potential approach, novel to music generation, would be control mechanisms using either spectral weight or inner metric weight. Here the weight profile is passed as a control mechanism either globally or on a per bar or per section level. *Comment: Introducing the control at a per bar level would be a bit odd since the idea of metric weight is about extracting weights not reflected in the symbolic grid, but it is a sensible way to introduce it to the model in bite-sized chunks able to reflect local change*. Inner metric and spectral weight weight has been successfully applied to dance music classification **Chew_Volk_Lee_Dance_metric_weight_2005**, meter detection **Haas_Volk_2016** and music retrieval **Volk_Garbers_VanKranenburg_Wiering_Grijp_Veltkamp_2009**. This suggests that it is a powerful feature for explaining various rhythmic aspects, which may make it a good guiding mechanism.

2.8 Implementation of control

Adding control to generative models can be split into three approaches. 1: Choice of architecture and training data, 2: fine-tuning approaches, and 3: external conditioning or guidance.

2.8.1 Control through architecture

The choice of architecture lends itself to different types of control. Transformers are next token predictors, that predict based on the prior sequence. The default training paradigm allows for

conditioning with a user defined musical (or audio) sequence. This is true for both audio based models such as MusicGen **copet2023simple**, Jukebox **Dhariwal_Jun_Payne_Kim_Radford_Sutskever_2020** and MusicLM **Agostinelli_Denk_Borsos_Engel_Verzetti_Caillon_Huang_Jansen_Roberts_Tagliasacchi_et_** as well as symbolic music models such as MMT **Dong_Chen_MMT_Kirkpatrick_2023**, Music-Transformer **Huang_Vaswani_Uszkoreit_Shazeer_Simon_Hawthorne_Dai_Hoffman_Dinculescu_Eck_201** and MusicBERT **Zeng_Tan_Wang_MUSICBERT_2021**.

Diffusion models are quite flexible compared to transformers, the same model can be used for inpainting, continuation and depending on the representation melody and accompaniment generation through masking.**Min_Jiang_Xia_Zhao_polyffusion_2023Rombach_Blattmann_Lorenz_Esser_C**

2.8.2 Control through training conditioning

Control can be added through training data. MusicGen**copet2023simple**, a recent text-to-music (audio) transformer is trained on 20000 hours of licensed music from shutterstock and pond5 [3] which includes textual descriptions and tags for genre, tempo, and other factors such as instrumentation. Control is achieved through the joint training of a text description and music. An example description is provided below:

Inspirational dramatic background music! Perfect for trailer, background, advertising, historical film, movie about superheroes, teaser and many other projects![4]

Text-based control, while user-friendly and accessible to non-musicians, is inherently vague. Levels of detail and choice of words vary widely by dataset, even with standardized tags such as genre and tempo. Specialized datasets such as MusicCaps **Agostinelli_Denk_Borsos_Engel_Verzetti_Caill** which contains 5500 text music pairs, with 10 seconds of music alongside a free text description and a list of aspect tags created by human experts still suffer substantially from subjectivity **Lee_Doh_Jeong_2023_subjectivity_musiccaps**.

For this reason, the creators of MusicGen **copet2023simple** add melody conditioning alongside text conditioning and train their model jointly with the chromagram of the melody alongside the text.

In MusicGenStyle **Rouard_Adi_Copet_Roebel_DÃl'fossez_musicgenstyle_2024** perform classifier-free guidance to add style conditioning to MusicGen. They train a music-style encoder that transforms a random subsample of a given reference audio track into tokens that are combined with the embeddings of the text-description. Both the style tokens and text tokens are provided as prefix to the model. The conditioner and the MusicGen transformer are trained

[3] https://www.shutterstock.com/music and https://www.pond5.com/

[4] https://www.pond5.com/royalty-free-music/item/95908062-inspiring-dramatic-epic-background-cinematic-music

jointly on the entire dataset.

The creators of FIGARO**RÃijtte_figaro_2023** enable fine grained control over instrumentation, note density, average pitch and volume on a bar-by-bar basis, in a symbolic music generator through joint conditioning while training.

2.8.3 Adding control through fine-tuning

Both the melody conditioning of MusicGen **copet2023simple** and the style conditioning of MusicGenStyle **Rouard_Adi_Copet_Roebel_DÃľfossez_musicgenstyle_2024** retrain the entire MusicGen model on the entire dataset which comes at considerable cost. Fine-tuning or transfer learning is another method through which models can be trained but at considerably smaller cost and using less data. This is particularly useful and widely used in the language domain to adjust large language models for niche use-cases, where the available data may simply not be sufficient to train a large model from scratch. In the examples of MusicGen and MusicGenStyle the availability of data was not a limiting factor since the controlling elements, melody and style can be inferred from the training data. However fine-tuning is also helpful for adding new control mechanisms.

MusiConGen **Lan_Hsiao_Cheng_Yang_musicongen_2024** is a fine-tuned variation of MusicGen which adds rhythm and chord control. They propose the jump-finetuning mechanism, where the original model with 1.5 Billion parameters and 48 self-attention layers, is split model into blocks consisting of 4 self-attention layers. They refine the first layer of each block, freezing the remaining layers. Additionally, they apply adaptive in-attention to the first 9 blocks, where the output of the transformer is augmented with copies of the original condition. As a result, only a quarter of the original parameters are tunable, which enables training on consumer GPUs on just 250 hours of music sourced from YouTube (as opposed to 20000 hours). In Coco-Mula **Lin_cocomulla_2024** the authors adjust a LLAMA adapter with just 4% of parameters, keeping all original MusicGen parameters frozen, and training only the adapter on a small dataset of 300 songs to add drum and chord conditioning.

MuseBarControl **Shu_Xu_Musebarcontrol_2024** is a fine-tuned version of MuseCoco**Lu_Xu_Kang_Yu_Xi**
which extends the global controls with fine-grained bar level control for music-generation. They compare several approaches. In the first they augment the prompt (which is generated from text) with additional tokens for bar-wise control of chords, and adjust the loss function to incoperate that. In the second approach they introduce two novel methods, first, they pre-adapt the new parameters (introduced by the lora adapter) to a separate classification task, an auxiliary task . The model classifies whether the a section of music corresponds with the control tokens, the body of the model is trained together with a classification head (which is removed after auxiliary task training. In the third step they introduce counterfactual loss where

the difference in negative log likelyhood conditioned on the original and changed attribute is maximized, which reinforces the models attention to the control. They find that the combination of the three strategies, pre-adaptation on a separate task followed with counterfactual-loss and prompt augmentation yields the strongest model.

2.8.4 Adding control through guidance

There are also other methods that do not involve any finetuning or retraining of the original model. Adding control through rule labels, or control tokens as in **RÃijtte_figaro_2023Lan_Hsiao_Cheng_Yan** does require some amount retraining, which is not always feasible, and adding many different types of control may deterioate the model. In these cases, guidance can be used to steer the model towards a certain output. In SMITIN **Koo_Wichern_Germain_SMITIN_2024** the authors use inference time intervention to guide a model towards a certain output with respect to certain goals. The goals explored are the presence of certain instruments (piano/drums/bass/guitar) in the mix and the quality/realism of the music. The authors train linear probes that learn to associate a state of the attention heads with the stated goal. Then the attention heads are steered in the direction of the probe's output, which increases the probability of the desired quality being present in the generated music.

In Diffusion models, the output is sampled over several steps, at each of these steps it is possible to intervene with guidance to direct the sampling towards a certain goal. In **Huang_rule_diffusion_2024**, each sampling step is repeated several times, and each time the sample that follows a set of rules most closely is chosen. ControlNet **Zhang_Rao_Agrawala_2023** adds spacial control to image generators allowing the guidance of image generation using sketches, poses, edges and depth maps without retraining. MusicControlNet **musicwellbeing_agres_2021** adapts this approach to music generation adding control for time varying factors, melody, dynamics and rhythm.

2.9 Evaluation

How to evaluate generated music is still an open research question. There are no standardized methods according to which evaluations happen**Yin_Reuben_Stepney_Collins_2023**. In the context of music generation there are several proposed frameworks to evaluate music. Typically we differentiate between subjective and objective evaluations.

For subjective approaches the methods vary widely **Xiong_Wang_ai_eval_methods_2023**. There are simple Turing-type evaluations that test how distinguishable generated and human written music are. Then there are subjective query metrics, where typically likert ratings of different parameters are collected.**Min_Jiang_Xia_Zhao_polyffusion_2023**. There are tourna-

ment style surveys, where the number of winning pieces are tallied for each approach.**Huang_Vaswani_Uszko** Finally there are expert evaluations (which can also include likert ratings) but also analysis of the produced score and musical structure. **Sturm_Ben-Tal_2016** These evaluations are often paired with statistical hypothesis testing. **RÃijtte_figaro_2023**

Automatic evaluation of generated music include model specific metrics and different musical metrics **Xiong_Wang_ai_eval_methods_2023**. Model specific metrics are generic evaluations of a models success to approximate training data, these will vary depending on the model and are not indicative of stylistic success. Examples of this are Negative Log Likelyhood **Huang_Vaswani_Uszkoreit_Shazeer_Simon_Hawthorne_Dai_Hoffman_Dinculescu_Eck_2018**, Root Mean Square Error **RÃijtte_figaro_2023** or Perplexity**RÃijtte_figaro_2023**. Musical metrics typically involve comparing a set of generated music to a set of real music, there are plenty of musical similarity measure techniques**Gurjar_Moon_similarity_2018** for a large variety of different use-cases i.e music retrieval, cover, genre and artist detection. A popular comparative metric is calculating the Kulback Leibler (KL) divergence between two datasets with respect to certain metrics i.e count of intervals or unique pitch-classes. However to obtain the divergence one has to select specific features that may only capture a subset of the desired properties. Similar issues arise with other distance metrics i.e cosine similarity, earth movers distance or maximum overlapping area.

Especially in the audio domain, additional AI models are often used for evaluation. MusicGen **copet2023simple** uses additional classifiers to generate labels for the music and calculates the KL-divergence between the generated labels. Additionally they calculate the Frachet Audio distance, a measure devised to calculate the plausibility of audio (for music enhancement purposes) compared to a large set of studio recordings**Kilgour_Frachet_2019**. Finally they use the CLAP-score which compares the corresponding text description to the latent representation of the generated audio, with text-description of the generated audio with the reference audio. **Elizalde_Deshmukh_Ismail_Wang_2023**

For this thesis we are interested in two factors, first the plausibility of the generated music, and second the success of the control. How the success of control is evaluated depends on what is controlled for, Examples of controlled parameters and how they are evaluated are as follows:

Note Density. (how many notes per bar). Root mean square error (RME) between generated vs target notes per bar. This is the approach to compare note density used in **RÃijtte_figaro_2023**

Note Variability (number of unique pitch classes, normalized by number onsets) - RME between generated and target.

Rhythmic patterns. Partial Similarity **Volk_Garbers_VanKranenburg_Wiering_Grijp_Veltkamp_2009**

between target and generated music:

Evaluating the plausibility of generated music is more complex, and there is no one method that has been proven superior. Possibly the plausibility of the music will be evaluated with a (small) subjective study. In this scenariou we would follow **Dong_Chen_MMT_Kirkpatrick_2023**, **Yu_Lu_Wang_Hu_Tan_Ye_Zhang_museformer_2022** and **Chen_Smith_Spijkervet_Wang_Zou_Li_Kong_D** and collect likert ratings on questions targeting Coherence Richness Arrangement and Consistency . Theese are would be paired with questions on musical background and preferences of the participant. For objective rating of the plausibility of the music we will follow the approach by **Min_Jiang_Xia_Zhao_polyffusion_2023** where the KL-divergence between the corpus of generated and a corpus of original music is calculated, most likely KL divergence over a set of extracted features, such as a pitch classes and chords. For a more complete list of different evaluation methods used on symbolic music generators, refer to the notes in the abstract **??**.

3. Research Questions

In the previous section we established the potential of generated music in serious games (section **??**), and the need of adequate time-varying controls over the generation process (section **??**) to reliably ensure the usability of the generated music in the context of MACT. We identified inner metric weight **??** as a promising target feature for control, due to its established link to rhythmic entrainment, and perceived and observed difficulty for a player to follow the music. Additionally it is extends prior work on controlled music generation with a powerful rythmic feature that is interpretable, concise and calculated symbolically. We collected crucial technical considerations for music generation including overall approach in section **??**, architecture in section **??**, representation in section **??** and tokenization in section **??**. Finally, we discussed promising methods of adding control to a model in section **??** and how to evaluate a model and its outputs **??**.

The focus of the thesis will be developing a model that generates a complete musical piece with time-varying controls for inner metric weight. The output of the model will be investigated for successfull integration of control, for player enjoyment and for interactive potential in the context of MACT.

1. Research Question 1: Can we effectivly control for inner metric weight in generated music?.
2. Research Question 2: Do shifts in metric weight provide a recognizable in-game cue?
3. Research Question 3: Does the generated music improve player enjoyment and engagement over Chalkiadakis**Chalkiadakis_2022** rule based system.

4. Methodology

To answer **RQ1** we develop, train and evaluate a model conditioned with inner metric weight. **RQ2** and **RQ3** are both answered using a user study that collects metrics on player interactions with the music alongside survey responses.

4.1 The Model

Sections **??** and **??** discussed the potential approaches to music generation with particular attention to state of the art approaches using deep learning. The developed model will be a transformer due to its ability to effectivley model sequences whether it be language or music as well as its potential to incooperate additional controls. We will be using a symbolic representation of music (as discussed in section **??**) due to its lightweight datasets and the ability to make changes and incooperate the output into a music production environment, enabling a cooperative co-composition process as opposed to replacing the composer. More specifically we will be using a representation similar to REMI+ (see section **??**), a representation that extends the standard MIDI-events with tokens indicating form, such as meter, bar-lines and note-duration. This representation will be compounded using byte pair encoding (see section **??**) to decrease the sequence length and improve the capability and efficienty of the model. For adding control (section **??**) we will focus on methods that don't require full training, specifically parameter efficient fine tuning and and guidance. This is more cost-effective and environmentally friendly. We will use the Lakh Midi Dataset **Raffel_2016**, a widely used open source and licensed dataset for symbolic music generation. This will help us avoid ethical pitfals around privacy and copyright. (section **??**). Finally. we are not attempting to add control by training custom large foundation models, rather we are using parameter efficient fine tuning or guidance (section **??**) to add contro. As a result we are using no or only a fraction of the model parameters, with substantially less need for data, computation, and energy. All relevant code, including code for training, configuration and data-preparation will be made available online alongside the model weights. Finally I use and extend an open-source model MusicLang in collaboration with it's maintainers. If successfull theese extensions will contribute to the MusicLang project and be widely available in a well documented and continously maintained ecosystem, with integrations into mainstream music production and composition software.

4.2 Developing the Model

4.2.1 Preliminary Experiments - proof of concept

The current methods of adding musical control to an existing model are poorly systemetized and rarely compared to each other. While this thesis does not aim to provide a systemic comparison and experimental evaluation of different control methods, some preliminary experiments are necessary to establish the best course of action. We use BassCraft a smaller model, and start by controlling for note-density (the number of notes in a bar), a feature that is more easily calculated, tokenized and verified as opposed to inner metric weight.

4.2.1.1 BassCraft - a tiny transformer model

To avoid wasting valuable compute resources and energy we perform the preliminary experiments on a smaller model: Basscraft. BassCraft is a small transformer model based on GPT2 **Radford_Wu_Child_Luan_gpt2_2019**. It has an embedding size of 256, 4 attention heads, 4 hidden transformer layers, and a total of 7 million trainable parameters. For contrast the target LLAMA 2 based model MusicLang has over 100 million trainable parameters. Basscraft is trained to generate a bassline to a provided piece of music using the lakh-midi dataset **Raffel_2016**. For training, songs with bass-lines are selected (based on the presence of particular MIDI-instrument channels). The tracks are partitioned into snippets between 1 and 16 bars long. The bass-lines are separated from the remaining track and matched as potential output.

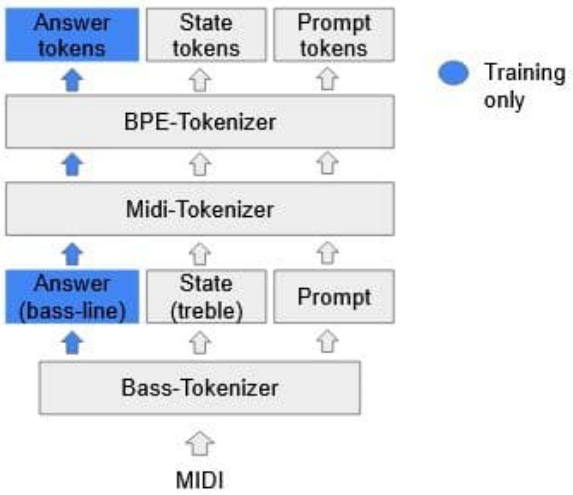


Figure 4.1: Preprocessing/tokenization of the original basscraft model

4.2.1.2 Fine-Tuning 1 - Vocabulary Extension

Vocabulary expansion is the process to add new vocabulary to a transformer model. MusicLang achieves it's extraordinary controlability similarly to FIGARO **RÃijtte_figaro_2023** us-

ing control tokens, that summarize features of the music that go beyond simply representing MIDI-events. In vocabulary extension it is critical to ensure that additional tokens do not overwrite or otherwise collide with the existing training, otherwise the benifit of using a priorly trained model is eliminated. Since we are using a BPE tokenizer, is difficult to simply add new tokens here, as it would require retraining the BPE tokenizer, which will transform the embedding layer making the entire model unusable. Instead we investigate whether there are unused tokens, and reassign them to our new control tokens. Theese new tokens will not be included in any of the coumpound tokens generated by the BPE process, which does negativley effect the sequence length.

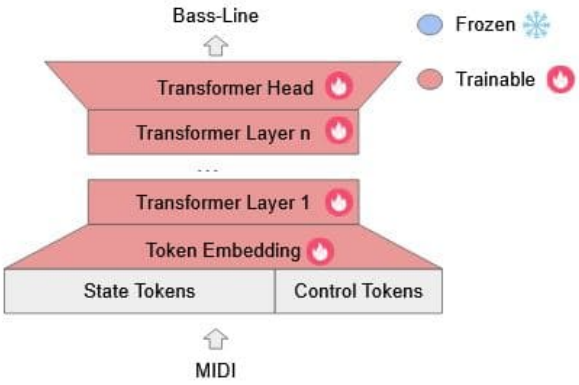


Figure 4.2: Vocabulary transfer with full fine tuning

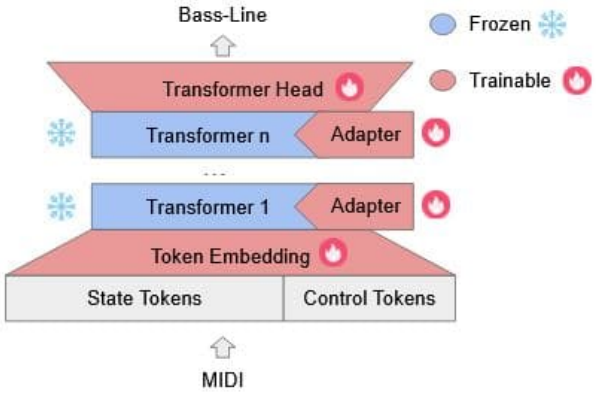


Figure 4.3: Vocabulary transfer with parameter efficient fine tuning

4.2.1.3 Fine-Tuning 2 - Integrating of control tokens

This approach is slightly different from the vocabulary extension, because it processes the additional control tokens as a paralell stream. This is similar to the the approach used in Coco-Mulla **Lin_cocomulla_2024**. The paralell stream of control tokens is inserted into the model after c layers (with bypasses that insert the embedding into every following layer), after passing through a custom learned embedding. The benifit of this method is that it doesn't require

editng the model's vocabulary.

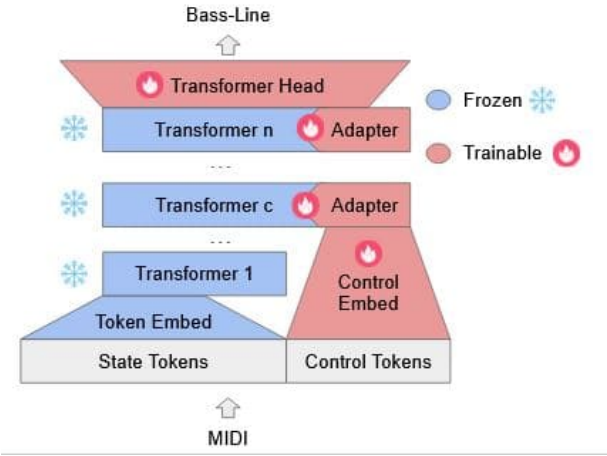


Figure 4.4: Integration of control tokens with parameter efficient fine tuning

4.2.1.4 Post-Hoc Guidance and other improvements

This is the approach used by SMITIN**Koo_Wichern_Germain_SMITIN_2024**. This type of sampling based guidance has been very successfull in diffusion models, however in transformers it produces mixed results **language_guide_rutte_2024**. One potential problem: This may be very difficult to implement. Both SMITIN and Rütte**language_guide_rutte_2024** only use one dimensional variables (probability of a concept being present or not).

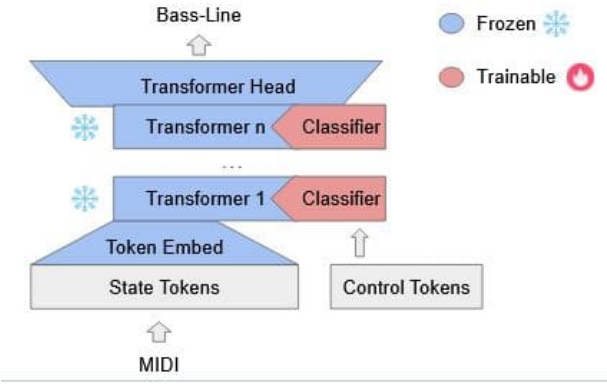


Figure 4.5: Integration of control using inference time interference

If these experiments are not successful, we can follow the approach of **Shu_Xu_Musebarcontrol_2024** and try additional training using auxiliary tasks and using a modified (counterfactual) loss function.

4.2.2 MusicLang - The foundation Model

After concluding the preliminary experiments and identifying a promising set of methods. We will proceed with the larger MusicGen model. MusicLang's core model is a transformer

based on LLAMA 2 trained on the Lakh Midi Dataset**Raffel_2016**. It can generate relativley long multi-track instrumental pieces, (a couple of minutes) with control for chord progression, instrumentation and range. Additionally it can generate interpolations and continuations of a user provided piece. It is trained on an extended vocabulary of tokens similar to REMI[1] with additional tokens detailing harmonic structure and voice characteristics i.e instrumentation or range (see figure **??**). This base vocabulary is additionally extended using the BPE tokenizer. The goal of this section is to extend MusicLang with control for inner metric weight.

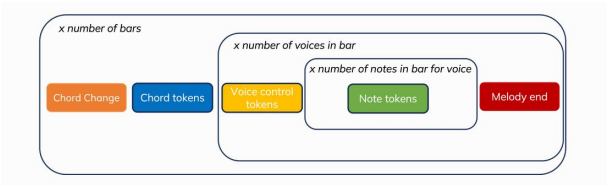
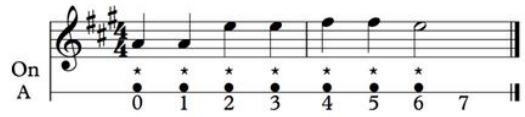


Figure 4.6: MusicLang tokenisation on a high level

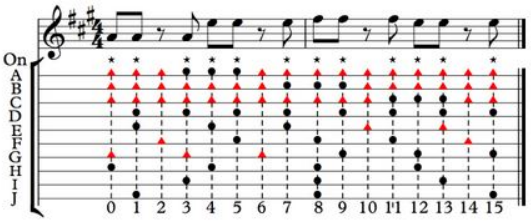
4.2.3 Using inner metric weight as controlable feature

Inner metric analysis (IMA) creates metric weight profiles which we use as guiding feature passed to the model bar by bar, which allows us to induce shifts in metric weight in the output. For this we use the globally smallest available note grid of the lakh-midi dataset g_min. Theese distributions are normalized and provided to the model as vectors of length g_{min}. The model then learns embeddings of the distribution alongside positional embeddings **Lin_cocomulla_2024**. Theese embeddings are then added to the model starting from a certain stop. For inference the user can provide a reference track from which the inner metric profile is extracted and then passed to the model.

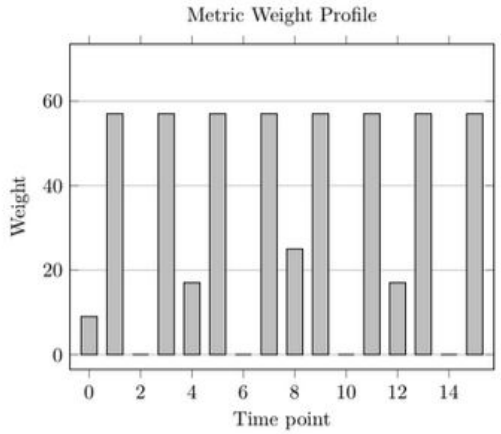
[1] https://musiclang.github.io/tokenizer/



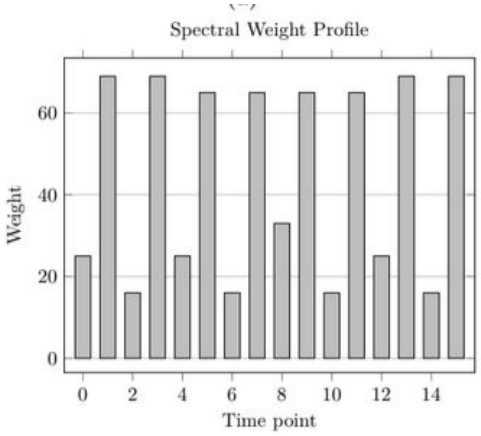
(a) Single local meter and its pulses (A) from the melody "Twinkle, Twinkle Little Star"



(b) 10 local meters and their pulses produced by IMA of a syncopated variation of "Twinkle, Twinkle Little Star"



(c) Metric weight profile of syncopated "Twinkle, Twinkle Little Star"



(d) Spectral weight profile of syncopated "Twinkle, Twinkle Little Star"

Figure 4.7: Visualisation of metric weight analysis **Bemman2024**

4.2.4 Evaluating Control-Effectivness

As discussed in section **??**, calculating whether or not the control is effective, depends on the controlled feature, but can happen automatically.

For the preliminary experiments targeting note density this can be done as follows, if note density is measured as a categorical variable i.e low, medium, high, the error would be calculated similarly to a multilabel classifier, where the predicted is label is the note$_d ensity of the generated music, and the tru$ $density is continuous(i.e notes per bar), then the error would be calculated as mean square error.$

$error_{continuous} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_{generated} - y_{prompt})}$ Inner metric weight analysis generates metric weight profiles, which we use as guidance mechanism. Following the approach by **Bemman2024** we can compare the the generated and the target rhythmic weight profiles using chi-squared distance. Given target distribution T and generated distribution G the distance is given by

$$D = \sum_{i=0}^{n-1}\left(\frac{(T_i - G_i)^2}{T_i + G_i}\right)$$

4.3 User Study

RQ2 and **RQ3** are evaluated through a user study. The goal is too recruit $n = 20$ participants to complete an interactive test of interactability and a survey comparativley evaluating the generated music.

4.3.1 Interactive Study

The interactive study aims to evaluate the fittness of the controlled generated music from an MACT perspective. Specifically, in the MACT protocoll used by Chalkiadakis **Chalkiadakis_2022**, the patient is supposed to listen to changes in the music, and change their playing. In our simplified interactive sessions the player is asked to hit a button whenever they hear a change in the music. The button-presses are registered alongside the starting time, tempo and registered changing points (determined from the prompt). Finally, the timing of the button presses and registered changes is correlated with each other. High correlation indicates that the participants noticed the changes.

4.3.2 Survey

The survey will compare the listening experience of different symbolic generative models including our model, FIGARO **RÃijtte_figaro_2023** and Polyfussion **Min_Jiang_Xia_Zhao_polyffusion_2023** as well as the improved rule based generator of "Last Minute Gig" **Chalkiadakis_2022Schlette_2022**. The questionnaire will include questions on personal details of the participant, gender, age, educational background, musical experience and gaming experience.

4.4 Thesis Timeline

Figure 4.8: Thesis project plan

5. Appendix

5.1 Comparing Tokenization lengths

Pretokenized Sequence Lengths - Mean: 47976.146146146144, Median: 43692.0, Std Dev: 23159.918513030065
Tokenized Sequence Lengths - Mean: 14519.587587587588, Median: 13159.0, Std Dev: 7197.064427699848

Applying the BPE tokenizer results in sequence lengths roughly 1/3 of the original sequences. The sample is taken from 1000 random music pieces from the lakh-midi dataset.

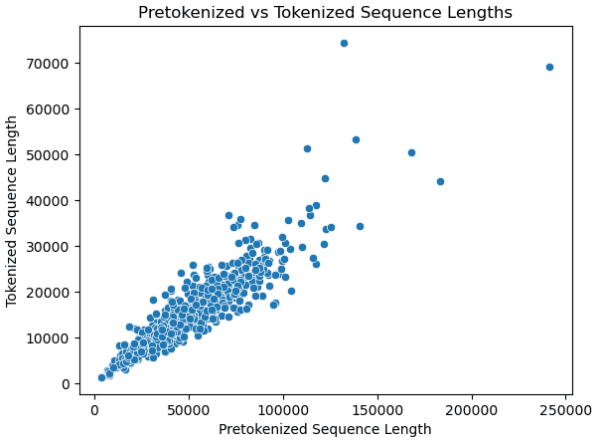


Figure 5.1: Scatterplot of Pre-tokenization vs bpe-tokenization sequence lengths

5.2 Integration into LastMinuteGig 2 months

This project is motivated by extending the music engine of LastMinuteGig. **Chalkiadakis_2022**. The application currently employs a very simple algorithm. I will outline it below.

- 1: Randomly choose key, rhythmic pattern, chord progression and tempo from a pool of possible values.
- 2: Play corresponding percussion audio clip (depends on tempo and rhythmic pattern)
- 3: When user plays the button trigger the guitar sound of the current chord.
- 4: After 8 bars - make a random change (to rhythm, tempo, pause)

This music engine will be changed in the following way. Most of the generative process will be done asynchronously.

- 1: Create $n = 100$ musical pieces with different control settings
- 2: For each piece
- 3: Extract chords from prompt (assuming high control success) - save mapping $m1$
- 4: Extract change of rhythmic pattern (or tempo or meter or instrumentation since these are already controllable in the musiclang model) - save mapping
- 5: Render symbolic music to audio

During game play

- 1: Load random audio clip and mappings.
- 2: Play associated guitar chords when player taps.
- 3: Register changes in tapping on musical changes.
- 4: Repeat when audio clip has finished playing.

Name
DeepBach (2017) **Hadjeres_Pachet_Nielsen_2017**
FolkRNN (2015) **Sturm_Ben-Tal_2016**
MT (2018) **Huang_Vaswani_Uszkoreit_Shazeer_Simon_Hawthorne_Dai_Hoffman_Dinculescu_Eck_2018**
MidiNet (2019) **midinet**
Polyfussion 2023 **Min_Jiang_Xia_Zhao_polyffusion_2023**
FIGARO 2023 **RÃijtte_figaro_2023**
MMT 2023 **Dong_Chen_MMT_Kirkpatrick_2023**
OMT 2020 **Huang_Yang_remi_pop_transformer_2020**
MuseNet 2019 **Christine_2019**
MMM 2020 **Ens_Pasquier_2020_MMM**
Sympack 2024 **Chen_Smith_Spijkervet_Wang_Zou_Li_Kong_Du_2024**
MuseCoco 2023 **Lu_Xu_Kang_Yu_Xing_Tan_Bian_MuseCoco_2023**
MBC 2024 **Shu_Xu_Musebarcontrol_2024**
Museformer 2022 **Yu_Lu_Wang_Hu_Tan_Ye_Zhang_museformer_2022**
NTT **Ryu_Dong_nested_2024**
FTG **Zhu_Liu_Jiang_Zheng_texture_2024**
Fader Nets
NDRD **Huang_rule_diffusion_2024**

Table 5.1: Overview of music generation models, their architectures, and control mechanisms. Empty columns indicate missing information on fine-grained control, multitrack capabilities, dataset, and evaluation methods.

5.3 Notes Comparing Symbolic Music Generators and their evaluation methods

- DeepBach **Hadjeres_Pachet_Nielsen_2017** - RNN - inpainting. Evaluation Turing
- FolkRNN **Sturm_Ben-Tal_2016** - RNN - control for meter and mode - Expert Evaluation + Performance Practice
- MusicTransformer **Huang_Vaswani_Uszkoreit_Shazeer_Simon_Hawthorne_Dai_Hoffman_Dincules** - Transformer - Evaluation: Subjective - Tournament Style between different generated and natural music. Objective: Validation NLL
- MidiNet**midinet** - GAN - Control for Chords/Priming melody. Human (how pleasing, how real, and how interesting)
- Polyfussion **Min_Jiang_Xia_Zhao_polyffusion_2023** - Diffusion Model - supports inpainting, interpolation, melody/accompaniment generation, control for chord progression, texture. Subjective Evaluation Questionnaire for naturalness, creativity, musicality. Objective Control success.
- FIGARO **RÃijtte_figaro_2023** - Transformer Model - bar-wise control for chords, instrumentation, time-signature, note-density, mean-pitch. Evaluation: Perplexity (improvement over NLL for sequences of different length), Discription Fidelity (i.e accuracy in regards binary controls instruments, chords, time-signature). Macro Overlapping Area -

comparison of feature histograms. Normalized Mean Root Square Area for note-density. Cosine similarity for chroma (melodic) and groove (rhythmic) feature vectors. Ablation study - effect of turning off controls. Extensive Subjective evaluation: 7569 comparisons by 691 participants - tournament style.

- Multi Track Music Transformer (MMT) **Dong_Chen_MMT_Kirkpatrick_2023** - Transformer - Instrument control. Comparison of different tokenisation techniques, REMI+ and Compound Tokens. Compound tokens are more condensed and the generated samples are longer, achieves significant speedups and reduces memory usage (2.6 * MMM, or 3.5 * REMI+). Objective evaluations: Inference time, pitch class entropy, scale consistency, groove consistency, Human evaluation (90 comparisons by 9 participants) on Coherence Richness Arrangement Overall.

  Additional: Analysis of self attention as explanation avenue, which notes are most important.

- REMI pop music transformer - **Huang_Yang_remi_pop_transformer_2020** - Transformer - continuation, control local control over chord and tempo
- MuseNet **Christine_2019** - Transformer -,Instrument control, style control. No evaluation, only showcase.
- MMM **Ens_Pasquier_2020_MMM** - Transformer model - inpainting, instrument control, note-density. Introduce novel representation No Rigorous Evaluation
- SymPAC **Chen_Smith_Spijkervet_Wang_Zou_Li_Kong_Du_2024** - Transformer - Control for Chords, structure, instrumentation, single notes. Train with both symbolic and transcribed audio data. Fine grained control. Constrained generation with a Finite State Machine. Comparison of three separate models trained on three datasets. **Evaluation** of controlability with KL-divergence on different controlled features over chords, structure, and individual notes. - KL divergence decreases with dataset size. 800 samples are generated and compared against a validation set of 3000 songs. Subjective evaluation (12 expert participants MIR researchers and music producers) on parameters of Coherence Richness Arrangement Structure
- MuseCoco **Lu_Xu_Kang_Yu_Xing_Tan_Bian_MuseCoco_2023** - Transformer - Control via text for following attributes instrumentation, ambitus, rhythm (intensity and "dancability"), number of bars, time signature, key, tempo, duration, artist, emotion, genre. All of these controls are global controls, a description is converted into a list of attributes which is then used for generation. Combination of many datasets. Creation of text descriptions from attribute list with data from the dataset, and ChatGPT. Evaluation: Objective - text to attribute list. Subjective evaluation 19 participants with at least basic music knowledge - questions to Musicality, Controlability (adherence of sample to music description), Overal Impression. Comparison with LLM generated music (with no special training for music generation)
- MBD **Shu_Xu_Musebarcontrol_2024** Extension of MuseCoco for time varying chord controls. Counterfactual Loss and Auxiliary task training improve controllability.
- Museformer **Yu_Lu_Wang_Hu_Tan_Ye_Zhang_museformer_2022** - Transformer - No controls. Goal improve long-term structure with fine and coarse attention. Captures structure well. Objective Evaluation: Perplexity: prediction accuracy of next token, Similarity Error the error between the similarity distribution of training data and generated music Subjective Evaluation 10 Participants Musicality, Long Term Structure, Short Term Structure. Ablation study - evaluate objective effect of coarse and fine-grained attention. + Case study and detailed look at model.

- NMT **Ryu_Dong_nested_2024** - Transformer improve longterm structure and reduce sequence length through compound tokens. Application to both symbolic and audio tokens. Cross attention vs self-attention comparison Evaluation: FAD, CLAP, KL and NLL over audio tokens. NLL over symbolic tokens. Subjective Evaluation Coherence, Richness, Consistency, Overall 29 participants. 8 selected prompts, 4 different continuations with REMI, Compound Word and 2 NMT variations.
- FTG - Fine Grained Texture Control - Diffusion - control over texture, rhythm and chords.
- Fader Nets**Tan_Herremans_2020** - VAE - Control over rhythm, arousal, Idea: develop a "fader" representing a high-level abstract feature i.e arousal. Arousal is disentangled using a VAE into lower level features (i.e rhythmic density).

  Evaluation of the influence of latent features is on generated (style transfer) music.: Consistency, Restrictiveness (one latent dimension does not influence other musical features), Linearity (linear change in latent feature - linear change in musical feature) Subjective listening test to indicate success of arousal shift -> 48 participants, evaluate agreement with arousal direction.
- NDRD Symbolic Music Generation with Non-Differentiable Rule Guided Diffusion. Guidance of diffusion sampling with non-differentiable rules.