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

Second examiner:
Peter van Kranenburg

Candidate:
Efraim Dahl

March 1, 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 RhythmLang, a transformer based music generator, which will evaluated for interactive potential in the context of MACT.

Contents

1. Introduction

Music is a cornerstone of modern video games. Music evokes emotional responses, triggers memories, can direct attention, and improves 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.[1] 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 [2]. Ideally, MACT is a personalized experience with dynamic adjustments considering a patient's abilities and preferences. In the context of gamified self-directed therapy, this is crucial to keep patients engaged and progressing.[1] 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 few years, music generation has grown from a primarily academic endeavor to a billion-dollar industry. Large technology companies are exploring music generation with foundation models such as Meta's MusicGen[3] or Google's MusicLM [4]. The commercial startup music generator Suno is valued at 500 million dollars. A German producer used Udio to generate a chart-topping song [5][6]. This breakthrough is powered by advances in language modeling, bringing about Large Language Models such as GPT-4 or LLAMA3 applied to music.

In games, generative music is promising to cater to different user preferences and enable a richer, more varied musical experience than conventional approaches to adaptive audio. Video games can include hundreds of hours of gameplay and branching storylines. Adapting the music to individual preferences adds additional complexity. 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 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 or information on genre, style, and instrumentation. Control can also include time-varying, composable musical parameters such as a melody, chord progression, or musical structure. Generative models are often controlled with human-composed

[1]This type of therapy is not aimed at replacing traditional guided therapy. Ideally, it is used as a supplement, providing relief in situations where regular treatment is not immediately available

music. They can continue provided music, interpolate between, or inpainted - generating accompaniments, melodies, or additional musical lines. In LastMinuteGig[2] the music periodically provides stimuli, a noticeable change in the music, based on which the user changes their playing, specifically their rhythm. One way to achieve this stimulus in generated music is by adding a 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 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 resources as the equivalent model without controls.

A more economical method of adding control to a deep learning generative model is fine-tuning and post-hoc guidance. In fine-tuning (also 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. Typically fine-tuning uses considerably fewer resources and data than training a model from scratch. In post-hoc guidance, a generative model is adjusted while running to increase the likelihood of the output to fit the desired constraints. The goal of this thesis is to steer a pre-trained generative model to generate music fit to a MACT context, without training a large model from scratch but using efficient fine-tuning to introduce new control mechanisms to a pre-trained model. For this, We will first explore and evaluate different methods of adding control using a small generative music model. We will then apply the most successful techniques of adding control 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 evaluate the interactive potential and the potential benefits in enjoyment and player engagement of using AI-generated music over more simple generation processes [2] in a game such as Last Minute Gig.

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[7]) were justified as methods to inspire composers and music makers, including enabling novice composers to write music. Composer David Cope[8] 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" [9]. Many current and past efforts highlight music generation as a process that assists composers. In DeepBach[10], 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[11], 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[12] and Ben-Tal's musical dialog system[13].

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

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

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[14], [15]. 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 [1] 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[14]. Furthermore, AI music generators currently lack proper integration into video game environments and engines[15].

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[16]. In music therapy, music can be used for emotion regulation, motivation, adherence, motor coordination, rhythmic entrainment, and facilitating social interactions[17]. Musical attention control training has been shown to help individuals with Parkinson's[18], ADHD[19], autism[20], and psychosis[21] improve their mental capabilities for selective and switching attention. Serious games have the potential to supplement music therapy. "Last Minute Gig"[2] 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[22] attempted to address these issues by introducing 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. [6] [23]. However, AI companies are facing backlash 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[24]. 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 [25]. While leakage may raise privacy concerns in other generative models such as speech and image-generators [26] 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 [27] 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 [28], 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.[29] 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 [27] 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 [30], 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 [31]. 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 [32]. The performance of current large language models often scales with simultaneous increases in model size, training data, and computation time.[33] Each of these three factors requires considerable resources. Music generation is no exception. In a recent ISMIR publication [34] 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 inference. 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.

2.3 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 [35] and providing musical material themselves, such as in the 1957 Illiac suite [36]. 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 [36][37] to evolutionary algorithms [38] to feed-forward [39], recursive [40], and convolutional neural networks [41]. Some composers in the classical tradition, such as Iannis Xenakis [42] and David Cope[8] use computer algorithms in their creative work. In the 2010s a small ecosystem of commercial generative music startups such as Jukedeck, PopGun, and Ampermusic [43] 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 [44], DeepBach [10] and FolkRNN [45]. With the development of the transformer architecture in 2017 [46], large technology companies start experimenting with music generators including OpenAI's Jukebox [47] and Musenet[48], Meta's MusicGen[3] and Google's MusicLM [4] 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[6], 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 2.2), it is worthwhile considering alternative approaches to music generation. A recent study [49] performs a comprehensive listening survey comparing neural net and non-neural net systems. The top-performing systems a Markov Model - MAIA Markov [50] and a deep learning system - MusicTransformer [51] perform similarly well in the listening study. The choice of one of the earliest transformer-based music from 2018 [51] 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 [52] to Fux's Gradus ad Parnassum [53] 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 compositions, the first and second movements are generated following the rules of first species counterpoint [53], 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 2.1). Rule-based generation can be highly complex, such as CHORAL [37] (for which the developer also built a custom programming language), which encodes over 300 rules to realize bach-style chorales from a given melody.

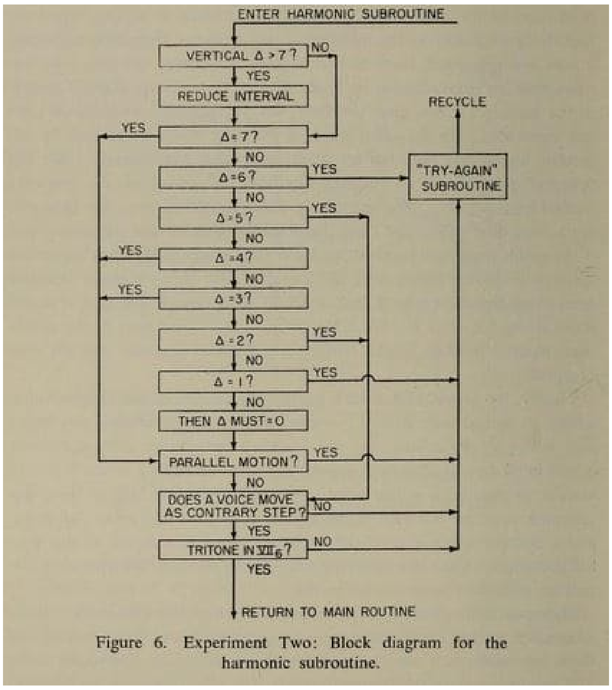


Figure 6. Experiment Two: Block diagram for the harmonic subroutine.

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" [54]. Markov chains can be nested or constrained for more complex interactions [50]. In Hiller and Isaacson's [36] 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 [12], but training over a whole corpus is also viable. Other systems configure Markov chains to take additional inputs into account, such as Allan, [55] 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 [56]. In Morpheus, [57] 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 [38] 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. [39] 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,[40] 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 [58], 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 [45], 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. [59]

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.[46] They keep the self-attention mechanism already deployed in LSTMs for seq2seq tasks [60] 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 2.3. Often, tasks relating to sequence understanding, such as music classification, use an encoder-only architecture. The BERT series of language models [61] and music understanding models such as MusicBERT[62] are examples. On the other hand, sequence generation tasks often employ a decoder-only architecture, this includes the GPT-series [63] and many music generators such as MusicGen[3]. 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.

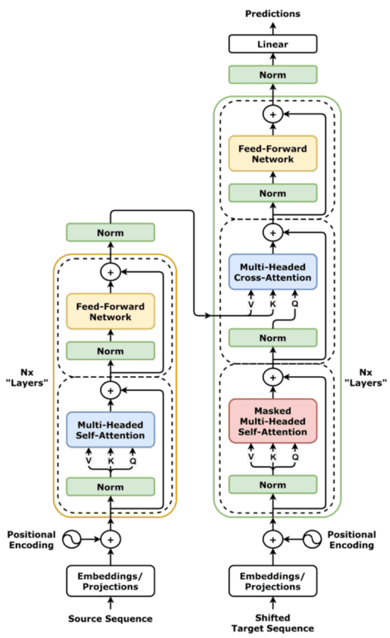


Figure 2.3: Schema of the full transformer encoder-decoder architecture

[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 [64], and StableAudio [65] 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 [66] 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.

2.5.3 Token Sampling

Transformer models are next token predictors, essentially large classification model that choose the next probable token given the prior sequence. There are different methods of how the transformer chooses the next token. Both the input and output have the size of the vocabulary which depends on the tokenisation chosen. Typically the last layer is a so called softmax layer, which transforms the output vector into an equally sized vector of probabilities adding up to one. [63]. This is calculated as follows.

$$\sigma(z_i) = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)} \tag{2.1}$$

This last layer then forms the basis of differnet sampling methods. The most simple method is greedy sampling, simply choosing the most probable token. More typical sampling methods are probabilistic, where the next token is sampled from the vector of probabilities. The temperature parameter T in the above equation, can often be controlled by the user to shape the probability distribution. At a high temperature the probabilties are distributed more uniformly across the vector, the output becomes more noisy. At a low temperature only the highest probability tokens remain, making the output more deterministic. A common variant is top k sampling, where only the most probable k tokens are taken into consideration. For top p sampling, the most probable tokens are added up until hitting a target percentage p.

2.6 Deep Learning and Symbolic Music Generation

There are various different approaches to symbolic music generation described that use deep learning. However there are considerable differences in the implementation such as choice of architecture and representation that influence what features can be controlled for. The following table summarizes theese key characteristics across a selection of approaches that inspire our approach. For a comprehensive survey of representation, tasks, and evaluation methods used in symbolic deep learning music generation see [67].

Name	Architecture	Control	TV	MI	Dataset	Representation
FolkRNN 2015 [45]	RNN	meter, mode	No	No	TheSession[68]	REMI-Like
DeepBach 2017 [10]	RNN	Inpainting	Yes	Yes	JSB-chorales[69]	Midi-Like
MusicTransformer (2018) [51]	Transformer	-	No	No	Maestro[70], JSB-chorales[69]	Midi-Like
MuseNet 2019 [48]	Transformer	instrument, genre	Yes	Yes	Mastro[70], CX[71], BM[72]	?
MidiNet 2019 [44]	GAN	chords, melody	Yes	Yes	HTPM[73]	Midi-Like
Fader Nets 2020[74]	VAE	arousal	No	No	Maestro [70]	Custom
MultitrackMusicMachine 2020 [75]	Transformer	inpainting, instrument, note-density	Yes	Yes	Lakh MIDI[76]	MMM
PopTransformer 2020 [77]	Transformer	chords, tempo	Yes	No	Custom	REMI
Museformer 2022 [78]	Transformer	-	No	Yes	LakhMIDI [76]	REMI-Like
Polyfussion 2023 [66]	Diffusion	inpainting, texture	Yes	No	POP90 [79]	Piano-Roll
FIGARO 2023 [80]	Transformer	chords, instrument, meter, note-density	Yes	Yes	Lakh MIDI[76]	REMI+
MMT 2023 [81]	Transformer	instrumentation	No	Yes	Lakh MIDI[76], SOD [82]	Midi-Tuple
Sympack 2024 [83]	Transformer	chords, structure, notes	Yes	Yes	Lakh MIDI[76], [84], Custom	-
MuseCoco 2023 [85]	Transformer	various	No	No	Custom	REMI-Like
MBC 2024 [86]	Transformer	chords	Yes	No	POP90[79]	REMI-Like
NTT 2024[87]	Transformer	-	No	Yes	Lakh MIDI[76], POP90[79], SOD[82]	Compound
FTG 2024[88]	Diffusion	texture, rhythm, chords	Yes	No	POP90 [79]	Piano-Roll
NDRD 2024[89]	Diffusion	chord, pitch, note-density	Yes	No	Maestro [70], POP90[79]	Piano-Roll

Table 2.1: Overview of symbolic, deep learning based music generation models, their architectures, and control mechanisms, fine-grained control, multitrack capabilities, dataset, and evaluation methods. A more complete Overview is found in [67]

2.7 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.

2.7.1 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.[67][83] 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.

2.7.2 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 [47] uses a variational autoencoder[90] with a discretizing bottleneck (VQ-VAE) to create tokens from audio. Musicgen [3] tokenizes audio using the previously developed Encodec model for audio compression which similarly to VQ-VAE, learns a highly condensed discrete representation of audio [91]. These condensed encodings are crucial for generative modeling on audio. In symbolic music depending on the representation a similar teqnique is used to encode piano-rolls for symbolic music diffusion. [66][88]).

2.7.3 Symbolic Tokenisation

Symbolic music can be represented in different ways, as text (i.e the ABC notation), piano-roll, graph and sequence. The most widely used representation however is the event-based MIDI-representation and this is also reflected in the tokenisation techniques. In symbolic music tokenizers are feature extracters, and they extend extends the standard MIDI-vocabulary with additional tokens that help models better capture different aspects of the music.[92]. In table **??** the tokenisation approach of different symbolic models is summarized.

MIDI-Like tokenization closely emulates the MIDI vocabulary, translating a MIDI file into a single stream of tokens such as note-on, note-off. The **MMM** tokenizer is designed to aid track inpainting. **REMI** [77] tokenization expands on MIDI-based tokenization with tokens for duration, bar and position, designed to help capture recurring musical patterns. **REMI+** [80] tokenization extends **REMI** with an instrument token to better encode multi-instrument tracks. 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 of tokens can become very long, which adversely affects the model[67]. Ongoing attempts are made to condense token sequences, such as compound words or nested tokens.[87]. Dong et al.[81] combine six different MIDI-like events (type, beat, position, pitch, duration, instrument) into single tokens. Hsiao et al.[93] 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) [94]. BPE is used widely in language modeling, including the GPT series of models[63] and has been successfully applied to symbolic music generation.[95] 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 2.4, 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.

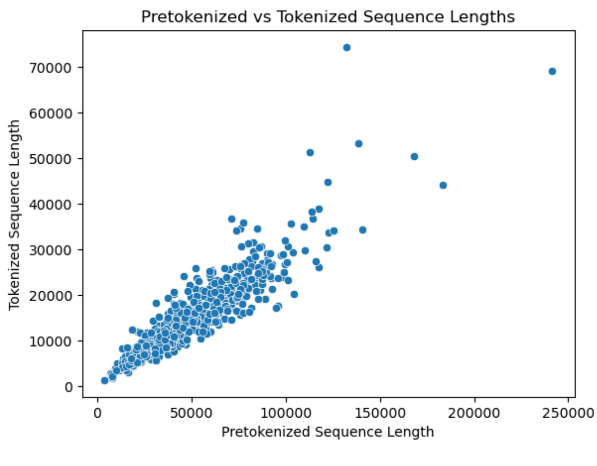


Figure 2.4: Scatterplot of sequence length before and after BPE tokenisation

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

2.8 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. One can differentiate between global and local features (discussed in the appendix 5.2) [96], deep vs surface-level features [97], high-level vs low-level features [74] and global vs fine-grained or time-varying features.

In the context of this thesis we differentiate between global features, and time-varying features [80]. What is time-varying or global is highly context dependent, a piece may have one time-signature and tempo as is assumed in [85] or it may vary over time as is suggested in [80] or [77]. Other time varying controls could be chords [80][98][99][66], melody [3][66] or texture [66]. 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.

2.8.1 Rhythmic control

The types of control exercised over rhythmic components varies by representation as discussed in **??**. In CocoMulla [100] generated audio is controlled with drum tracks and a piano-roll. Similarly in JASCO[101] drum-audio is used for conditioning. In MusicConGen.[99]control for rhythm is added through tracking beats and downbeat. MusicControlNet[98] adds beat and downbeat conditioning to an audio diffusion model. For symbolic systems control of tempo and meter is relatively common [80], [77], [85]. Time-varying control over rhythm is often deployed through note-density (both vertical and horizontal)[80],[89]. Another approach [88] involves passing the piano-roll as factor to guide the diffusion process. In Polyffusion[66] Min et al successfully encode texture, that is disentangled into harmony and rhythm using a pre-trained variational auto-encoder [102]. Herremans et al [74] control rhythm through the high-level feature arousal, that is disentagled using a variational autoencoder into rhythm and note-density.

2.8.2 Inner Metric Analysis

Using a variational auto-encoder to disentangle rhythmic descriptions into lower level features is an option, but introduces additonal complexity of training a second model, and combining both to be used for inference. Instead, we use Inner Metric Analysis (IMA) to create metric profiles of a sequence, and use theese profiles as guiding features. Inner Metric Analysis identifies strong and week pulses and their periods within note onset in symbolic music.**<empty citation>** It is used to identify *local meters* as opposed to *outer meter* given by time-signatures, which can be useful in the study of syncopation [103][104]. IMA is also used in the classification of dance-music [105], automatic detection of meter [106] and music retrieval [107].

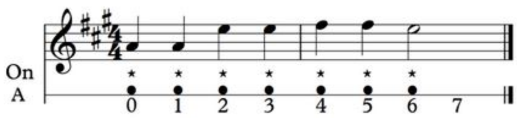
A local meter is a set of evenly spaced onsets with a minimum length of three, not able to be extended forward or backwards in time, and not contained within any other local meter. (see figure 2.5). Let $M(l)$ be the set of local meters with a length of at least l. The parameter p is variable and determines how much the length of a local meter influences its weight. Intuitivly, longer and more established local meters should carry more weight. This is given by k_m^p. The weight of an onset $W_{l,p}(o)$ is defined as the weighted sum of the local meters. The metric weight is calculated as follows [104].

$$W_{l,p}(o) = \sum_{m \in M(l): o \in m} k_m^p \tag{2.2}$$

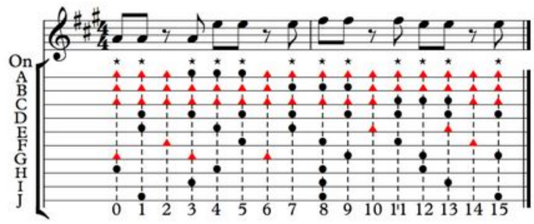
Spectral weight is a variation of the metric weight, that consider the extension of each local meter: $ext(m)$. The red triangles in figure 2.5 are and example of extensions. The calculation is similar, but it assigns a weight to each time point (instead of only to onsets) and considers the extensions. This feature is less sensitive to local changes.

$$SW_{l,p}(t) = \sum_{m \in M(l): t \in ext(m)} k_m^p \tag{2.3}$$

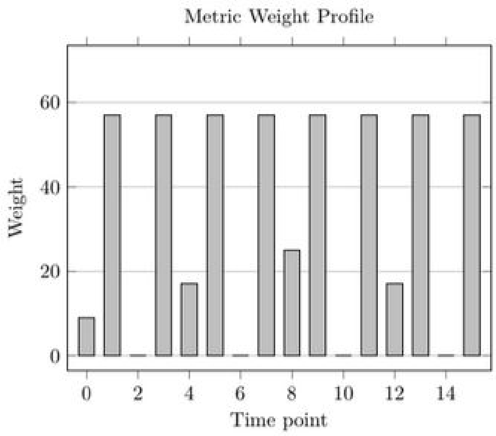
In the context of MACT, inner metric analysis is interesting because it can indicate rhythmic complexity, which in turn influences ease of tapping along/following [104] and rhythmic entrainment. This could be extended to generate music that is more difficult to follow to allow for difficulty adjustment in the context of an MACT game.



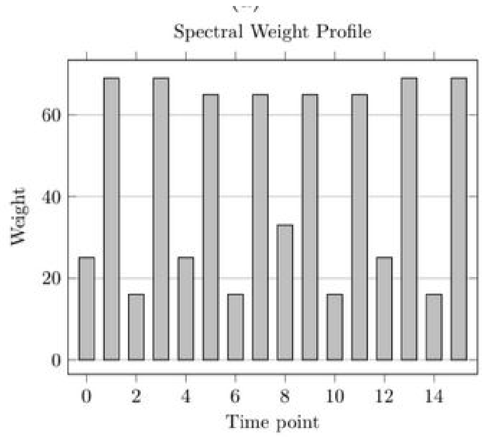
(a) Single local meter and its pulses (A) from the melody "Twinkle, Twinkle Little Star"



(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 2.5: Visualisation of metric weight analysis [103]

2.9 Implementation of control

Methods of enabling control in a generative models can be split into four approaches. 1: Choice of architecture, 2: training data, 3: fine-tuning and 4: post-hoc guidance.

2.9.1 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 control/conditioning with a user defined musical (or audio) sequence. This is true for both audio based models such as MusicGen [3], Jukebox [47] and MusicLM [4] as well as symbolic music models such as MMT [81], MusicTransformer [51] and MusicBERT [62]. 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.[66][108]Transformers have to be explicitly trained for theese tasks.

2.9.2 Control through training data

Joint training of a model with the desired control, is the most common and robust method of enabling control in a generative model. MusicGen[3], 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. This is also true of specialized music-text datasets such as MusicCaps [4],[109]. For this reason, the creators of MusicGen [3] add melody conditioning alongside text conditioning and train their model jointly with the chromagram of the melody alongside the text.

In MusicGenStyle [110] 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 jointly on the entire dataset. The creators of FIGARO[80] 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.

2.9.3 Adding control through fine-tuning

Both the melody conditioning of MusicGen [3] and the style conditioning of MusicGenStyle [110] 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-

[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

tuning may achive additional control at a lower cost.

MusiConGen [99] 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 [100] 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 [86] is a fine-tuned version of MuseCoco[85] which extends the global controls with time-varying 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.

2.9.4 Adding control through guidance

There are also other methods that do not involve any finetuning or retraining of the original model. Adding control with additional model inputs does require at least some amount retraining, which is not always feasible, and adding many different types of control may deterioate the model performance. In these cases, guidance can be used to steer the model towards a certain output. In SMITIN [111] the authors intervene at inference time, while the trained model is generating, to guide MusicGen[3] towards a certain goal. They explore guidance for ensuring the presence of certain instruments (piano/drums/bass/guitar) and to increase the quality/realism of the generated audio. For this, the authors train linear probes that learn to associate the state of each transformer layer in the networke with the goal (i.e drums being present in the output). Then the attention heads are steered in the direction of the probe's output, which increases the probability of the desired quality (drums) being present in the generated music. Guidance of transformer models has been explored in other contexts [112], i.e influencing truthfullness, humor, and appropriatness, with mixed results.

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 [89], each sampling step is repeated several times, and each time the sample that follows a set of rules most closely is chosen. ControlNet [113] adds spacial control to image generators allowing the guidance of image generation using sketches, poses, edges and depth maps without retraining. MusicControlNet [98] adapts this approach to music generation adding control for time varying factors, melody, dynamics and rhythm.

2.10 Evaluation

How to evaluate generated music is still an open research question. There are no standardized methods according to which evaluations happen[49]. In the context of music generation there are several proposed frameworks to evaluate music. Music generation literature often distinguished between "objective" and "subjective" evaluation. Despite what the name suggest objective evaluation of generated music does not (usually) claim to evaluate the aesthetic quality or beauty of the music objectivly, instead it encompasses a series of automated, statitistical methods of analysing and comparing generated music to human composed music. Subjective evaluation encompasses evaluation methods that center human judgement.

2.10.1 Subjective Evaluation

For subjective approaches the methods vary widely [114]. There are simple turing-type evaluations that test how distinguishable generated and human written music are. There are tournament style surveys, where the number of winning pieces are tallied for each approach.[51][80] Another typical approach is using (likert) ratings along different dimensions to separate different qualities of the music. [81], [78] and [83] collect likert ratings on coherence, richness, arrangement and consistency. Specifically in [81] this takes the following form: coherence—"Is [the music] temporally coherent? Is the rhythm steady? Are there many out-of-context notes?"; richness—"Is [the music] rich and diverse in musical textures? Are there any repetitions and variations? Is [the music] too boring?"; arrangement—"Are the instruments used reasonably? Are the instruments arranged properly?". The specific questions asked vary by what exactly the project aimed to achieve. In [49] likert ratings are collected along the dimensions of stylistic success, aesthetic pleasure, repetition, melody, harmony, rhythm. Here, the questions for stylistic success are relevant due to their use of generative models to produce music in a certain style specifically classical string quartets, and classical piano improvisations. These evaluations are often paired with statistical hypothesis testing, to investigate relationships between the various participant ratings of the different models. An example would be: There is no difference between ModelA and ModelB on ratings of stylistic success. Or ratings of melodic success are positively correlated with ratings of aesthetic pleasure. Finally there are expert evaluations (which can also include likert ratings) but also detailed analysis or even performance of the produced music [45] [49].

2.10.2 Objective Evaluation

Objectuve evaluation of generated music include model specific metrics and different musical metrics [114]. 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 [51], Root Mean Square Error [80] or Perplexity[80]. Musical metrics typically involve comparing a set of generated music to a set of real music, there are plenty of musical similarity measure techniques[115] 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 [3] 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[116]. 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. [117]

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 [80]

Rhythmic patterns. Partial Similarity [107] between target and generated music:

3. Research Questions

In the previous section, we established the potential of generated music in serious games (section 2.1), and the need for adequate time-varying controls over the generation process (section 2.8) to reliably ensure the usability of the generated music in the context of MACT. We identified inner metric weight (section 2.8.1) as a promising target to describe rhythmic structure. Research links inner metric weight to rhythmic entrainment, and perceived and observed difficulty for a player to follow/tap along to the music. Additionally, it extends prior work on controlled music generation with a powerful rhythmic feature that is interpretable, relatively concise, and calculated symbolically. We collected crucial technical considerations for music generation including the overall approach in section 2.4, architecture in section 2.5, representation in section 2.7.1 and tokenization in section 2.7.3. Finally, we discussed promising methods of adding control to a model in section 2.9 and how to evaluate a model and its outputs (section 2.10).

The focus of the thesis is to develop a model that generates a complete musical piece with time-varying controls for rhythmic structure. The output of the model will be investigated for successful integration of control, player enjoyment, and interactive potential in the context of MACT.

1. Research Question 1: Can we effectively control for rhythmic structure 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' [2] rule based system.

4. Methodology

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

4.1 RhythmLang

4.1.1 Approach

Sections 2.4 and 2.5 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 effectively model various sequences and its potential to incorporate additional controls. We will use a symbolic representation of music (as discussed in section 2.7.1) due to its lightweight datasets and the ability to make changes to and incorporate the output into a music production environment, enabling a cooperative co-composition process as opposed to replacing the composer. More specifically, we will use a representation similar to REMI+ (see section 2.7.3), which extends the standard MIDI events with tokens indicating forms, such as meter, bar lines, and note-duration. This representation is compounded using byte pair encoding (see section 2.7.3) to decrease the sequence length and improve the capability and efficiency of the model. For adding control (section 2.9) we focus on methods that don't require full training, specifically parameter-efficient fine-tuning. This is more cost-effective and environmentally friendly. We will use the Lakh MIDI Dataset [76], a widely used open-source and licensed dataset for symbolic music generation, which will help us avoid ethical pitfalls around privacy and copyright (see section 2.2). Finally, we are not attempting to add control by training custom large foundation models, rather we use parameter-efficient fine-tuning or guidance to add control. (section 2.9) As a result, we only train 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 trained models. Finally, we use and extend an open-source model MusicLang in collaboration with its maintainers. If successful these extensions will contribute to the MusicLang project and be more widely available in a well-documented and continuously maintained ecosystem, with potential integrations into mainstream music production and composition software.

4.1.2 MusicLang - The foundation Model

MusicLang's core model is a transformer based on LLAMA 2 trained on the Lakh MIDI Dataset[76]. It can generate relatively long multi-track instrumental pieces (1-3 minutes) with control for chord progression, instrumentation, and range. Additionally, it can create 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 the harmonic structure and voice characteristics such as instrumentation or range (see figure 4.1). This base vocabulary is extended using a BPE tokenizer. **Addition** For inference there are multiple different modes of generation, 1) free generation, 2) continuation, 3) controlled generation and 4) controlled continuation. For free generation, the user can indicate the following settings.

- Number of tokens: Number of tokens to generate, this influences the length of the music depending on the number of instruments. More instruments and higher note density means more tokens per second of music.
- Temperature: Temperature parameter for softmax sampling (see section 2.5.3)
- Top p: Target percentage: (see section 2.5.3)

For controlled generation, the user can indicate a chord progression as a string. This includes extended chord variations including Major (M), minor (m), 7, m7b5, sus2, sus4, m7, M7, dim, dim0. One can also specify a bass note, i.e CM7/D. The length of the chord progression influences the length of the generation. For *continuation*, the user provides a MIDI-track and the section of measures that are used as the basis for generation. For *conrolled continuation* the user provides both a chord progression and a MIDI-track.

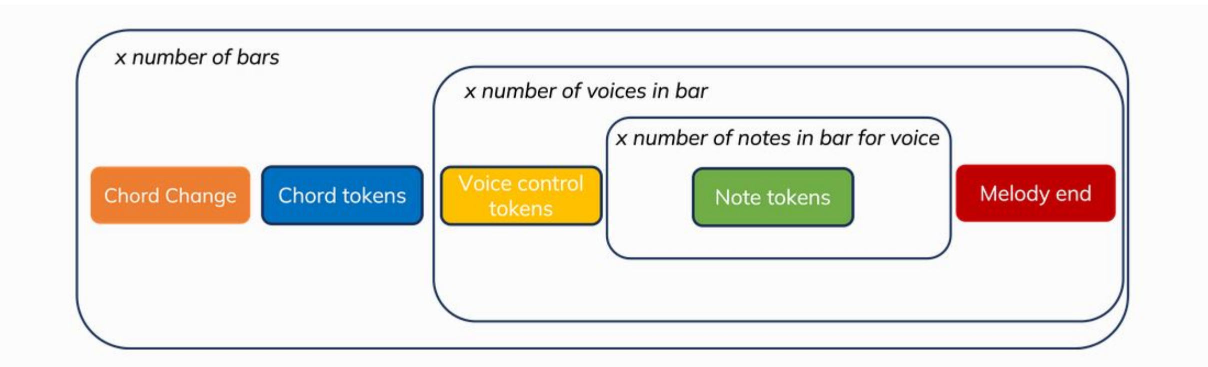


Figure 4.1: MusicLang's tokenisation on a high level, shows an hierachical tokenisation process. At the highest level are chord-related tokens they describe the harmonic structure of a section. This is followed by instrument level tokens that encode summarizing features such as instrumentation, octave, range, note-density of a voice within a section. Finally, there are note level tokens that encode pitch and duration of an event within a voice within a section

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

4.1.3 RhythmLang

RhythmLang is a fine-tuned variant of MusicLang with controls for inner metric weight. The fine-tuning targets the *controlled continuation* and the *controlled generation* modes of the model. Here the user provides a MIDI-file as input from which inner metric weight is extracted and passed to the model. In *controlled continuation* the user provides two MIDI-files, one from which the model continues and one from which the model extracts the target metric weight profile.

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 systematized 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 a good course of action. We use BassCraft, a smaller model, and start by controlling for note density. Note density is more easily calculated, tokenized, and verified than inner metric weight. Once control for note density is established, we will move on to inner metric weight. **Addition** BassCraft is used as a proof of concept for the methods of adding control, it is not part of the final model and evaluation. The most promising approach, or combination of approaches will be applied to fine-tune MusicLang.

4.2.1.1 BassCraft - a tiny transformer model

To avoid wasting computing resources and energy, we perform the preliminary experiments on a smaller model: Basscraft. BassCraft is a small transformer model based on GPT2 [63]. It has an embedding size of 256, 4 attention heads, four hidden transformer layers, and 7 million trainable parameters. In contrast, the target LLAMA 2-based model MusicLang has over 100 million trainable parameters. Basscraft generates a bassline to a provided piece of music and is trained on the Lakh MIDI dataset [76]. For training, songs with bass lines are selected (based on the presence of particular MIDI-instrument channels). The tracks are divided into snippets between 1 and 16 bars long. The bass lines are separated from the remaining track and matched as potential output. **Addition** For inference the user provides a target MIDI file *target* and a MIDI bass instrument such as Cello, Electric Bass, see the appendix for a full list of MIDI-bass instruments. The target MIDI file *target* can be a single instrument or multi-instrumental track. The model generates a corresponding bass-line of the same length as the target file (up to 16 bars). When control is added for note density, the user provides a single integer or a list of integers. If a single integer is passed this is replicated across all generated bars, if a list of integers X of size $|X|$ is provided, then the note density of bar m is given by $X_{m\%|X|} \forall m \in target$

When control is added for inner metric weight, the user provides an additional reference MIDI file $reference$. The metric weight profile of a m W is given per bar m w_m. The target rhythmic weight of a bar is given by $W_{m\%|W|} \forall m \in target$. The modulo operation ensure that for note-density and rhythmic weight control the length of the control input does not have to match the length of the target midi file.

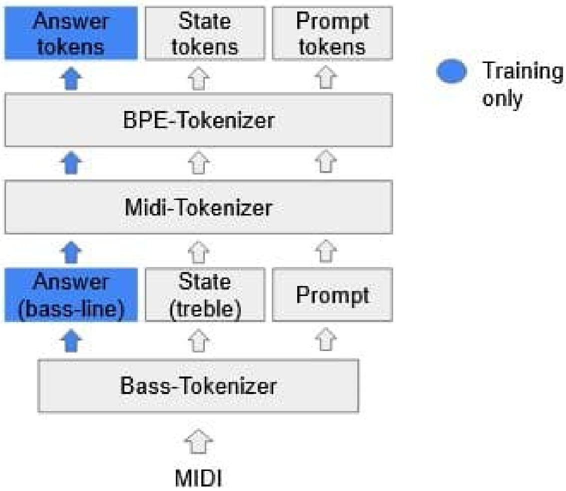


Figure 4.2: Preprocessing and tokenization of the original basscraft model

4.2.1.2 Method 1 - Vocabulary Expansion

Vocabulary expansion is the process of adding new vocabulary to a transformer model. MusicLang achieves its extraordinary controllability similarly to FIGARO [80] using control tokens that summarize features of the music that go beyond simply representing MIDI-like events. In vocabulary extension, it is critical to ensure that additional tokens do not overwrite or otherwise collide with the existing training. Otherwise, the benefits of using a pre-trained model disappear. Since we use a BPE tokenizer, it is difficult to add new tokens, as it would require retraining the BPE tokenizer, which will transform the embedding layer, making the pretrained model unusable. Instead, we investigate whether there are unused tokens and reassign them to our new control tokens. These new tokens are not included in any compound tokens generated by the BPE process, which increases the sequence length.

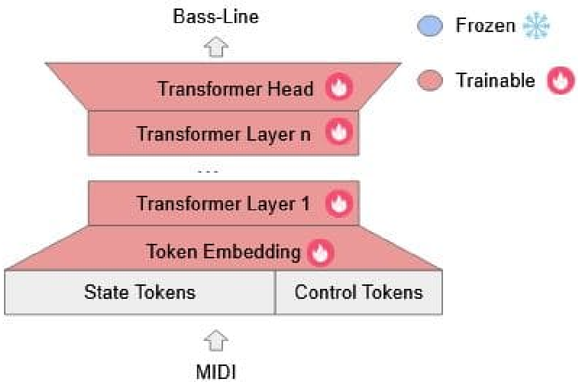


Figure 4.3: Vocabulary transfer with full fine tuning

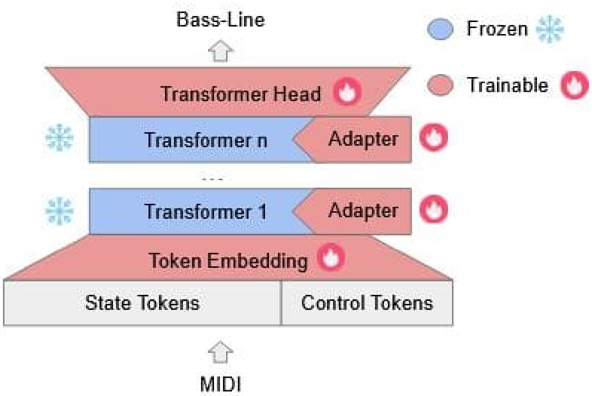


Figure 4.4: Vocabulary transfer with parameter efficient fine tuning

4.2.1.3 Method 2 - Integrating of control tokens

This approach differs from vocabulary expansion because it processes the control tokens as a parallel stream. This adopted from the approach used in Coco-Mulla [100]. After passing through a trainable positional embedding, the parallel stream of control tokens is inserted into the model at a layer c. The benefit of this method is that it doesn't require editing the model's vocabulary.

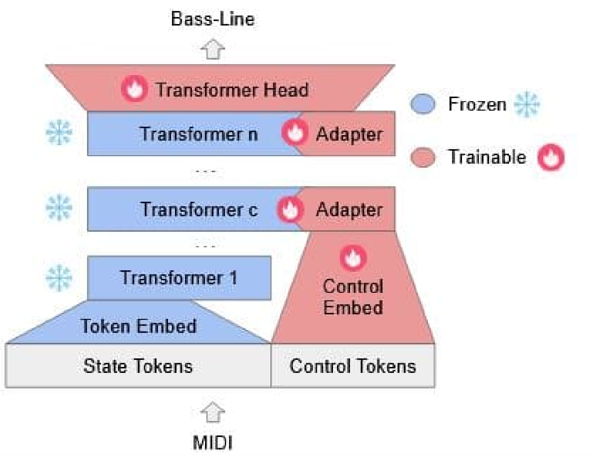


Figure 4.5: Integration of control tokens with parameter efficient fine tuning

4.2.1.4 Method 3 - Post-Hoc Guidance and other improvements

SMITIN[111] uses post-hoc guidance on a trained model to influence the generation process without retraining the model. This type of sampling-based guidance has been very successful in diffusion models. In transformers, however, it produces mixed results [112]. Additionally, this may be difficult to implement and transfer to inner metric weight. Both SMITIN and Rütte[112] only use one-dimensional variables that indicate the probability of a concept being present or not.

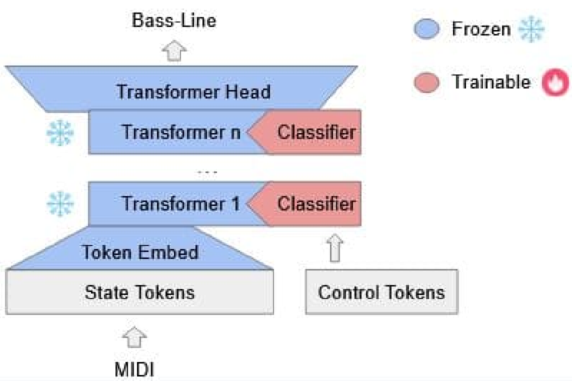


Figure 4.6: Integration of control using inference time interference

If these experiments are unsuccessful, we can follow the approach of [86] and try additional training using auxiliary tasks or use a modified (counterfactual) loss function.

4.2.2 Using inner metric weight as controllable feature

Inner metric analysis (IMA) creates metric weight profiles we use as guiding features 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 $g_m in$. These 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 [100]. These embeddings are then incorporated into the model. For inference, the user can provide a reference track from which the inner metric profile is extracted and passed to the model.

4.2.3 Training RhythmLang

Methods 1, 2 and 3 are sorted by expected difficulty of implementation. First BassCraft is extended using each of the methods with control for note density. When a method is found to introduce sufficient control we extend this to metric weight and try generating basslines with a provided metric profile. Finally, the most promising method (or combination of methods) is adopted to MusicLang, creating RhythmLang.

4.3 Model Evaluation

4.3.1 Datasets

Addition For fine-tuning the models with new control mechanisms we use a small random subset of 1000 songs from the Lakh MIDI Dataset $lmd1000$. For BassCraft we make sure that all songs in $lmd1000$ contain bass-lines. Both BassCraft and RhythmLang are evaluated quantitavly using a second random subset of 500 songs from which a random subsection between 4 and 16 bars is selected for the *continuation* mode of RhythmLang and the bass-generation of BassCraft. This is the $lmd500$ evaluation dataset. The references for inner metric weight come from a small set of $n = 30$ manually chosen prototypical rhythms *srhythm* using the set by [105] of prototypical rhythms for Tango, Rumba BossaNova, Merengue and March as a starting point. This can be expanded as desired for showcasing the control effectivness. To evaluate combined control of chords and rhythm we create a small set $n = 30$ of different two, three and four bar chord progressions. Finally we create three datasets containing inferences from the model for evaluation, $bcraft\_inf$, $mlang\_inf\_con$, *srhythm*. Each contains 500 entries, the model weights remain frozen during this process. To evaluate BassCraft we create $bcraft\_inf$ by running BassCraft on $lmd500$ with a randomly chosen rhythm from *srhythm* as control condition for each section in $lmd500$. To evaluate RhythmLangs's *controlled generation* mode we create $mlang\_inf\_con$ by running RhythmLang on 500 different pairwise permutations of chords from *schord* and rhythms from *srhythm*. To evaluate RhythmLang's *controlled continuation* we run RhythmLang on $lmd500$ with randmly chosen rhythms from *srhythm* for each track to create $mlang\_inf$, where RhythmLang continues from the excerpt from $lmd500$.

4.3.2 Evaluating Control-Effectivness

As discussed in section 2.10, calculating whether or not the control is effective depends on the controlled feature but can happen automatically. The first set of experiments target note density: If note density is a categorical variable such as low, medium, or high, the error is calculated similarly to a multilabel classifier. The predicted label is the note density of the generated music, and the ground-truth label is the note density given in the prompt on a bar-by-bar level. The metrics include accuracy, precision, recall, and F1 scores. If note density is continuous (the number of notes per bar), then the error would be calculated as mean square error:

$$error_{continuous} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_{generated} - y_{prompt})} \tag{4.1}$$

Inner metric weight analysis generates metric weight profiles, which we use as a guidance mechanism. Following the approach by [103], we can compare 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

$$D = \sum_{i=0}^{n-1}\left(\frac{(T_i - G_i)^2}{T_i + G_i}\right) \tag{4.2}$$

4.3.2.1

To evaluate the success of inner metric weight control we compare the inference datasets

4.4 User Study

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

4.4.1 Interactive Study

The interactive study aims to evaluate the fitness of the controlled generated music from an MACT perspective. Specifically, in the MACT protocol used by Chalkiadakis [2], the patient is supposed to listen to changes in the music and change their playing. There are two options that we are deciding between. **Option 1** 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 are correlated with each other. A high correlation indicates that the participants noticed the changes. For this we will generate $n = 10$ long pieces using RhythmLang, with varying rhythm controls. As control we pass a MIDI-file consisting of different template rhythms from *srhythm* at random intervals concatonated into one file.

4.4.2 Survey

The survey will compare the listening experience our model to the rule-based music generation system used in the original game to answer RQ3, whether the music by RhythmLang improves listener experience. The survey will also include the original MusicLang, to asess whether or not the fine-tuning for rhythm control decreases other capabilities of the model. Optionally we can include, other symbolic music generators such as FIGARO [80], this will help provide a point of comparison. Following the recommendation of [49] we could also include a state of the art rule-based generator such as MayaMarkov [50] and human-composed music from the Lakh MIDI dataset. The questionnaire will be adopted from [49], and each musical excerpt will be rated on a 7 point likert scale along the following dimensions: aesthetic pleasure, repetition, melody, harmony, rhythm.[2] Optionally there is space for the participant to comment on each excerpt. The participant will not see the source of the excerpt (i.e which model, or whether or not it is generated). The music provided for comparison will be excerpts cropped to about 30 seconds of rendered MIDI. While we don't cherry pick the tracks, we will follow [49] and filter the music to prevent 1) excessive repeatition of a small sequence (the model getting "stuck"), 2) long stretches of silence, 3) verbatim copying of training data or the prompt. The questionnaire also includes questions on demographic information such as gender, age, and educational background and musical experience.

4.4.3 Evaluating the survey, additional points of discussion

For each of the musical dimensions aesthetic pleasure, repetition, melody, harmony, rhythm, the difference in responses between the models is evaluated and significant differences $\alpha = 0.05$ are highlighted. Beyond simple statistical evaluation I would include listening examples, and an interactive code-space on google-collab where the reader can experiment with RhythmLang.

[2]From [49]: Aesthetic pleasure (Ap) The extent to which someone finds beauty in something Repetition or self-reference (Re) The reuse, in exact or inexact form, of musical material (e.g., notes, melody, harmony, rhythm) within a piece. Melody (Me) A succession of notes, varying in pitch, which have an organised and rec- ognisable shape. Melody is horizontal, i.e. the notes are heard consecutively. Harmony (Ha) The simultaneous sounding of two or more notes; synony- mous with chords. The organisation and arrangement of chords and their relationships to one another, vertically (at the same time) and horizontally (across time) over the course of a piece Rhythm: Everything pertaining to the time aspect of music (as distinct from the aspect of pitch), including event or note beginnings and endings, beats, accents, meas- ures, and groupings of various kinds.

Additionally, discussing participants comments, and investigating the model output musicologically, could be worthwhile depending on the results.

4.5 Thesis Timeline

Figure 4.7: Thesis project plan

Bibliography

[1] K. Rogers and M. Weber, "Audio habits and motivations in video game players," in *Proceedings of the 14th International Audio Mostly Conference: A Journey in Sound*, ser. AM '19, New York, NY, USA: Association for Computing Machinery, Sep. 2019, pp. 45–52, ISBN: 978-1-4503-7297-8. DOI: 10.1145/3356590.3356599. [Online]. Available: https://doi.org/10.1145/3356590.3356599.

[2] E. Chalkiadakis, "Developing and evaluating a musical attention control training computer game application.," M.S. thesis, Utrecht University, Utrecht, NL, Jan. 2022. [Online]. Available: https://studenttheses.uu.nl/bitstream/handle/20.500.12932/500/Master%20Thesis%20-%20Developing%20and%20evaluating%20a%20Musical%20Attention%20Control%20Training%20computer%20game%20application%20%285892252%29.pdf?sequence=1%5C&isAllowed=y.

[3] J. Copet, F. Kreuk, I. Gat, *et al.*, "Simple and controllable music generation," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. DOI: 10.48550/arXiv.2306.05284. [Online]. Available: https://openreview.net/forum?id=jtiQ26sCJi.

[4] A. Agostinelli, T. I. Denk, Z. Borsos, *et al.*, "Musiclm: Generating music from text," *arXiv*, no. arXiv:2301.11325, Jan. 2023. DOI: 10.48550/arXiv.2301.11325. [Online]. Available: http://arxiv.org/abs/2301.11325.

[5] B. R. Ferdinand Meyen, ""verknallt in einen talahon": So reagiert der talahon-produzent auf die rassismus-vorwürfe," *Bayern 2 Zuendfunk*, Aug. 2024. [Online]. Available: https://www.br.de/radio/bayern2/sendungen/zuendfunk/verknallt-in-einen-talahon-produzent-rassismus-vorwuerfe-charts-udio-ki-100.html.

[6] M. Stassen, "Suno, with a $500m valuation, has admitted training its ai on copyrighted music," *Music Business Worldwide*, Apr. 2024. [Online]. Available: https://www.musicbusinessworldwide.com/suno-with-a-500m-valuation-has-admitted-training-its-ai-on-copyrighted-music-it-just-named-timbaland-as-a-strategic-advisor1/.

[7] G. Nierhaus, *Algorithmic Composition (pp 36&38)*, en. Vienna: Springer, 2009, ISBN: 978-3-211-75539-6. DOI: 10.1007/978-3-211-75540-2. [Online]. Available: http://link.springer.com/10.1007/978-3-211-75540-2.

[8] D. Cope, *New directions in music*. Dubuque, Iowa : W.C. Brown, 1989, ISBN: 978-0-697-03342-0. [Online]. Available: http://archive.org/details/unset0000unse_x4d7.

[9] en. [Online]. Available: https://suno.com/about.

[10] G. Hadjeres, F. Pachet, and F. Nielsen, "Deepbach: A steerable model for bach chorales generation," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, PMLR 70:1362-1371, Jun. 2017. DOI: 10.48550/arXiv.1612.01010. [Online]. Available: http://arxiv.org/abs/1612.01010.

[11] M. E. Malandro, "Composer's assistant: An interactive transformer for multi-track midi infilling," in *2024 International Society for Music Information Retrieval*, arXiv:2301.12525 [cs], San Francisco, USA: ISMIR, Jul. 2023. DOI: 10.48550/arXiv.2301.12525. [Online]. Available: http://arxiv.org/abs/2301.12525.

[12] F. Pachet, "The continuator: Musical interaction with style," *Journal of New Music Research*, vol. 32, no. 3, pp. 333–341, Sep. 2003, ISSN: 0929-8215. DOI: 10.1076/jnmr.32.3.333.16861.

[13] J. Kite-Powell, *See how this ai improvised with a musican to create a musical dialogue in real-time*, en, Feb. 2023. [Online]. Available: https://www.forbes.com/sites/jennif

erhicks/2023/02/25/see-how-this-ai-improvised-with-a-musican-to-create-a-musical-dialogue-in-real-time/.

[14] C. Plut and P. Pasquier, "Generative music in video games: State of the art, challenges, and prospects," *Entertainment Computing*, vol. 33, p. 100 337, Mar. 2020, ISSN: 1875-9521. DOI: 10.1016/j.entcom.2019.100337.

[15] K. Worrall and T. Collins, "Considerations and concerns of professional game composers regarding artificially intelligent music technology," *IEEE Transactions on Games*, vol. 16, pp. 586–597, Sep. 2024, ISSN: 2475-1502, 2475-1510. DOI: 10.1109/TG.2023.3319085.

[16] D. Djaouti, J. Alvarez, and J.-P. Jessel, "Classifying serious games: The g/p/s model," in *Handbook of Research on Improving Learning and Motivation through Educational Games: Multidisciplinary Approaches*, P. Felicia, Ed., IGI Global, 2011, pp. 118–136. DOI: 10.4018/978-1-60960-495-0.ch006. [Online]. Available: https://www.igi-global.com/chapter/classifying-serious-games/52492.

[17] K. R. Agres, R. S. Schaefer, A. Volk, *et al.*, "Music, computing, and health: A roadmap for the current and future roles of music technology for health care and well-being," *Music & Science*, vol. 4, p. 2 059 204 321 997 709, Jan. 2021, ISSN: 2059-2043. DOI: 10.1177/2059204321997709.

[18] J.-K. Park and S. J. Kim, "Dual-task-based drum playing with rhythmic cueing on motor and attention control in patients with parkinson's disease: A preliminary randomized study," eng, *International Journal of Environmental Research and Public Health*, vol. 18, no. 19, p. 10 095, Sep. 2021, ISSN: 1660-4601. DOI: 10.3390/ijerph181910095.

[19] M. Martin-Moratinos, M. Bella-Fernández, and H. Blasco-Fontecilla, "Effects of music on attention-deficit/hyperactivity disorder (adhd) and potential application in serious video games: Systematic review," *Journal of Medical Internet Research*, vol. 25, e37742, May 2023, ISSN: 1439-4456. DOI: 10.2196/37742.

[20] V. Pasiali, A. B. LaGasse, and S. L. Penn, "The effect of musical attention control training (mact) on attention skills of adolescents with neurodevelopmental delays: A pilot study," *Journal of Music Therapy*, vol. 51, no. 4, pp. 333–354, 2014, ISSN: 0022-2917. DOI: 10.1093/jmt/thu030.

[21] R. van Alphen, G. J. J. M. Stams, and L. Hakvoort, "Musical attention control training for psychotic psychiatric patients: An experimental pilot study in a forensic psychiatric hospital," *Frontiers in Neuroscience*, vol. 13, p. 570, 2019, ISSN: 1662-4548. DOI: 10.3389/fnins.2019.00570.

[22] L. Schlette, "Increasing engagement in a mact game through feedback and adaptability," en, M.S. thesis, Utrecht University, 2022.

[23] D. Tencer, "New ai-powered 'instant' music-making app udio raises $10m; launches with backing from will.i.am, common, unitedmasters, a16z," en-US, *Music Business Worldwide*, Apr. 2024. [Online]. Available: https://www.musicbusinessworldwide.com/new-ai-powered-instant-music-making-app-udio-raises-10m-launches-with-backing-from-will-i-am-common-unitedmasters-a16z/.

[24] M. M. Kaba, M. N. River, and A. R. Perry, *Demand for jury trial*, Jun. 2024. [Online]. Available: https://storage.courtlistener.com/recap/gov.uscourts.mad.272063/gov.uscourts.mad.272063.1.0.pdf.

[25] M. Duan, A. Suri, N. Mireshghallah, *et al.*, "Do membership inference attacks work on large language models?" In *Conference on Language Modeling (COLM)*, arXiv:2402.07841 [cs], arXiv, Sep. 2024. DOI: 10.48550/arXiv.2402.07841. [Online]. Available: http://arxiv.org/abs/2402.07841.

[26] N. Carlini, J. Hayes, M. Nasr, *et al.*, "Extracting training data from diffusion models," in *Proceedings of the 32nd USENIX Security Symposium*, arXiv:2301.13188 [cs], Anaheim,

CA, USA: arXiv, Jan. 2023. DOI: 10.48550/arXiv.2301.13188. [Online]. Available: http://arxiv.org/abs/2301.13188.

[27] E. Newton-Rex, "Suno is a music ai company aiming to generate $120 billion per year. but is it trained on copyrighted recordings?" en-US, *Music Business Worldwide*, Apr. 2024. [Online]. Available: https://www.musicbusinessworldwide.com/suno-is-a-music-ai-company-aiming-to-generate-120-billion-per-year-newton-rex/.

[28] R. Reed, "Does chatgpt violate new york times' copyrights?" en-us, *Harvard Law School*, May 2024. [Online]. Available: https://hls.harvard.edu/today/does-chatgpt-violate-new-york-times-copyrights/.

[29] *Copyright law of the united states title 17, chapter 1, section 107*. [Online]. Available: https://www.copyright.gov/title17/92chap1.html#107.

[30] M. Singh, *Indian filmmaker ram gopal varma abandons human musicians for ai-generated music*, en-US, Sep. 2024. [Online]. Available: https://techcrunch.com/2024/09/19/indian-filmmaker-ram-gopal-varma-abandons-human-musicians-for-ai-generated-music/.

[31] M. Stassen, ""there are now 120,000 new tracks hitting music streaming services each day"," *Music Business Worldwide*, May 2023. [Online]. Available: https://www.musicbusinessworldwide.com/there-are-now-120000-new-tracks-hitting-music-streaming-services-each-day/.

[32] A. Marechal, "Generative ai: Energy consumption soars," en-GB, *Polytechnique Insights*, Nov. 2024. [Online]. Available: https://www.polytechnique-insights.com/en/columns/energy/generative-ai-energy-consumption-soars/.

[33] J. Kaplan, S. McCandlish, T. Henighan, *et al.*, "Scaling laws for neural language models," *arXiv*, no. arXiv:2001.08361, Jan. 2020, arXiv:2001.08361 [cs]. DOI: 10.48550/arXiv.2001.08361. [Online]. Available: http://arxiv.org/abs/2001.08361.

[34] A. Holzapfel, A.-K. Kaila, and P. Jääskeläinen, "Green mir? investigating computational cost of recent music-ai research in ismir," in *25th Int. Society for Music Information Retrieval Conf*, San Francisco, United States, 2024. [Online]. Available: https://drive.google.com/file/d/1rAepoJk1U2R4g3S_AcxdqWDGzRGd7xPg/view?usp=embed_facebook.

[35] A. France-Presse, "First recording of computer-generated music – created by alan turing – restored," *The Guardian*, Sep. 2016, ISSN: 0261-3077. [Online]. Available: https://www.theguardian.com/science/2016/sep/26/first-recording-computer-generated-music-created-alan-turing-restored-enigma-code.

[36] L. Hiller and L. M. (M. Isaacson, *Experimental music; composition with an electronic computer*. New York, McGraw-Hill, 1959. [Online]. Available: http://archive.org/details/experimentalmusi00hill.

[37] K. Ebcioğlu, "An expert system for harmonizing chorales in the style of j. s. bach," *Understanding Music with AI: Perspectives on Music Cognition*, vol. eds. M. Balaban, K. Ebcioğlu, and O. Laske, pp. 145–185, 1994.

[38] J. Polito, J. M. Daida, and T. F. Bersano-Begey, "Musica ex machina: Composing 16th-century counterpoint with genetic programming and symbiosis," *Evolutionary Programming*, Lecture Notes in Computer Science, vol. 1213, P. J. Angeline, R. G. Reynolds, J. R. McDonnell, and R. Eberhart, Eds., pp. 113–123, 1997, Book Title: Evolutionary Programming VI. DOI: 10.1007/BFb0014805.

[39] P. M. Todd, "A connectionist approach to algorithmic composition," *Computer Music Journal*, vol. 13, no. 4, pp. 27–43, 1989, ISSN: 0148-9267. DOI: 10.2307/3679551.

[40] M. C. Mozer, "Neural network music composition by prediction: Exploring the benefits of psychoacoustic constraints and multi-scale processing," *Connection Science*, vol. 6, no. 2–3, pp. 247–280, Jan. 1994, ISSN: 0954-0091. DOI: 10.1080/09540099408915726.

[41] T. Chakraborti, B. McCane, S. Mills, and U. Pal, "Coconet: A collaborative convolutional network," no. arXiv:1901.09886, Nov. 2020, arXiv:1901.09886 [cs]. DOI: 10.48550/arXiv.1901.09886. [Online]. Available: http://arxiv.org/abs/1901.09886.

[42] I. Xenakis, *Formalized Music Thought and Mathematics in Composition*. Pendragon Press, 1992. [Online]. Available: https://monoskop.org/images/7/74/Xenakis_Iannis_Formalized_Music_Thought_and_Mathematics_in_Composition.pdf.

[43] E. Featherstone, "Introducing the next generation of music makers," *The Guardian*, Aug. 2017, ISSN: 0261-3077. [Online]. Available: https://www.theguardian.com/small-business-network/2017/aug/29/computer-write-music-jukedeck-artificial-intelligence.

[44] L.-C. Yang, S.-Y. Chou, and Y.-H. Yang, "Midinet: A convolutional generative adversarial network for symbolic-domain music generation," in *18th International Society for Music Information Retrieval Conference (ISMIR)*, Suzhou, China, Jul. 2017. DOI: 10.48550/arXiv.1703.10847. [Online]. Available: http://arxiv.org/abs/1703.10847.

[45] B. L. T. Sturm and O. Ben-Tal, "Folk the algorithms: (mis)applying artificial intelligence to folk music," in E. R. Miranda, Ed. Cham: Springer International Publishing, 2016, pp. 423–454, ISBN: 978-3-030-72115-2. DOI: 10.1007/978-3-030-72116-9_16. [Online]. Available: https://link.springer.com/10.1007/978-3-030-72116-9_16.

[46] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, arXiv, 2017. DOI: 10.48550/arXiv.1706.03762. [Online]. Available: http://arxiv.org/abs/1706.03762.

[47] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever, "Jukebox: A generative model for music," *arXiv*, no. arXiv:2005.00341, Apr. 2020. DOI: 10.48550/arXiv.2005.00341. [Online]. Available: http://arxiv.org/abs/2005.00341.

[48] P. Christine, *Musenet*, Apr. 2019. [Online]. Available: openai.com/blog/musenet.

[49] Z. Yin, F. Reuben, S. Stepney, and T. Collins, "Deep learning's shallow gains: A comparative evaluation of algorithms for automatic music generation," en, *Machine Learning*, vol. 112, no. 5, pp. 1785–1822, May 2023, ISSN: 1573-0565. DOI: 10.1007/s10994-023-06309-w.

[50] T. Collins and R. Laney, "Computer-generated stylistic compositions with long-term repetitive and phrasal structure," None, *Journal of Creative Music Systems*, vol. 1, no. 22, Mar. 2017, ISSN: 2399-7656. DOI: 10.5920/JCMS.2017.02. [Online]. Available: https://www.jcms.org.uk/article/id/510/.

[51] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, *et al.*, "Music transformer," *arXiv*, no. arXiv:1809.04281, Dec. 2018, arXiv:1809.04281. DOI: 10.48550/arXiv.1809.04281. [Online]. Available: http://arxiv.org/abs/1809.04281.

[52] S. Mirelman, "Tuning procedures in ancient iraq," en, *Analytical Approaches to World Music 2*, no. 2, pp. 43–56, 2013.

[53] J. J. Fux, *Gradus ad Parnassum*. Vienna, Austria: Catholicae Majestati Aulae-Typographi, 1725. [Online]. Available: https://www.loc.gov/item/16002789/.

[54] R. C. Pinkerton, "Information theory and melody," *Scientific American*, vol. 194, no. 2, pp. 77–87, 1956.

[55] M. Allan, "Harmonising chorales in the style of johann sebastian bach," en, Ph.D. dissertation, Edinborough, 2002. [Online]. Available: https://www.researchgate.net/publication/238242337_Harmonising_Chorales_in_the_Style_of_Johann_Sebastian_Bach.

[56] E. V. Altay and B. Alatas, "Music based metaheuristic methods for constrained optimization," in *2018 6th International Symposium on Digital Forensic and Security (ISDFS 2018)*, Mar. 2018, pp. 1–6. DOI: 10.1109/ISDFS.2018.8355355. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8355355.

[57] D. Herremans and E. Chew, "Morpheus: Generating structured music with constrained patterns and tension," en, *IEEE Transactions on Affective Computing*, vol. 10, no. 4, pp. 510–523, Oct. 2019, arXiv:1812.04832 [cs], ISSN: 1949-3045, 2371-9850. DOI: `10.1109/TAFFC.2017.2737984`.

[58] H. Hild, J. Feulner, and W. Menzel, "Harmonet: A neural net for harmonizing chorales in the style of l.s.bach," en, *Advances in Neural Information Processing Systems 4 (NIPS 1991)*, 1991.

[59] M. Civit, J. Civit-Masot, F. Cuadrado, and M. J. Escalona, "A systematic review of artificial intelligence-based music generation: Scope, applications, and future trends," *Expert Systems with Applications*, vol. 209, p. 118 190, Dec. 2022, ISSN: 0957-4174. DOI: `10.1016/j.eswa.2022.118190`.

[60] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," en, *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, Dec. 2014, arXiv:1409.3215 [cs]. DOI: `10.48550/arXiv.1409.3215`. [Online]. Available: `http://arxiv.org/abs/1409.3215`.

[61] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," en, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (ACL 2019)*, arXiv:1810.04805 [cs], arXiv, May 2019. DOI: `10.48550/arXiv.1810.04805`. [Online]. Available: `http://arxiv.org/abs/1810.04805`.

[62] M. Zeng, X. Tan, R. Wang, Z. Ju, T. Qin, and T.-Y. Liu, "Musicbert: Symbolic music understanding with large-scale pre-training," en, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 791–800, Jun. 2021, arXiv:2106.05630 [cs]. DOI: `10.18653/v1/2021.findings-acl.70`.

[63] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019. [Online]. Available: `https://api.semanticscholar.org/CorpusID:160025533`.

[64] H. Liu, Z. Chen, Y. Yuan, *et al.*, "Audioldm: Text-to-audio generation with latent diffusion models," in *Proceedings of the 40th International Conference on Machine Learning (PMLR)*, arXiv:2301.12503 [cs], arXiv, Sep. 2023. DOI: `10.48550/arXiv.2301.12503`. [Online]. Available: `http://arxiv.org/abs/2301.12503`.

[65] Z. Evans, J. D. Parker, C. J. Carr, Z. Zukowski, J. Taylor, and J. Pons, "Stable audio open," *arXiv*, no. arXiv:2407.14358, Jul. 2024, arXiv:2407.14358 [cs]. DOI: `10.48550/arXiv.2407.14358`. [Online]. Available: `http://arxiv.org/abs/2407.14358`.

[66] L. Min, J. Jiang, G. Xia, and J. Zhao, "Polyffusion: A diffusion model for polyphonic score generation with internal and external controls," in *24th Int. Society for Music Information Retrieval Conference (ISMIR 2023)*, arXiv:2307.10304 [cs], Milan, Italy: arXiv, Jul. 2023. [Online]. Available: `http://arxiv.org/abs/2307.10304`.

[67] S. Ji, X. Yang, and J. Luo, "A survey on deep learning for symbolic music generation: Representations, algorithms, evaluations, and challenges," en, *ACM Computing Surveys*, vol. 56, no. 1, pp. 1–39, Jan. 2024, ISSN: 0360-0300, 1557-7341. DOI: `10.1145/3597493`.

[68] J. Keith, *Thesession data*. [Online]. Available: `https://github.com/adactio/TheSession-data`.

[69] *Canonical jsb dataset chorals 389*. [Online]. Available: `https://github.com/czhuang/JSB-Chorales-dataset`.

[70] C. Hawthorne, A. Stasyuk, A. Roberts, *et al.*, "Enabling factorized piano music modeling and generation with the MAESTRO dataset," in *International Conference on Learning Representations*, 2019. [Online]. Available: `https://openreview.net/forum?id=r1lYRjC9F7`.

[71] *Classical archives dataset donation*. [Online]. Available: https://www.classicalarchives.com/newca/#!/.

[72] *Bit midi dataset donation*. [Online]. Available: https://bitmidi.com/.

[73] *Hook theory collection of pop-song transcriptions*. [Online]. Available: https://www.hooktheory.com/theorytab.

[74] H. Tan and D. Herremans, "Music fadernets: Controllable music generation based on high-level features via low-level feature modelling," en, in *Proc. of 21st International Society of Music Information Retrieval Conference, ISMIR 2020*, Montreal QC Canada, 2020. [Online]. Available: https://www.researchgate.net/publication/343500818_Music_FaderNets_Controllable_Music_Generation_Based_On_High-Level_Features_via_Low-Level_Feature_Modelling.

[75] J. Ens and P. Pasquier, "Mmm: Exploring conditional multi-track music generation with the transformer," *arXiv*, no. arXiv:2008.06048, Aug. 2020, arXiv:2008.06048 [cs]. DOI: 10.48550/arXiv.2008.06048. [Online]. Available: http://arxiv.org/abs/2008.06048.

[76] C. Raffel, "Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching," en, Ph.D. dissertation, Columbia University, New York, NY, USA, 2016.

[77] Y.-S. Huang and Y.-H. Yang, "Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions," in *Proceedings of the 28th ACM International Conference on Multimedia*, ser. MM '20, New York, NY, USA: Association for Computing Machinery, Oct. 2020, pp. 1180–1188, ISBN: 978-1-4503-7988-5. DOI: 10.1145/3394171.3413671. [Online]. Available: https://doi.org/10.1145/3394171.3413671.

[78] B. Yu, P. Lu, R. Wang, *et al.*, "Museformer: Transformer with fine- and coarse-grained attention for music generation," en, in *Thirty-sixth Conference on Neural Information Processing Systems (NeurIPS 2022)*, arXiv:2210.10349 [cs], arXiv, Oct. 2022. DOI: 10.48550/arXiv.2210.10349. [Online]. Available: http://arxiv.org/abs/2210.10349.

[79] Z. Wang, K. Chen, J. Jiang, *et al.*, "Pop909: A pop-song dataset for music arrangement generation," en, in *21st International Conference on Music Information Retrieval (ISMIR 2020)*, 2020.

[80] D. v. Rütte, L. Biggio, Y. Kilcher, and T. Hofmann, "Figaro: Generating symbolic music with fine-grained artistic control," *International Conference on Learning Representations*, 2023, arXiv:2201.10936. DOI: 10.48550/arXiv.2201.10936. [Online]. Available: http://arxiv.org/abs/2201.10936.

[81] H.-W. Dong, K. Chen, S. Dubnov, J. McAuley, and T. Berg-Kirkpatrick, "Multitrack music transformer," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2023)*, arXiv:2207.06983, arXiv, May 2023. DOI: 10.48550/arXiv.2207.06983. [Online]. Available: http://arxiv.org/abs/2207.06983.

[82] L. Crestel, P. Esling, L. Heng, and S. McAdams, "A database linking piano and orchestral midi scores with application to automatic projective orchestration," in *18th International Society for Music Information Retrieval Conference*, arXiv:1810.08611 [cs], Suzhou, China: arXiv, Oct. 2018. DOI: 10.48550/arXiv.1810.08611. [Online]. Available: http://arxiv.org/abs/1810.08611.

[83] H. Chen, J. B. L. Smith, J. Spijkervet, *et al.*, "Sympac: Scalable symbolic music generation with prompts and constraints," in *25th Int. Society for Music Information Retrieval Conference*, arXiv:2409.03055 [cs], San Fancisco, CA, USA, Sep. 2024. DOI: 10.48550/arXiv.2409.03055. [Online]. Available: http://arxiv.org/abs/2409.03055.

[84] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proceedings of the 12th international conference on music information retrieval (ISMIR 2011)*, 2011.

[85] P. Lu, X. Xu, C. Kang, *et al.*, "Musecoco: Generating symbolic music from text," *arXiv*, no. arXiv:2306.00110, May 2023, arXiv:2306.00110 [cs]. DOI: 10.48550/arXiv.2306.00110. [Online]. Available: http://arxiv.org/abs/2306.00110.

[86] Y. Shu, H. Xu, Z. Zhou, A. v. d. Hengel, and L. Liu, "Musebarcontrol: Enhancing fine-grained control in symbolic music generation through pre-training and counterfactual loss," *arXiv*, no. arXiv:2407.04331, Jul. 2024, arXiv:2407.04331 [cs]. DOI: 10.48550/arXiv.2407.04331. [Online]. Available: http://arxiv.org/abs/2407.04331.

[87] J. Ryu, H.-W. Dong, J. Jung, and D. Jeong, "Nested music transformer: Sequentially decoding compound tokens in symbolic music and audio generation," in *25th Int. Society for Music Information Retrieval Conference (ISMIR 2024)*, arXiv:2408.01180 [cs], Aug. 2024. DOI: 10.48550/arXiv.2408.01180. [Online]. Available: http://arxiv.org/abs/2408.01180.

[88] T. Zhu, H. Liu, Z. Jiang, and Z. Zheng, "Symbolic music generation with fine-grained interactive textural guidance," en, *arXiv*, no. arXiv:2410.08435, Oct. 2024, arXiv:2410.08435 [cs]. DOI: 10.48550/arXiv.2410.08435. [Online]. Available: http://arxiv.org/abs/2410.08435.

[89] Y. Huang, A. Ghatare, Y. Liu, *et al.*, "Symbolic music generation with non-differentiable rule guided diffusion," in *The Forty-First International Conference on Machine Learning*, arXiv:2402.14285 [cs], Vienna, Austria: arXiv, Sep. 2024. DOI: 10.48550/arXiv.2402.14285. [Online]. Available: http://arxiv.org/abs/2402.14285.

[90] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations (ICLR)*, arXiv:1312.6114 [stat], arXiv, 2014. DOI: 10.48550/arXiv.1312.6114. [Online]. Available: http://arxiv.org/abs/1312.6114.

[91] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, "High fidelity neural audio compression," en, *Transactions on Machine Learning Research*, 2023, arXiv:2210.13438 [eess]. DOI: 10.48550/arXiv.2210.13438. [Online]. Available: http://arxiv.org/abs/2210.13438.

[92] N. Fradet, J.-P. Briot, and F. Chhel, "Miditok: A python package for midi file tokenization," en, *Extended Abstracts for the Late-Breaking Demo Session of the 22nd International Society for Music Information Retrieval Conference (ISMIR 2021)*, 2021.

[93] W.-Y. Hsiao, J.-Y. Liu, Y.-C. Yeh, and Y.-H. Yang, "Compound word transformer: Learning to compose full-song music over dynamic directed hypergraphs," in *AAAI Conference on Artificial Intelligence*, arXiv:2101.02402 [cs], arXiv, Jan. 2021. DOI: 10.48550/arXiv.2101.02402. [Online]. Available: http://arxiv.org/abs/2101.02402.

[94] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Association for Computational Linguistics (ACL 2016)*, arXiv:1508.07909 [cs], Berlin, Germany: arXiv, Jun. 2016. DOI: 10.48550/arXiv.1508.07909. [Online]. Available: http://arxiv.org/abs/1508.07909.

[95] N. Fradet, N. Gutowski, F. Chhel, and J.-P. Briot, "Byte pair encoding for symbolic music," in *Conference on Empirical Methods in Natural Language Processing*, arXiv:2301.11975 [cs], arXiv, Nov. 2023. DOI: 10.48550/arXiv.2301.11975. [Online]. Available: http://arxiv.org/abs/2301.11975.

[96] P. Van Kranenburg, A. Volk, and F. Wiering, "A comparison between global and local features for computational classification of folk song melodies," en, *Journal of New Music Research*, vol. 42, no. 1, pp. 1–18, Mar. 2013, ISSN: 0929-8215, 1744-5027. DOI: 10.1080/09298215.2012.718790.

[97] J. Blacking, "Deep and surface structures in venda music," *Yearbook of the International Folk Music Council*, vol. 3, pp. 91–108, 1971, ISSN: 0316-6082. DOI: 10.2307/767458.

[98] S.-L. Wu, C. Donahue, S. Watanabe, and N. J. Bryan, "Music controlnet: Multiple time-varying controls for music generation," en, *arXiv*, no. arXiv:2311.07069, Nov. 2023, arXiv:2311.07069 [cs]. [Online]. Available: http://arxiv.org/abs/2311.07069.

[99] Y.-H. Lan, W.-Y. Hsiao, H.-C. Cheng, and Y.-H. Yang, "Musicongen: Rhythm and chord control for transformer-based text-to-music generation," en, in *25th Int. Society for Music Information Retrieval Conference*, arXiv:2407.15060 [cs], San Fancisco, CA, USA: arXiv, Jul. 2024. [Online]. Available: `http://arxiv.org/abs/2407.15060`.

[100] L. Lin, G. Xia, J. Jiang, and Y. Zhang, "Content-based controls for music large language modeling," en, *arXiv*, no. arXiv:2310.17162, Oct. 2024, arXiv:2310.17162 [cs]. DOI: `10.48550/arXiv.2310.17162`. [Online]. Available: `http://arxiv.org/abs/2310.17162`.

[101] O. Tal, A. Ziv, I. Gat, F. Kreuk, and Y. Adi, "Joint audio and symbolic conditioning for temporally controlled text-to-music generation," en, in *25th Int. Society for Music Information Retrieval Conference*, arXiv:2406.10970 [cs], San Fancisco, CA, USA, Jun. 2024. [Online]. Available: `http://arxiv.org/abs/2406.10970`.

[102] Z. Wang, D. Wang, Y. Zhang, and G. Xia, "Learning interpretable representation for controllable polyphonic music generation," in *21st International Conference on Music Information Retrieval (ISMIR 2020)*, arXiv:2008.07122 [cs], Montreal QC Canada: arXiv, Aug. 2020. DOI: `10.48550/arXiv.2008.07122`. [Online]. Available: `http://arxiv.org/abs/2008.07122`.

[103] B. Bemman and J. Christensen, "Inner metric analysis as a measure of rhythmic syncopation," in *Proceedings of the 25th International Society for Music Information Retrieval Conference (ISMIR)*, San Francisco, USA, Nov. 2024, pp. 389–396. [Online]. Available: `https://ismir2024program.ismir.net/poster_251.html`.

[104] A. Volk, "The study of syncopation using inner metric analysis: Linking theoretical and experimental analysis of metre in music," *Journal of New Music Research*, vol. 37, no. 4, pp. 259–273, 2008. DOI: `10.1080/09298210802479213`. [Online]. Available: `https://webspace.science.uu.nl/~veltk101/publications/art/JNMR08.pdf`.

[105] E. Chew, A. Volk, and C.-Y. Lee, "Dance music classification using inner metric analysis," en, in *The Next Wave in Computing, Optimization, and Decision Technologies*, B. Golden, S. Raghavan, and E. Wasil, Eds., Boston, MA: Springer US, 2005, pp. 355–370, ISBN: 978-0-387-23529-5. DOI: `10.1007/0-387-23529-9_23`.

[106] W. B. Haas and A. Volk, "Meter detection in symbolic music using inner metric analysis," in *International Society for Music Information Retrieval Conference (ISMIR 2016)*, Aug. 2016. [Online]. Available: `https://www.semanticscholar.org/paper/Meter-Detection-in-Symbolic-Music-Using-Inner-Haas-Volk/8a72fc0874e0476f3dc0b92a109ebf854a4551cc`.

[107] A. Volk, J. Garbers, P. Van Kranenburg, F. Wiering, L. Grijp, and R. C. Veltkamp, "Comparing computational approaches to rhythmic and melodic similarity in folksong research," en, in *Mathematics and Computation in Music* (Communications in Computer and Information Science), T. Klouche and T. Noll, Eds., Communications in Computer and Information Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, vol. 37, pp. 78–87, ISBN: 978-3-642-04578-3. DOI: `10.1007/978-3-642-04579-0_8`. [Online]. Available: `http://link.springer.com/10.1007/978-3-642-04579-0_8`.

[108] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," en, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695. [Online]. Available: `https://openaccess.thecvf.com/content/CVPR2022/html/Rombach_High-Resolution_Image_Synthesis_With_Latent_Diffusion_Models_CVPR_2022_paper.html`.

[109] M. Lee, S. Doh, and D. Jeong, "Annotator subjectivity in the musiccaps dataset," en, *HCMIR23: 2nd Workshop on Human-Centric Music Information Research)*, vol. 1, Nov. 2023.

[110] S. Rouard, Y. Adi, J. Copet, A. Roebel, and A. Défossez, "Audio conditioning for music generation via discrete bottleneck features," in *25th Int. Society for Music Information Retrieval Conference*, arXiv:2407.12563, San Fancisco, CA, USA: arXiv, Jul. 2024. DOI: 10.48550/arXiv.2407.12563. [Online]. Available: http://arxiv.org/abs/2407.12563.

[111] J. Koo, G. Wichern, F. G. Germain, S. Khurana, and J. L. Roux, "Smitin: Self-monitored inference-time intervention for generative music transformers," *IEEE Open Journal of Signal Processing 2025*, Apr. 2024, arXiv:2404.02252. DOI: 10.48550/arXiv.2404.02252. [Online]. Available: http://arxiv.org/abs/2404.02252.

[112] D. von Rütte, S. Anagnostidis, G. Bachmann, and T. Hofmann, "A language model's guide through latent space," in *Proceedings of the 41st International Conference on Machine Learning*, ser. ICML'24, Vienna, Austria: JMLR.org, 2024.

[113] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *IEEE/CVF International Conference on Computer Vision (ICCV 2023)*, arXiv:2302.05543 [cs], arXiv, Nov. 2023. DOI: 10.1109/ICCV51070.2023.00355. [Online]. Available: http://arxiv.org/abs/2302.05543.

[114] Z. Xiong, W. Wang, J. Yu, Y. Lin, and Z. Wang, "A comprehensive survey for evaluation methodologies of ai-generated music," *arXiv*, no. arXiv:2308.13736, Aug. 2023, arXiv:2308.13736 [cs]. DOI: 10.48550/arXiv.2308.13736. [Online]. Available: http://arxiv.org/abs/2308.13736.

[115] K. Gurjar and Y.-S. Moon, "A comparative analysis of music similarity measures in music information retrieval systems," en, *Journal of Information Processing Systems*, vol. 14, no. 1, pp. 32–55, Feb. 2018. DOI: 10.3745/JIPS.04.0054.

[116] K. Kilgour, M. Zuluaga, D. Roblek, and M. Sharifi, "Fréchet audio distance: A metric for evaluating music enhancement algorithms," en, *INTERSPEECH 2019*, Jan. 2019, arXiv:1812.08466 [eess]. DOI: 10.48550/arXiv.1812.08466. [Online]. Available: http://arxiv.org/abs/1812.08466.

[117] B. Elizalde, S. Deshmukh, M. A. Ismail, and H. Wang, "Clap: Learning audio concepts from natural language supervision," en, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2023)*, arXiv:2206.04769 [cs], arXiv, 2023. DOI: 10.48550/arXiv.2206.04769. [Online]. Available: http://arxiv.org/abs/2206.04769.

[118] C. McKay, "Automatic genre classification using large high-level musical feature sets.," en, Ph.D. dissertation, McGill, Montreal QC Canada, 2004. [Online]. Available: https://www.researchgate.net/publication/220723648_Automatic_Genre_Classification_Using_Large_High-Level_Musical_Feature_Sets.

5. Appendix

5.1 Comparing Tokenization lengths

Pretokenized Sequence Lengths - Mean: 47976, Median: 43692.0, Std Dev: 23160.
Tokenized Sequence Lengths - Mean: 14519, Median: 13159.0, Std Dev: 7197

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.

5.2 Feature categorisation

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 [118] 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 [96]. **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 [96].

Also relevant in the music generation context are **high-level** vs **low-level** features. Music generation can take high-level concepts such as genre [110], [85] or emotion [74] [85] into account, which in turn effects more complex dynamics of concrete features. To illustrate: In Music FaderNets [74] 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.

5.3 LastMinuteGig - music algorithms

This project is motivated by extending the music engine of LastMinuteGig. [2]. 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)

The following is a description of an improved algorithm utilizing generated music by Rhythm-Lang

- 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.

5.4 Notes Comparing Symbolic Music Generators and their evaluation methods

- DeepBach [10] - RNN - inpainting. Evaluation Turing
- FolkRNN [45] - RNN - control for meter and mode - Expert Evaluation + Performance Practice
- MusicTransformer [51] - Transformer - Evaluation: Subjective - Tournament Style between different generated and natural music. Objective: Validation NLL
- MidiNet[44] - GAN - Control for Chords/Priming melody. Human (how pleasing, how real, and how interesting)
- Polyfussion [66] - Diffusion Model - supports inpainting, interpolation, melody/accompaniment generation, control for chord progression, texture. Subjective Evaluation Questionnaire for naturalness, creativity, musicality. Objective Control success.
- FIGARO [80] - 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) [81] - 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 - [77] - Transformer - continuation, control local control over chord and tempo
- MuseNet [48] - Transformer -,Instrument control, style control. No evaluation, only show-

case.

- MMM [75] - Transformer model - inpainting, instrument control, note-density. Introduce novel representation No Rigorous Evaluation
- SymPAC [83] - 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 [85] - 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 [86] Extension of MuseCoco for time varying chord controls. Counterfactual Loss and Auxiliary task training improve controllability.
- Museformer [78] - 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 [87] - 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[74] - 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.