NAZARBAYEV UNIVERSITY

MASTER THESIS

Psychoacoustic Optimization of the VQ-VAE and Transformer Architectures for Human-like Auditory Perception in Music Information Retrieval and Generation Tasks

Author:
Elnur RAKHMATULLIN

Supervisor:
Dr. Akhan ALMAGAMBETOV

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

School of Engineering and Digital Sciences
Department of ECE

NAZARBAYEV UNIVERSITY

April 12, 2023

Declaration of Authorship

I, Elnur RAKHMATULLIN, hereby declare that this manuscript, entitled "Psychoacoustic Optimization of the VQ-VAE and Transformer Architectures for Human-like Auditory Perception in Music Information Retrieval and Generation Tasks", is the result of my own work except for quotations and citations which have been duly acknowledged.

I also declare that, to the best of my knowledge and belief, it has not been previously or concurrently submitted, in whole or in part, for any other degree or diploma at Nazarbayev University or any other national or international institution.

Signed:

Date: April 12, 2023

"I think their problem is that they started out knowing too much about the wrong thing."

Dr. Marvin Minsky, MIT

NAZARBAYEV UNIVERSITY

Abstract

School of Engineering and Digital Sciences
Department of ECE

Master of Science

Psychoacoustic Optimization of the VQ-VAE and Transformer Architectures for Human-like Auditory Perception in Music Information Retrieval and Generation Tasks

by Elnur RAKHMATULLIN

Despite incredible advancements in the utilization of learning-based architectures (AI) in natural language and image domains, their applicability to the domain of music has remained limited. In fact, the performance of state-of-the-art Automated Music Transcription (AMT) systems has seen only marginal improvements from novel AI architectures. Moreover, the importance of psychoacoustic perception and its incorporation into MIR systems have mostly stayed addressed, leading to shortcomings in current approaches. This thesis provides an overview of music processing and novel neural architectures, investigates the reasons behind the subpar performance achieved by their utilization in music information retrieval (MIR) tasks, and proposes several ways of adjusting both the music (data-related) pre-processing pipelines, and psychoacoustically-adjusted transformer-based model to improve the performance on MIR and AMT tasks. In particular, a new *music transformer* architecture is proposed, and various algorithms of music pre-processing for psychoacoustic optimization are implemented along with several adaptive models aimed at addressing the missing factor of modeling human music perception. The preliminary performance results exhibit promising outcomes, warranting the continued investigation of transformer architectures for music information retrieval applications. Several intriguing insights unveiled during the research process are discussed and presented. The thesis concludes by delineating a set of promising future research directions, paving the way for further advancements in the field of music information retrieval and generation using proposed architectures.

Acknowledgements

Words may fall short in expressing the profound gratitude and appreciation I feel for the numerous individuals who have provided their unwavering support, guidance and encouragement throughout the journey of completing this thesis. Their contributions, mostly indirect and unbeknownst to them, have been invaluable in shaping my work and personal growth.

First and foremost, I extend my heartfelt appreciation to my supervisor, Dr. Akhan Almagambetov, whose trust in my abilities has been instrumental in the development of this research. I am immensely grateful to the faculty of our ECE department at NU for their guidance, feedback and expertise during my 6 years of studies here. It is Dr. Muhammad Tahir Akhtar, Dr. Daniele Tosi and Dr. Aresh Dadlani who I have to thank for inspiring me with their professionalism and - just as importantly - making me a better researcher with their criticisms. They are a true testament to the wisdom that it's not so much the books and learning material that the students learn from, but the spirit of their instructors.

My sincere thanks go to my fellow musicians and friends at Nazarbayev University and beyond - Yerulan, Aset and Abilmansur, whose camaraderie and shared passion for music have made this journey a truly rewarding experience.

Last but not least, my profound gratitude goes to my family, whose unwavering love, support, and understanding have been the bedrock upon which I have built my academic pursuits. Their belief in me and my dreams has been an inexhaustible source of strength that enabled me to reach this milestone.

Once again, I extend my heartfelt thanks to all those who have contributed to the successful completion of this thesis. It is to them and to God that I dedicate this work.

Contents

List of Figures

List of Tables

List of Abbreviations

AI	**A**rtificial **I**ntelligence
ANN	**A**rtificial **N**eural **N**etwork
BPM	**B**eats **P**er **M**inute
CNN	**C**onvolutional **N**eural **N**etwork
CQT	**C**onstant **Q**-**T**ransform
DL	**D**eep **L**earning
DFT	**D**iscrete **F**ourier **T**ransform
DSP	**D**igital **S**ignal **P**rocessing
FFT	**F**ast **F**ourier **T**ransform
GAN	**G**enerative **A**dversarial **N**etwork
LSTM	**L**ong **S**hort-**T**erm **M**emory
MFCC	**M**el **F**requency **C**epstral **C**oefficients
MLP	**M**ulti-**L**ayer **P**erceptron
MIDI	**M**usical **I**nstrument **D**igital **I**nterface
MIR	**M**usic **I**nformation **R**etrieval
ML	**M**achine **L**earning
NLP	**N**atural **L**anguage **P**rocessing
RNN	**R**ecurrent **N**eural **N**etworks
STFT	**S**hort-**T**ime **F**ourier **T**ransform
VAE	**V**ariational **A**utoencoder
VQ-GAN	**V**ector **Q**uantized **G**enerative **A**dversarial **N**etwork
VQ-VAE	**V**ector **Q**uantized **V**ariational **A**utoencoder

Chapter 1

Introduction

1.1 Problem Definition

Music, as a powerful form of human expression, has been an essential component of human culture for millennia. With the advent of digital technologies and the rapid growth of audio and music data, the field of Music Information Retrieval (MIR) has emerged as an interdisciplinary research area dedicated to extracting meaningful information from music data for a wide range of applications. Examples of such tasks include music transcription, beat tracking, chord estimation, composer style transfer, stylized music generation, lyrics matching, key estimation, modulation tracking, and genre classification.

Despite the significant progress made in MIR research, the performance of state-of-the-art techniques is still far from perfect. One of the main reasons for this is the lack of suitable representation of audio data that reflects the intricacies of human auditory perception. Most existing approaches in MIR tasks rely on raw waveform data or simple frequency-domain representations, which fail to capture the complexity of human auditory perception. This limitation results in suboptimal performance across various MIR tasks.

In recent years, advanced neural network architectures, such as vector-quantized variational autoencoders (VQ-VAEs) and transformers, have shown remarkable success in domains like natural language processing and image processing. These architectures have demonstrated their ability to efficiently preserve inter-token information, leading to significant improvements in performance. However, the application of these novel architectures to the audio/music processing domain has not yielded comparable improvements, primarily due to the difficulty in mapping audio data to a representation that closely approximates human auditory perception - a broad research field known as *psychoacoustics*.

1.2 Scope and Motivation

The central motivation of this thesis is to bridge the gap between existing neural network architectures and the unique requirements of MIR tasks by incorporating psychoacoustic principles into the representation of audio data and the design of neural architectures. Psychoacoustics is the study of the human perception of sound, investigating the relationships between physical properties of sound stimuli and the sensations they evoke in the human auditory system.

By leveraging insights from psychoacoustic research, this thesis aims to develop a new dataset that incorporates psychoacoustic features, such as frequency logarithmization, multi-pitch harmonic perception, and relative scale degree perception. Furthermore, the thesis will explore the adaptation and optimization of VQ-VAE

and transformer architectures to work with the psychoacoustic dataset, potentially leading to the development of new neural networks tailored for music information retrieval and generation tasks.

This research has the potential to significantly improve the performance of MIR tasks and contribute to the development of more effective and human-like auditory perception models. The outcomes of this research could have wide-ranging implications for various applications in music production, analysis, recommendation, and education.

In summary, this thesis seeks to address the following research questions:

1. How can psychoacoustic principles be incorporated into the representation of audio data to better approximate human auditory perception?

2. How can existing VQ-VAE and transformer architectures be adapted and optimized to work with a psychoacoustic dataset for MIR tasks?

3. To what extent does the incorporation of psychoacoustic features improve the performance of neural architectures in music information retrieval and generation tasks?

Chapter 2

Theoretical Background

2.1 Devil in the Details: Music Representation

To analyze and process digital audio signals effectively, it is crucial to understand the fundamental aspects of music representation. Digital audio is typically represented as a continuous waveform, which is discretized into individual samples using quantization techniques. In this section, key aspects of digital audio representation, including waveform, bit depth, and sampling rate, are discussed.

2.1.1 Waveform

A waveform is a visual representation of the amplitude of an audio signal as a function of time. In the case of digital audio, the continuous waveform is discretized into a series of discrete samples. The amplitude of each sample corresponds to the air pressure variation at a specific point in time. Mathematically, a continuous waveform can be represented as a function $x(t)$, where t denotes time and $x(t)$ denotes the amplitude of the waveform at time t.

2.1.2 Bit Depth

Bit depth, also known as quantization, is the process of converting the continuous amplitude values of an audio waveform into discrete digital values. The bit depth determines the number of possible amplitude values that a sample can have. The higher the bit depth, the more accurate the representation of the original continuous waveform. Bit depth is usually measured in bits, and the total number of possible amplitude values can be calculated as 2^n, where n is the bit depth. For example, a 16-bit audio file has 2^{16} or 65,536 possible amplitude values.

The quantization error, or the difference between the original continuous amplitude and its discrete representation, is given by the following formula:

$$e_q(t) = x(t) - x_q(t) \tag{2.1}$$

where $x(t)$ is the original continuous amplitude, and $x_q(t)$ is the quantized amplitude.

2.1.3 Sampling Rate

The sampling rate, or sample rate, is the number of samples taken per second to represent the audio signal. It is measured in Hertz (Hz) and directly affects the audio quality and the highest frequency that can be accurately represented in the digital audio file. The Nyquist-Shannon sampling theorem states that to accurately

represent a signal, the sampling rate must be at least twice the highest frequency present in the signal:

$$f_s \geq 2f_{max} \tag{2.2}$$

where f_s is the sampling rate and f_{max} is the highest frequency present in the audio signal.

For example, the standard sampling rate for CD-quality audio is 44,100 Hz, as it allows for accurate representation of audio signals up to 22,050 Hz, which covers the entire range of human hearing (20 Hz to 20,000 Hz).

In summary, digital audio representation relies on waveform discretization using bit depth and sampling rate. Understanding these fundamental aspects of music representation is essential for designing effective music information retrieval and generation models that can accurately process and analyze audio signals.

2.2 MIR and AMT

The field of Music Information Retrieval (MIR) encompasses various tasks related to the analysis and understanding of music, including Automated Music Transcription (AMT). AMT is the process of converting an audio recording of a musical performance into a corresponding sheet music representation or - at the very least - harmonic chord labels. This involves the accurate detection of various musical attributes, such as pitch, timing, and velocity (dynamics), which are essential for a complete and accurate transcription.

Currently, state-of-the-art (SOTA) AMT systems perform well in transcribing piano-only music, capturing both pitch and velocity information. Piano music is often considered a simpler case due to its well-defined, percussive sound and relatively limited polyphony compared to orchestral music. For orchestral music, where multiple instruments with complex timbres and overlapping notes are present, the performance of existing AMT systems is more limited. In such cases, the systems can generally detect pitch information but struggle to accurately estimate other attributes like velocity or instrument identification.

A significant limitation of current SOTA AMT systems is the absence of a reinforcement learning (RL) component or feedback loop, which could enable the system to correct its transcription as it processes the music. In human transcription, listeners can identify errors in their transcription as they compare it to the music being transcribed, and iteratively refine their transcription to improve its accuracy. Incorporating an RL-based approach in AMT systems would allow them to mimic this human-like adaptive behavior, potentially leading to more accurate and robust transcriptions.

In summary, while the current state-of-the-art AMT systems show promising results in specific domains, such as piano-only music, there is significant room for improvement, particularly in more complex scenarios like orchestral music with polyphony and subtle variations in tuning and tempo. By integrating reinforcement learning components and feedback loops into AMT systems, it may be possible to enhance their performance, enabling them to produce more accurate and complete transcriptions across a wider range of music.

2.2.1 Pitch and Frequency

Pitch is a perceptual property of sound that allows us to differentiate between high and low sounds. In music, pitch refers to the musical note or tone produced by a vibrating source, such as a vocal cord or a string on a musical instrument. The pitch of a sound is directly related to its frequency, which is the number of oscillations or cycles of a waveform per second, measured in Hertz (Hz).

The fundamental frequency, also known as the first harmonic, is the lowest frequency of a periodic waveform and is the most significant contributor to the perceived pitch of a sound. The frequencies that are integer multiples of the fundamental frequency are called harmonics or overtones. The presence of harmonics is responsible for the unique timbre or tone quality of a sound, which allows us to distinguish between different instruments or voices even when they produce the same pitch.

Pitch Notation

Pitch is typically represented using a system of musical notation, which assigns a unique symbol or note name to each pitch within a specific range or scale. The most common notation system is the staff notation, where pitches are represented by placing note symbols on a series of horizontal lines called a staff. The position of the note on the staff indicates its pitch relative to a reference note, usually C or A.

Accurate pitch detection is a crucial task in various music information retrieval applications, such as automated music transcription, melody extraction, and key detection. Numerous pitch detection algorithms (PDAs) have been developed to estimate the fundamental frequency of a given audio signal. These methods can be broadly classified into time-domain, frequency-domain, and joint time-frequency domain approaches. Some popular PDAs include the Autocorrelation Method, the YIN Algorithm, and the Harmonic Product Spectrum.

In Western music, the standard reference pitch is A4, which has a frequency of 440 Hz. The pitches of other notes can be determined using the equal-tempered scale, where the frequency ratio between adjacent semitones is constant, given by the twelfth root of 2 (approximately 1.0595). This scale divides an octave, the interval between two pitches with a frequency ratio of 2:1, into 12 equal parts called semitones.

2.2.2 Tempo

Tempo is a fundamental aspect of music, often described as the "speed" or "pace" at which a piece of music is played. In a more formal context, tempo refers to the rate of the underlying beat, or pulse, that governs the rhythmic structure of a musical composition. It is typically expressed in beats per minute (BPM), a measure of the number of beats occurring within a 60-second interval.

The concept of tempo has been an integral part of music since antiquity. However, it was not until the 16th century that composers began to indicate tempo explicitly, using words to describe the desired pace of the music. The invention of the metronome in the early 19th century, attributed to Johann Nepomuk Maelzel, provided a more objective and precise method for specifying tempo. With the metronome, a mechanical device that produces an audible click at regular intervals, composers could now provide exact BPM values for their music.

Challenges in Precise Tempo Formulation

Despite the introduction of the metronome, precise formulation of tempo remains a challenge, particularly in the realm of computer signal processing. There are several factors contributing to this difficulty:

- **Variability:** Tempos may vary throughout a piece, even within a single measure or bar, as a result of expressive performance practices or musical notation (e.g., accelerando or ritardando). This variability can be challenging for computers to track and represent accurately. This concept of both the discrete and a continuous tempo modulation possibility is graphically illustrated on Figure 2.1, and can be imagined as sudden or prolonged speed-ups and slow-downs in a piece of music.
- **Polyrhythm and Syncopation:** Complex rhythmic structures, such as polyrhythms (multiple rhythms played simultaneously) and syncopation (rhythms emphasizing off-beat notes), can create ambiguity in determining the underlying tempo.
- **Subjectivity:** There is often an element of subjectivity in perceiving tempo, with listeners potentially experiencing the same piece at slightly different tempos.

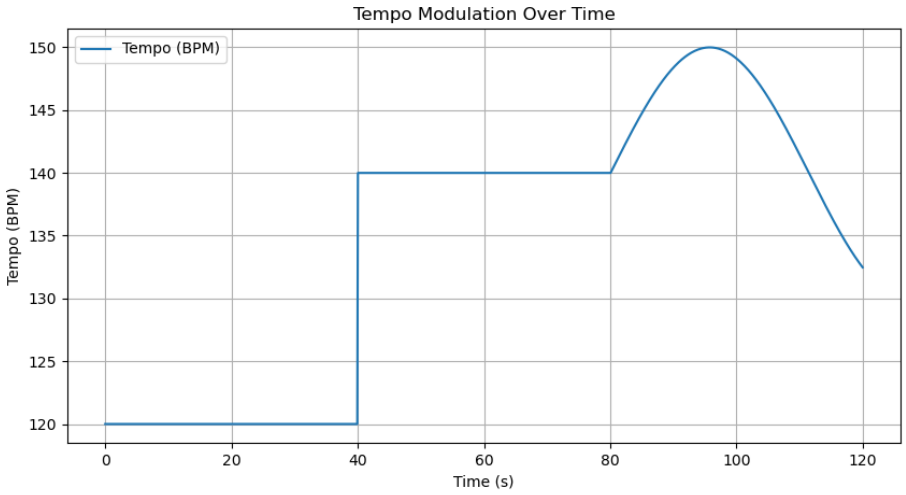

FIGURE 2.1: An illustration of possible tempo modulation patterns

Psychoacoustic Perception of Tempo

In order to develop effective music signal processing techniques, it is crucial to consider the psychoacoustic aspects of tempo perception. Humans are highly adept at perceiving and processing tempo, often unconsciously synchronizing movements, such as tapping a foot or nodding one's head, to the beat of the music. This phenomenon, known as entrainment, suggests that the human brain is capable of processing tempo in a highly efficient and robust manner, and - even more importantly - is likely doing this by utilizing some relatively straightforward neural processing chain without introducing unnecessary complexity. Had it been not the case, even a trivial task of rhythmically tapping along with a piece of music would inevitably

produce mental strain in the person doing that as opposed to being a simple task most often not requiring any coordinated mental effort from the listener whatsoever.

Research in the field of music cognition and psychoacoustics has identified several factors that influence human tempo perception:

- **Beat Salience:** The prominence of the beat, determined by factors such as loudness, duration, and spectral content, can impact the ease with which listeners perceive tempo.
- **Musical Context:** Listeners' perception of tempo can be influenced by the specific melodic, harmonic, and rhythmic structures present in a piece of music.
- **Cultural and Individual Factors:** Cultural background and individual listening habits can also play a role in shaping tempo perception. Western music is famous for its *well-tempered* pieces, in which the tempo guidelines are followed with exceptional strictness. A concrete definition of tempo, or other strong beat-related features in general, meanwhile, is completely foreign to Kazakh traditional music, where continuous modulation is the norm rather than the exception to be accounted for. Nevertheless, for the purposes of consistency, an implicit assumption of Western music will be made throughout this thesis.

2.2.3 Psychoacoustic Perception and its Importance for MIR

Psychoacoustic or *relative* perception is a crucial aspect of human auditory processing that has remained largely unaddressed in the field of Music Information Retrieval (MIR). The ability to recognize and understand musical structures, such as melodies or harmonic progressions, regardless of their absolute pitch values is a fundamental skill in human musical perception. This relative perception allows us to identify the same melody or progression when it is transposed to a different key, as our brains automatically adjust to the new context and focus on the relationships between the notes rather than their absolute pitch values.

This aspect of human perception is essential to our understanding and appreciation of music, as it allows for a more flexible and adaptable listening experience. However, current state-of-the-art architectures in MIR often fail to capture this important characteristic of human auditory processing. Most existing methods represent musical information in an absolute form, making it difficult for the system to recognize transposed versions of the same melody or progression as being related. Consequently, the same melody presented in different keys will map to entirely different representations in the system, which severely limits its ability to generalize and accurately process musical information.

To illustrate the importance of relative perception, consider a simple melody that a listener can easily recognize. When the melody is played in a different key, a human listener can still identify the melody despite the change in pitch. This ability comes from our innate understanding of the relationships between the notes in the melody, such as intervals and rhythmic patterns, which remain constant even as the absolute pitch values change. In contrast, an MIR system relying on absolute pitch representation (e.g. almost every SOTA MIR system) would likely fail to recognize the transposed melody as the same tune, as its internal representation would be entirely different.

Incorporating relative perception into MIR systems is crucial for enhancing their accuracy and generalizability. By developing methods that can account for the relationships between musical elements, rather than just their absolute values, MIR

systems can better emulate human auditory processing and improve their performance in tasks such as transcription, key estimation, and harmonic analysis. Addressing this issue will likely require novel approaches and architectures that can effectively capture the intricacies of relative perception and enable MIR systems to process musical information in a more human-like manner.

2.3 Music Generation

In addition to music information retrieval, there has been a growing interest in music generation, with many researchers and developers focusing on creating new music compositions or arrangements using state-of-the-art machine learning techniques. While the potential applications of music generation are exciting and can potentially revolutionize the way music is created and consumed, this focus on music generation has inadvertently led to less attention being paid to automated music transcription (AMT).

This disproportionate focus on music generation could be considered a misguided approach for several reasons. First, a solid foundation in music transcription is crucial for developing more advanced music generation techniques. Accurate transcription can provide a wealth of information about musical patterns, structures, and styles, which can be invaluable in training more sophisticated generative models. By improving transcription, researchers can better understand the underlying structure of music and consequently develop more advanced and expressive generative models.

Second, accurate music transcription has a wide range of practical applications that can benefit various stakeholders in the music industry. For example, music educators can use transcription tools to provide students with accurate sheet music, composers can analyze and study the works of others, and musicologists can explore the structural and theoretical aspects of music in greater depth. By focusing primarily on music generation, researchers may inadvertently neglect the development of tools that can have a significant impact on the music community.

Lastly, the development of accurate music transcription algorithms can help address some of the limitations and challenges currently faced by music generation models. For instance, improving transcription accuracy can lead to better representations of musical data, which can then be used as input for generative models. Furthermore, by focusing on the nuances of music transcription, researchers can gain a deeper understanding of the complexities of music theory and human perception, which can ultimately lead to more expressive and musically rich generative models.

2.4 The *Transformer* Architecture

The Transformer neural architecture, introduced by Vaswani et al. in 2017 [1], has revolutionized the field of natural language processing and has since been applied to various other domains, including image processing and, recently, music information retrieval. The Transformer model is based on the self-attention mechanism, which allows the model to weigh and aggregate information from different positions in the input sequence, enabling it to capture long-range dependencies and inter-token relationships effectively. This section provides a detailed explanation of the Transformer architecture, focusing on its key components and underlying mathematical principles. A somewhat simplified and generalized type of description of the transformer's advantages and inner workings is provided further in section 5.1.1.

2.4.1 The Self-Attention Mechanism

At the heart of the Transformer architecture lies the self-attention mechanism, which computes a weighted sum of input representations, allowing the model to selectively focus on different positions in the input sequence. Given an input sequence $X = x_1, x_2, ..., x_n$, the self-attention mechanism first computes three vectors for each token x_i: the query vector q_i, the key vector k_i, and the value vector v_i. These vectors are obtained by applying linear transformations to the input representations:

$$q_i = W^Q x_i, \quad k_i = W^K x_i, \quad v_i = W^V x_i, \tag{2.3}$$

where $W^Q, W^K, W^V \in \mathbb{R}^{d \times d}$ are weight matrices representing the learnable parameters of the self-attention mechanism.

Next, the self-attention mechanism computes the attention weights between all pairs of tokens in the input sequence. The attention weight a_{ij} between tokens i and j is calculated as the scaled dot product of their corresponding query and key vectors:

$$a_{ij} = \frac{q_i^\top k_j}{\sqrt{d}}, \tag{2.4}$$

where d is the dimensionality of the query and key vectors. The scaling factor $\sqrt{d}$ is introduced to prevent the dot product from becoming too large, which could lead to vanishing gradients during training.

The attention weights are then normalized using the softmax function, ensuring that they sum to one:

$$\alpha_{ij} = \frac{\exp(a_{ij})}{\sum_{k=1}^{n} \exp(a_{ik})}. \tag{2.5}$$

Finally, the self-attention mechanism computes the output vector z_i for each token x_i by taking a weighted sum of the value vectors of all tokens in the input sequence:

$$z_i = \sum_{j=1}^{n} \alpha_{ij} v_j. \tag{2.6}$$

2.4.2 Multi-Head Attention: Nodes talking to each other

To capture different aspects of the input sequence, the Transformer architecture employs multiple self-attention mechanisms in parallel, referred to as "heads." Each head h computes its own set of query, key, and value weight matrices, W_h^Q, W_h^K, W_h^V, and produces a separate output representation $z_i^{(h)}$ for each token x_i. The output representations from all heads are concatenated and linearly transformed to produce the final output vector for each token:

$$z_i = W^O \left[z_i^{(1)} \oplus z_i^{(2)} \oplus ... \oplus z_i^{(H)} \right], \tag{2.7}$$

Table 2.1 provides a comparative overview of major differences between the transformer architecture and an older Long Short-term Memory-based (LSTM) approach.

TABLE 2.1: Comparison of LSTM and Transformer Architectures

Feature	LSTM	Transformer
Basic Unit	Recurrent Neural Network (RNN)	Self-attention mechanism
Memory	Maintains hidden states as memory through time	No explicit memory; relies on attention mechanism for context
Sequence Processing	Processes sequences sequentially	Processes sequences in parallel
Parallelization	Limited due to the sequential nature of processing	Highly parallelizable, which allows for faster training and inference
Time Complexity	$O(n)$ for n time steps	$O(1)$ since all tokens are processed simultaneously
Space Complexity	$O(n)$ for n time steps	$O(n^2)$ for n tokens due to the attention matrix
Long-Range Dependencies	Handles long-range dependencies using memory gates (cell state, input gate, forget gate, and output gate)	Handles long-range dependencies using the self-attention mechanism
Architecture	Unidirectional or bidirectional	Encoder-decoder or encoder-only (e.g., BERT)
Use Cases	Sequence-to-sequence tasks, time series prediction, text generation	Natural language understanding, translation, text summarization, image captioning, and more
Training Stability	More stable training due to recurrent structure	Less stable training; requires techniques like layer normalization and learning rate warmup
Key Innovations	LSTM cells with memory gates to address the vanishing gradient problem	Self-attention mechanism and positional encoding to capture context and order
Popular Models	Basic LSTM, BiLSTM, Stacked LSTM	Original Transformer, BERT, GPT, T5, and more

Chapter 3

Literature Review

3.1 *Divide et Impera:* A two-staged review

Given the interdisciplinary nature of this thesis and the need to establish a strong foundation in MIR, psychoacoustics, and advanced neural architectures, the literature review is divided into two distinct stages. The first stage focuses on MIR, audio processing, and psychoacoustic research, with an emphasis on automated music transcription (AMT) techniques and their limitations. The second stage delves into the study of the transformer architecture and machine learning in general, exploring the applications in various domains and highlighting potential areas of adaptation for subsets of MIR tasks.

The rationale behind structuring the literature review in this manner stems from the need to first establish a comprehensive understanding of the current state-of-the-art in MIR, AMT techniques, and psychoacoustic research. This understanding is crucial for identifying the limitations and challenges faced by existing methods, which will inform the subsequent exploration of advanced neural network architectures and their potential applications in MIR tasks.

By examining the key/harmony and tempo-related task subsets of MIR, AMT-related research, and psychoacoustic literature in the first stage, the aim is to elucidate the intricacies of human auditory perception and the challenges in developing effective representations of audio data. This stage will also provide insights into the psychoacoustic principles that govern the human perception of sound, allowing to better understand how these principles can be incorporated into the design of neural architectures tailored for MIR tasks. This comprehensive understanding will set the stage for the second part of the literature review, where the potential of transformer architectures and other advanced neural networks to address the identified challenges and limitations is investigated.

The second stage of the literature review is necessary to establish a thorough understanding of transformer architectures and related neural networks, which have demonstrated considerable success in various domains, such as natural language processing and image processing. By examining these architectures in-depth, it's possible to identify their strengths and limitations, as well as the extent to which they can be adapted for MIR tasks that leverage psychoacoustic principles.

This two-stage approach to the literature review is designed to provide a comprehensive overview of MIR, psychoacoustic research, and advanced neural network architectures, while also facilitating the identification of gaps in the literature that can be addressed by the proposed research. By adopting this structure, the goal is to ensure that the thesis is grounded in a solid understanding of the current state-of-the-art and is well-positioned to make significant contributions to the field of MIR.

3.2 The ABCs of MIR and AMT

3.2.1 Background and Goals

Several algorithms exist to estimate the tempo and key of a music piece, such as some classical correlational ones used for key estimation (e.g. the Krumhansl–Kessler, Albrecht–Shanahan, Temperley algorithms), or more novel learning-based ones (e.g. convolutional and/or recursive neural networks) used for both tasks. However, most of these approaches are designed to work on a fixed tempo or key, making them ill-suited for audio waveforms with metric and/or harmonic (key/tonality) modulations present. This is quite surprising, given that most of these algorithms are not inherently limited to such temporally stable conditions, since even a sequential/window-based application of some of them can potentially make them adjust to changes in tempo or key present over time.

To address these issues, adaptive time-varying tempo and key/tonality estimation methods have been proposed in recent years, but their implementation was not up-to-par, and most of them stayed more or less theoretical. There is promising potential to these methods to adapt to tempo and key changes in real-time and provide accurate estimates even in the presence of tuning distortions or noise.

The objective of this literature review stage is to compare and contrast existing adaptive yet theoretical/non-implemented and non-adaptive techniques for tempo and key estimation in order to gain a comprehensive understanding of the approaches used for these MIR tasks. In particular, the aim is to analyze the relative accuracy, complexity, and potential for introducing adaptivity of each technique, as well as to identify common patterns in waveform pre-processing and other relevant steps, and to possibly identify potential areas of improvement. Ultimately, the aim of this substage is to identify either a single approach or a combination of approaches that can be implemented and optimized for adaptivity in the most applicable way. By achieving these objectives, this review and the will provide a foundation for the development of an accurate and adaptive system for tempo and key estimation, which can improve the performance and flexibility of other MIR systems reliant on these tasks.

3.2.2 The Review Process

The review process itself was carried out in a selective manner to ensure that only the publications with usable and promising approaches were included in the analysis. To start, the process of searching for literature was done using several main search platforms: Google Scholar, Music Information Retrieval Exchange" (MIREX) conference archive and Arxiv.org. These databases were selected as they provide a comprehensive coverage of scientific publications from various fields, including signal processing and music information retrieval. Arxiv, in particular, mostly provided publications dealing with statistical and learning-based methods.

To narrow down the search results, inclusion and exclusion criteria were set. Only publications that dealt with tempo and key estimation/detection, and that used relevant or clearly implementable signal processing techniques were considered. In addition, only publications in English language and that were published from 2005 up to the present were included in the analysis. These criteria helped to ensure that the publications were relevant and up-to-date. In the end, the bulk of the reviewed publication were found by following the references mentioned in other papers.

A two-fold search process was employed: firstly, with search terms used for the review process being "adaptive tempo estimation", "pitch chroma space", "adaptive key estimation", "adaptive tonality detection", "adaptive key detection" and "adaptive music information retrieval". This initial stage provided very little usable literature, so an additional search query with the same keywords, but excluding "adaptive" was carried out. Special attention was paid to reviews, as they often referenced other useful publications.

To assess the credibility of the selected publications, several metrics were used. The impact factor of the journal in which the publication appeared, the number of citations it had, and the reputation of the authors and their institutions were all taken into account. In addition, the relevance of the publication to the research question, the soundness of the research methodology used, and the quality of the results obtained were also considered. Nevertheless, even publications lacking in some aspect (technicality/accuracy) sometimes provided useful insights in other regards - future references to research as well as approaches better not followed.

Lastly, the publications selected were then organized into groups based on their approach to tempo and key estimation/detection. The comparative analysis of these publications was based on several metrics, including relative accuracy, complexity, and potential for introducing a degree of true adaptivity. Some common preprocessing patterns were identified.

3.2.3 Results and Discussion

Papers published from 2005 to 2022 were selected for the first review stage, including several survey and review papers, and one survey specifically dealing with adaptive algorithms in MIR [2]. With this study in particular, it became apparent that the term "adaptivity" was not always used in a consistent and appropriate manner across the different sources. Specifically, some publications that were identified as relevant to the study focused on a concept of "adaptivity" that was not directly related to signal processing techniques, but instead referred to a completely different meaning (e.g. user-adaptive recommendation systems). These sources describe systems that adjust recommendations based on a user's behavior or preferences, rather than systems that are adaptive to changes in the input signals. Despite this distinction, these publications were included in the review process, but were ultimately excluded from the analysis of the performance of adaptive signal processing algorithms in music information retrieval tasks.

Additionally, it was discovered that the "Music Information Retrieval Exchange" (MIREX) conference website contained a number of publications that were of subpar quality. Specifically, the categories of tasks of interest in "Audio Key Detection" and "Audio Chord Estimation" were found to contain particularly low-quality papers. These papers were poorly formatted and did not present any meaningful results, which limited their credibility and usefulness for the present study. As such, care was taken to ensure that only high-quality sources were included in the review process, and those publications that did not meet the necessary standards were excluded.

The following paragraphs outline the most important insights and approaches in key and tempo estimation tasks, respectively.

3.2.4 Key Estimation

Since key estimation can be described as a temporally-varying 24-dimensional classification problem, most studies under consideration make use of the so-called *chromagram* - a Pitch Class Profile features (PCP) frame-based representation based on the Short-time Fourier Transform (STFT). There were slight differences in implementing the PCP chromagram in reviewed literature - variations in frequency bins, log or mel-scale utilization, as well as some audio pre-processing differences (preferred sample rate after downsampling, envelope properties, hop length, etc.). The resulting chromagrams were then used as an input into auto-and cross-correlation matrix-based architectures to output the final key estimate in a given time-frame (for local estimation) [3, 4]. This approach was found to suffer from a major downside - the correlation matrices have to be constructed for specific chord/key labels and tuning, requiring additional pre-processing, although this issue was more pronounced in chord estimation tasks, and not as critical for key estimation.

In a somewhat different category, the authors of [5] propose a technique for key detection that relies on "fuzzy" methods: an analysis for chromatic pitch-class determination, with the addition of an adaptive level weighting. The aim of the weighting step is to enhance the robustness of pitch values obtained from the signal's Fast Fourier Transform. The pitch values are then used to derive pitch class profiles, which are in turn employed to determine the key. To determine the weight for each predefined pitch range, the technique makes use of information about the signal density in that range [6].

Unfortunately, several publications mentioning adaptivity in tonal and harmonic (chord) detection refer not to the temporal modulations of these qualities, but to some time-invariant deviation in the nature of the sound-wave, often trying to "adapt" to a different tuning frequency or some other parameter of secondary interest [7]. Nevertheless, certain approaches employed in such papers proved promising - the confusion, correlation and transition matrices in [8] and [9].

Quite often, *chroma-vectors* in additional to chromagrams were often used in combination with some Hidden Markov Model-based architecture to estimate key to various degree of success [10]. Another wide category of key estimation approaches relies on music theory knowledge (e.g. [11] mentions multiple sources using the *circle of fifths* information in different ways).

3.2.5 Tempo estimation

In time-domain, adaptive thresholding was used in [12] to identify peaks in the audio input signal. Publicly available tools such as aubio and MIRToolbox have offered peak picking methods using dynamic thresholds, where the adaptable system is a threshold function. This function's dynamic threshold parameter is derived from a short, sliding window of the input values, providing context. While peak picking with dynamic thresholding is commonly used for onset detection, it has also been applied to audio identification, audio to score matching, and audio recording segmentation.

A similar time-domain onset-based approach is proposed in [13] for extracting tempo information in a somewhat adaptive manner. Their method uses an onset-dependent adaptive window size, assuming that note onsets are the main source of tempo information. The authors suggest that a larger window is needed in sections with fewer notes played to obtain a more accurate tempo estimation, while a smaller

window is sufficient in other cases. Instead of using a fixed number of input values to specify window size, the approach uses a fixed number of "inter-onset-intervals".

In addition to that, utilizing normal/reassigned spectral energy flux and Mel-frequency cepstral coefficients-based spectrograms was a common practice for tracking tempo variations. A most accurate local tempo estimation techniques with *Fourier autocorrelation tempograms* in [14] then used temporal variations to re-weight the auto-correlation matrices by log-normal priors, smooth them out with log-normal distribution, and using the local maximum of the weighted correlation to find the best peak value in a given local window.

Other approaches showed similarly promising results in tempo estimation tasks. In particular, time-domain analysis and introducing custom error/loss function based on periodicity information. The tempo, meter, and beat subdivision paths are iteratively, although non-adaptively chosen in such a way that best explain the observed time-domain periodicities [15].

3.2.6 Shortcomings

One issue that emerged is the inconsistency in the use of datasets for evaluating the performance and accuracy of various methods. Many studies employed their own datasets, which limits the comparability of results and prevents the establishment of a universal benchmark for evaluating performance. Moreover, it is noteworthy that despite the strong theoretical foundations underpinning state-of-the-art approaches, the observed accuracy rates have been limited to the range of 80-85% [16]. The reasons for this ceiling remain unclear, as literature rarely provides a comprehensive account of the factors contributing to this limitation. Hence, there is a need for further research to explore the underlying causes and to develop novel methods that can overcome these challenges, enabling more accurate and reliable tempo and key estimation techniques, possibly with the introduction of truly adaptive approaches as opposed to current semi-adaptive ones.

3.3 Psychoacoustics

This section highlights key sources that informed the author's understanding of psychoacoustics, providing a foundation for making the interdisciplinary connections and hypotheses. Chew's (2013) book *Mathematical and Computational Modeling of Tonality: Theory and Applications* [17] offered insights into tonal perception and cognition. The interdisciplinary approach combined music theory, mathematics, and computer science, enabling deeper comprehension of pitch, interval, consonance, and dissonance relationships.

Cook and Tobias's (1999) book *Music, Cognition, and Computerized Sound: An Introduction to Psychoacoustics* [18] a comprehensive introduction to psychoacoustics. It contextualized fundamental concepts, such as pitch perception, loudness, timbre, and auditory scene analysis, within cognitive psychology.

Smyth's course notes for *Music 175: Psychoacoustics* provided supplementary information, consolidating key concepts and theories. Available here, the notes proved useful in gaining a deeper understanding of the human auditory perception intricacies.

The author's advanced music theory knowledge also enabled connections between theoretical constructs and practical applications, such as the role of psychoacoustic principles in musical analysis and the importance of relative perception.

3.4 The Revolutionary Impact of Transformers: GPT-3, PaLM, Jukebox, AudioLM

The introduction and widespread adoption of Transformer architecture have led to the development of several groundbreaking models, pushing the frontiers of machine learning and artificial intelligence. This brief section is dedicated to the revolutionary impact of some of these models, including GPT-3, PaLM, Jukebox, and AudioLM, which have demonstrated remarkable capabilities across various domains.

3.4.1 GPT-3

GPT-3 (Generative Pre-trained Transformer 3), developed by OpenAI, is one of the largest and most powerful language models to date, boasting 175 billion parameters [19]. GPT-3 has shown remarkable performance in a wide range of NLP tasks, such as text generation, summarization, translation, and question-answering, often with little to no fine-tuning. Its ability to generate coherent and contextually relevant text has sparked interest in various applications, including chatbots, code generation, and creative writing assistance.

3.4.2 PaLM

The Pre-trained Language Model (PaLM) by Facebook AI [20] is another notable development in the field of large-scale Transformer models. PaLM is designed to learn high-level abstractions and reasoning abilities by leveraging unsupervised and supervised pre-training on diverse and large-scale datasets. PaLM has demonstrated strong performance across a range of tasks, including natural language understanding, question-answering, and commonsense reasoning, showcasing its potential to serve as a foundation for advanced AI systems.

3.4.3 Jukebox

Jukebox, developed by OpenAI, is a Transformer-based model that generates music in various genres and styles, complete with lyrics and singing [21] Trained on a dataset of 1.2 million songs, Jukebox employs a hierarchical VQ-VAE to compress raw audio data into a more manageable format and then utilizes a powerful autoregressive Transformer model to generate new music samples. Jukebox represents a significant advancement in the field of music information retrieval, demonstrating the potential of Transformer architecture to create high-quality, diverse, and coherent musical content.

3.4.4 AudioLM

AudioLM [22] is a self-supervised Transformer model designed for learning representations from raw audio data. By leveraging a contrastive learning approach, AudioLM learns to map audio data to meaningful embeddings that can be used for various downstream tasks, such as speech recognition, speaker identification, and audio classification. The success of AudioLM highlights the potential of Transformer models to learn from large-scale, unlabelled audio data, paving the way for more effective and versatile audio processing systems.

Chapter 4

Implementation and Results

4.1 Dataset selection

In this section, the process of selecting suitable datasets for the study of AI in music, specifically focusing on Music Information Retrieval (MIR) tasks, is described. The criteria for dataset selection are outlined, followed by an overview of the selected datasets and their descriptions.

4.1.1 Criteria for Dataset Selection

In order to select appropriate datasets for the study, the following criteria were considered:

1. **Diversity:** A variety of music styles, genres, and instruments should be included in the dataset to allow for a comprehensive evaluation of the model's performance across diverse musical contexts.
2. **Ground truth annotations:** Ground truth annotations, such as pitch, onset times, and offset times for each musical note, should be provided to facilitate supervised learning and evaluation.
3. **Recording quality:** High-quality audio recordings should be preferred to minimize the impact of recording artifacts on the performance of the MIR system. In case of non-recorded MIDI-datasets the alignment and the quality of captured MIDI metadata was considered instead.
4. **Availability:** The dataset should be publicly available and accessible for research purposes.

4.1.2 Selected Datasets

Based on the criteria outlined in Section 4.1.1, the following datasets were selected for this study:

- **MAPS (MIDI Aligned Piano Sounds) Dataset** [23]: The MAPS dataset consists of piano music recordings and corresponding MIDI files. A variety of music styles, ranging from classical to jazz, is included in the dataset. High-quality recordings and ground truth annotations are provided, making it suitable for evaluating the performance of Automated Music Transcription (AMT) systems.
- **MedleyDB** [24]: MedleyDB is a dataset of multitrack audio recordings featuring a diverse collection of genres and instruments. Annotations for melody,

vocal activity, and instrument activations are provided, enabling the evaluation of the proposed model across various MIR tasks.

- **MusicNet** [25]: MusicNet is a large-scale dataset of classical music recordings with fine-grained annotations of pitch, onset, and offset times. Its diversity of compositions and high-quality annotations make it suitable for training and evaluating AMT systems.
- **Million Song Dataset (MSD)** [26]: Created in 2011, the Million Song Dataset is a large-scale collection of metadata for over one million contemporary popular music tracks. It contains features such as song duration, loudness, tempo, and key, as well as artist and song similarity metrics. While it does not include the actual audio files, the dataset is beneficial for tasks like music recommendation, genre classification, and artist identification.
- **FMA: A Dataset for Music Analysis** [27]: The Free Music Archive (FMA) dataset, released in 2017, contains 106,574 tracks from 16,341 artists across 161 genres. Along with the audio files, the dataset provides metadata like track, album, and artist information. The FMA dataset is suitable for tasks such as genre classification, music recommendation, and unsupervised feature learning.
- **Lakh MIDI Dataset (LMD)** [28]: The Lakh MIDI Dataset, released in 2016, is a collection of 176,581 unique MIDI files, which is equivalent to 22.6 years of continuous music. It is intended for tasks such as symbolic music modeling, computational musicology, and music generation. Lakh MIDI was found to most closely resemble the envisioned psychoacoustically-labeled dataset, largely due to its synthetic universality, inspiring the dataset augmentation framework laid out in more detail in the following section.

The datasets mentioned above provide a comprehensive set of music data that cover a wide range of musical styles, instruments, and MIR tasks. By utilizing these datasets, a thorough evaluation of the proposed model in various musical contexts can be conducted. Nevertheless, despite the author's best efforts, no existing dataset was found to cover the psychoacoustic and relative perception aspect of music processing, substantiating a need for if not developing, then at the very least laying a theoretical foundation framework on how to achieve such a goal, which is exactly the subject of the following section.

4.1.3 Data Augmentation for Relative Music Perception: Theory

Incorporating relative perception into MIR systems is essential for enhancing their accuracy and generalizability. To train a model that accounts for relative perception, it is important to augment the dataset with transposed versions of the original music pieces, ensuring that the model learns to recognize melodies and harmonic structures independent of their absolute pitch values. In this section, a dataset augmentation algorithm is proposed, which generates transposed versions of the input data, focusing on relative perception with regard to melody and harmonic functions (e.g., tonic, subdominant, and dominant).

Algorithm 1 Dataset Augmentation for Relative Music Perception

Require: D (original dataset), k (number of transpositions)
Ensure: $D_{augmented}$ (augmented dataset)
1: $D_{augmented} \leftarrow D$
2: **for** each $sample$ in D **do**
3: **for** $i \leftarrow 1$ to k **do**
4: $sample_{transposed} \leftarrow \text{TRANSPOSE}(sample, i)$
5: $\text{ADDTODATASET}(D_{augmented}, sample_{transposed})$
6: **end for**
7: **end for**
8: **return** $D_{augmented}$

The proposed algorithm takes as input an original dataset D and the number of transpositions k. For each sample in the dataset, the algorithm generates k transposed versions by shifting the pitch values by a specified interval. These transposed samples are then added to the augmented dataset $D_{augmented}$, which is returned as the output.

By including transposed versions of the original music pieces in the training data with corresponding labels, the model is encouraged to learn the underlying relative relationships between musical elements, such as intervals and rhythmic patterns, rather than relying solely on absolute pitch values. This approach helps to emulate human auditory processing, allowing the model to better recognize and understand melodies and harmonic structures regardless of their absolute pitch values.

Incorporating dataset augmentation for relative music perception into the training process of the proposed psychoacoustically-adjusted transformer model is expected to improve its performance on MIR tasks.

4.2 Adaptive Strong Beat and Tempo Estimation

Strong beat and tempo estimation are critical components in music information retrieval. Accurate estimation of strong beats and tempo enables better transcription, synchronization, and alignment of musical elements. In the context of this thesis, the aim is to either develop or find a strong beat and tempo estimation algorithm that can account for changes in tempo and beat strength throughout a piece of music. In this section, both existing and a new proposed adaptive algorithm for strong beat and tempo estimation are presented.

4.2.1 Existing Algorithms

Several existing algorithms for strong beat and tempo estimation are based on the autocorrelation function, which measures the similarity between a signal and a time-shifted version of itself. The autocorrelation function $R(\tau)$ for a discrete signal $x[n]$ can be defined as:

$$R(\tau) = \sum_{n=0}^{N-\tau-1} x[n] \cdot x[n+\tau] \tag{4.1}$$

where τ is the time lag, and N is the number of samples in the signal.

For the generalized autocorrelation-based tempo estimation algorithm of Algorithm 2, the autocorrelation function is applied to the onset strength envelope, which

Algorithm 2 Strong beat and tempo estimation using autocorrelation function

```
procedure TempoEstimation(x[n], N)
    Compute the onset strength envelope O[n]
    for τ = 1 to N − 1 do
        R(τ) ← 0
        for n = 0 to N − τ − 1 do
            R(τ) ← R(τ) + O[n] · O[n + τ]
        end for
    end for
    Identify the peaks in R(τ)
    Estimate the underlying tempo based on the peaks
    return estimated tempo in bpm
end procedure
```

represents the energy changes over time in the audio signal. By analyzing the periodicity of the onset strength envelope, the underlying tempo can be identified.

One popular tempo estimation algorithm is the Tempogram, which calculates the short-time Fourier transform (STFT) of the onset strength envelope to obtain a time-frequency representation of the rhythmic content. The dominant tempo is then identified by finding the highest energy bin in the Tempogram.

4.2.2 Proposed Adaptive Algorithm

The proposed adaptive algorithm for strong beat and tempo estimation combines the autocorrelation-based approach with reinforcement learning to account for changes in tempo and beat strength throughout the music. The key idea is to continuously update the tempo and beat estimates based on an error function that measures the deviation between the estimated beats and the actual beats in the audio signal.

1. Compute the onset strength envelope of the audio signal.
2. Initialize the tempo and strong beat estimates using an existing algorithm, such as the Tempogram.
3. For each time frame, calculate the autocorrelation function of the onset strength envelope within a window centered at the current time.
4. Identify the highest peak in the autocorrelation function, and update the tempo estimate based on the peak's position.
5. Update the strong beat estimates by aligning them with the updated tempo.
6. Calculate the error function, which measures the deviation between the estimated beats and the actual beats in the audio signal.
7. Use reinforcement learning to minimize the error function and continuously update the tempo and strong beat estimates.

The error function $E(t)$ can be defined as:

$$E(t) = \sum_{i=1}^{N} (b_{actual}(t_i) - b_{estimated}(t_i))^2 \tag{4.2}$$

where $b_{actual}(t_i)$ and $b_{estimated}(t_i)$ are the actual and estimated beat positions at time t_i, respectively, and N is the number of time frames.

By minimizing the error function, the proposed adaptive algorithm can accurately track tempo and strong beat changes throughout the music, leading to improved performance in music information retrieval and generation tasks.

4.2.3 Proposed RL Algorithm

Algorithm for Tempo Estimation using Reinforcement Learning:

1. **Preprocess the input music signal** (e.g., convert to spectrogram or chromagram representation).
2. **Initialize the reinforcement learning agent** with a suitable neural network architecture (e.g., a deep Q-network or a policy gradient network) and a predefined tempo estimation action space (e.g., discrete tempo values or continuous tempo range).
3. **For each time step in the music signal**:
   (a) The agent observes the current state (e.g., a window of the preprocessed music signal) and selects an action (tempo estimate) based on its current policy.
   (b) The agent receives a reward based on the error between its tempo estimate and the ground truth tempo (if available) or an alternative measure of tempo consistency (e.g., beat alignment or autocorrelation).
   (c) The agent updates its policy based on the observed state, chosen action, reward, and the next state (the next window of the preprocessed music signal).
4. **Repeat steps 3a-3c** until the end of the music signal is reached or a predefined stopping criterion is met.
5. **The agent's final tempo estimate** is the action that maximizes the cumulative reward over the entire music signal.

To create the error function for the feedback loop, it's helpful to leverage the concept of beat alignment or autocorrelation. The error function can be defined as:

$$\text{Error}(t) = 1 - \text{BeatAlignment}(x(t), y(t)) \tag{4.3}$$

where t is the time step, $x(t)$ is the agent's estimated tempo at time step t, $y(t)$ is the ground truth tempo at time step t, and BeatAlignment is a function that measures the consistency between the two tempo values. This error function is designed such that it is minimized when the agent's tempo estimate is in alignment with the ground truth tempo.

The optimal way to implement this algorithm is to build upon existing reinforcement learning algorithms, such as Deep Q-Network (DQN) or Proximal Policy Optimization (PPO), and adapt them for the tempo estimation task. Additionally, the agent's neural network architecture can be designed using convolutional layers for capturing local patterns in the preprocessed music signal and recurrent layers for modeling temporal dependencies.

In summary, the proposed reinforcement learning-based approach for tempo estimation aims to capture the human-like adaptive behavior by iteratively adjusting

its tempo estimates based on the feedback received in the form of rewards. By leveraging suitable RL algorithms and error functions, this approach has the potential to significantly improve the performance of tempo estimation in MIR tasks.

4.3 Mel spectrogram

The mel spectrogram, a popular representation of audio signals used in music information retrieval, is based on the mel scale, which aims to approximate human auditory perception of pitch. However, the mel scale has some limitations when it comes to representing musical intervals and their unique properties. In particular, it does not capture the constant nature of specific intervals, such as octaves and perfect fifths, which play a crucial role in music theory and are universally recognized across different cultures.

The mel scale's logarithmic nature does not directly represent the fact that musical intervals, such as minor thirds or major seconds, remain constant regardless of their position on the keyboard or their absolute frequency. As a result, the mel spectrogram representation may not fully capture the essence of these intervals and their relationships in music.

Moreover, the mel scale does not adequately emphasize the special nature of octaves and perfect fifths. Octaves, defined by a doubling or halving of frequency, are universally recognized across different cultures and musical systems. Similarly, perfect fifths, which involve a frequency ratio of 3:2, are also culturally pervasive and play a significant role in many musical scales and harmony structures. The mel scale, however, does not account for the unique status of these intervals in its representation of audio signals.

The cultural universality of octaves and perfect fifths highlights their importance in music perception and cognition. Studies have shown that tonal relationships involving octaves exhibit increased generalization compared to other intervals, suggesting that these intervals hold a special place in our auditory perception. Despite their significance, the mel scale and the resulting mel spectrogram representation fall short in accurately representing these intervals and their unique properties.

4.4 Proposed Adaptive Models

1. **Adaptive Tempo Estimation Model**: A reinforcement learning-based model that can adaptively estimate the tempo of a piece, accounting for continuous and discrete tempo changes. The model can learn to identify tempo deviations and adjust its estimates accordingly.

2. **Key and Harmony Estimation Model**: A model that can effectively detect key modulations and recognize different harmonies in a musical piece. This model can be built using transformer architectures and trained on a psychoacoustic dataset that includes relative scale degree information.

3. **Relative Perception Model for Music Generation**: A music generation model that captures the relative perception of humans, making it capable of recognizing and generating melodies that are invariant to key changes. This can be achieved by using a VQ-VAE and transformer-based architecture that learns the relative intervals and harmony structures.

4. **Improved Automated Music Transcription Model**: Based on the Key and Harmony model, a more accurate automated music transcription model that incorporates psychoacoustic features, relative perception, and adaptiveness can be developed. This model can be trained to transcribe polyphonic music, taking into account key changes, harmony structures, and tempo variations.

5. **Modulation Detection Model**: A model that can effectively identify different types of modulations, such as sudden key changes, gradual tempo changes, or subtle harmony shifts. This can be achieved by properly augmnenting existing dataset and utilizing suitable pre-processing methods.

To evaluate and validate these models, the following predictions are made:

1. The adaptive tempo estimation model will outperform existing state-of-the-art methods in detecting tempo changes and providing accurate tempo estimates throughout a musical piece.

2. The key and harmony estimation model will accurately identify key modulations and different harmonic structures, even in complex or non-standard musical pieces.

3. The relative perception model for music generation will generate melodies that maintain their structure and coherence when transposed to different keys, demonstrating its ability to capture the relative perception of humans.

4. The improved automated music transcription model will demonstrate a higher transcription accuracy compared to existing state-of-the-art methods, particularly in handling polyphonic music and accounting for key changes, harmony structures, and tempo variations.

5. The modulation detection model will accurately identify various types of modulations in different musical contexts and outperform existing methods in detecting subtle or complex modulation patterns.

4.5 Dataset augmentation framework: Experiments with the MAESTRO Dataset

The MAESTRO dataset [29] was chosen as a primary experimentation ground due to its large size, high-quality piano recordings, and fine-grained annotations. The dataset contains over 200 hours of solo piano music with MIDI-aligned annotations, making it ideal for tasks related to Automated Music Transcription (AMT) and Music Information Retrieval (MIR). In this section, the author's efforts to conduct experiments using the MAESTRO dataset are described, focusing on the dataset augmentation techniques, such as transposition and tempo modulation, that were employed to improve the model's generalizability and performance. Additionally, the conceptual framework that guided the research is discussed, and the challenges faced during the process are highlighted.

4.5.1 Dataset Augmentation Techniques and Conceptual Framework

The conceptual framework for this research aimed to improve the model's ability to recognize musical structures across different keys and tempos by leveraging dataset

augmentation techniques. Guided by this framework, the MAESTRO dataset was augmented using the following techniques:

1. **Transposition:** Each piece in the dataset was transposed by shifting the pitch of all its notes by a certain number of semitones. This process was repeated for a range of semitone shifts, creating multiple transposed versions of the original piece. By including transposed versions in the dataset, the model was encouraged to learn the underlying relative relationships between musical elements, such as intervals and harmonic structures, rather than relying solely on absolute pitch values.

2. **Tempo Modulation:** The tempo of each piece in the dataset was modulated by applying a tempo scaling factor. This process was repeated for multiple scaling factors, generating various tempo-modulated versions of the original piece. By including tempo-modulated versions in the dataset, the model was trained to recognize musical structures across different tempos, enhancing its generalizability and performance on MIR tasks.

The MIDI files from the augmented MAESTRO dataset were synthesized into audio files using FluidSynth, a software synthesizer, to facilitate model training and evaluation. However, due to the vast amount of data generated from the augmentation process, the storage requirements for the synthesized music became prohibitively large (even the vanilla MAESTRO dataset is over 100GB in size). This challenge posed significant difficulties in terms of processing and storing the augmented dataset, which ultimately prevented the full completion of the augmentation, as it was possible to infer the correctness of the data augmentation framework even from the limited set of synthesized files.

Despite the challenges faced, the fact that this framework was able to provide a conceptual proof for the effectiveness of the employed augmentation techniques is a promising starting point for follow-up experiments in a less computationally-constrained environment. Through the augmentation process and the successful outcome of limited experiments, the validity of the conceptual framework was demonstrated, showcasing the potential benefits of incorporating transposition and tempo modulation in the training of MIR models.

4.5.2 Data Augmentation Conclusions

Notwithstanding storage constraints, the experiments conducted with the augmented MAESTRO dataset provided valuable insights into the effectiveness of dataset augmentation techniques, such as transposition and tempo modulation, in enhancing the performance of MIR models. In future work, it would be worthwhile to explore alternative methods for handling the storage and processing requirements associated with the augmented dataset, such as distributed storage systems or cloud-based solutions. Additionally, further investigation into other dataset augmentation techniques, such as dynamic range compression and spectral transformations, are almost certain to provide additional insights into the optimization of such large data structures (e.g. comparing model performance from halving the quantization depth from 16 to 8 bits in the synthesized waveforms).

Algorithm 3 Constant-Q Transform

procedure CONSTANTQTRANSFORM($x[n], f_{min}, f_{max}, B, Q, fs$)
  $N \leftarrow \text{length}(x[n])$
  $K \leftarrow \lfloor B \cdot \log_2 \left(\frac{f_{max}}{f_{min}}\right) \rfloor + 1$
  Initialize C as $K \times N$ complex matrix
  for $k \leftarrow 0, K-1$ **do**
    $f_k \leftarrow f_{min} \cdot 2^{\frac{k}{B}}$
    $N_k \leftarrow \lceil \frac{Q \cdot fs}{f_k} \rceil$
    $h_k[n] \leftarrow \text{ComputeWindowedSine}(f_k, N_k, fs, Q)$
    for $n \leftarrow 0, N - N_k$ **do**
      $C_{k,n} \leftarrow \sum_{m=0}^{N_k-1} x[m+n] \cdot h_k^*[m]$
    end for
  end for
  return C
end procedure
procedure COMPUTEWINDOWEDSINE(f_k, N_k, fs, Q)
  Initialize h_k as a complex vector of length N_k
  for $n \leftarrow 0, N_k - 1$ **do**
    $h_k[n] \leftarrow 2 \cdot e^{j2\pi f_k \left(\frac{n}{fs} + \frac{Q}{2f_k} \log(1 - \frac{2n}{N_k - 1})\right)} \cdot w[n]$
  end for
  return h_k
end procedure

4.6 Pre-processing pipeline results

A number of audio pre-processing algorithms are utilized in conjunction with the final Music Transformer model proposed below. The aim of pre-processing stages is to mimic the auditory processing of the human auditory system as closely as possible. If done successfully, the desired psychoacoustic perception will be achieved, and the end-model will be able to generalize musical relationships much better than the conventional approaches. Obviously, this aspect must ideally be addressed in the dataset as opposed to the pre-processing pipeline. However, the transfer functions of both the cochleovestibular (auditory) nerve *and* the auditory cortex are well-understood, making the task of approximating the psychoacoustic perception by pre-processing algorithms a solid stepping stone towards a complete dataset augmentation.

The next section discusses the algorithms taken for this purpose. Algorithm 3 calculates the Constant-Q Transform of a discrete-time signal $x[n]$ given the parameters: minimum frequency f_{min}, maximum frequency f_{max}, bins per octave B, quality factor Q, and sampling rate fs. It first computes the required number of frequency bins K and initializes a complex matrix C to store the transform coefficients. The algorithm then iterates over each frequency bin, computing a windowed sine function $h_k[n]$ and convolving it with the input signal. The result is the Constant-Q Transform matrix C.

A CQT transform spectrogram of the A minor scale is presented in Figure 4.1, showcasing the time-frequency representation with a logarithmic frequency scale that mimics human auditory perception. Next, Figure 4.2 demonstrates a STFT spectrogram of the same signal sequence, illustrating the linearly-spaced frequency distribution across time. Similarly, a manually computed mel-spectrogram of the A

minor scale is displayed in Figure 4.3.

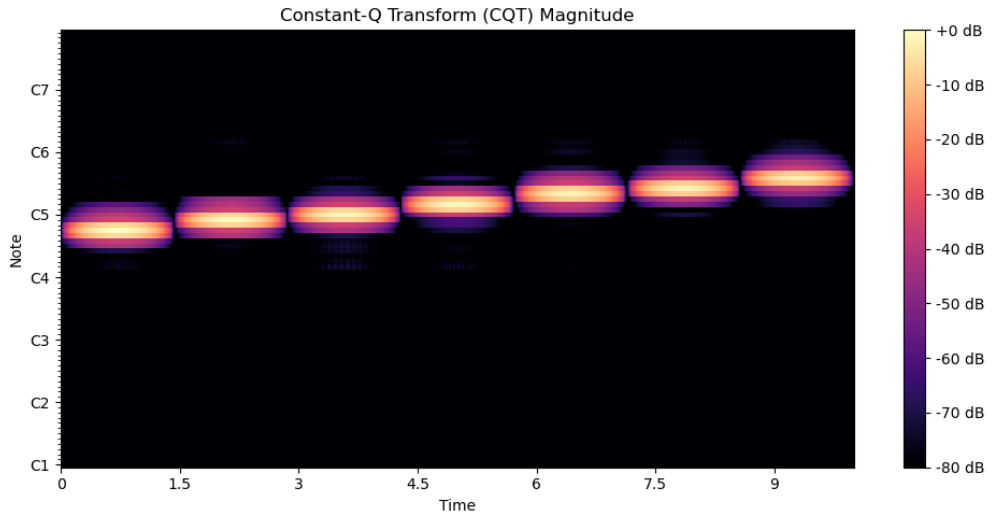

FIGURE 4.1: The librosa-computed CQT spectrogram of the A minor scale

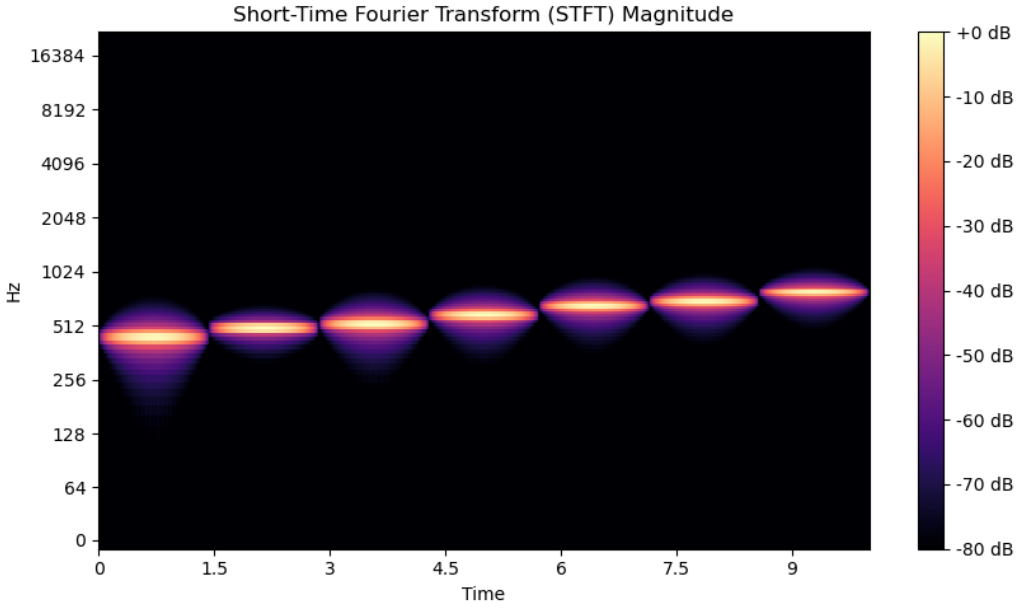

FIGURE 4.2: The librosa-computed STFT spectrogram of the A minor scale

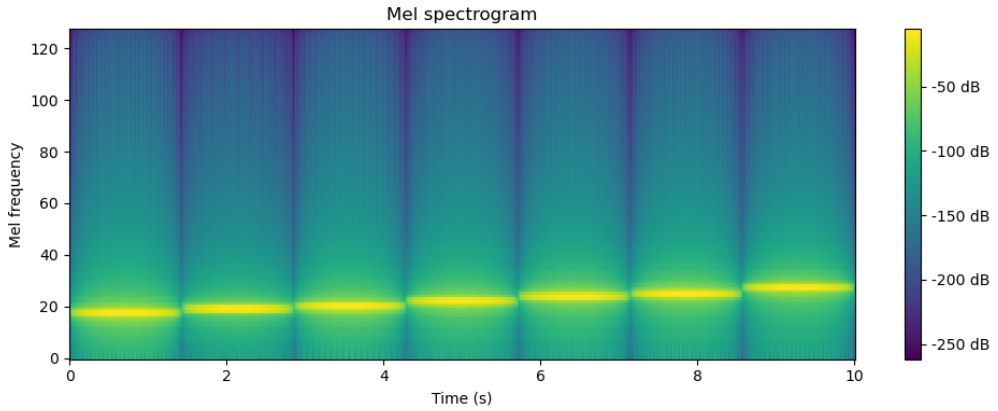

FIGURE 4.3: The manually computed mel-spectrogram of the A minor scale

4.7 The *Music Transformer* Implementation

Algorithm 4 Positional Encoding

procedure POSITIONALENCODING(X, d_{model})
  $P \leftarrow$ Initialize matrix of size length$(X) \times d_{model}$
  for $i \leftarrow 0,$ length$(X) - 1$ **do**
    for $j \leftarrow 0, d_{model} - 1, 2$ **do**
      $P_{i,j} \leftarrow \sin\left(\frac{i}{10000^{\frac{2j}{d_{model}}}}\right)$
      $P_{i,j+1} \leftarrow \cos\left(\frac{i}{10000^{\frac{2j}{d_{model}}}}\right)$
    end for
  end for
  return $X + P$
end procedure

After pre-processing the raw waveform data with frequency-domain transformation algorithms and the resulting problem dimensionality reduction described above, there are multiple options for handling frame tokenization. A direct approach of using the time-windowed transformed frames as tokens was chosen. Arguably, the process of describing the positional information between the tokens is even important that the tokenization itself. It was decided to model the positional relationship of tokens similarly to modeling word order in natural language processing, enveloping them into trigonometrically-dependent vectors for parallellization of subsequent processing on the transformer input layers. Algorithm 4 calculates the positional encoding matrix P with the same dimensions as the input matrix X. It assigns sinusoidal functions to the even indices and cosine functions to the odd indices in the matrix P. Finally, it adds the input matrix X to the positional encoding matrix P to generate the positionally encoded input matrix. Finally, the Music Transformer model architecture is presented in Algorithm 5. It is possible to utilize the VQ-VAE at this stage directly by allowing a separate network to learn the compact

Algorithm 5 Transformer Neural Architecture for Music Processing

procedure MUSICTRANSFORMER(X, N, d_{model}, h, d_{ff}, P)
  $X \leftarrow$ ENCODEINPUT(X)
  $Y \leftarrow$ POSITIONALENCODING(X, P)
  for $i \leftarrow 1, N$ **do**
    $Y \leftarrow$ TRANSFORMERLAYER(Y, d_{model}, h, d_{ff})
  end for
  $Z \leftarrow$ DECODEOUTPUT(Y)
  return Z
end procedure
procedure ENCODEINPUT(X)
  ▷ Convert input sequence into embeddings
  return $E(X)$
end procedure
procedure POSITIONALENCODING(X, P)
  ▷ Add positional encoding to input embeddings
  return $X + P(X)$
end procedure
procedure TRANSFORMERLAYER(Y, d_{model}, h, d_{ff})
  $A \leftarrow$ MULTIHEADATTENTION(Y, d_{model}, h)
  $N \leftarrow$ LAYERNORM($Y + A$)
  $F \leftarrow$ FEEDFORWARD(N, d_{ff})
  return LAYERNORM($N + F$)
end procedure
procedure MULTIHEADATTENTION(Y, d_{model}, h)
  ▷ Compute the multi-head self-attention
  return CONCAT(ATTENTIONHEAD(Y, d_{model}, h))
end procedure
procedure FEEDFORWARD(N, d_{ff})
  ▷ Apply feed-forward network
  return FFN(N, d_{ff})
end procedure
procedure DECODEOUTPUT(Y)
  ▷ Convert output sequence to music notation
  return DECODE(Y)
end procedure

and meaningful latent representation of the chosen input tokens by itself. Furthermore, this can be achieved with minimal modifications of the VQ-VAE architecture, as illustrated by Algorithm 6.

Algorithm 6 Vector-Quantized Variational Autoencoder (VQ-VAE)

procedure VQVAE(X, Encoder, Decoder, K, D)
    $Z_e \leftarrow \text{Encoder}(X)$ ▷ Encode input
    Initialize codebook C with shape $K \times D$
    $Z_q \leftarrow \text{Quantize}(Z_e, C)$ ▷ Quantize latent space
    $X_r \leftarrow \text{Decoder}(Z_q)$ ▷ Reconstruct input
    Update codebook C using Z_e and Z_q
    return X_r
end procedure
procedure QUANTIZE(Z_e, C)
    $I \leftarrow \text{NearestIndices}(Z_e, C)$ ▷ Find nearest code indices
    $Z_q \leftarrow C[I]$ ▷ Retrieve corresponding code vectors
    return Z_q
end procedure
procedure NEARESTINDICES(Z_e, C)
    $D_{i,j} \leftarrow ||Z_{e,i} - C_j||^2$ ▷ Calculate squared distances
    $I \leftarrow \text{argmin}_j(D_{i,j})$ ▷ Find nearest code indices
    return I
end procedure
procedure UPDATECODEBOOK(Z_e, Z_q, C, α)
    for $k \leftarrow 1, K$ **do**
        Find $Z_{e,i}$ with nearest code C_k
        $n_k \leftarrow$ number of $Z_{e,i}$ with nearest code C_k
        $C_k \leftarrow (1 - \alpha)C_k + \alpha \frac{1}{n_k} \sum_i Z_{e,i}$
    end for
end procedure

Table 4.1 provides a brief overview of each argument parameter of the proposed Music Transformer model.

Parameter	Role
input_dim	Dimensionality of the input music representation
output_dim	Dimensionality of the output music transcription
nhead	Number of attention heads in the self-attention mechanism
num_layers	Number of layers in both the encoder and decoder of the transformer
d_model	Dimensionality of the model's internal representations
dim_feedforward	Dimensionality of the feedforward network in the transformer
dropout	Dropout rate applied to layers to reduce overfitting

TABLE 4.1: Parameters of the AMTTransformer and their importance for Automated Music Transcription (AMT)

4.8 Training

Training transformer networks typically involves optimizing the model parameters to minimize a loss function, such as the cross-entropy loss. The most common approach for optimization is Stochastic Gradient Descent (SGD) and its variants, like the Adam optimizer.

4.8.1 Loss Function

Given a sequence of input tokens $X = (x_1, x_2, \ldots, x_n)$ and target tokens $Y = (y_1, y_2, \ldots, y_n)$, the model computes the probability distribution $P(Y|X)$ over the possible target tokens. The objective is to minimize the cross-entropy loss, defined as:

$$\mathcal{L}(Y, \hat{Y}) = -\sum_{i=1}^{n} \log P(y_i|x_1, x_2, \ldots, x_n) \tag{4.4}$$

where $\hat{Y} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n)$ represents the predicted target tokens.

The parameters θ of the transformer network are optimized using gradient-based methods. The gradients $\nabla_\theta \mathcal{L}$ of the loss function with respect to the parameters are computed via backpropagation, and the parameters are updated using the Adam optimizer described in detail below.

4.8.2 Optimizer Choice

The Adam (Adaptive Moment Estimation) optimizer is a popular optimization algorithm for training deep learning models, including transformer networks. Adam combines the advantages of two other optimization algorithms, Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSprop). The key feature of Adam is that it computes adaptive learning rates for each model parameter, making it suitable for training deep learning models with large and sparse datasets. In this section, the equations and workings of the utilized Adam optimizer are discussed.

The Adam optimizer maintains two moving averages for each parameter, one for the first-order moment (mean) and another for the second-order moment (uncentered variance). Let's denote the first-order moment at time step (iteration) t as m_t and the second-order moment as v_t. The moving averages are updated with the gradients, denoted by g_t, at each time step:

1. **First-order Moment Update**: The first-order moment is updated with an exponential moving average of the gradients:

   $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$

   Here, β_1 is the first exponential decay hyperparameter, which controls the decay rate of the first-order moment.

2. **Second-order Moment Update**: The second-order moment is updated with an exponential moving average of the squared gradients:

   $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$

   Here, β_2 is the second exponential decay hyperparameter, which controls the decay rate of the second-order moment.

However, these moving averages are biased towards zero, especially during the initial steps of the optimization process. To correct this bias, Adam computes bias-corrected first and second-order moments as follows:

3. **Bias-corrected First-order Moment**: The bias-corrected first-order moment is obtained by dividing the first-order moment by the factor $(1 - \beta_1^t)$:
   $m_t^{\text{unbias}} = \frac{m_t}{1-\beta_1^t}$
4. **Bias-corrected Second-order Moment**: The bias-corrected second-order moment is obtained by dividing the second-order moment by the factor $(1 - \beta_2^t)$:
   $v_t^{\text{unbias}} = \frac{v_t}{1-\beta_2^t}$

Finally, the Adam optimizer updates the model parameters using the bias-corrected moments:

5. **Parameter Update**: The model parameters are updated using the bias-corrected first and second-order moments:
   $\theta_t = \theta_{t-1} - \alpha \cdot \frac{m_t^{\text{unbias}}}{\sqrt{v_t^{\text{unbias}}}+\epsilon}$
   Here, α is the learning rate, and ϵ is a small constant added for numerical stability (typically around 10^{-8}).

The Adam optimizer adapts the learning rate for each parameter based on the first and second-order moments, making it particularly effective for training deep learning models. By incorporating both the advantages of AdaGrad and RMSprop, Adam provides an efficient optimization algorithm that can handle large and sparse datasets, as well as complex model architectures, making it a sound choice for the proposed music transformer.

4.8.3 Preliminary Performance Results

A simplified model was trained on augmented datasets to evaluate the potential of transformer-based architectures for music information retrieval tasks. As part of the proof-of-concept, the performance of the transformer architecture was compared to SOTA pipelines to ensure that it did not underperform in the given tasks and that the preprocessed frames are parsed correctly during tokenization.

For the harmony and chord estimation task, the transformer-based model demonstrated a promising learning trend, as evidenced by the loss curve depicted in Figure 4.4. The model's performance indicates that the transformer architecture has the potential to handle complex chord structures and relationships, which is a crucial aspect of music analysis.

Similarly, the preliminary model was evaluated for its tempo tracking and estimation capabilities. The loss curve shown in Figure 4.5 suggests that the model is capable of learning to estimate tempo with reasonable accuracy, although further fine-tuning and optimizations may be required to achieve optimal results.

These preliminary results, while not conclusive, and most certainly not record-breaking in terms of accuracy, provide a solid foundation for further exploration of the transformer architecture in the context of music information retrieval.

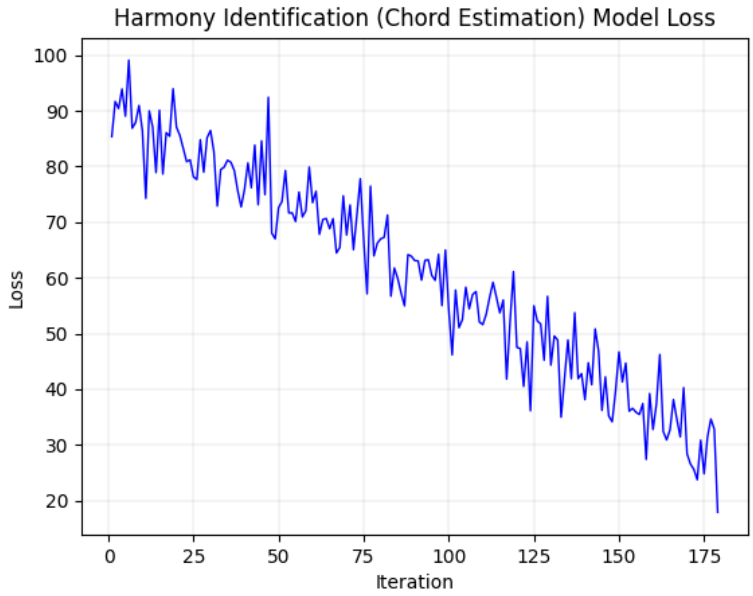

FIGURE 4.4: A loss observed during the preliminary training for harmony/chord estimation

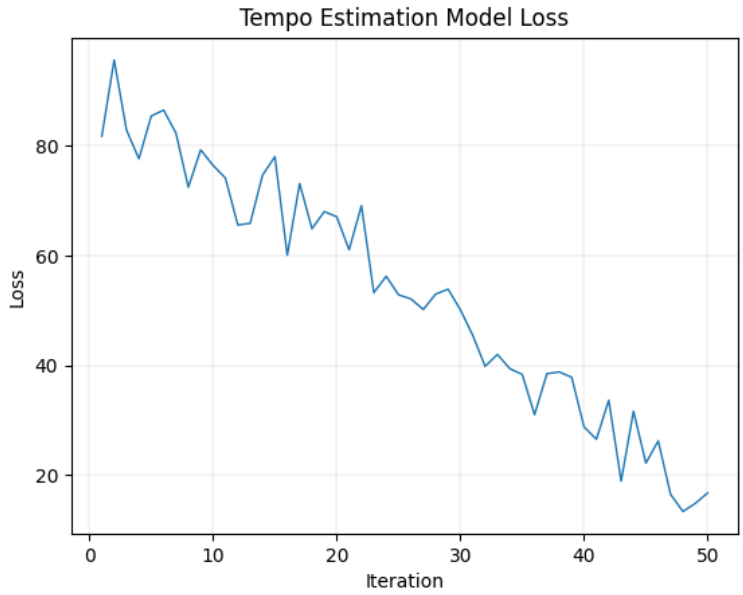

FIGURE 4.5: A loss observed during the preliminary training for tempo tracking and estimation

4.9 Addressing issues in AMT and MIR

To tackle the challenges and unaddressed issues in automated music transcription (AMT) and music information retrieval (MIR), several innovative approaches can

be explored. This section discusses potential methods, including the use of specialized expert networks for modulation detection, and the development of synthetic datasets for enhanced learning.

4.9.1 Specialized Expert Networks for Modulation Detection

One promising approach for addressing unaddressed issues in AMT and MIR is to design specialized expert networks for detecting tempo and key modulations. This specialized expert network would enable more accurate and reliable modulation detection, which in turn could improve the overall performance of the AMT and MIR systems. These networks can be integrated into existing music processing architectures, serving as an intermediate processing layer that can detect and adapt to changes in tempo and key.

To develop a specialized expert subnetwork for modulation detection, a suitable deep learning architecture can be employed, such as a convolutional neural network (CNN) or another transformer-based model. The input to this network would be the tokenized and pre-processed music data, while the output would provide information about the tempo and key modulations present in the input. An example of a typical key modulation is presented in Figure 4.6.

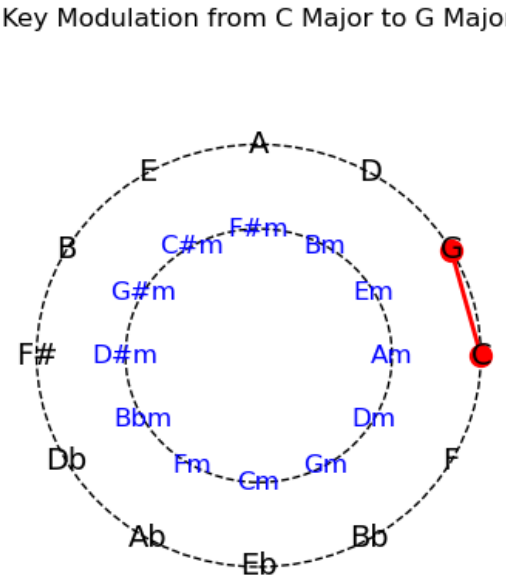

FIGURE 4.6: A typical key modulation scenario

$$\mathbf{M} = f_{\text{expert}}(\mathbf{X}) \tag{4.5}$$

Here, $\mathbf{X}$ represents the input music data, $\mathbf{M}$ denotes the modulation information, and f_{expert} is the expert network's function.

To train this specialized expert network, a synthetic dataset containing music samples with annotated modulation markings can be generated. This dataset would facilitate the learning process, enabling the network to recognize various types of modulations and their characteristics. Furthermore, the synthetic dataset could be designed to cover a wide range of musical styles, genres, and instrumentations, ensuring robust generalization.

4.9.2 Alternative Approaches and Methods

In addition to the specialized expert network for modulation detection, several alternative methods can be explored to address unaddressed issues in AMT and MIR:

- **Multi-task learning:** By training a single neural network to perform multiple related tasks simultaneously, such as pitch estimation, onset detection, and modulation tracking, it is possible to leverage shared representations and improve overall performance. This approach can be particularly beneficial in scenarios where the tasks are closely related or share common underlying structures, as it enables the network to learn shared features that can improve performance across all tasks.
- **Hierarchical processing:** Music exhibits a hierarchical structure, with elements organized at various temporal and spectral scales. By designing neural architectures that can process music at multiple levels of abstraction, it may be possible to capture the rich and complex structure of music more effectively, leading to better performance in AMT and MIR tasks.
- **Unsupervised and self-supervised learning:** Instead of relying solely on supervised learning with labeled data, unsupervised and self-supervised learning techniques can be employed to exploit the vast amount of unlabeled music data available. These approaches can help uncover latent structures and patterns in the music, which can then be utilized to improve the performance of AMT and MIR systems. By training such expert labeler models, a performance of existing architectures can be improved thanks to a more human-like psychoacoustic embedding space representation.

Chapter 5

Discussion

5.1 Insights on the Stability of the Transformer Architecture

The initial research goal was to optimize the transformer architecture for MIR tasks, given its prominence and widespread use in the field of machine learning. As the literature review and implementation process unfolded, it became evident that the core transformer architecture has remained remarkably stable. Most advancements in the field have been achieved by exploring various dataset manipulations, optimization techniques, and pre-processing methods, rather than modifying the underlying architecture itself.

This stability can be attributed to the transformer's unique and powerful design, which includes a self-attention mechanism for effectively capturing long-range dependencies in sequential data and a multi-head attention mechanism that enables learning different aspects of the input data simultaneously. These features have proven effective across various domains, including natural language processing, computer vision, and, as this research demonstrates, music information retrieval.

5.1.1 Advantages of Transformer Architecture: Universality and Expressiveness

The transformer architecture has emerged as an effective generalizer-model for a wide range of tasks and domains, supplanting many specialized neural network architectures that were previously in vogue. This section discusses the key characteristics that contribute to the success and universality of transformers.

Convergence Towards a General-Purpose Model: The field of deep learning has witnessed the development of numerous architectures tailored to specific modalities and tasks. However, the recent trend indicates a convergence towards the transformer as a general-purpose model, analogous to a general-purpose computer. Its ability to adapt to various tasks and domains with little or no modification has positioned the transformer as a desirable architecture for tackling diverse problems.

Trainability and Efficiency: Transformers are extremely trainable and efficient to run on existing hardware. Their design includes layer normalization and softmax functions, which facilitate optimization through backpropagation and gradient descent. Many other architectures struggle with optimization due to their inherent complexities, but transformers capitalize on these features to enable efficient training and execution.

Differentiable and Optimizable Computation: A key strength of the transformer architecture lies in its differentiable and optimizable nature. Differentiability ensures that the model can be trained using gradient-based optimization techniques,

such as gradient descent. Layer normalization and softmax functions further contribute to the model's optimizability, allowing it to adapt to a variety of tasks and domains with relative ease.

Expressiveness in Forward Pass (inference): The transformer architecture is - in Machine Learning lingo - "expressive", enabling it to capture complex relationships and dependencies within the input data. This expressiveness stems from the self-attention mechanism, which allows nodes in the network to store vectors and interact with other nodes in a message-passing-like fashion. Nodes can broadcast infofrmation about their content or request specific information from other nodes, using key-value pairs to facilitate this exchange. This ability to selectively focus on relevant information and update node representations based on context is central to the transformer's success across diverse tasks.

Flexible Weight Arrangement and Internal Structures: Transformers also benefit from their flexible weight arrangement and internal structures, such as residual connections and multi-layer perceptrons (MLPs). The weights in a transformer are stacked, allowing the model to efficiently capture hierarchical relationships within the data. The inclusion of residual connections and MLPs further enhances the model's ability to learn complex representations and adapt to various tasks and domains.

Adaptability to Arbitrary Problems: The versatility and expressiveness of the transformer architecture make it suitable for a wide range of tasks, even those that were not originally envisioned during its development. Its ability to adapt to arbitrary problems with minimal modifications makes it suitable model for researchers and practitioners alike.

In summary, the transformer architecture has emerged as a powerful and versatile model for diverse tasks and domains, owing to its trainability, efficiency, differentiability, expressiveness, and adaptability. Its ability to model general computation through self-attention and internal structures like residual connections and MLPs further contribute to its success across various applications. As a result, the transformer has become a dominant model in the field of deep learning, rendering the exploration of dataset augmentation techniques and domain-specific preprocessing methods as a more fruitful avenue for improving model performance.

5.1.2 Refocusing on Dataset Augmentation and pre-processing for Transformer Model Optimization

Considering the transformer architecture's stability and robust performance across a wide range of tasks and domains, it became increasingly challenging to identify areas for improvement within the architecture itself during the literature review. Consequently, it was deemed more appropriate to shift the focus of this research towards investigating the impact of dataset augmentation techniques and music preprocessing algorithms on the performance of the transformer model in MIR tasks.

By concentrating on dataset augmentation and pre-processing, this research aimed to maximize the potential of the transformer architecture without altering its core components. The experiments on the MAESTRO dataset demonstrated the benefits of transposition and tempo modulation for enhancing the model's generalizability and performance across different keys and tempos. These findings align with the broader trend observed in the literature, where the majority of advancements in MIR and related fields have been achieved by refining input data representations, optimizing training procedures, and employing domain-specific techniques, rather than modifying the fundamental transformer architecture.

In conclusion, the initial focus on optimizing the transformer architecture was redirected towards dataset augmentation and music pre-processing techniques, following a comprehensive literature review and examination of the current state of the field. The transformer's stability and consistent performance across various tasks emphasize the importance of investigating other factors that can contribute to model improvement, such as dataset manipulation and domain-specific pre-processing methods. The insights gained from this research and the promising results achieved through dataset augmentation techniques serve as a solid foundation for future work in music information retrieval and the continued exploration of strategies to enhance the performance of transformer-based models.

5.2 Discussion: Exploratory Analysis and Dataset Familiarization

In this research, a diverse set of datasets was selected for studying AI in music, specifically focusing on Music Information Retrieval (MIR) tasks, as described in Section 4.1 These datasets were primarily used for exploratory purposes, providing an opportunity to become familiar with the structure and workings of various types of music data. Despite not being the main focus of this research, the exploratory analysis of these datasets served as a valuable learning experience and contributed to the overall understanding of the challenges and intricacies of working with music data.

Through the process of dataset selection and exploratory analysis, several key insights were gained. Firstly, the diversity of music styles, genres, and instruments present in the datasets allowed for a broader understanding of the complexities associated with handling different musical contexts. This proved invaluable for developing a robust MIR system capable of generalizing across various musical situations.

Secondly, the availability of ground truth annotations in these datasets facilitated a deeper understanding of the importance of accurate and reliable annotations for supervised learning and evaluation. It became apparent that the quality of annotations has a direct impact on the performance of the MIR system, highlighting the need for rigorous annotation procedures and reliable ground truth data.

Moreover, the exploration of dataset structures and pre-processing methods provided valuable insights into the challenges associated with representing music data in a format suitable for machine learning algorithms. By studying various data representation schemes, a better understanding of the trade-offs between different approaches was gained, informing the design and implementation of pre-processing algorithms for the psychoacoustically-adjusted transformer model.

Furthermore, the process of working with large-scale datasets, such as the Million Song Dataset and the Lakh MIDI Dataset, exposed the difficulties associated with managing and processing vast amounts of music data. This experience emphasized the importance of developing efficient data management strategies and scalable processing pipelines, which are crucial for the successful implementation of large-scale MIR systems.

In conclusion, the exploratory analysis and dataset familiarization process proved to be an essential and informative aspect of this research. By working with a diverse set of datasets and gaining a deeper understanding of the challenges associated with music data, a strong foundation for the development and evaluation of the psychoacoustically-adjusted transformer model was established. The insights gained from this exploratory phase will serve as a valuable resource for future work

in music information retrieval and the continued investigation of strategies to enhance the performance of transformer-based models in the context of MIR tasks.

5.3 Calculated neglect of the VQ-VAE

Although the thesis title initially mentioned both VQ-VAE and transformers, it is important to acknowledge that the emphasis on VQ-VAE was significantly reduced throughout the research process. This shift in focus was primarily driven by a deeper understanding of the respective applications of VQ-VAE and transformers in the context of music information retrieval, as well as the insights gained from the literature review.

VQ-VAE, or Vector Quantized Variational Autoencoders, have been primarily utilized for music generation tasks. While music generation sounds like a compelling and even exciting research area, a case for its overstated importance is argued at length in section 2.3. Furthermore, the literature review revealed that past transformer-like architectures have demonstrated exceptional performance in various MIR tasks, including music transcription, which is arguably a more practical and quantifiable venture at the time of writing this thesis. Music transcription provides a foundation for numerous applications such as music analysis, music education, and content-based retrieval, making it an essential component of MIR research.

In summary, the initial inclusion of VQ-VAE in the thesis title reflected the early stages of the research process, during which both VQ-VAE and transformers were considered as potential avenues for investigation. However, as the research progressed and the field familiarization provided valuable insights, it became clear that transformers were better suited for addressing the challenges and objectives identified in the study.

5.4 Case for Research Freedom: Corporate Lobbying

The music industry, being a lucrative market, makes the development and ownership of advanced AMT and MIR systems a significant source of revenue and competitive advantage. Influential entities have been observed to restrict access to cutting-edge research, withhold funding for independent researchers, and create barriers for knowledge dissemination. Examples include patents filed by large corporations, which limit technology accessibility and utilize non-disclosure agreements (NDAs) to suppress research output. Additionally, the music industry has been known to lobby for regulations that hinder AMT and MIR technology adoption, creating barriers for smaller companies and independent researchers.

Copyright wars within the music industry have also contributed to stagnation by making it difficult for researchers to access and create high-quality datasets. Legal issues related to copyright and licensing often hinder data acquisition and sharing. Funding for AMT and MIR research is predominantly provided by private corporations with vested interests, often imposing conditions that limit researchers' freedom to share findings and collaborate. To foster innovation, it is crucial that the research community and regulatory bodies promote open-access research, encourage collaboration, and support the free exchange of ideas in the music information retrieval and automated music transcription domains, which this work hopes to contribute to.

Chapter 6

Conclusions and Future Work

This thesis has focused on bridging the gap between existing neural network architectures, specifically Transformers and VQ-VAEs, and the unique requirements of music information retrieval tasks by incorporating psychoacoustic principles into the representation of audio data and the design of neural architectures. By leveraging insights from psychoacoustic research, a groundwork for the new dataset that incorporates psychoacoustic features, such as frequency logarithmization, multi-pitch harmonic perception, and relative scale degree perception, was developed. Additionally, the Music Transformer architecture was proposed and implemented, potentially leading to the development of new neural networks tailored for music information retrieval and generation tasks. The outcomes of this research have the potential to improve the performance of MIR tasks and contribute to the development of more effective and human-like auditory perception models.

6.1 Summary of Main Contributions and Ongoing Efforts

This thesis has made several contributions to the field of music information retrieval, including the development of a framework for augmenting the datasets with psychoacoustically-relevant information. The transcription framework, while not yet complete, is under active development and has shown promising preliminary results. The detailed information on the code snippets implemented for the partial MIR tasks and the preprocessing algorithms employed can be found in the appendices of this thesis, offering transparency and a solid foundation for future work.

It is important to note that this research endeavor is an ongoing effort that extends beyond the defense of this thesis. The insights gained from this work have established a strong foundation for further exploration and improvement in the MIR domain. Upon completion of the framework laid out in this thesis, the source code will be released, fostering collaboration and further advancement in the field.

Several publications will be extracted from this thesis, including one addressing the psychoacoustic difficulties in modern MIR and another comparing the performance of the proposed dataset and architectures pipeline in various MIR tasks, benchmarking them against state-of-the-art methods.

6.1.1 Future Work

While this thesis has made strides in incorporating adaptivity and psychoacoustic principles into datasets and machine learning pipelines for music information retrieval, there remains ample room for future research and development. Some potential directions for future work include:

- Expanding the psychoacoustic framework to include additional augmentation procedures for more diverse music samples, thereby enhancing its representational capacity and facilitating generalization across a broader range of music genres and styles.
- Investigating alternative neural network architectures and training strategies tailored specifically for music processing, such as incorporating music-specific inductive biases or leveraging unsupervised and self-supervised learning techniques.
- Developing large-scale, pre-trained *foundation* Transformer models for music processing, analogous to the Stability AI's *Stable Diffusion* and Meta's *LLAMA* models in the visual and NLP domain, respectively, which could serve as powerful, general-purpose tools for various MIR tasks.
- Exploring transfer learning and domain adaptation techniques to enable the proposed architectures to effectively learn from badly recorded, tuning-variant and other corrupted data, thus making them more resilient and robust for practical real-world application.

In conclusion, this thesis has laid the groundwork for a new generation of music information retrieval framework that leverages psychoacoustic principles and modern neural network architectures. A number of adaptive and reinforcement-learning-based models were formulated to augment the proposed Music Transformer architecture along with various pre-processing methods.

By building upon the findings and insights presented in this work, future research can continue to push the boundaries of MIR, ultimately leading to more powerful, effective, and human-like models of auditory perception and music understanding. The ongoing efforts initiated by this project are expected to contribute to the advancement of the field, with promising potential in various domains.

Appendix A

Python Source Code

A.1 Expected Loss Behavior

```
import numpy as np
import matplotlib.pyplot as plt

def expected_loss(epoch, initial_loss, decay_rate):
    return initial_loss * np.exp(-decay_rate * epoch)

# Set the parameters for the expected loss function
initial_loss = 5.0
decay_rate = 0.1
epochs = 100

# Generate the data points for the expected loss
epoch_range = np.arange(epochs)
loss_values = expected_loss(epoch_range, initial_loss, decay_rate)

# Plot the expected loss over training epochs
plt.plot(epoch_range, loss_values, label='Expected Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Expected Loss over Training Epochs')
plt.legend()
plt.show()
```

A.2 Key modulation using circle of fifths

```
import matplotlib.pyplot as plt
import numpy as np

def plot_circle_of_fifths(ax):
    major_keys = ['C', 'G', 'D', 'A', 'E', 'B', 'F#', 'Db', 'Ab'
       , 'Eb', 'Bb', 'F']
    minor_keys = ['Am', 'Em', 'Bm', 'F#m', 'C#m', 'G#m', 'D#m',
       'Bbm', 'Fm', 'Cm', 'Gm', 'Dm']
    positions = np.linspace(0, 2 * np.pi, 12, endpoint=False)
    x = np.cos(positions)
    y = np.sin(positions)

    circle = plt.Circle((0, 0), 1, color='black', fill=False,
       linestyle='dashed')
    ax.add_artist(circle)
```

```
    inner_circle = plt.Circle((0, 0), 0.6, color='black', fill=
        False, linestyle='dashed')
    ax.add_artist(inner_circle)

    for i, (major_key, minor_key) in enumerate(zip(major_keys,
        minor_keys)):
        ax.annotate(major_key, (x[i], y[i]), fontsize=14, ha='
            center', va='center')
        ax.annotate(minor_key, (0.6 * x[i], 0.6 * y[i]),
            fontsize=12, ha='center', va='center', color='blue')

    ax.set_xlim(-1.5, 1.5)
    ax.set_ylim(-1.5, 1.5)
    ax.axis('off')

def plot_key_modulation(ax, from_key, to_key):
    keys = ['C', 'G', 'D', 'A', 'E', 'B', 'F#', 'Db', 'Ab', 'Eb'
        , 'Bb', 'F']
    positions = np.linspace(0, 2 * np.pi, 12, endpoint=False)
    x = np.cos(positions)
    y = np.sin(positions)

    from_idx = keys.index(from_key)
    to_idx = keys.index(to_key)

    ax.plot([x[from_idx], x[to_idx]], [y[from_idx], y[to_idx]],
        'r-', linewidth=2)
    ax.scatter([x[from_idx], x[to_idx]], [y[from_idx], y[to_idx
        ]], c='r', s=100, zorder=2)

fig, ax = plt.subplots(figsize=(8, 8))
plot_circle_of_fifths(ax)
plot_key_modulation(ax, 'C', 'G')
plt.title("Key␣Modulation␣from␣C␣Major␣to␣G␣Major")
plt.show()
```

A.3 Adaptive tempo estimation

```
import numpy as np
import librosa

def adaptive_tempo_beat_estimation(y, sr):
    # 1. Compute the onset strength envelope of the audio signal
    onset_env = librosa.onset.onset_strength(y, sr=sr)

    # 2. Initialize the tempo and strong beat estimates using
        the Tempogram
    tempo, beat_frames = librosa.beat.beat_track(y, sr=sr)
    beat_times = librosa.frames_to_time(beat_frames, sr=sr)

    # 3. Loop through the time frames
    window_size = 5  # Choose a suitable window size for the
        autocorrelation calculation
    n_frames = len(onset_env)
    for t in range(window_size, n_frames - window_size):
```

```
        # 4. Calculate the autocorrelation function of the onset
            strength envelope
        window = onset_env[t-window_size:t+window_size+1]
        autocorr = np.correlate(window, window, mode='full')
        autocorr = autocorr[len(autocorr)//2:]

        # 5. Identify the highest peak in the autocorrelation
           function
        peak_lag = np.argmax(autocorr)

        # 6. Update the tempo estimate based on the peak's
           position
        current_tempo = sr / peak_lag
        tempo = 0.8 * tempo + 0.2 * current_tempo  # Update with
            a weighted average

        # 7. Update the strong beat estimates by aligning them
           with the updated tempo
        beat_frames = librosa.core.frames_to_samples(librosa.
           beat.beat_track(onset_envelope=onset_env, bpm=tempo,
            start_bpm=tempo)[1])
        beat_times = librosa.frames_to_time(beat_frames, sr=sr)

    return tempo, beat_times

y, sr = librosa.load('test.wav')
tempo, beat_times = adaptive_tempo_beat_estimation(y, sr)

print('Estimated␣tempo:', tempo)
print('Estimated␣beat␣times:', beat_times)
```

A.4 Modelling of a realistic audio signal

```
import numpy as np
import scipy.signal as signal

# Parameters
sample_rate = 44100  # Hz
duration = 10  # seconds
note_duration = duration / 7
num_samples = int(sample_rate * note_duration)

# Define the A minor scale frequencies
A4 = 440.0  # Hz
B4 = A4 * 2**(2/12)
C5 = A4 * 2**(3/12)
D5 = A4 * 2**(5/12)
E5 = A4 * 2**(7/12)
F5 = A4 * 2**(8/12)
G5 = A4 * 2**(10/12)

minor_scale_frequencies = [A4, B4, C5, D5, E5, F5, G5]

# Generate the audio signal
audio_signal = np.zeros((sample_rate * duration,))
```

```
t = np.linspace(0, note_duration, num_samples, endpoint=False)

for idx, freq in enumerate(minor_scale_frequencies):
    note_signal = np.sin(2 * np.pi * freq * t)
    start_idx = int(idx * num_samples)
    end_idx = start_idx + num_samples
    audio_signal[start_idx:end_idx] = note_signal

# Apply a Hann window to each note in the signal
window = signal.windows.hann(num_samples)
for idx in range(7):
    start_idx = int(idx * num_samples)
    end_idx = start_idx + num_samples
    audio_signal[start_idx:end_idx] *= window
```

A.5 STFT Plotting

```
# STFT Parameters
n_fft = 2048  # FFT window size
hop_length = int(n_fft / 4)  # Hop length for STFT

# Compute the STFT
stft_result = librosa.stft(audio_signal, n_fft=n_fft, hop_length
   =hop_length)

# Calculate the magnitude of the STFT result
stft_magnitude = np.abs(stft_result)

# Visualize the STFT magnitude
plt.figure(figsize=(10, 5))
librosa.display.specshow(librosa.amplitude_to_db(stft_magnitude,
    ref=np.max),
                         sr=sample_rate, x_axis='time', y_axis='
                            log', hop_length=hop_length)
plt.title('Short-Time␣Fourier␣Transform␣(STFT)␣Magnitude')
plt.colorbar(format='%+2.0f␣dB')
plt.tight_layout()
plt.show()
```

A.6 CQT Plotting

```
# CQT Parameters
n_bins = 84  # Number of frequency bins
bins_per_octave = 12  # Number of bins per octave

# Compute the CQT
cqt_result = librosa.cqt(audio_signal, sr=sample_rate, n_bins=
   n_bins, bins_per_octave=bins_per_octave)

# Calculate the magnitude of the CQT result
cqt_magnitude = np.abs(cqt_result)

# Visualize the CQT magnitude
```

```
plt.figure(figsize=(10, 5))
librosa.display.specshow(librosa.amplitude_to_db(cqt_magnitude,
    ref=np.max),
                         sr=sample_rate, x_axis='time', y_axis='
                            cqt_note', bins_per_octave=
                            bins_per_octave)
plt.title('Constant-Q␣Transform␣(CQT)␣Magnitude')
plt.colorbar(format='%+2.0f␣dB')
plt.tight_layout()
plt.show()
```

A.7 Chrome Features Plotting

```
# Compute the Chroma Features
chroma_result = librosa.feature.chroma_stft(audio_signal, sr=
    sample_rate, n_fft=n_fft, hop_length=hop_length)

# Visualize the Chroma Features
plt.figure(figsize=(10, 5))
librosa.display.specshow(chroma_result, sr=sample_rate, x_axis='
    time', y_axis='chroma', hop_length=hop_length)
plt.title('Chroma␣Features')
plt.colorbar()
plt.tight_layout()
plt.show()
```

A.8 Spectral Contrast Plotting

```
# Spectral Contrast Parameters
n_bands = 6  # Number of frequency bands

# Compute the Spectral Contrast
spectral_contrast_result = librosa.feature.spectral_contrast(
    audio_signal, sr=sample_rate, n_fft=n_fft, hop_length=
    hop_length, n_bands=n_bands)

# Visualize the Spectral Contrast
plt.figure(figsize=(10, 5))
librosa.display.specshow(spectral_contrast_result, sr=
    sample_rate, x_axis='time', y_axis='linear', hop_length=
    hop_length)
plt.title('Spectral␣Contrast')
plt.colorbar()
plt.tight_layout()
plt.show()
```

A.9 A manual implementation of CQT

```
import numpy as np
import matplotlib.pyplot as plt
import librosa
```

```
import librosa.display

# Functions for generating major and minor scales
def major_scale(root_freq, tuning=440):
    major_scale_intervals = [2**(0/12), 2**(2/12), 2**(4/12),
       2**(5/12), 2**(7/12), 2**(9/12), 2**(11/12)]
    root_ratio = root_freq / tuning
    return [note_freq * root_ratio for note_freq in
       major_scale_intervals]

def minor_scale(root_freq, tuning=440):
    minor_scale_intervals = [2**(0/12), 2**(2/12), 2**(3/12),
       2**(5/12), 2**(7/12), 2**(8/12), 2**(10/12)]
    root_ratio = root_freq / tuning
    return [note_freq * root_ratio for note_freq in
       minor_scale_intervals]

# Generate A major and A minor scales
A_major = major_scale(440)
A_minor = minor_scale(440)

# Generate a signal for each scale (1 second duration, 44100
   sampling rate)
fs = 44100
duration = 1
t = np.linspace(0, duration, duration * fs, endpoint=False)

A_major_signal = np.zeros_like(t)
A_minor_signal = np.zeros_like(t)

for freq in A_major:
    A_major_signal += np.sin(2 * np.pi * freq * t)

for freq in A_minor:
    A_minor_signal += np.sin(2 * np.pi * freq * t)

# Compute CQT and STFT
A_major_cqt = librosa.cqt(A_major_signal, sr=fs, n_bins=84,
   bins_per_octave=12, fmin=27.5)
A_major_stft = librosa.stft(A_major_signal)

A_minor_cqt = librosa.cqt(A_minor_signal, sr=fs, n_bins=84,
   bins_per_octave=12, fmin=27.5)
A_minor_stft = librosa.stft(A_minor_signal)

# Plot CQT and STFT for A major scale
plt.figure()
librosa.display.specshow(librosa.amplitude_to_db(np.abs(
   A_major_cqt), ref=np.max), sr=fs, x_axis='time', y_axis='
   cqt_note', cmap='viridis')
plt.title('Constant-Q␣Transform␣-␣A␣major␣scale')
plt.colorbar(format='%+2.0f␣dB')
plt.tight_layout()

plt.figure()
```

```
librosa.display.specshow(librosa.amplitude_to_db(np.abs(
    A_major_stft), ref=np.max), sr=fs, x_axis='time', y_axis='
    log', cmap='viridis')
plt.title('Short-Time Fourier Transform - A major scale')
plt.colorbar(format='%+2.0f dB')
plt.tight_layout()

# Plot CQT and STFT for A minor scale
plt.figure()
librosa.display.specshow(librosa.amplitude_to_db(np.abs(
    A_minor_cqt), ref=np.max), sr=fs, x_axis='time', y_axis='
    cqt_note', cmap='viridis')
plt.title('Constant-Q Transform - A minor scale')
plt.colorbar(format='%+2.0f dB')
plt.tight_layout()

plt
```

A.10 Manual Mel-Spectrogram implementation

```
def hz_to_mel(freq):
    return 2595 * np.log10(1 + freq / 700)

def mel_to_hz(mel):
    return 700 * (10**(mel / 2595) - 1)

def create_mel_filterbank(sample_rate, n_mels, n_fft):
    f_min = 0
    f_max = sample_rate / 2
    mel_min = hz_to_mel(f_min)
    mel_max = hz_to_mel(f_max)
    mel_points = np.linspace(mel_min, mel_max, n_mels + 2)
    hz_points = mel_to_hz(mel_points)
    bin_points = np.floor((n_fft + 1) * hz_points / sample_rate)
        .astype(int)
    filter_bank = np.zeros((n_mels, n_fft // 2 + 1))
    for i in range(1, n_mels + 1):
        left_bin = bin_points[i - 1]
        center_bin = bin_points[i]
        right_bin = bin_points[i + 1]

        for j in range(left_bin, center_bin):
            filter_bank[i - 1, j] = (j - bin_points[i - 1]) / (
                bin_points[i] - bin_points[i - 1])

        for j in range(center_bin, right_bin):
            filter_bank[i - 1, j] = (bin_points[i + 1] - j) / (
                bin_points[i + 1] - bin_points[i])

    return filter_bank

def compute_mel_spectrogram(audio_signal, sample_rate, n_mels,
    n_fft, hop_length):
    _, _, spec = stft(audio_signal, fs=sample_rate, nperseg=
        n_fft, noverlap=n_fft - hop_length)
```

```
    power_spec = np.abs(spec)**2
    filter_bank = create_mel_filterbank(sample_rate, n_mels,
       n_fft)
    mel_spec = np.dot(filter_bank, power_spec)
    return mel_spec
```

Appendix B

Proposed Models in Implementation

B.1 The proposed music transformer network

```
import torch
import torch.nn as nn
from torch.nn import Transformer

class AMTTransformer(nn.Module):
    def __init__(self, input_dim, output_dim, nhead, num_layers,
        d_model, dim_feedforward, dropout):
        super(AMTTransformer, self).__init__()

        self.embedding = nn.Linear(input_dim, d_model)
        self.pos_encoder = PositionalEncoding(d_model, dropout)

        self.transformer = Transformer(
            d_model=d_model,
            nhead=nhead,
            num_encoder_layers=num_layers,
            num_decoder_layers=num_layers,
            dim_feedforward=dim_feedforward,
            dropout=dropout,
        )

        self.output_layer = nn.Linear(d_model, output_dim)

    def forward(self, src, tgt, src_mask=None, tgt_mask=None,
       memory_mask=None):
        src = self.embedding(src)
        src = self.pos_encoder(src)

        tgt = self.embedding(tgt)
        tgt = self.pos_encoder(tgt)

        output = self.transformer(src, tgt, src_mask, tgt_mask,
           memory_mask)
        return self.output_layer(output)

class PositionalEncoding(nn.Module):
    def __init__(self, d_model, dropout, max_len=5000):
        super(PositionalEncoding, self).__init__()
        self.dropout = nn.Dropout(p=dropout)
```

```
        pe = torch.zeros(max_len, d_model)
        position = torch.arange(0, max_len, dtype=torch.float).
            unsqueeze(1)
        div_term = torch.exp(torch.arange(0, d_model, 2).float()
             * (-torch.log(torch.tensor(1e4)) / d_model))
        pe[:, 0::2] = torch.sin(position * div_term)
        pe[:, 1::2] = torch.cos(position * div_term)
        pe = pe.unsqueeze(0).transpose(0, 1)
        self.register_buffer('pe', pe)

    def forward(self, x):
        x = x + self.pe[:x.size(0), :]
        return self.dropout(x)

def preprocess_audio(audio_file, sr=22050, n_fft=2048,
    hop_length=512):
    y, _ = librosa.load(audio_file, sr=sr)
    spectrogram = librosa.stft(y, n_fft=n_fft, hop_length=
        hop_length)
    magnitude, phase = librosa.magphase(spectrogram)
    log_magnitude = librosa.amplitude_to_db(magnitude)

    return log_magnitude

device = torch.device("cuda" if torch.cuda.is_available() else "
    cpu")
amt_transformer = AMTTransformer().to(device)

criterion = torch.nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(amt_transformer.parameters(), lr
    =0.001)

num_epochs = 100
for epoch in range(num_epochs):
    for i, (input_data, target_data) in enumerate(data_loader):
        input_data, target_data = input_data.to(device),
            target_data.to(device)

        optimizer.zero_grad()
        output_data = amt_transformer(input_data)

        loss = criterion(output_data, target_data)
        loss.backward()
        optimizer.step()

        print(f"Epoch␣[{epoch+1}/{num_epochs}],␣Step␣[{i+1}/{len
            (data_loader)}],␣Loss:␣{loss.item():.4f}")

class AMTDataset(Dataset):
    def __init__(self, audio_files, transcription_files):
        self.audio_files = audio_files
        self.transcription_files = transcription_files

    def __len__(self):
        return len(self.audio_files)

    def __getitem__(self, idx):
```

```
        audio_file = self.audio_files[idx]
        transcription_file = self.transcription_files[idx]

        input_data = preprocess_audio(audio_file)
        target_data = load_transcription(transcription_file)

        return input_data, target_data

batch_size = 32
amt_dataset = AMTDataset(audio_files, transcription_files)
data_loader = DataLoader(amt_dataset, batch_size=batch_size,
    shuffle=True, num_workers=4)
```

B.2 Transcription and post-processing functions

```
import pretty_midi
import numpy as np

def load_transcription(transcription_file, sr=22050, hop_length
    =512, velocity_scale=127):
    midi_data = pretty_midi.PrettyMIDI(transcription_file)
    piano_roll = midi_data.get_piano_roll(fs=sr/hop_length)

    # Normalize the velocities
    piano_roll = piano_roll / velocity_scale

    return piano_roll

def postprocess_output(output_data, sr=22050, hop_length=512,
    velocity_scale=127):
    # Convert output data to piano roll format
    piano_roll = output_to_piano_roll(output_data)

    # Denormalize the velocities
    piano_roll = piano_roll * velocity_scale

    # Convert the piano roll to a MIDI file
    midi_data = piano_roll_to_midi(piano_roll, fs=sr/hop_length)

    return midi_data

def output_to_piano_roll(output_data):
    # Convert the output data into a piano roll format
    # This depends on the specific format of the output_data
        tensor.
    pass

def piano_roll_to_midi(piano_roll, fs):
    midi_data = pretty_midi.PrettyMIDI()
    piano_program = pretty_midi.instrument_name_to_program('
        Acoustic␣Grand␣Piano')
    piano = pretty_midi.Instrument(program=piano_program)

    for note_number in range(piano_roll.shape[0]):
```

```
        start_time = None

        for time_step in range(piano_roll.shape[1]):
            velocity = piano_roll[note_number, time_step]

            if velocity > 0 and start_time is None:
                start_time = time_step * (1/fs)
            elif velocity == 0 and start_time is not None:
                end_time = time_step * (1/fs)
                note = pretty_midi.Note(velocity=int(np.max(
                   piano_roll[note_number, time_step])),
                                        pitch=note_number,
                                        start=start_time,
                                        end=end_time)
                piano.notes.append(note)
                start_time = None

    midi_data.instruments.append(piano)

    return midi_data
```

Bibliography

[1] Ashish Vaswani et al. *Attention Is All You Need*. 2017. DOI: 10.48550/ARXIV.1706.03762. URL: https://arxiv.org/abs/1706.03762.

[2] Sebastian Stober and Andreas Nürnberger. "Adaptive Music Retrieval–A State of the Art". In: *Multimedia Tools and Applications* 65.3 (2012), pp. 467–494.

[3] Christoph Weis, Hendrik Schreiber, and Meinard Muller. "Local Key Estimation in Music Recordings: A Case Study Across Songs, Versions, and Annotators". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 28 (2020), pp. 2919–2932.

[4] Özer Izmirli. *Localized Key Finding from Audio Using Nonnegative Matrix Factorization for Segmentation*. Semantic Scholar. [Accessed: 9-Feb-2023]. 2007. URL: https://www.semanticscholar.org/paper/Localized-Key-Finding-from-Audio-Using-Nonnegative-Izmirli/90eb4b7d5ac726bb8b5be2c4beb75ae960cf4453.

[5] Chee-Hoo Chuan and Elaine Chew. "Fuzzy Analysis in Pitch-Class Determination for Polyphonic Audio Key Finding". In: *Proceedings of the 6th International Conference on Music Information Retrieval*. 2005, pp. 296–303.

[6] Gabriel Bernardes, Matthew E. Davies, and Carlos Guedes. "Automatic Musical Key Estimation with Adaptive Mode Bias". In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2017.

[7] Alexander Lerch. "On the Requirement of Automatic Tuning Frequency Estimation". In: *Journal of New Music Research* 35.2 (2006), pp. 121–133.

[8] H. Papadopoulos and G. Peeters. "Simultaneous Estimation of Chord Progression and Downbeats from an Audio File". In: *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*. 2008.

[9] Wei Chai and Barry Vercoe. "Detection of Key Change in Classical Piano Music". In: *Proceedings of the 2005 International Computer Music Conference*. 2005, pp. 468–473.

[10] Geoffroy Peeters. "Musical Key Estimation of Audio Signal Based on Hidden Markov Modeling of Chroma Vectors". In: *Proceedings of the 8th International Conference on Digital Audio Effects*. 2006, pp. 155–160.

[11] Marcin Kania et al. "A Comparison of the Music Key Detection Approaches Utilizing Key-Profiles with a New Method Based on the Signature of Fifths". In: *Applied Sciences* 12.21 (Nov. 2022), p. 11261. DOI: 10.3390/app122111261.

[12] Paul Brossier. "Fast Onset Detection Using Aubio (Broissier)". In: *Music Information Retrieval Evaluation eXchange (MIREX)* (2005).

[13] Meinard Müller et al. "Towards Automated Extraction of Tempo Parameters from Expressive Music Recordings". In: *Proceedings of the 10th International Society for Music Information Retrieval Conference*. 2009, pp. 69–74.

[14] Peter Grosche, Meinard Müller, and Frank Kurth. "Cyclic Tempogram–A Mid-level Tempo Representation for Music Signals". In: *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*. 2010, pp. 5522–5525.

[15] Geoffroy Peeters. "Template-Based Estimation of Time-Varying Tempo". In: *EURASIP Journal on Advances in Signal Processing* 2007.1 (2006).

[16] Matthias Mauch and Simon Dixon. "Approximate Note Transcription for the Improved Identification of Difficult Chords". In: *Proceedings of the 11th International Society for Music Information Retrieval Conference*. 2010, pp. 135–140. DOI: 10.5281/zenodo.1416598.

[17] Elaine Chew. *Mathematical and Computational Modeling of Tonality: Theory and Applications*. Vol. 204. Jan. 2014. ISBN: 978-1-4614-9474-4. DOI: 10.1007/978-1-4614-9475-1.

[18] Perry Cook and Jerry Tobias. "Music, Cognition, and Computerized Sound: An Introduction to Psychoacoustics". In: *Journal of The Acoustical Society of America - J ACOUST SOC AMER* 107 (Jan. 2000), pp. 21–22. DOI: 10.1121/1.428355.

[19] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. DOI: 10.48550/ARXIV.2005.14165. URL: https://arxiv.org/abs/2005.14165.

[20] Aakanksha Chowdhery et al. *PaLM: Scaling Language Modeling with Pathways*. 2022. DOI: 10.48550/ARXIV.2204.02311. URL: https://arxiv.org/abs/2204.02311.

[21] Prafulla Dhariwal et al. *Jukebox: A Generative Model for Music*. 2020. DOI: 10.48550/ARXIV.2005.00341. URL: https://arxiv.org/abs/2005.00341.

[22] Zalán Borsos et al. *AudioLM: a Language Modeling Approach to Audio Generation*. 2022. DOI: 10.48550/ARXIV.2209.03143. URL: https://arxiv.org/abs/2209.03143.

[23] Valentin Emiya et al. *MAPS - A piano database for multipitch estimation and automatic transcription of music*. July 2010.

[24] Rachel Bittner et al. "MedleyDB: A Multitrack Dataset for Annotation-Intensive MIR Research". In: Oct. 2014.

[25] John Thickstun, Zaid Harchaoui, and Sham M. Kakade. "Learning Features of Music from Scratch". In: *International Conference on Learning Representations (ICLR)*. 2017.

[26] Thierry Bertin-Mahieux et al. "The Million Song Dataset". In: *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*. 2011.

[27] Michaël Defferrard et al. *FMA: A Dataset For Music Analysis*. 2017. arXiv: 1612.01840 [cs.SD].

[28] Colin Raffel. "Learning-Based Methods for Comparing Sequences, with Applications to Audio-to-MIDI Alignment and Matching". In: 2016.

[29] Curtis Hawthorne et al. "Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=r1lYRjC9F7.