BRNO UNIVERSITY OF TECHNOLOGY

Faculty of Electrical Engineering
and Communication

DOCTORAL THESIS

Brno, 2024 Ing. Matěj Ištvánek

BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF TELECOMMUNICATIONS
ÚSTAV TELEKOMUNIKACÍ

COMPUTATIONAL METHODS FOR COMPARATIVE MUSIC PERFORMANCE ANALYSIS

VÝPOČETNÍ METODY PRO SROVNÁVACÍ HUDEBNÍ ANALÝZU INTERPRETAČNÍHO VÝKONU

DOCTORAL THESIS
DIZERTAČNÍ PRÁCE

AUTHOR AUTOR PRÁCE	Ing. Matěj Ištvánek
SUPERVISOR ŠKOLITEL	prof. Ing. Zdeněk Smékal, CSc.

BRNO 2024

ABSTRACT

The availability of digital music content and various interpretations of musical pieces is increasing rapidly. Simultaneously, even though the computational methods of Music Information Retrieval (MIR) are evolving at a quick pace, they are not always reflected in related fields such as Music Performance Analysis (MPA). The main topic of this dissertation is the utilization of computational methods and digital music processing for the goals of MPA. It aims to combine MIR principles to analyze and compare differences in musical performances and their parameters. The thesis examines the limitations of conventional and machine learning-based onset and beat detectors to correctly estimate the ground-truth data under the effect of input audio degradations, sampling rate reductions, or in complex musical structures. Furthermore, this work shows the possibilities of music synchronization, parameter extraction, and feature selection application on novel string quartet data to provide a semi-automated strategy for binary classification of performers' origin. Finally, it demonstrates a software tool for comparative music analysis by combining a comfortable user environment for playback, navigation, and visualization of music performance data with the computation methods of MIR.

KEYWORDS

music information retrieval; music performance analysis; music processing; onset detection; beat detection; synchronization; comparative analysis; string quartets; software

ABSTRAKT

Zvyšující se dostupnost digitálního hudebního obsahu a různých interpretací hudebních děl zároveň podněcuje vývoj výpočetních metod oboru získávání hudebních informací (MIR). Tento rozvoj ale nemusí být vždy reflektován v souvisejících oblastech – například v analýze interpretačního výkonu (MPA). Hlavním tématem této disertační práce je využití výpočetních metod a digitálního zpracování hudby pro cíle MPA. Tato práce se zabývá principy MIR pro analýzu a porovnání rozdílů hudebních výkonů a jejich parametrů. Práce analyzuje limitace detektorů nástupů tónů a dob založených na konvenčních přístupech a na strojovém učení, při degradaci vstupního signálu, snížení vzorkovací frekvence, nebo v komplexnějších hudebních strukturách. Dále ukazuje možnosti použití hudební synchronizace, extrakce parametrů a výběru příznaků na originálních datech smyčcových kvartetů pro binární klasifikaci původu interpretů. Na závěr demonstruje vyvíjený software pro komparativní analýzu hudby, který kombinuje přívětivé uživatelské prostředí pro přehrávání, navigaci a vizualizaci dat hudebního výkonu s výpočetními metodami MIR.

KLÍČOVÁ SLOVA

získávání hudebních informací; analýza interpretačního výkonu; zpracování hudby; detekce nástupů; detekce dob; synchronizace; srovnávací analýza; smyčcové kvartety; software

IŠTVÁNEK, Matěj. *Computational Methods for Comparative Music Performance Analysis*. Brno, 2024, 177 p. Doctoral thesis. Brno University of Technology, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Advised by prof. Ing. Zdeněk Smékal, CSc.

Typeset by the `thesis` package, version v1.4.0; `http://latex.feec.vutbr.cz`

DECLARATION

I declare that I have written the Doctoral Thesis titled “Computational Methods for Comparative Music Performance Analysis” independently, under the guidance of the advisor and using exclusively the technical references and other sources of information cited in the thesis and listed in the comprehensive bibliography at the end of the thesis.

As the author, I furthermore declare that, with respect to the creation of this Doctoral Thesis, I have not infringed any copyright or violated anyone’s personal or ownership rights. In this context, I am fully aware of the consequences of breaking Regulation § 11 of the Copyright Act No. 121/2000 Coll. of the Czech Republic, as amended, and of any breach of rights related to intellectual property or introduced within amendments to relevant Acts such as the Intellectual Property Act or the Criminal Code, Act No. 40/2009 Coll., Section 2, Head VI, Part 4.

Brno . . . .

author’s signature

Contents

List of Figures

List of Tables

List of Abbreviations

DSP	Digital Signal Processing
ABD	Average Beat Deviation
A/D	Analog-to-Digital
ADT	Audio Degradation Toolbox
ADSR	Attack—Decay—Sustain—Release
ae	annotation efficiency
AMT	Automatic Music Transcription
BLSTM	Bidirectional LSTM
BPM	Beats Per Minute
BW	Bowed String
CBR	Constant Bit Rate
CC	Creative Commons
CENS	Chroma Energy Normalized Statistic
CM	Complex Mixtures
CNN	Convolutional Neural Network
CQT	Constant-Q Transform
DBN	Dynamic Bayesian Network
DFT	Discrete Fourier Transform
DS	Detection System without the TKEO
DSP	Digital Signal Processing
DTW	Dynamic Time Warping
EAT	Estimated Average Tempo
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FN	False Negative
FP	False Positive
fps	frames per second
FT	Fourier Transform
GAN	Generative Adversarial Network
GT	Global Tempo or Ground-Truth

GTT	Ground-Truth Tempo
HMM	Hidden Markov Model
IEEE	Institute of Electrical and Electronics Engineers
ISMIR	International Society of Music Information Retrieval
LGD	Local Group Delay
LSTM	Long Short-Term Memory
LUFS	Loudness Unit Full Scale
MDCT	Modified Discrete Cosine Transform
MIDI	Musical Instrument Digital Interface
MIR	Music Information Retrieval
MIREX	Music Information Retrieval Evaluation eXchange
ML	Machine Learning
mRMR	minimum-Redundancy Maximum-Relevance
MrMsDTW	Memory restricted Multi-scale DTW
MPA	Music Performance Analysis
NN	Neural Network
NPP	Non-Pitched Percussive
ODF	Onset Detection Function
PCM	Pulse Code Modulation
PP	Pitched Percussive
RBF	Radial Basis Function
RNN	Recurrent Neural Network
RMS	Root Mean Square
RPM	Rounds Per Minute
SF	Spectral Flux or Super Flux
STFT	Short-Time Fourier Transform
SVM	Support Vector Machines
TCN	Temporal Convolutional Network
TISMIR	Transactions of ISMIR
TKEO	Teager–Kaiser Energy Operator
TN	True Negative
TP	True Positive
TS	Detection System with the TKEO
WI	Wind Instruments

List of Basic Symbols and Notions

$\mathbb{N}$	natural numbers $\{1, 2, 3, \dots\}$
$\mathbb{Z}$	integers $\{\dots - 1, 0, 1, \dots\}$
$\mathbb{R}$	real numbers
$\mathbb{R}^N$	real coordinate space of dimension N
$\mathbb{R}_{>0}$	positive real numbers
$\mathbb{C}$	complex numbers
j	imaginary unit; equivalent to i
$[a : b]$	integers from a to b, where $a, b \in \mathbb{Z}$
$[a, b]$	interval from a to b, where $a, b \in \mathbb{R}$
$\langle x \vert y \rangle$	inner product of two vectors x, y
$\vert x \vert$	absolute value of a number x
$\Vert x \Vert$	norm of a vector x
$\lfloor x \rfloor$	floor of a number x, i.e., greatest integer $\leq x$

Preface

Music is a universal language across all parts of the world. Its purpose, features, and underlying properties can be understood differently based on tradition and cultural variance. The ability to perceive harmonic, melodic, and rhythmic patterns, including their relationships within a given musical piece, begins to develop already in the earliest stage of one's life. It may be gradually refined into musical knowledge and experience, allowing individual music analysis at different levels. Each educated musician analyzes the music, even unconsciously, when listening or performing. The perception of music is a complex mechanism, from changes in local air pressure and transfer of vibrations to the cochlea, a spiral-shaped cavity in the bony labyrinth of our ears, up to evaluation in the brain. Our ears process the sound waves—from amplification and transformation into the frequency representation—and send them via electrical signals into our control unit. But what happens next? Are musical education, knowledge, and experience correlated with how we listen and perform music? Is it possible to distinguish musicians or a group of musicians with the same cultural or musical background objectively based on their performance? Can we measure and compare differences in music expressivity?

Transferring the music analysis capabilities of a human into the computing domain contains many obstacles that we have not yet overcome. When describing music subjectively, one may start with expressions of our emotional state or feelings. However, in today's digitized world, there is a strong need for an objective description, although even the technical descriptions of sound parameters, such as dynamics or timbre, are related to our subjective perception. This results in many studies focused on music information retrieval, extraction of relevant music features, or digital music analysis. As technology and computational methods evolve, we find new ways of dealing with music data, including analysis, sorting, processing, recommending, and changing and transforming their properties. By extracting the underlying features of music, we may be able to understand the context of music and its differences across various cultural backgrounds or music schools. Furthermore, we could build on previous analytic approaches and provide new forms of comparative music analysis, focusing on distinguishable traits of individual performances in large-scale datasets.

In this work, I contribute to some of the questions raised; however, due to the state of current knowledge in the related fields, all chapters and discussed topics are far from exhausted and resolved. I aim to provide some insight into performance data and comparative performance analysis topics by utilizing methods of the music information retrieval field, focusing on a more technical view of music processing and leaving out the processing of the human brain. Due to the nature of music performance data and the limitations of automated extraction systems in the past (and even today), most performance studies use a very limited number of recordings. Researchers rely on manual annotations of data, hand-crafted features, or very specific scenarios. In this thesis, I address some tasks of music information retrieval and performance analysis from an application point of view. I briefly describe the history and principles of both interdisciplinary fields and present the motivation, ideas, and some of the methods I used during experiments. The rest is covered in a given literature or articles I authored or co-authored (Chapters 2–9). The main topic, present throughout the thesis, is the utilization of retrieval methods for comparing music performances and decreasing the time of the annotation process for music analysis scenarios.

Structure of the Thesis

This thesis is divided into nine chapters and two parts. The first part summarizes my contributions to low-level and mid-level detection systems and their evaluation based on the principles of music performance analysis. The second part analyzes and compares music performance differences using information retrieval methods. Note that the chapters are not always chronological (compared to the year in which the corresponding articles were published) but are arranged to follow a more consistent thesis structure.

Chapter 1 presents the two related fields, Music Information Retrieval and Music Performance Analysis, and the related fundamentals of music processing, including audio signals, time-frequency transformations, music synchronization based on dynamic time warping, and its modified variants.

Chapter 2 starts the part I of the thesis. It is based on the article "The Effect of Audio Degradation on Onset Detection Systems" [1] and introduces the notion of various degradations in the pre-processing phase of onset detectors.

Chapter 3 follows Chapter 2 by focusing on MPA-oriented global tempo computation and introduces the article "Enhancement of Conventional Beat Tracking System Using Teager–Kaiser Energy Operator" [2]. This article is an updated version (nominated for further publishing in a journal) of two previous consecutive conference articles [3] and [4].

Chapter 4 introduces the article "Beat Tracking: Is 44.1 kHz Really Needed?" [5] that won first place in the Audio, Speech, and Language Processing category at the EEICT 2023 student conference[1]. It experiments with and evaluates different input sampling rates of beat tracking models. I later used one of the models with

[1] `www.eeict.cz/eeict_download/archiv/vysledky/EEICT_2023_vysledky_v2.pdf`, p. 18

slight modifications in the follow-up studies [6, 7] and the MemoVision software (see Chapter 9 and Appendix D) for a combined synchronization approach inspired by the collaboration (and my internship topic) and the research in Audio Labs institute in Germany.

Chapter 5 is based on the article “The Application of Tempo Calculation for Musicological Purposes” [8], which also won first place in the Signal, Image, and Data Processing category at the EEICT 2021 student conference[2]. It evaluates conventional and machine learning-based beat tracking systems on two challenging string quartet motifs. I tried to explain the problems of applying beat detectors in music analysis scenarios and the standard beat tracking and tempo evaluation metrics.

Chapter 6 presents the article “Exploring the Possibilities of Automated Annotation of Classical Music with Abrupt Tempo Changes” [9] that won second place in the Analog and Digital Signal Processing category at the EEICT 2022 student conference[3]. It follows the ideas of the previous chapter by jointly evaluating a synchronization method, beat detector, and downbeat detector utilizing user-driven metrics.

Chapter 7 starts part II of the thesis and introduces the article “Towards Automatic Measure-Wise Feature Extraction Pipeline for Music Performance Analysis” [10]. I deployed synchronization strategy and parameter extraction based on previous studies to create representative matrices for analyzing interpretation differences.

Chapter 8 presents the article “Classification of Interpretation Differences in String Quartets Based on the Origin of Performers” [11] that uses the ideas of [10] and trains the machine learning classifier to distinguish between two separate groups of recordings based on performance parameters. In the study, we predicted the Czech and non-Czech origin of interpretations with relatively high accuracy within large corpora of Czech string quartet music.

Chapter 9 follows with the article “Application of Computational Methods for Comparative Music Analysis” [6], introducing the MemoVision software, its functions, and contributions to MIR and MPA communities.

Additional experiments with the application of the MemoVision software on a piano composition from Bedřich Smetana are demonstrated in Appendix D.

[2]`www.eeict.cz/eeict_download/archiv/vysledky/EEICT_2021_vysledky_new.pdf`, p. 22
[3]`www.eeict.cz/eeict_download/archiv/vysledky/EEICT_2022_vysledky.pdf`, p. 20

Publications Related to the Ph.D. Thesis

Chapters 2–9 of this thesis were published in peer-reviewed conferences and journals in the fields of audio processing, digital signal processing, music information retrieval, machine learning, or music performance analysis. Furthermore, supplementary code and/or additional materials were released on GitHub repositories (see below). I also authored the “Methodology of Phonograph Cylinders Digitization” [12] and wrote a chapter about audio codecs for the book “Saving the audio cultural heritage” by M. Lorenz et al. [13] together with the Czech Museum of Music, Prague. I successfully submitted and finished two projects:

- “Identification of the Czech origin of digital music recordings using machine learning” grant was realized within the project Quality Internal Grants of BUT (KInG BUT), Reg. No. CZ.02.2.69 / 0.0 / 0.0 / 19 073 / 0016948 and financed from the OP RDE. Related publications are: [9, 10, 14, 11].
- TL05000527 “Memories of Sound: The Evolution of the Interpretative Tradition Based on the Works of Antonin Dvorak and Bedrich Smetana” was co-financed with the state support of the Technology Agency of the Czech Republic within the ETA Program. Related publications are: [6, 7, 15, 16].

First author publications:

[1] M. Ištvánek, “The effect of audio degradation on onset detection systems,” *Elektrorevue*, vol. 24, no. 1, pp. 1–13, 2022.

[2] M. Ištvánek, Z. Smékal, L. Spurný, and J. Mekyska, “Enhancement of conventional beat tracking system using Teager–Kaiser energy operator,” *Applied Sciences*, vol. 10, no. 1, pp. 1–20, 2020.

[5] M. Ištvánek and Š. Miklánek, “Beat tracking: Is 44.1 khz really needed?” in *Proceedings of the conference STUDENT EEICT 2023 Selected Papers*, Brno, Czech Republic, 2023, pp. 227–231.

[8] M. Ištvánek, “The application of tempo calculation for musicological purposes,” in *Proceedings of the conference STUDENT EEICT 2021 Selected Papers*, Brno, Czech Republic, 2021, pp. 265–269.

[9] M. Ištvánek and Š. Miklánek, “Exploring the possibilities of automated annotation of classical music with abrupt tempo changes,” in *Proceedings of the conference STUDENT EEICT 2022 Selected Papers*, Brno, Czech Republic, 2022, pp. 286–290.

[10] M. Ištvánek and Š. Miklánek, “Towards automatic measure-wise feature extraction pipeline for music performance analysis,” in *45th International Conference on Telecommunications and Signal Processing (TSP)*, 2022, pp. 192–195.

[11] M. Ištvánek, Š. Miklánek, and L. Spurný, “Classification of interpretation differences in string quartets based on the origin of performers,” *Applied Sciences*, vol. 13, no. 6, pp. 1–20, 2023.

[6] M. Ištvánek, Š. Miklánek, K. H. Mühlová, L. Spurný, and Z. Smékal, “Application of computational methods for comparative music analysis,” *2023 4th International Symposium on the Internet of Sounds*, 2023, pp. 1–6.

[7] M. Ištvánek and Š. Miklánek, “Memovision: a tool for feature selection and visualization of performance data,” in *Extended Abstracts for the Late-Breaking Demo Session of the 24th International Society for Music Information Retrieval Conference (ISMIR)*, Milan, Italy, 2023, p. 3.

[16] M. Ištvánek, K. H. Mühlová, L. Spurný, and M. Mejzr, “Metodika analýzy hudebně-interpretačního výkonu s podporou the memovision software,” certified methodology, p. 56, 2023

Second author publications:

[14] Y. Özer, M. Ištvánek, V. Arifi-Müller, and M. Müller, “Using activation functions for improving measure-level audio synchronization,” in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Bengalúru, India, 2022, pp. 750–756.

[15] L. Spurný, M. Ištvánek, and J. Jiraský, “Paměť zvuku (smetana–dvořák–janáček). k metodě zpracování zvukových nahrávek pomocí nástroje memovision,” *Hudební věda*, vol. 55, no. 4, pp. 501–533, 2023.

GitHub repositories and software:

[17] GitHub: The Effect of Audio Degradation on Onset Detection Systems, supplementary data
[18] GitHub: Beat Tracking TCN, supplementary data
[19] GitHub: Classification of Interpretation Differences, supplementary data
[20] GitHub: The Audio Duplicate Finder Software
[21] GitHub: The MemoVision Software

Acknowledgments

First, I would like to thank my supervisor, Prof. Zdeněk Smékal. He never doubted me—he always encouraged and motivated me. He was a pillar of strength, even in moments of doubt.

I am grateful that I had the opportunity to study at Brno University of Technology and meet many interesting people. I would like to explicitly express my gratitude to my colleague and friend-in-battle Štěpán Miklánek. He was always hardworking, resourceful, and creative. Furthermore, I would like to thank Prof. Meinard Müller for constructive feedback and an aura of motivation during my internship in Audio Labs and Yigitcan Özer, a colleague and, especially, a great friend.

I am truly grateful for my family and closest friends. They have been the reason I do what I like and enjoy it. They create a place I love to call home. I wish everybody could experience this.

Lili, I would like to explain everything once you are old enough to understand.

1 Basics of Music Processing and Analysis

This chapter explains the basics of music processing related to the thesis. This covers a quick overview of music performance analysis and music information retrieval fields (Sections 1.1 and 1.2), audio representations (Section 1.3), and music synchronization (Section 1.4). The last two sections are substantially inspired by [22] and [23].

1.1 Music Performance Analysis

Music Performance Analysis (MPA) is an interdisciplinary field that focuses on extracting information, evaluating, and analyzing musical performances. It provides a connection between the subjective experience of music and objective data analysis [24]. The performance or interpretation[1] plays a critical role in how listeners perceive a given composition. Variations of musical parameters shape the expressivity of the resulting interpretation. The blueprint, usually in the form of a musical score or sheet music, is not the performed music itself, as it first requires acoustic realization [25, 26]. Different performers may interpret the same blueprint uniquely, creating variations of performance data. Furthermore, the availability of music increased radically in previous years thanks to the internet, lossy audio data compressions, and streaming portals, leading to more options for inspiration. The performers are usually musicians, but in recent years, the advances in computational models such as neural networks have allowed us to consider new systems as possible performers with human-like deviations. Music as a performing art requires a performer or multiple performers to render the musical ideas into a signal domain, usually in the form of physical sound that is then perceived by the listener's ears. However rich, for all purposes of this thesis, we assume and use Western musical concepts, tuning, and notation.

There are multiple forms of relevant music data such as sheet music, midi transcription, audio signal, or derived representations like spectrograms and chroma vectors. However, some aspects of music performance can also be described by visual information, e.g., by gestures and facial expressions of musicians [27, 28, 29]. Such data are usually unavailable for further processing or do not convey performance cues. The focus of MPA research is mostly on symbolic data and audio recordings. The preprocessing adjustments (recording equipment, position of microphones), processing workflow (digital effects, mixing, mastering), and postprocessing choices (effects, audio carriers, and formats) of music recordings affect the performance data

[1]For the purpose of this thesis, I use both terms interchangeably.

and the objectivity of the analysis. If we use audio recordings, we cannot simply separate the performer's intentions from all other aspects of the data. However, the parameter variations are sometimes subtle, and in the case of MPA, they are usually evaluated in reference to the same performer, different interpretations, or symbolic score representation [24]. The research drive to generalize performance principles naturally leads to the analysis of classical music, where a large number of interpretations of the same piece may be available compared to other genres of music. This work focuses mostly on the string quartet and piano music. The MPA research covers but is not limited to:

- expressivity: the term itself is vague and may cover multiple research topics,
- timing: time-related features such as onsets, offsets, beats, downbeats,
- intonation: intentional and unintentional changes in pitch, vibrato,
- timbre: a spectrum of instruments, the relation between timbre, dynamics, and expressivity,
- dynamics: loudness and tempo, structure-related dynamic choices,
- musical structure: traits of compositions, relation of the structure and performance,
- psychological aspects: communication between performers, the influence of music,
- computational analysis: retrieving of music-related information from audio signals,
- pedagogy and education: aiding the pedagogical process by understanding underlying music material,
- historical styles: how music evolved in time.

1.2 Music Information Retrieval

Thanks to the digital revolution in distributing and storing audio data, music has become easily available and popular multimedia content [30]. Music processing is one of the fields that thrives from advances in digital signal processing (DSP), computer science, and machine learning (ML). It extends our capabilities of understanding, accessing, analyzing, and manipulating music. The music industry is pushing towards the continuous production of new musical pieces, especially in the pop genre, so the number of listeners increases along with cloud services and streaming portals. The need for sorting, retrieval, and recommendation algorithms is unavoidable. The first sign of Music Information Retrieval (MIR) emerged in the paper [31] in 1966 as "Musical Information Retrieval". The growth of the MIR community was not rapid; it slowly increased with the availability of data, computing power, and new communication options. The ISMIR (International Society for Music Information Retrieval) conference[2] has been held annually since 2000 and is currently the leading forum for processing, retrieving, sorting, and accessing music data. It supports and follows the open-access guidelines and includes topics from musicology, cognitive science, psychoacoustics, computer science, computational intelligence, machine learning, and electrical engineering. Furthermore, the open-access journal Transactions of the International Society for Music Information Retrieval (TISMIR)[3] complements the conference and publishes substantial scientific research in the field of MIR. There used to be the Music Information Retrieval Evaluation eXchange (MIREX) initiative[4] with a focus on comparing and evaluating detection systems of MIR-related challenges, but the last official meeting of this community was at ISMIR 2021. The MIR topics cover but are not limited to:

- feature extraction and modeling: low-, mid-, and high-level features (for example, onset, beat, and downbeat detections),
- music classification: music identification, pattern matching, music structure analysis,
- music recommendation: recommendation systems, database handling,
- source separation: sound source identification and separation (for example, voice or instrument separation), instrument recognition,
- music generation: automatic generation, autoencoders, generative adversarial networks,
- automatic music transcription: creating symbolic music notation from audio recordings,
- differentiable signal processing: modeling of audio effects and systems,
- retrieval: fingerprinting (Shazam-like applications), query-related tasks (query by humming or singing).

[2]`https://www.ismir.net`
[3]`https://transactions.ismir.net`
[4]`https://www.music-ir.org/mirex/wiki/MIREX_HOME`

1.3 Audio Representations

1.3.1 Digital Signals

In the audio processing sense, sound is a deviation of the air pressure from the average atmospheric air pressure in time. The vibration of a physical object with a frequency between circa 20 Hz to 20 kHz or 16 Hz to 16 kHz (limitation of human capabilities) causes oscillations of air molecules, resulting in regions of compression and rarefaction. The oscillation propagation through the transmission environment is called a sound wave, which can be converted via a microphone into electrical voltage levels. The function of continuous-time analog signal $f : \mathbb{R} \to \mathbb{R}$ assigns a value $f(t) \in \mathbb{R}$ to each point in time $t \in \mathbb{R}$ [32] and is often represented as a waveform (Figure 1.1). Note that all discrete values are interpolated, resulting in a smooth and seemingly continuous curve. In this thesis, to follow the vocabulary of DSP, we use the term "amplitude" only as a maximum value of a harmonic signal (yielding one value for a given function), which is not the case when displaying a time-varying waveform. Note that the thesis does not differentiate between round and square brackets for continuous or discrete signals to follow the standards and notions of the MIR community.

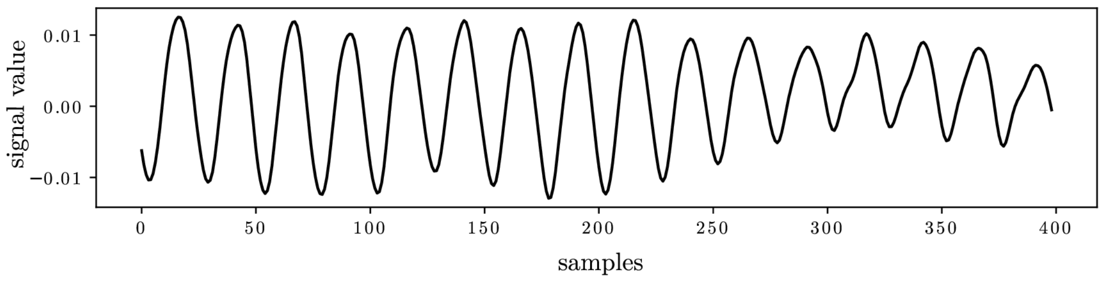


Fig. 1.1: Waveform visualization (400 samples) of a piano recording.

A signal is the carrier of information and can be transformed into a digital signal by analog-digital (A/D) conversion with two standard processes—sampling and quantization. First, a sampling process transforms the analog signal into, e.g., an equidistant discrete set of values $x(n)$ with a sample index $n \in \mathbb{Z}$ with sampling period $T \in \mathbb{R}_{>0}$:

$$x(n) = f(n \cdot T). \tag{1.1}$$

The sampling rate f_s (or sr in some literature), which is one of the key parameters of discrete and digital signals, is defined as $f_\mathrm{s} = 1/T$. The signal is defined on a discrete set of points, but the value can be any real number and may be of infinite length. This sampling procedure can be considered downsampling as we no longer have information about values between discrete points in time, inevitably losing some information. However, the Nyquist–Shannon theorem states that the analog signal can be perfectly reconstructed from its discrete version if it does not contain any frequencies above the Nyquist frequency $\omega \in \mathbb{R}_{>0}$ defined as $\omega = f_{\mathrm{s}}/2$. Common sampling rates for music recordings are 44.1, 48, and 96 kHz. Theoretically, we need at least 40 kHz to sample the audio material (about twice the highest frequencies one can hear, although they do not usually convey any useful information, and the overall ability to hear higher frequencies is also individual and depends, e.g., on one's age). In music analysis and for music-related feature extraction, the common sampling rate is 22.05 kHz or 16 kHz because most of the relevant information is contained in the lower frequency band. For example, a sampling rate of 8 kHz may be sufficient for some speech processing and telecommunications applications.

The last step to obtain a digital signal from a time-discrete variant is to discretize the values of $x(n)$ in a process called quantization. As a simple example with uniform quantization, we can map the values a of $x(n)$ to the quantized values $Q(a)$ as follows:

$$Q(a) = \mathrm{sqn}(a) \cdot \delta \cdot \left\lfloor \frac{|a|}{\delta} + \frac{1}{2} \right\rfloor, \tag{1.2}$$

where $\mathrm{sqn}(a)$ is the signum function that yields the sign of the value a, δ is the quantization stepsize, and $\delta \in \mathbb{R}_{>0}$. Furthermore, $a \in \mathbb{R}$ and $\delta = 1/2^d$, where d is a bit depth, a property of an A/D converter that describes how many bits are available to store the values. For example, an 8-bit converter has a resolution of $2^8 = 256$ quantization levels. The common bit depths for digital audio recordings are 16 and 32 bits. We end up with a digital signal that a computer can process, analyze, and modify.

1.3.2 Fourier Transform

The Fourier transform (FT) was named after the French mathematician Jean-Baptiste Joseph Fourier (1768–1830) [33], and it is one of the most used algorithms in the world. The name *Fourier transform* occurred for the first time in an article by Edward Charles Titchmarsh [34] in 1923, although it was used earlier in 1915 in an article by Michel Plancherel [35] but not in the same sense as we use it today. It was first described probably by Carl Friedrich Gauss (1777–1855) and discovered/re-discovered independently by many people. Johann Peter Gustav Lejeune-Dirichlet (1805–1854) published a famous article showing for which conditions the convergence of the Fourier series (Fourier series deals only with periodic signals) holds [36]. There are more definitions of Fourier transform but in this thesis, we use the complex definition for its elegant explanation of magnitude and phase components.

The main concept of FT is a harmonic analysis—decomposition of a signal into its frequency components (spectrum). In our analysis scenarios, we use a discrete variant of the Fourier transform called Discrete Fourier Transform (DFT). We can define the complex DFT $X(k)$ of a signal $x(n)$ as:

$$X(k) = \sum_{n=0}^{N-1} x(n) \mathrm{e}^{-\frac{2\pi \mathrm{j} k n}{N}}, \tag{1.3}$$

where $k \in [0 : N-1]$ is a frequency index and $N \in \mathbb{N}$. We can also define the inverse DFT to obtain the original signal $x(n)$ from its transformation $X(k)$:

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) \mathrm{e}^{\frac{2\pi \mathrm{j} k n}{N}}, \tag{1.4}$$

with the same range of k. Note that the sign in the exponent has to be inverse to the forward DFT, and the resulting spectrum $X(k)$ is non-periodic (as if it was only one period of a periodic DFT) and symmetric. The Fourier coefficients $X(k)$ are complex values that yield magnitude and phase information of physical frequencies $k \cdot f_\mathrm{s}/N$, but one usually considers the frequency range between 0 and $f_\mathrm{s}/2$ Hz. Computing DFT is computationally expensive, and the complexity is $\mathcal{O}(N^2)$ compared to the elegant and efficient Fast Fourier Transform (FFT) algorithm from 1965 by Coolie and Tukey [37] with $\mathcal{O}(N \log_2 N)$ complexity that is used in most of the applications. In such a case, the length of the sequence N has to be a power of 2. It was first published as a method of effectively computing Fourier series coefficients and later used for the fast DFT. Figure 1.2 shows a single side spectrum of a short audio excerpt. Negative frequencies are omitted as they carry no additional information for real signals. We can observe dominant peaks in the spectrum, distinguishing frequencies that correspond to tones or their overtones. However, we have no information about the timing of the onset or offset of individual tones. We need a transformation that captures the timing of musical events while providing sufficient frequency resolution.

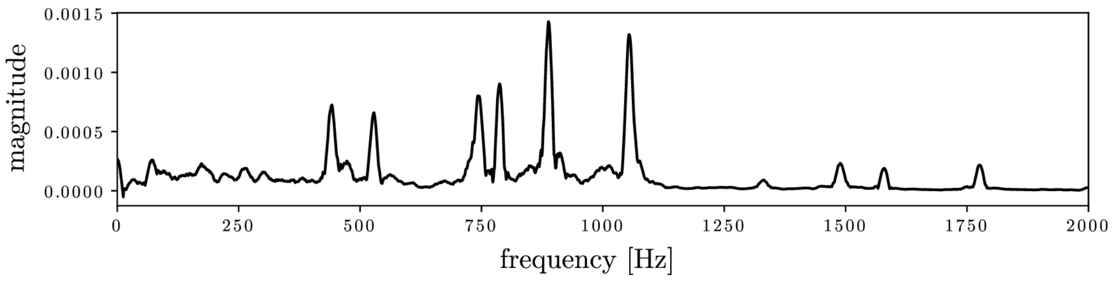


Fig. 1.2: A single side spectrum of a short audio excerpt with several tones and their overtones.

1.3.3 Short-Time Fourier Transform

Fourier transform describes frequencies within a given audio signal. However, we have no information about when these frequencies occur. For instance, two notes played in succession or the same notes played simultaneously would have a very similar spectrum using DFT. The only parameter is the length of the signal in samples. However, we can use only one segment defined by its length in samples, compute the Fourier transform, and then proceed to the next segment and repeat. To retrieve time and frequency information from the signal, we divide it into frames of length $N \in \mathbb{N}$ using a window function $w \in [0 : N-1]$ and calculate DFT on each frame, resulting in discrete Short-Time Fourier Transform (STFT). The discrete time-frequency representation $X(m,k)$ can be defined as:

$$X(m,k) = \sum_{n=0}^{N-1} x(n+mH)w(n)\mathrm{e}^{\frac{-2\pi \mathrm{j} kn}{N}}, \tag{1.5}$$

where $m \in \mathbb{Z}$ is the frame index, $k \in [0 : N/2]$ is the frequency index, and $n \in [0 : N-1]$. $H \in \mathbb{N}$ represents the hop size, hop factor, or window overlap. It defines the number of samples between consecutive frames. Here, we shift the signal in time by the hop factor, but in other definitions, the window is shifted instead (for instance, in [23]). There are more definitions that should be equivalent but vary in the practical implementation. Instead of frequency indexes $X(k)$ as in DFT, we end up with a matrix $X(m,k)$, where all m time positions contain $\lfloor N/2 \rfloor$ frequency indexes. The time coefficients $T(m)$ represent the physical time in seconds, and frequency coefficients $F(k)$ represent the physical frequency in Hertz:

$$T(m) = \frac{m \cdot H}{f_\mathrm{s}}, \tag{1.6}$$

$$F(k) = \frac{k \cdot f_\mathrm{s}}{N}. \tag{1.7}$$

We can use many different window functions (rectangle, triangle, Hamming, Hann, Bartlett, etc.), each with different frequency properties (abrupt changes lead to specific artifacts that propagate in the spectrum) that slightly change the spectral estimate of the STFT. We can consider the window as a Finite Impulse Response (FIR) filter with the corresponding frequency response. The most common window for music processing applications is Hann window $w(x)$ defined as [22]:

$$w(x) = \frac{1+\cos(\pi x)}{2} \text{ if } -0.5 \leq x \leq 0.5, \text{ else } 0. \tag{1.8}$$

It has a smoother shape and no discontinuities compared to, e.g., rectangular window, leading to attenuation of ripple artifacts. However, it also blurs or smears the frequencies in the windowed signal. Note that the window length denotes the time and frequency resolution of STFT. Frequency resolution increases with a longer window while time resolution decreases and vice versa. The notion is called the uncertainty principle of signal analysis, based on Heisenberg's uncertainty principle, and is referred to as a time-frequency trade-off. The length always depends on the application. Denis Gabor combined the Fourier transform with the Gaussian window function and computed the first STFT in 1946 [38]. The Gabor transformation is a special case of Short-Time Fourier Transform with a Gaussian window and laid the foundation for another time-frequency representation that addresses the problem of a static window length through the analysis—wavelet transform. The spectrum of the SFTF is usually described by complex numbers, including magnitude and phase information. The magnitude information corresponds to the absolute value of $X(m,k)$ and the phase information from the argument of $\mathrm{j}X(m,k)$ as follows:

$$X(m,k) = |X(m,k)|\mathrm{e}^{\mathrm{j}X(m,k)}. \tag{1.9}$$

The visual representation of $|X(m,k)|$ is called a magnitude spectrogram. Most music-related information is usually contained in the lower bands of the spectrum, so in many applications, a logarithmic compression is applied. Furthermore, the human perception of intensity and frequency is almost logarithmic. It makes sense to use the log-frequency spectrogram if the goals of our application support that. In most experiments in this thesis, we use only logarithmic magnitude spectrograms, discarding the phase information.

1.3.4 Chroma features

We need a different time-frequency representation to analyze the harmony and melodic parts of music representations. We leverage the fact that two pitches are perceived as similar in their color (containing similar higher frequencies or overtones) when they differ in an integer multiple of their frequencies. For instance, a pitch of 440 Hz (A4) is perceived as similar to 880 Hz (A5), which is one octave higher, doubling the frequency. We can separate the pitch into two components: tone height and chroma. Tone height represents the octave number (such as 4 in A4) and chroma describes the tone class as in international Western music notation based on equal temperament, leaving out enharmonic equivalents and creating a vector of 12 values [C, C♯, D, D♯, E, F, F♯, G, G♯, A, A♯, B]. The main idea of chroma features is to combine all spectral information related to a given tone or pitch class into a single chroma coefficient [22]. It is possible to derive chroma features from a pitch-based log-frequency spectrogram, which has specifically crafted bandwidths

BW(p) for standard MIDI pitches $p \in [0 : 127]$:

$$\mathrm{BW}(p) = F_{\mathrm{pitch}}(p + 0.5) - F_{\mathrm{pitch}}(p - 0.5), \tag{1.10}$$

where $F_{\mathrm{pitch}}(p)$ corresponds to the center frequency of a pitch p:

$$F_{\mathrm{pitch}}(p) = 2^{\frac{p-69}{12}} \cdot 440. \tag{1.11}$$

The chroma features $C(m, c)$ can be defined as:

$$C(m, c) = \sum_{p \bmod 12 = c} X_{\mathrm{LF}}(m, k), \tag{1.12}$$

where X_{LF} stands for the log-frequency spectrogram. There are, however, many ways how to compute the chroma representation, including filter banks, Constant-Q Transform (CQT), or a deep neural network-based approach called DeepChroma [39]. The CQT approach usually gives better resolution in lower frequency bands than STFT but is more computationally expensive. DeepChroma is trained on a specific set of recordings and musical genres. For most experiments in this thesis, we used the Chroma Energy Normalized Statistics (CENS) feature, a normalized and smoothed variant of STFT chroma features [40]. To understand the similarity of chroma features and a MIDI transcription (visualized as a piano roll) from a real audio recording, we show the comparison in Figure 1.3. Furthermore, Figure 1.4 shows the comparison of three chroma representations of a piano recording (20 s excerpt): STFT chroma, CENS, and DeepChroma. We can observe the downsampling and smoothing of the features in the CENS and DeepChroma approaches. This usually leads to better synchronization robustness while providing sufficient temporal resolution. The darker the area, the more that tone or its corresponding frequencies are present in the signal.

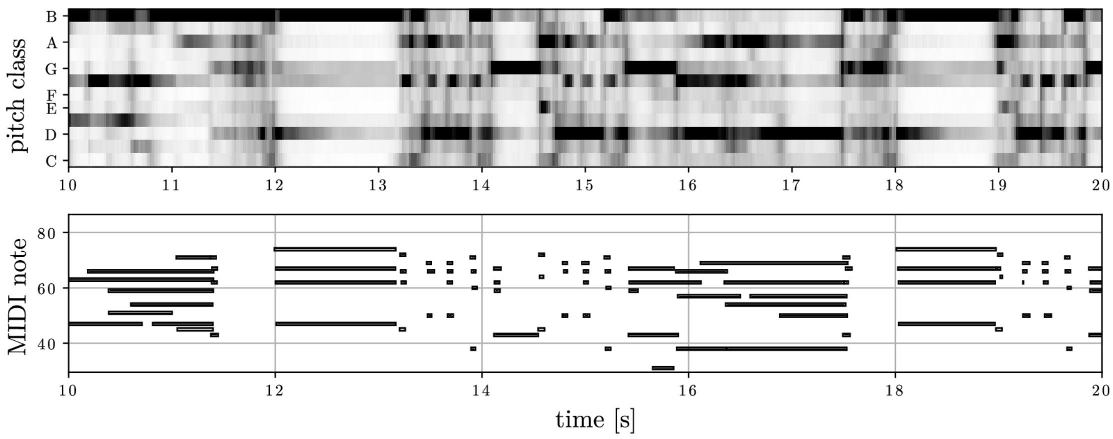


Fig. 1.3: The CQT chroma features and a piano roll visualization of a piano recording. From top to bottom: CQT chroma and corresponding piano roll.

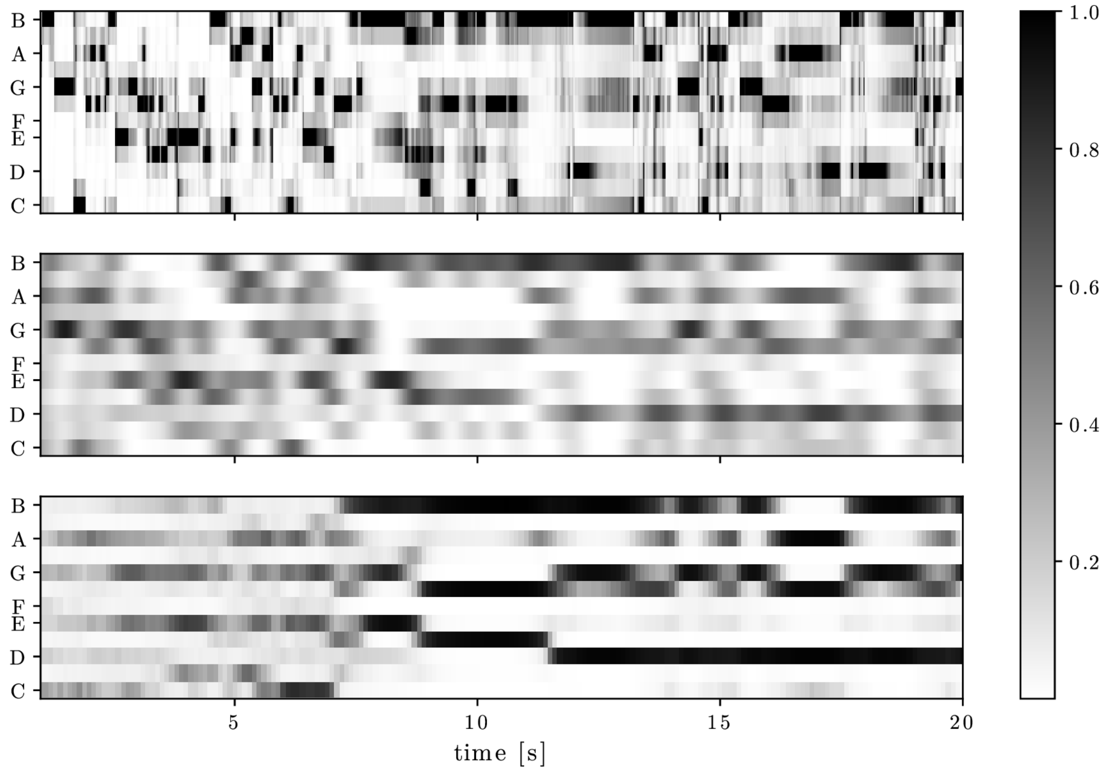


Fig. 1.4: Three chroma time-frequency representations of a piano recording excerpt. From top to bottom: STFT chroma, CENS, and DeepChroma.

1.4 Synchronization

As mentioned before, we can represent digital audio material as a waveform, spectrum based on DFT, or a time-frequency representation such as spectrograms based on STFT, CQT, wavelets, or chroma vectors. However, music is multimodal, meaning we have many ways to represent the musical information—either a symbolic score or an actual physical rendering. For example, we can represent a piece of sheet music or a score as an image (a single matrix consisting of rows and columns, values 0–255 represent the greyscale color of pixels) or transform the audio recording to a magnitude spectrogram and again to the image. Furthermore, we can modify the note timings of a piece's score to correspond to the recorded performance and render it as a symbolic MIDI file.

In the experiments reported in this thesis, we often use datasets containing multiple performances (interpretations) of the same musical piece. We usually assume that the performers play the piece using the same score material. The harmonic and melodic structure should be the same for all performances. Ideally, performances should vary only in performance parameters such as local and global tempo (position of tones, beats, and measures in time), interpretation style (expressivity, legato, staccato, vibrato, tremolo, etc.), dynamics, and timbre. However, the underlying harmonic and melodic progressions and changes should not differ. Even though this assumption may often be wrong, as performers can make mistakes (often during live versions), use different scores, or play/skip repetitions, we can leverage the music synchronization technique to determine the corresponding time positions of two music representations. Figure 1.5 shows the synchronization idea using sheet music, chroma vectors, and corresponding audio recording. If we compute chroma features from the digital audio, we can compare them with the binary chroma of sheet music to obtain corresponding time positions of both representations (score-to-audio synchronization).

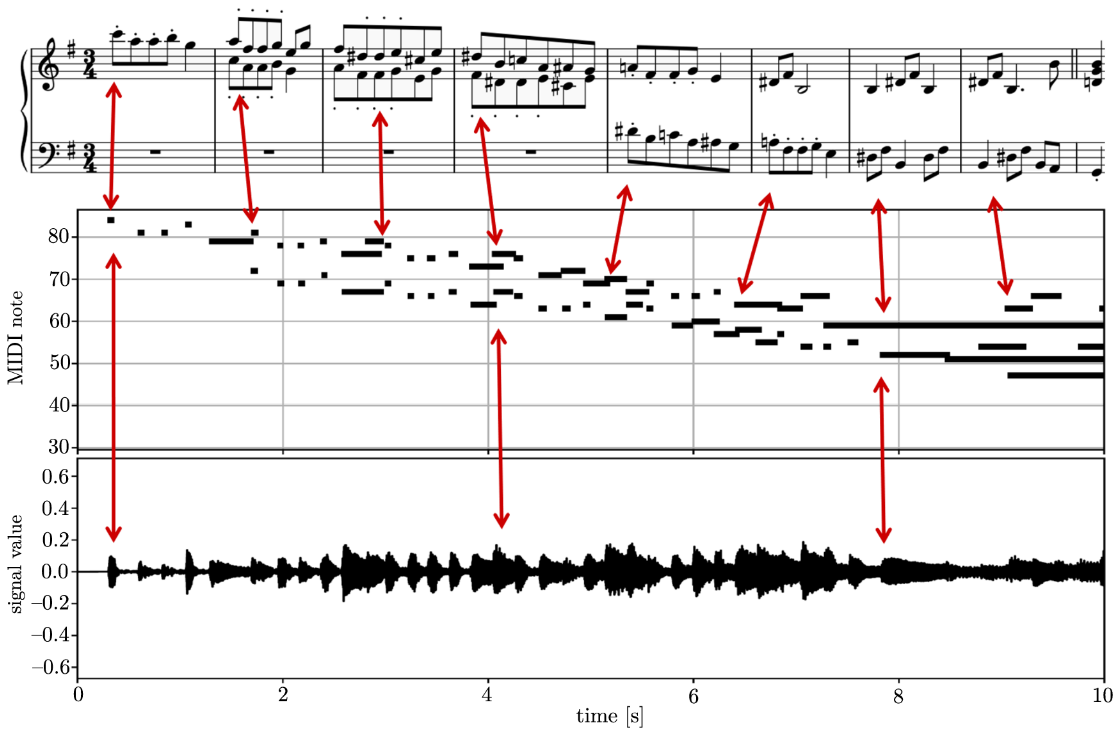


Fig. 1.5: The example of the synchronization idea with corresponding time stamps (red arrows): sheet music, MIDI piano roll, and waveform representations of the same audio excerpt.

1.4.1 Dynamic Time Warping

The main idea of music synchronization is to find time positions in one representation (reference) and the corresponding time positions in the second representation (target). First, one has to compute suitable features for each music representation, such as chroma features, and then deploy a synchronization technique. The most common synchronization method in MIR is called Dynamic Time Warping (DTW). We can use chroma vectors from the reference recording and synchronize them with chroma vectors from the target audio recording (audio-to-audio synchronization) [41, 42]. Similarly, the reference could be MIDI (symbolic score-to-audio) [43] or sheet music (sheet music-to-audio) [44].

The DTW compares two sequences $X = (x_1, \ldots, x_N)$ and $Y = (y_1, \ldots, y_M)$ with $N \in \mathbb{N}$ and $M \in \mathbb{N}$. In our scenario, X and Y feature sequences correspond to the chroma vectors of reference and target recordings, respectively. Both sequences usually vary in length N and M (tempo), but their chroma vectors should be similar. First, we assume that $x_n, y_m \in \mathcal{F}$ for $n \in [1 : N]$ and $m \in [1 : M]$, where $\mathcal{F}$ is a feature space. Then we define compute local cost measure function c as $c : \mathcal{F} \times \mathcal{F} \to \mathbb{R}$.

If $c(x, y)$ is small (small cost), both input features are similar; otherwise, they are different. Computing local cost measure on all pairs of X and Y, we obtain cost matrix $\mathbf{C} \in \mathbb{R}^{N \times M}$:

$$\mathbf{C}(n, m) = c(x_n, m_y), \tag{1.13}$$

where $n \in [1 : N]$ and $m \in [1 : M]$. Then, we compute distance using cosine distance for nonzero values of x and y:

$$c(x, y) = 1 - \frac{\langle x|y \rangle}{||x|| \cdot ||y||}. \tag{1.14}$$

If x or y is zero, $c(x, y) = 0$. If x and y are orthogonal, $c(x, y) = 1$. The cosine distance does not depend on the length of input sequences—only the energy distribution across all twelve chroma pitch classes is considered, potentially leaving out the dynamics or timbre elements of input recordings. Next, we need to obtain an optimal alignment path called the warping path. In a standard DTW, the warping path $P = (p_1, \ldots, p_T)$ follows three conditions: boundary conditions (the path starts at the (1, 1) position and ends in (N, M) position; monotonicity condition (the n and m are always the same or increasing, never decreasing), and step size (no x or y can be omitted and there are no duplicates in the alignment path). Next, we can define path cost $c_P(X, Y)$ of a warping path P:

$$c_P(X, Y) = \sum_{t=1}^{T} \mathbf{C}(n_t, m_t). \tag{1.15}$$

The goal is to minimize the cost $\mathrm{DTW}(X, Y)$ of the optimal warping path P_o:

$$\mathrm{DTW}(X, Y) = \min \left(c_{P_o}(X, Y) \right), \tag{1.16}$$

which is usually done using dynamic programming. The idea is to segment a given problem into smaller subproblems and, by solving and concatenating small subproblems, solve the original problem. In the DTW case, we can derive a global warping path by solving smaller subsegments of feature sequences X and Y. First, the accumulated cost matrix $\mathbf{D}$ is computed [45]:

$$\mathbf{D}(n, m) = \mathbf{C}(n, m) + \min \mathbf{D}(n - i, m - j) \tag{1.17}$$

with $\mathbf{D}(n, 1) = \sum_{k=1}^{n} \mathbf{C}(k, 1)$ for $n \in [1 : N]$ and $\mathbf{D}(1, m) = \sum_{k=1}^{m} \mathbf{C}(1, k)$ for $m \in [1 : M]$ using backtracking. The complexity of the algorithm is $\mathcal{O}(NM)$. We refer to [22, 46] or a recent Ph.D. thesis [47] for an extensive description of the DTW algorithm. Figure 1.6 shows the example of standard DTW synchronization on two sets of chroma vectors. There are, however, other approaches to compute the time alignment, such as Hidden Markov Models (HMMs) and particle filters [43, 44].

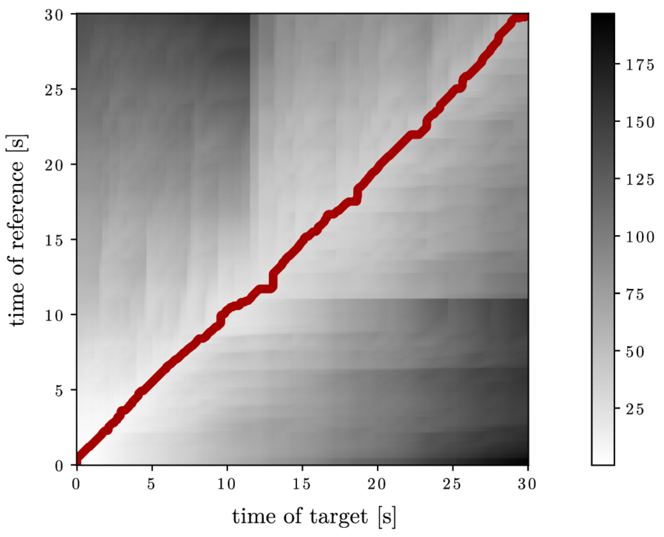


Fig. 1.6: The example of synchronization of two audio recordings using chroma vectors and DTW.

The DTW can be further modified to decrease the computational cost or memory consumption. For example, one can change the step size conditions, adjust local weights, and deploy global constraints or multiscale methods. One way to decrease the memory requirements of DTW is to use the Sakoe-Chiba band or Itakura parallelogram, the constant global constraint regions that were introduced in [48] and [49], respectively. Their comparison is given, e.g., in [50]. Instead of constant constraints, one can use adaptive global constraints such as multiscale DTW (MsDTW) [41] and a variant called FastDTW [51]. Multiscale means that the potentially non-optimal alignment is computed first on the coarse resolution, projected onto a finer feature resolution level and refined using a tubular constraint region [45]. The alignment computation can be divided into two problems: online and offline alignments. In the online variant, we do not know the data in advance, and the alignment is usually computed by greedy forward path estimation [42] or by block-by-block processing methods [52]. Offline approaches can use backtracking as all data is known prior. In recent years, Cuturi et al. introduced the SoftDTW [53], which makes the DTW method differentiable. It was further used in [54] to stabilize pitch class estimation training with weakly aligned targets. Furthermore, Bükey et al. presented FlexDTW [55] with flexible boundary conditions to the alignment, dealing with some limitations of standard DTW approaches, where the warping path starts and ends in diagonal points of the similarity matrix.

In our experiments, we used memory-restricted multiscale dynamic time warping (MrMsDTW) [45] for its efficiency and availability via synctoolbox [56]. This method builds the global warping path by concatenating smaller local alignments, thus restricting memory usage. It utilizes local rectangular constraint regions in the refinement step with a defined size using anchor points, which makes the required memory dependent only on a restriction parameter τ. The memory requirement is constant instead of linear in MsDTW or quadratic in the case of standard DTW. The example of MrMsDTW from synctoolbox [56] applied to two interpretations of the same piano piece is shown in Figure 1.7. Note that the cost matrix values differ substantially from a standard DTW, in comparison with the colorbar of Figure 1.6, due to the path's boundary conditions that reduce the area of computation and the overall length of the example (30 s in both cases).

I attended an internship in Audio Labs, Erlangen, Germany, under the supervision of Prof. Meinard Müller, resulting in the collaboration and the paper at the ISMIR conference called "Using activation functions for improving measure-level audio synchronization" [14]. The International Audio Laboratories Erlangen is a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer Institut für Integrierte Schaltungen IIS. We combined multiple activation functions from neural networks (beat detector, downbeat detector, and several onset detectors) with chroma features and evaluated them based on measure transfer accuracy. We utilized combined cost matrix $\mathbf{C}_{\texttt{CH+ACT}}$ consisting of weighted chroma cost matrix $\mathbf{C}_{\texttt{CH}}$ with cosine distance and the beat, downbeat, and onset cost matrix $\mathbf{C}_{\texttt{ACT}}$ with Euclidean distance as follows:

$$\mathbf{C}_{\texttt{CH+ACT}} = \alpha \mathbf{C}_{\texttt{CH}} + (1-\alpha)\mathbf{C}_{\texttt{ACT}}, \tag{1.18}$$

where $\alpha \in [0, 1]$ is a weighting parameter. The sum $\mathbf{C}_{\texttt{CH+ACT}}$ accounts for both harmonic or melodic information of the $\mathbf{C}_{\texttt{CHROMA}}$ and additional temporal cues via $\mathbf{C}_{\texttt{ACT}}$. We evaluated the synchronization accuracy of multiple neural networks and

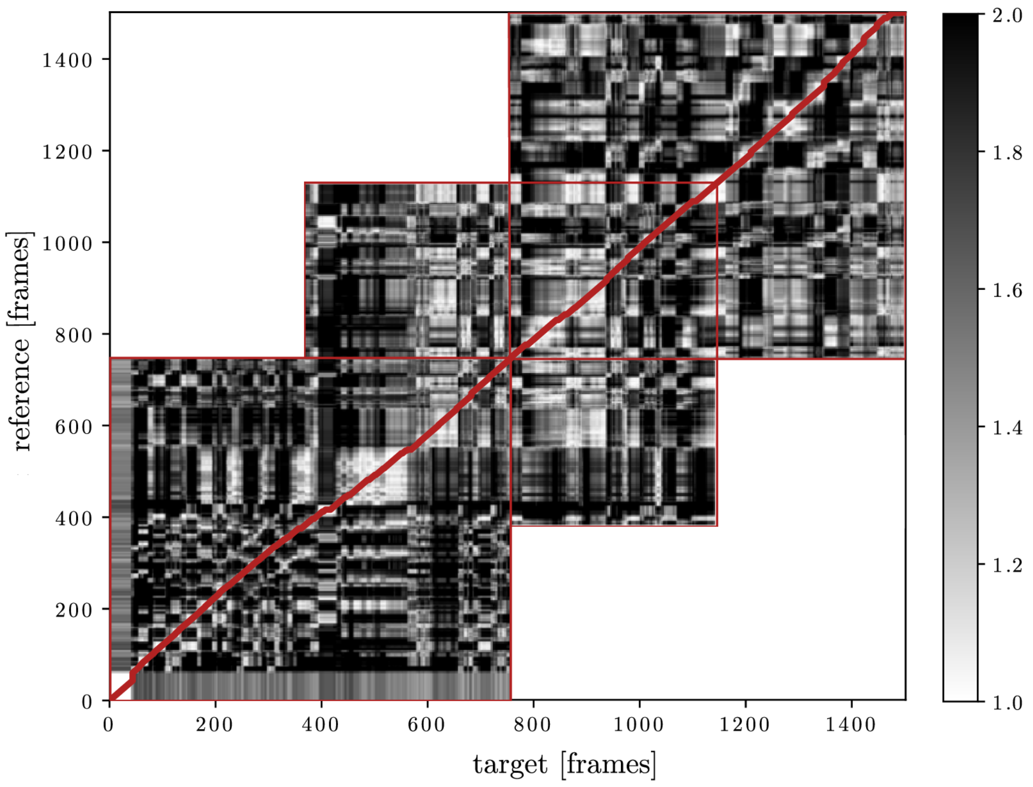


Fig. 1.7: The example of synchronization of two audio recordings using chroma vectors and MrMsDTW. The temporal resolution of chroma features is 50 fps.

MrMsDTW using several window settings (see Figure 1.8). The accuracy denotes the proportion of correctly transferred measure positions having an error below a given tolerance τ. Furthermore, in Chapters 9 and D (based on articles [6], [7] and the MemoVision software), we used a combined synchronization approach using our beat tracking model trained specifically for the synchronization pipeline.

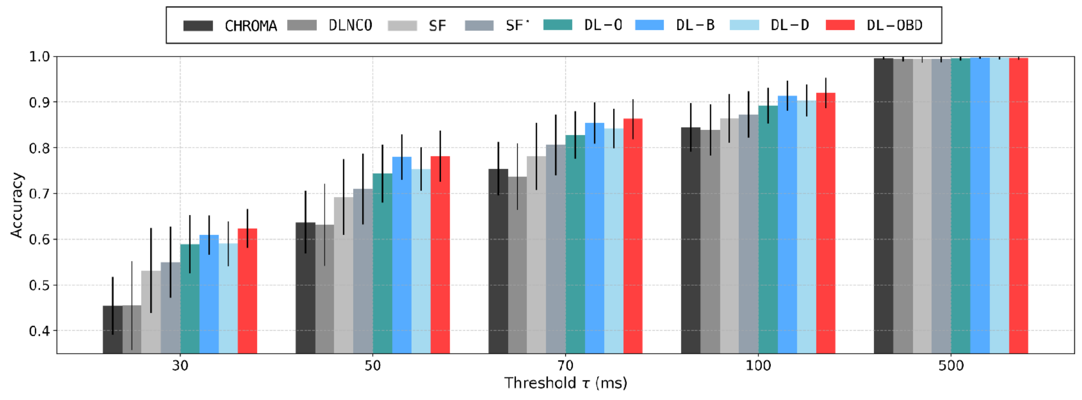


Fig. 1.8: Comparison of the average accuracy values for different synchronization approaches and different threshold parameters τ. The figure is taken from [14].

Author’s Contributions

Matěj Ištvánek (M.I.) substantially contributed to the data acquisition, implementation, testing, and evaluation of proposed experiments, training and evaluation of machine learning models, and the writing of all first-author articles [1, 2, 5, 8, 9, 10, 11, 6, 7, 16]. He presented a conference paper at the TSP 2019 conference [4] in Budapest and at the TSP 2022 [10] online, four publications at the EEICT student conference [3, 8, 9, 5] in Brno, and a late-breaking article [7] including a poster at ISMIR 2023 conference, Milan, Italy. Furthermore, he contributed to the data acquisition, visualizations, and writing of [15].

Štěpán Miklánek contributed to the implementation, testing, and metrics evaluation in [9] and training of NN models and writing of [5]. Furthermore, he contributed to the implementation of feature matrices and parameter detection in [10], the design of experiments, data acquisition and processing, implementation and writing of [11], and implementation, data processing, and writing of [6, 7]. He also presented the article [6] at the satellite event Multilayer Music Representation and Processing (MMRP) at the 4th International Symposium on the Internet of Sounds (IoS) in Pisa, Italy, and maintains the MemoVision Software [21].

Lubomír Spurný contributed to the writing and supervision of [2, 11, 7, 15, 16]. Klára Hedvika Mühlová contributed to the data acquisition and design of experiments of [6] and writing of [16]. Jiří Mekyska contributed to the statistical evaluation and design of experiments in [2]. Martin Mejzr contributed to the data acquisition and writing of [16]. Finally, Zdeněk Smékal supervised and contributed to the signal processing parts in [2] and [6] and supervised this Ph.D. thesis.

M.I. was in an internship in Audio Labs, Germany, in March 2022 under the supervision of Prof. Meinard Müller, which resulted in an article on the ISMIR 2022 conference [14]. M.I., as a second author, contributed to the string quartet data acquisition, conducted some parts of the experiments and method evaluations, and helped with the writing of the article. Yigitcan Özer substantially contributed to the design of experiments, implementation, evaluation, and writing. Vlora Arifi-Müller contributed to the data preparation and annotation and Meinard Müller supervised the article and contributed to the writing.

Part I: MPA-based Evaluation of Low-level and Mid-level Detectors

2 Audio Degradation for Onset Detection

This chapter is based on the journal article “The Effect of Audio Degradation on Onset Detection” [1] and introduces experiments with the degradation of input audio for the onset detection task in MIR.

Although many articles in the field of Music Information Retrieval have been introduced to improve onset detection systems, only the bare minimum focus on the degradation of input audio to increase detection accuracy. This article evaluates the accuracy of five onset detectors, including state-of-the-art machine and non-machine learning-based systems, and compares the influence of various types of audio signal degradation on musical onset detection. We used three different degradations based on impulse responses, a Teager–Kaiser energy operator, and two MP3 compression settings. The results suggest that if MP3 compression of any settings is applied, the accuracy of detection systems is very similar. Using the energy operator as degradation has not improved overall detection but may offer the potential of pre-processing the neural network input signal for easier identification of onsets in a training phase. Furthermore, radio broadcast degradation increases the number of all predicted onsets in general, both true and false positives, resulting in better recall but worse precision. This information could be used to modify the pre-processing phase of neural network-based detectors and to optimize the sensitivity trade-off.

2.1 Introduction

In the Music Information Retrieval (MIR), onsets are common low-level parameters. An onset is the beginning of any musical tone—it refers to the starting time point of the produced sound or note. An onset detection function (ODF) is the output of onset detectors and represents the probability of onset occurrence in a given time. Its peaks (local maxima) should correspond to the onset time positions and, thus, ideally, all tones in an audio recording. An onset detection system is usually divided into a few parts: pre-processing of the input signal, ODF, and peak-picking—finding onset positions from the onset curve [57]. Concerning the non-machine learning systems, there have been many works on developing and optimizing better ODFs [58, 59, 60], but very few on reducing the raw audio information. The exception is The Audio Degradation Toolbox [61], where the authors used audio degradations for different MIR tasks, such as beat tracking or score-to-audio alignment. Another musical data augmentation and degradation tool is the MUDA package [62]. Both studies present options for degradation to reduce or change information in an audio file to improve a particular MIR task. Other articles [63, 64] focus on audio or music identification. Furthermore, different degradations were used to study the robustness of cover song recognition [65]. Generally, a common approach is using the 16-bit (PCM) audio files in a *.wav* format with a 44.1 kHz or 22.05 kHz sample rate. However, this does not mean that audio degradation cannot positively affect the onset detection or the final accuracy.

Methods of audio onset detection are used in beat tracking, rhythm, and metric detection, Automatic Music Transcription (AMT), and many other high-level feature extraction tasks. As a basic low-level feature, onsets are the key parameter for performance analysis in musicology research. Therefore, achieving high accuracy and consistency in these systems is important. Common detectors are based on spectral, phase, or complex domain and supervised machine learning algorithms such as artificial neural networks. There are other options (such as pitch detection [66] or non-negative matrix factorization [67]), but they are not widely used for this task. To this date, the state-of-the-art detectors are using Convolutional Neural Networks (CNN) [68, 69, 70] or Long Short-Term Memory (LSTM) networks [71, 72], but some robust spectral methods such as SuperFlux [73] are still providing quite comparable results in the onset detection task (e.g., onset detection in Music Information Retrieval Evaluation eXchange (MIREX) 2018 evaluation [74]). In this study, we test six different degradations (and the original dataset with *.wav* files) and five different well-known offline detectors on the large *onsets_ISMIR_2012* dataset, introduced in [75], with a total of 321 recordings.

There is a distinction between audio degradation and pre-processing of an input signal in the detection systems. Audio degradation is a process of specific manipulation of input audio. Then, the system reacts to it (different spectral elements, activation function, final detection accuracy, etc.). On the other hand, pre-processing is one of the first parts of detection systems, e.g., separation to multiple frequency bands to modify the time-frequency representation [57], reducing information, dimension, or methods like filtering, adaptive whitening [75, 76], and noise suppression [77]. The degradation becomes the pre-processing step when it is implemented into the system. However, degradation is generally considered to be signal deterioration with negative effects on the detector's accuracy. This study suggests that it may not be the case for all music categories and onset detectors.

In this article, we test and evaluate different types of degradation of an input audio signal to achieve better onset detection results of both state-of-the-art machine and non-machine learning detectors. We test impulse response degradations and lossy compressions and propose the Teager–Kaiser Energy Operator (TKEO) as one of the degradations that, to our knowledge, has not been used for this MIR task yet. The rest of the article is organized as follows: Section 2.2 describes a method of onset annotation, audio degradations, and onset detectors. Section 2.3 introduces the dataset, segmentation of categories, and evaluation technique. The results are presented in Section 2.4. Section 2.5 discusses the analysis and evaluation results. Finally, the conclusion and plans for future work are given in Section 2.6.

2.2 Methods

First, we introduce the onset annotation problem and all the degradation methods we use. Then, a brief description of selected detection systems, chosen datasets, and evaluation techniques are given. Python 3.7.6 was used for all tests and types of degradations except the Audio Degradation Toolbox (MATLAB 2015a).

2.2.1 Onset Annotation

Onset is usually defined as the starting point when a note is being played or a tone is created. However, perceptual onset (when a listener hears the onset) can differ from the very beginning of a physical tone. Problems tend to occur in polyphonic structures due to difficult estimation of the correct onset positions caused by inaccuracies. The definition we use may differ from human perception, yet this method of labeling onsets is the most commonly used. Hence, there is no "clear" method of dealing with annotations, although recommendations do exist [78]. In this article, we use a dataset with merged onsets as in [69] to compensate for this phenomenon.

2.2.2 Impulse Response Degradation

For the impulse response degradation, Audio Degradation Toolbox (ADT) [61] was used. This MATLAB toolbox consists of a code for creating and handling the degradation of audio signals, including ground-truth annotation of the degraded audio. There are plenty of available degradations—we used radioBroadcast, smartPhonePlayback, and smartPhoneRecording options to simulate real-world scenarios [61]:

- **Radio Broadcast**: two degradation units:
  - dynamic range compression at a medium level to emulate the high loudness characteristic of many radio stations,
  - speed-up by 2% to shorten the music and create more advertisement time.
- **Smartphone Playback**: two degradation units simulating a user playing back audio on a phone:
  - impulse response of a smartphone speaker (Google Nexus One); highpass characteristic and a cutoff at 500 Hz,
  - additional light pink noise.

- **Smartphone Recording**: four degradation units, simulating a user holding a smartphone in front of a speaker:
  - impulse response of a smartphone microphone (Google Nexus One),
  - dynamic range compression to simulate the smartphone's auto-gain,
  - clipping, 3% of all given samples,
  - additional medium pink noise.

For a more detailed description, see [61]. Reference onsets were shifted according to the methods used. These degradations and MP3 compression could also show how well the proposed detection systems can deal with the different real-world input audio conditions.

2.2.3 MP3

MP3 (from MPEG, Audio Layer III) is a coding format for digital audio. It uses psychoacoustic principles to remove redundant data—*mp3* files take up just 10% of the storage space of the uncompressed lossless original file, depending on the settings [79]. The input audio signal is mapped into 32 subbands with the same bandwidth through a polyphase filterbank (simulation of critical bands in the human auditory system). Then, Modified Discrete Cosine Transform (MDCT) is used on each subband using the long and short windows for different frequency and time resolutions. Other algorithms and principles are applied to decrease the size of a file (e.g., Huffman coding). The MP3 format utilizes lossy compression, and depending on the settings used, it can create noise and specific degradation in the recording. For our evaluation, we tested the constant bit rate (CBR) settings of 64 kbps and 320 kbps using FFmpeg (libavcodec[1]), which simulates the audio quality of common online streaming portals and internet content. All files were then converted back to *.wav* format (while keeping the audio degradation).

2.2.4 Teager–Kaiser Energy Operator

The TKEO as the degradation unit is inspired by our previous work [2], where we studied the effect of TKEO on a conventional non-machine learning beat tracking method. In that method, the final beat positions depend on the occurrence of onsets—their reduction and the emphasis on onsets that achieve higher energy might be beneficial for the beat detector.

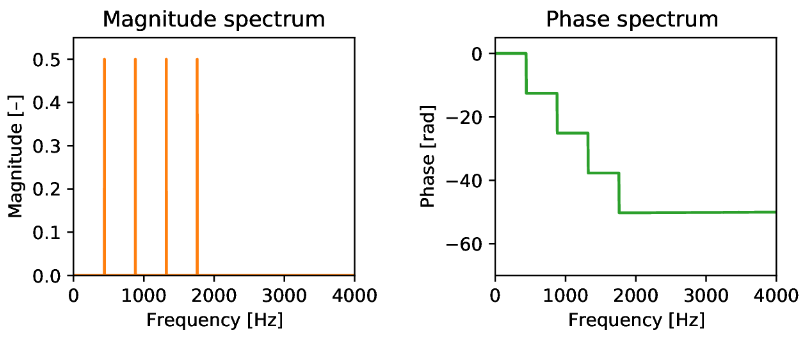
The TKEO is a non-linear time-invariant operator that includes both amplitude and frequency of an input signal [80]. Furthermore, TKEO for the lower frequency

[1] `https://ffmpeg.org/libavcodec.html`

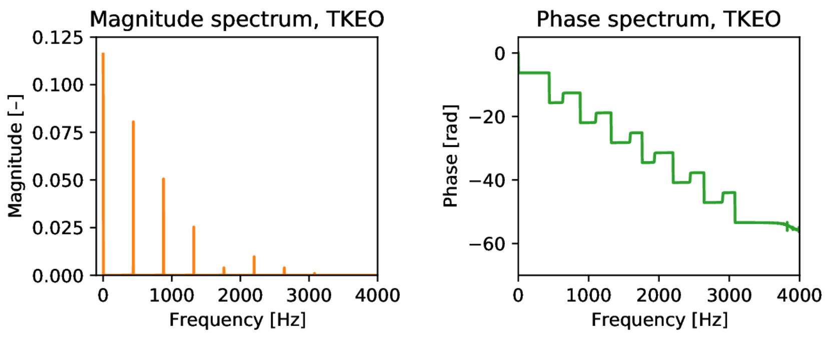
has a smaller value than for the higher with the same amplitude [81]. In the discrete version, the TKEO $\Psi(x[n])$ of a discrete-time signal $x(n)$ is defined as:

$$\Psi(x(n)) = x^2(n) - x(n-1) \cdot x(n+1). \tag{2.1}$$

Figure 2.1a shows the magnitude and phase spectrum of a tone composed of four pure frequencies (cosine waves) that start at 440 Hz, and each additional one is the next overtone. The amplitude of each frequency is 1. Then, TKEO was applied to the same four frequencies. The result is shown in Figure 2.1b:



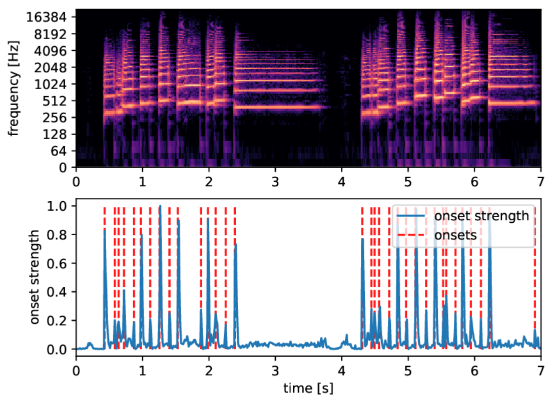
(a) Original synthetic tone consisting of four frequencies.



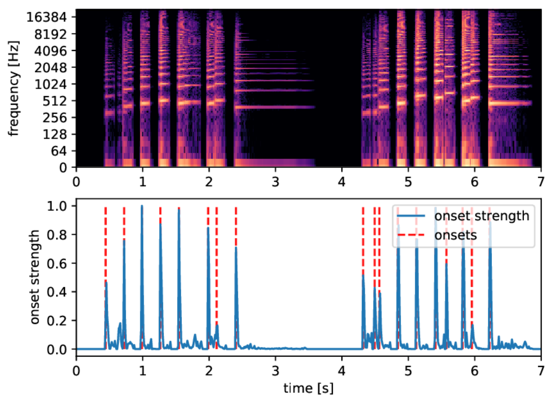
(b) The synthetic tone after TKEO degradation.

Fig. 2.1: A tone made of 440, 880, 1320, and 1760 Hz. Each frequency has an amplitude of 1. a) Before and b) after applying TKEO degradation.

We can see seven harmonics and a DC component after the modulation. A similar change is observed when applying TKEO on the actual music track. New overtones are created, and the energy of spectra is shifted towards higher frequencies, as shown in Figures 2.2a and 2.2b. These figures represent the difference between a spectrogram and the ODF of an original *.wav* file using *librosa* package and the same excerpt degraded by TKEO. Note that we also used normalization to compensate for low signal values when TKEO was applied.



(a) Original audio file: log-spectrogram, ODF, and detected onsets.



(b) Original file after TKEO degradation: log-spectrogram, ODF, and detected onsets.

Fig. 2.2: Log-spectrogram and the ODF with estimated onsets of an audio excerpt (solo trumpet). a) Before and b) after TKEO degradation.

The TKEO separated onsets, and the ODF is cleaner. Onsets gain clarity, and they are more easily distinguishable. Although this seems to be a big improvement in the detection function, it is not true for many other audio excerpts. The TKEO tracks the energy evolution, but when a peak of energy does not correspond to the onset time position, problems can arise (see Figure 2.3). A standard 50 ms evaluation window should cover the delay (see Section 2.3.2). The ADT calculates new onset times (from the reference onsets) for each degradation to fit the degraded audio excerpt.

The original audio signal in the time domain and the same excerpt using TKEO are shown in Figure 2.3. The green line indicates the onset position. We can see that TKEO suppressed the low-energy end of the previous tone (before the new onset occurs) and changed the structure of the audio signal. Furthermore, Figure 2.4 shows the difference in spectral components between all selected types of degradation (excluding the MP3 320 kbps version and the original *wav* file, which are very similar to the MP3 64 kbps).

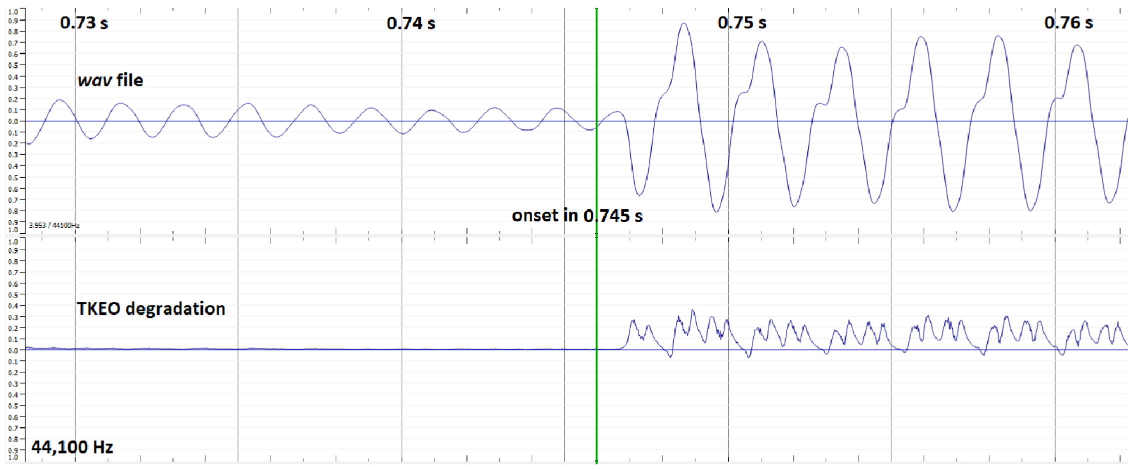


Fig. 2.3: An audio segment with an annotated ground-truth onset: before and after TKEO degradation. The green vertical line shows the onset time position.

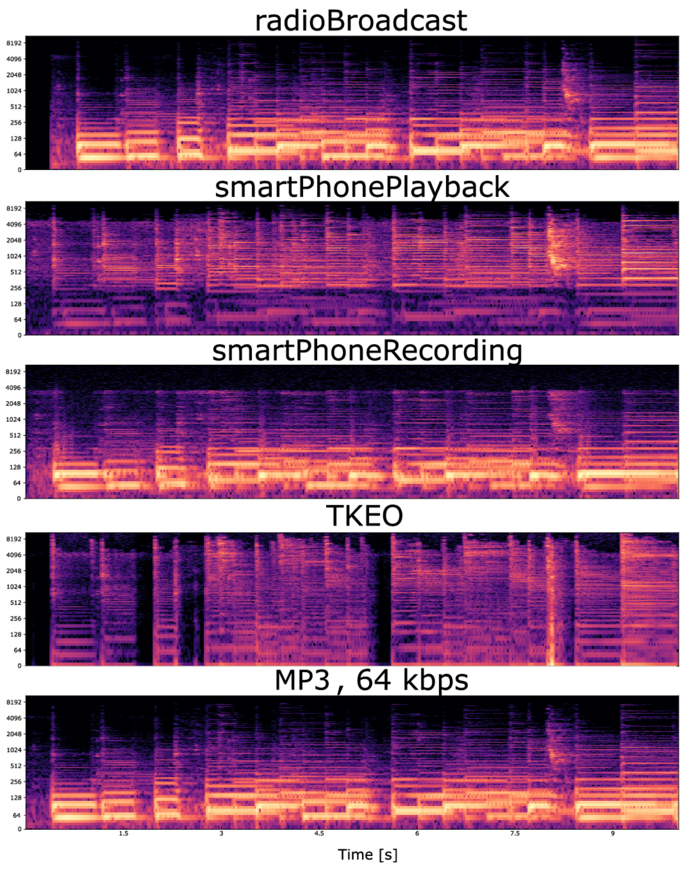


Fig. 2.4: Spectrograms of an audio excerpt, different degradations – radioBroadcast, smartPhonePlayback, smartPhoneRecording, TKEO, and MP3 (64 kbps).

2.2.5 Detectors

We selected five detectors for the evaluation. All systems use 44.1 kHz sample rate audio signal as their input. They are available via *librosa*[2] and *madmom*[3] modules [82]. We chose the state-of-the-art [74] machine and non-machine learning detectors:

- **Lib** – Librosa,
- **SF** – SuperFlux,
- **CF** – ComplexFlux,
- **CNN** – Convolutional Neural Network,
- **RNN** – Recurrent Neural Network.

Lib: This system utilizes spectral flux computation (detection of positive changes in the overall energy of a spectrum over time). We used parameters inspired by [73]. Instead of 22.05 kHz (*librosa* default), we use the 44.1 kHz sampling rate. Next, we chose the length of FFT = 2048, hop size 512 samples, and conversion to Mel spectrogram with 138 bands and frequency range of 27.5 Hz to 16 kHz.

SF: SuperFlux is a special extension of the standard spectral flux algorithm. The detector uses logarithmic frequency scale representation with quarter-tone spacing (again 138 bands), a frame rate of 200 fps for better temporal resolution, and contrary to the spectral flux, it includes a special-trajectory tracking stage for vibrato suppression [73, 83].

CF: This detector uses the core of the **SF** system [73] but introduces Local Group Delay (LGD) based difference weighting. The LGD gives information as to where the "gravitational" center of the magnitude in a spectrum is located. Combining the magnitude and phase information (complex domain) helps to avoid problems with loudness variations of steady tones, thus increasing the potential detection accuracy when tremolo is present [83].

CNN: This system is based on a CNN. First, the input audio stream is converted into 3 magnitude spectrograms with a hop size of 10 ms and windows of 23, 46, and 93 ms. Then, a logarithmic Mel filtering (80 bands, frequency range: 27.5 Hz to 16 kHz) is used and each frequency band is normalized to zero mean and unit variance. From the 3-channel spectrogram (15 frames by 80 bands), a convolution layer with filters of 7 frames by 3 bands (tanh unit) computes 10 feature maps. The next layer performs max-pooling, reducing the maps to 26 bands, and another convolutional layer of 3×3 filters (tanh unit) is used, followed by a max-pooling

[2]librosa python module, version 0.8.0 – DOI: 10.5281/zenodo.3955228, `https://github.com/librosa/librosa/tree/0.8.0`

[3]madmom python module, version 0.17.dev0, `https://madmom.readthedocs.io/en/latest/modules/features/onsets.html`

layer and a fully-connected layer (logistic sigmoid) of 256 units. The single output neuron also uses logistic sigmoid and predicts onsets [68, 69].

RNN: This system utilizes a bidirectional LSTM network (BLSTM) to incorporate a broader time context. However, it is referred to simply as RNN in the *madmom* documentation, so we keep the abbreviation. The input audio stream is transformed to the time-frequency domain via two parallel STFTs with different window sizes (1024 and 2048 samples). Then, conversion to the Mel spectrogram with 40 triangular filters is done, and spectral flux is calculated. The neural network has 3 hidden layers with 20 LSTM cells for each direction [72]. Contrary to the cited source, *madmom* implementation uses simple tanh units in the output neuron[3].

Finally, Table 2.1 shows the F-score for the first five state-of-the-art onset detectors in MIREX evaluation [70]. Note that the table keeps the acronyms that MIREX originally used. Our chosen detectors are partially from this list but may differ slightly in implementation (included in Section 2.2.5 and their documentation).

Tab. 2.1: Results for the first five state-of-the-art detectors evaluated by MIREX competition on a MIREX05 dataset. Bold numbers indicate the highest value for a given metric.

	avg. F-score	avg. Precision	avg. Recall
SB4	**0.873**	0.861	**0.898**
AR3	0.860	**0.889**	0.846
AR4	0.857	0.881	0.849
SB5	0.853	0.834	0.893
SB7	0.840	0.854	0.857

SB4 [68], AR3 [70], AR4 [70], SB5 [72], SB7 [75]

2.3 Dataset and Evaluation

2.3.1 Onsets_ISMIR_2012 Dataset

We used the *onsets_ISMIR_2012* dataset, also referred to as the Böck dataset[4] and introduced in [69], to evaluate the audio degradation effect. The dataset contains 321 audio excerpts taken from [57, 72, 84] and further enhanced by [69]. All onsets within 30 ms were combined, resulting in 25 966 onsets in total, as stated by the

[4]`https://gitlab.cp.jku.at/sebastian/onsets/-/tree/master`

authors. Contrary to the original article, some onsets were corrected by the authors. All files were manually divided into six main categories:

- bowed string (BS) – bowed string instruments (e.g., violin, viola, kemenche),
- complex mixtures (CM) – group of instruments together, complex mixtures and musical genres (e.g., classical, rock, pop, jazz),
- non-pitched percussive (NPP) – percussion without a pitch (e.g., snare and bass drum, cymbals),
- pitched percussive (PP) – instruments that create a sound of a specific pitch using a percussion mechanism (e.g., guitar, tanbur, piano),
- vocal – vocal music,
- wind instruments (WI) – brass and woodwind instruments (e.g., clarinet, sax, trumpet).

Table 2.2 shows the number of files for each category. It presents segmentation by the authors of the dataset and our segmentation. It differs slightly, e.g., we considered choir song (*ff123_duel.wav*) as a vocal (non-monophonic) category. We also classified the song in which the violas and cellos play as bowed string, although it was originally in the complex mixtures category because it is not a monophonic song (*SoundCheck2_80_Instrumental_Cellos_and_violas.wav*). We understand that this segmentation might be questionable because it does not follow strict rules but rather builds on the type of sound (sound texture). However, a few recordings (apart from the very poorly represented vocal category) should not significantly affect the final evaluation.

Tab. 2.2: The number of audio excerpts for the original and proposed segmentation of the *onsets_ISMIR_2012* dataset.

category	orig. seg.	our seg.
bowed strings	23	25
complex mixtures	193	185
non-pitched percussive	17	18
pitched percussive	60	64
vocal	3	4
wind instruments	25	25
Σ	321	321

The dataset was selected considering the number of tracks, open-source policy, and availability of the reference annotation. At the same time, it contains various musical instruments, textures, and different audio quality; it is not specialized in only one type of music or instrument.

2.3.2 Evaluation

First, we evaluated all systems (CF, Lib, CNN, RNN, and SF) on the dataset without any degradation or segmentation. Next, we used degradation methods, thus creating six separate datasets plus the original one (*wav* files): rBcast (radioBroadcast), SPPb (smartphonePlayback), SPRec (smartphoneRecording), 64kb (64 kbps MP3), 320kb (320 kbps MP3), and TKEO (with normalization). In total, there are 2 247 separate files. We labeled all categories in the dataset (Table 2.2) and tested each one separately. Using this method, we can evaluate the effect of each degradation on each system and category. We try to follow the recommendations of open-source practices [85]. The detailed results, including outputs of each system and ground-truth annotation for each degradation, are available on the GitHub repository [17].

When evaluating onset detection accuracy, it is first determined which estimated onsets are correct. The correctness of the estimated onset is defined as being within a small window of a reference onset [78]. The evaluation window indicates the length by which the detected onset is sought out according to the ground-truth onset position. The default parameter for the evaluation window is 50 ms [75]. Each estimated onset is first evaluated:

- TP – True Positives: correctly predicted positive values,
- TN – True Negatives: correctly predicted negative values,
- FP – False Positives: incorrectly predicted positive values,
- FN – False Negatives: incorrectly predicted negative values.

To find out, which degradation statistically provided the best results, we calculated precision, recall, and F-score (F-measure) as follows:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{2.2}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{2.3}$$

$$\text{F-score} = 2 \cdot \frac{\text{recall} \cdot \text{precision}}{\text{recall} + \text{precision}}. \tag{2.4}$$

Precision is defined as the number of all correctly predicted onsets divided by all retrieved onsets. Higher precision means more TPs. Recall is the number of all correctly predicted onsets divided by all ground-truth onsets that should have been predicted. Finally, the F-score is the harmonic mean of recall and precision. A high F-score means a low number of false positives and a low number of false negatives. If the detector detected many onsets, including incorrect ones, and often hit the correct time positions, it could achieve a high recall but, at the same time, a small precision. If both recall and precision are equal to 1, then the F-score is also equal to 1 (100%), and the detector determined all onsets correctly. In practice, it is usually about the precision-recall trade-off (optimizing the recall and precision ratio). Although this is the most widely used metric for evaluating the accuracy of onset detection, the F-score ignores TNs and gives equal importance to both recall and precision.

The evaluation depends on the dataset and method used; however, the default window for the F-score evaluation is used in most cases. As described in Section 2.2, the onset is not a perfectly defined parameter. We decided to test an additional window setting (100 ms) to see whether there is a significant difference in the detection accuracy of each degradation and system for different window settings. Note that we have not multiplied the results by 100, as is often done—this notation can sometimes be clearer with a large amount of numerical data in the tables.

2.4 Results

First, Tables 2.3 and 2.4 show the results of onset detection on the original dataset without any degradation or category segmentation for the 50 ms and 100 ms window, respectively. All metrics are average values of the corresponding data. The CNN provided the best results (F-score 0.889, recall 0.837, and precision 0.969) as expected (MIREX evaluation). The SF had the highest recall for all degradation cases, including the original dataset. A larger evaluation window increased the values of all metrics. However, for neural network-based detectors, this increase was much less pronounced. For instance, differences between both windows for CNN and RNN are from 0.889 to 0.891 and 0.764 to 0.767, respectively.

Tab. 2.3: Overall results for the original dataset; 50 ms window.

metrics	CF	Lib	CNN	RNN	SF
F-score	0.808	0.772	**0.889**	0.764	0.827
recall	0.774	0.800	0.837	0.667	**0.841**
precision	0.878	0.822	**0.969**	0.950	0.845

Then, we evaluated all systems on each category. The resulting F-score for each category of the dataset without any degradation is shown in Tables 2.5 and 2.6. The CNN again outperformed all other detection systems in every category.

Figures A.1 and A.2 in the Appendix show the corresponding box plots, i.e., range of values (boxes cover the 25–75 percentiles and whiskers the 5–95 percentiles), Tab. 2.4: Overall results for the original dataset, 100 ms window.

metrics	CF	Lib	CNN	RNN	SF
F-score	0.820	0.782	**0.891**	0.767	0.837
recall	0.786	0.812	0.839	0.670	**0.850**
precision	0.893	0.831	**0.971**	0.954	0.857

Tab. 2.5: The F-score for all categories of the original dataset; 50 ms window.

det.	BS	CM	NPP	PP	vocal	WI
CF	0.689	0.802	0.954	0.859	0.570	0.768
Lib	0.648	0.777	0.960	0.845	0.416	0.601
CNN	**0.807**	**0.879**	**0.990**	**0.937**	**0.740**	**0.870**
RNN	0.573	0.756	0.963	0.874	0.331	0.653
SF	0.695	0.846	0.942	0.873	0.583	0.657

median, and the average for individual degradations of all detectors together. Note that these plots were created from data of individual categories (Table 2.2) and not from testing on the whole dataset. Using a longer evaluation window results in less variance of the final F-score values, primarily for non-machine learning systems. A larger window means a higher recall and precision, which may not always indicate a better result; at the same time, it can reflect inaccuracies of data annotation.

Next, Table 2.7 shows the F-score, recall, and precision for each degradation—this is the average of all recordings in the database regardless of the category. This way, we wanted to test how degradation can affect the output of detection systems. Differences between F-score on *.wav* files and MP3 compression are again very low, and the final accuracy is similar (except for the 64 kbps MP3 with CNN detector). Considering the RNN detector, radio broadcast degradation improved the overall F-score. Radio degradation also increased recall except for the Lib detector.

Finally, Tables A.1 and A.2 in the Appendix present the F-score for all categories, degradations, and detectors. There is almost no difference between 320 kbps MP3 degradation and the original *wav* dataset. The TKEO degradation achieved the best results for the NPP category using the Lib system and also for the PP category using both CF and Lib systems. In all other cases, TKEO decreased the detectors' performance. Besides, the TKEO generally reported the worst values in the BS, vocal, and WI categories, where an energetic nature of onsets is less pronounced. The 64 kbps MP3 increased the F-score for the PP category in all non-machine learning detectors. Radio broadcast degradation seems to improve the detection

Tab. 2.6: The F-score for all categories of the original dataset; 100 ms window.

det.	BS	CM	NPP	PP	vocal	WI
CF	0.704	0.816	0.956	0.864	0.590	0.792
Lib	0.664	0.787	0.961	0.847	0.443	0.619
CNN	**0.808**	**0.881**	**0.990**	**0.938**	**0.740**	**0.877**
RNN	0.576	0.759	0.963	0.876	0.334	0.664
SF	0.705	0.856	0.944	0.877	0.603	0.684

in some cases, especially for the RNN system. However, the detection accuracy of the WI category for this detector decreased when a 50 ms window was used but increased significantly with a 100 ms evaluation window (from 0.533 to 0.716). The F-score of the CF system increases for the BS category when radio broadcast or smartphone playback degradation is present.

2.5 Discussion

In this study, we tested and evaluated different types of degradation of an input audio signal on state-of-the-art onset detectors. It was confirmed that the difference between the input signal in the form of a *.wav* file (16-bit, 44.1 kHz) and the 320 kbps CBR MP3 codec with the same sampling rate is essentially negligible for any onset detector. When 64 kbps MP3 was used, the statistical accuracy of detection for the pitched percussive category even slightly increased in some cases. An exception is the CNN system, which showed a worse F-score in all cases when the audio signal was degraded. As mentioned in Section 2.2.4, both neural network-based systems were trained on *.wav* files. If we degrade a recording and test a given system, we are actually testing the behavior of the system for that degradation, but there is no way to include this degradation in the pre-processing phase of the detector. We would have to re-train the network, this time on a degraded audio signal. Only then could a valid conclusion be reached as to whether degradation increases or decreases the accuracy of these systems and whether degradation could serve as part of the pre-processing phase of the detector.

This behavior is opposed by the results of the RNN system, where the introduction of radio broadcast degradation led to an increased F-score in three categories (bowed string, complex mixtures, and vocal) but also the whole dataset. The increase in the vocal category was most pronounced (from 0.331 to 0.521 for 50 ms window). However, it should be noted that the resulting number is still too small to be considered a real improvement.

Tab. 2.7: The F-score, recall, and precision for all degradations and detectors – 50 ms window.

F-score							
detector	rBcast	SPPb	SPRec	64kb	320kb	TKEO	wav
CF	0.799	0.794	0.732	0.805	0.807	0.759	**0.808**
Lib	0.635	0.760	0.715	0.770	**0.773**	0.748	0.772
CNN	0.874	0.805	0.804	0.873	**0.889**	0.765	**0.889**
RNN	**0.777**	0.563	0.712	0.757	0.764	0.661	0.764
SF	0.820	0.814	0.744	**0.827**	**0.827**	0.750	**0.827**
recall							
CF	**0.796**	0.754	0.763	0.769	0.774	0.755	0.774
Lib	0.589	0.789	0.725	0.796	0.799	0.771	**0.800**
CNN	**0.840**	0.735	0.769	0.815	0.837	0.712	0.837
RNN	**0.696**	0.455	0.622	0.658	0.667	0.575	0.667
SF	**0.871**	0.829	0.848	0.837	0.840	0.839	0.841
precision							
CF	0.831	0.871	0.742	**0.878**	0.877	0.815	**0.878**
Lib	**0.857**	0.807	0.783	0.823	0.822	0.807	0.822
CNN	0.926	0.935	0.863	0.964	0.968	0.874	**0.969**
RNN	0.920	0.915	0.881	**0.953**	0.950	0.847	0.950
SF	0.796	0.829	0.693	**0.848**	0.845	0.727	0.845

The TKEO showed the worst results from all degradation types except for the Lib system with a 100 ms evaluation window. The energy operator changes the signal significantly, resulting in a bad final detection. On the other hand, visualizations show (Figure 2.2 and 2.4) that onsets in modified spectrograms may sometimes be more visible or clearer—new network-based detectors might be trained on such degraded data to identify proper onset time positions successfully.

Radio degradation increased recall on all systems except for the Lib detector. Evaluation using two windows of different sizes also showed that machine learning-based systems are generally more robust in the correct time detection of onsets—with a larger window, F-score, recall, and precision increase very little. All results, including both evaluation windows, detected onsets from all systems, and the ground truth annotation for all categories, are given in the mentioned GitHub repository [17].

The data indicate that some systems respond better to the degradation of the input signal than others. Experiments show that if radio broadcast degradation is included in the pre-processing phase of the RNN detector, the detection success is, on average, increased. Radio broadcast degradation generally decreased the detectors' precision but increased recall. This means that the detector detected more possible onsets (TP but also FP) and thus achieved higher recall and lower precision. In the case of RNN, this trade-off was good enough to increase the final F-score. Raising the evaluation window can cause an undesirable effect—the evaluation can capture a detected onset within a window that belongs to another correct onset position, creating an error. The vast majority of annotated onsets are more than 100 ms apart; however, many FPs can cause statistical bias. We are aware that 100 ms is the limit value for the evaluation of onset detectors.

If the MP3 codec is set to at least 64 kbps CBR, it shows similar results as the *.wav* file for all tested detectors. Therefore, these systems are also suitable for the onset detection of recordings on streaming portals. However, the quality of audio recordings in the test database varies despite a unified *.wav* format. The 44.1 kHz with 16-bit resolution does not necessarily indicate the same audio quality, as the source material might have already been degraded somehow. This cannot be effectively compensated, especially if a large, freely available database containing various quality audio materials is used. Neural network-based detectors are very often partially trained on degraded audio material. A better understanding of audio degradation could help to find new pre-processing options. Furthermore, unless we create our dataset and ground-truth annotations, we cannot be sure that these systems will be tested on data the detectors have never seen before.

2.6 Conclusion and Future Work

This study presents an evaluation of the audio signal degradation on onset detectors. Five detectors, including machine and non-machine learning state-of-the-art systems, were selected for experiments. The chosen dataset contains 321 different recordings of various musical genres and instruments and was divided into several categories. Six different degradations were applied to create separate test subsets. Next, all systems were tested and evaluated using standard metrics (F-score, recall, and precision) and two evaluation windows. The results show that the difference between MP3 compression used in streaming portals and the most common lossless format is minimal regarding onset detection. The TKEO shows some potential in audio signal degradation with subsequent conversion to time-frequency representation for neural network training. However, networks would have to be trained on such modified data to confirm or refute this effect. A general improvement of onset detection by adding TKEO has not been confirmed. The F-score and recall with both evaluation windows were even increased for the RNN system when radio broadcast simulation was applied. Experiments suggest that radio broadcast degradation generally increases the number of detected onsets, both TP and FP, resulting in worse precision but better recall.

This phenomenon could be used to enhance the time-frequency representation input of neural network-based detectors. In future research, we would like to explore the use of degradations and specially modified spectrograms as a new type of pre-processing. Then, we may optimize the delicate recall-precision trade-off by changing the input for neural network training.

2.7 Further Notes

In the original study, we theorized that new network-based detectors might be trained on such degraded data to identify proper onset time positions more successfully. This idea was derived from the behavior of TKEO degradation and consecutive ODF on some recordings of woodwind instruments. In [86], the beat detector based on the state-of-the-art ML architecture was trained on audio recordings degraded by TKEO, but no improvement in overall beat detection accuracy was reported.

3 MPA-oriented Global Tempo Computation

This chapter is based on the journal article “Enhancement of Conventional Beat Tracking System Using Teager–Kaiser Energy Operator” [2], which is a continuation of our previous studies on the global tempo estimation using a modified beat tracking method [3, 4].

Beat detection systems are widely used in the music information retrieval (MIR) research field for the computation of tempo and beat time positions in audio signals. One of the most important parts of these systems is usually onset detection. There is an understandable tendency to employ the most accurate onset detector. However, there are options to increase the global tempo (GT) accuracy and also the detection accuracy of beat positions at the expense of less accurate onset detection. The aim of this study is to introduce an enhancement of a conventional beat detector. The enhancement is based on the Teager–Kaiser energy operator (TKEO), which pre-processes the input audio signal before the spectral flux calculation. The proposed approach is first evaluated in terms of the ability to estimate the GT and beat positions compared to the same conventional system without the proposed enhancement. The accuracy of the GT and average beat differences (ABD) estimation is tested on the manually labeled reference database. Finally, this system is used for the analysis of a string quartet music database. Results suggest that the presence of the TKEO lowers onset detection accuracy but also increases the GT and ABD estimation. The average deviation from the reference GT in the reference database is 9.99 BPM (11.28%), which improves the conventional methodology, where the average deviation is 18.19 BPM (17.74%). This study has a pilot character and provides some suggestions for improving the beat tracking system for music analysis.

3.1 Introduction

Onset time in audio signal analysis represents the time position of a relevant sound event, usually when a musical tone is created. Onset detection functions are algorithms that capture onsets (onset time positions), and thus ideally all tones in audio recordings. They can create a representation or an evolution of onset structure at a given time of a particular audio recording. There are also offsets of tones (indicating the end time position of a tone in a signal), e.g., see [77, 87], but beat tracking systems do not need such information to work properly. The conventional beat tracking system is usually based on the calculation of repetitiveness of the dominant components in an onset function (onset curve), and its output represents a temporal framework, i.e., time instances, where a person would tap when listening to the corresponding piece of music. That is why it is important to have a robust and computationally effective onset detector. Calculating the beat positions and global tempo (GT) is important for musicologists and music analysts. With such automated systems, tempo and agogic changes can be measured much faster than only with a manual approach alone. Musicologists would have to spend less time correcting calculated beat positions. Therefore, we set a new parameter—the average deviation of reference beat positions to the calculated beat positions as the average beat deviation (ABD).

Most of the onset detectors are based on energy changes in spectra, the variant of spectral flux. For bowed string instruments, there is a method called SuperFlux that can suppress vibrato in an expressive performance and reduce the amount of false-positive detections [73]. Some methods use logarithmic spectral compression to enhance the spectral flux onset detection and then compute the cyclic tempogram for a tempo analysis [88]. There is also a method that calculates tempograms using Predominant Local Pulse [89]. Besides, the onset detection and beat detection could be performed in several toolboxes and libraries such as Tempogram Toolbox [90], LibROSA [91], MIR Toolbox [92], etc. [93]. The state-of-the-art onset detectors are usually based on deep neural networks [68, 72], using spectral components and parameters as their inputs. Beat detection systems contribute from the solid onset detectors, where periodicity is identified [90, 92, 94, 95, 96, 89].

While onset detection in percussive music is considered to be highly accurate (already at MIREX 2012 conference [74], algorithms achieved F-measure values greater than 0.95 for percussive sounds), detection of soft onsets produced by bowed string or woodwind instruments is still challenging. Many improvements in onset detection have been made, but no system is truly universal for all musical instruments and all types of music.

This work aims to enhance the conventional beat tracking system while following the tempo analysis methodology published in [97, 98] using a more sophisticated approach of tempo estimation based on the automated beat tracking system with the Teager–Kaiser energy operator (TKEO) included. This nonlinear energy operator is used, e.g., for the improvement of onset detection in EMG signals (electromyography) [99], to decompose audio into amplitude and frequency modulation components [100], for the detection of Voice Onset Time [101], or the highly efficient technique for LOS estimation in WCDMA mobile positioning [102]. So far, there is no extensive study on using TKEO to analyze musical instruments.

3.2 Dataset and Methods

3.2.1 Onset Detection

Usually, onset detection algorithms use pre-processing steps to reduce redundant information and improve detection accuracy. In this study, we propose a new method of pre-processing based on the TKEO. The TKEO ($\Psi\{s(t)\}$) is a nonlinear energy operator that can be calculated using the following formula:

$$\Psi\{s(t)\} = \left(\frac{\mathrm{d}s(t)}{\mathrm{d}t}\right)^2 - s(t) \cdot \frac{\mathrm{d}^2 s(t)}{\mathrm{d}t^2}, \tag{3.1}$$

i.e., we compute the square of the first derivative (which denotes the square of the rate of signal change) and then subtract the signal multiplied by the second derivative (which determines the acceleration at that point). We speed up the temporal changes of the signal module by removing the slow changes because we consider the rate of change. It is known that the faster the time changes, the higher the frequency components appear in the spectrum. By taking the first derivative into account, we increase the magnitude of higher frequencies of the spectrum [103].

In our discrete approach, we first downsample the input signal $x(n)$ to 22 050 Hz. Next, we apply the TKEO, i.e., we calculate the corresponding discrete non-causal form:

$$\Psi\{x(n)\} = x^2(n) - x(n-1) \cdot x(n+1), \tag{3.2}$$

which creates an energy profile of the given audio sample. In comparison to the conventional squared energy operator, the TKEO takes into account the signal's frequency [104], and it can yield negative values, e.g., see Figure 3.1. Differences in spectra for the same audio track (clarinet recording) are shown in Figure 3.2. The dominant spectral components have changed—the clarinet has naturally strong odd harmonics, but the TKEO has changed their magnitude.

We calculate the onset envelope using the perceptual model in the following step. We use Short-Time Fourier Transform (STFT) with Hann window (hop factor: 512 samples) and then the conversion to the perceptual model with log-power mel-frequency representation: 120 mel bands, max frequency at 10 kHz and min frequency at 27.5 Hz. We get the matrix $|X(m,k)|$, where m denotes the index of the frame and k is the frequency bin or index of the mel band. These settings were inspired by SuperFlux calculation [73].

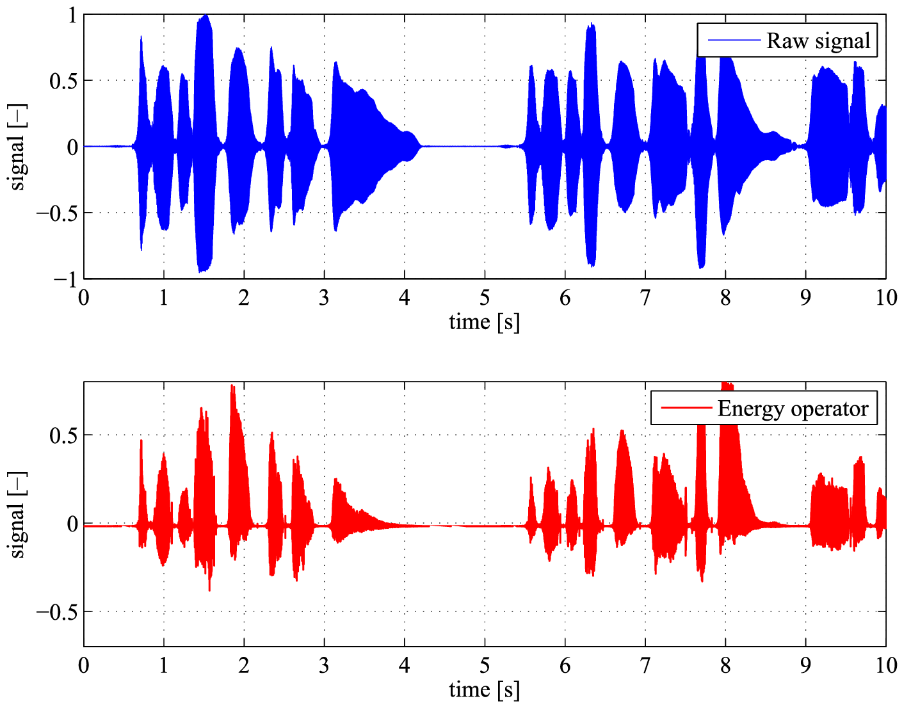


Fig. 3.1: Waveform of the clarinet recording before and after the application of TKEO.

In the next step, we calculated the spectral flux. The basic version of spectral flux is defined as the l_1-norm of consecutive frames [32]:

$$SF(m) = \frac{1}{K} \sum_{k=0}^{K-1} H(|X(m+1,k)| - |X(m,k)|), \tag{3.3}$$

for $m = 0, 1, 2, \ldots, M-2$, where $H(x) = (x + |x|)/2$ is the half-wave rectifier, M is the number of frames, and K is half of STFT frequency bins, or number of mel bands. A half-wave rectifier sets negative values to zero, and positive differences are summed across all frequency bands. Spectral flux gives us information, on how energy in spectra changes in time. Finally, a peak-picking function is applied [75] to identify time positions of onsets and, therefore, new tones in the audio signal.

An example of this system based on the mel-frequency representation, but without the use of TKEO, is shown in Figure 3.3. It represents a solo clarinet part. The onset function detected many false peaks and marked positions where tones were not played. For comparison, Figure 3.4 shows the same signal, but in this case, pre-processed by the TKEO. The peak-picking function now marked all real onsets with better accuracy and without any false positive detection. The colorbar in dBFS (decibels relative to full scale) (Figure 3.5) is presented separately because of the proper alignment of a spectrogram and onset function but is the same for all spectrograms in this paper.

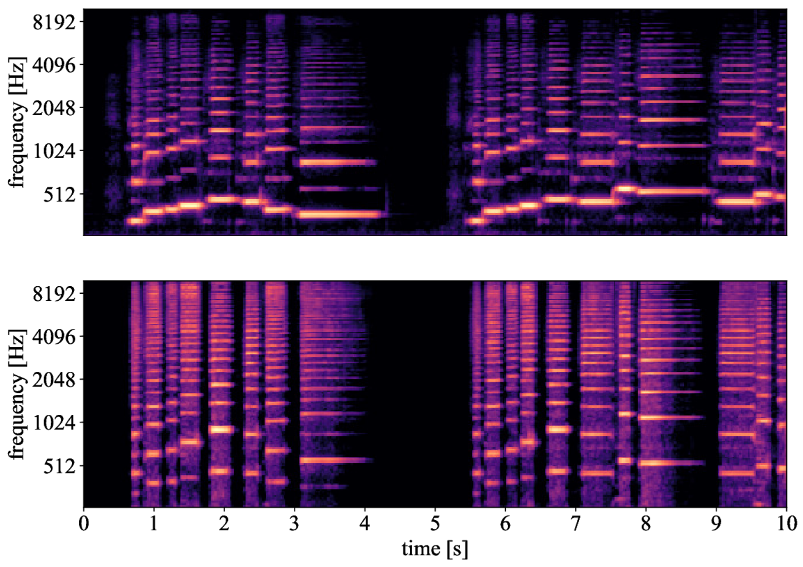


Fig. 3.2: Spectrograms of the same clarinet recording—the bottom one is using a TKEO step.

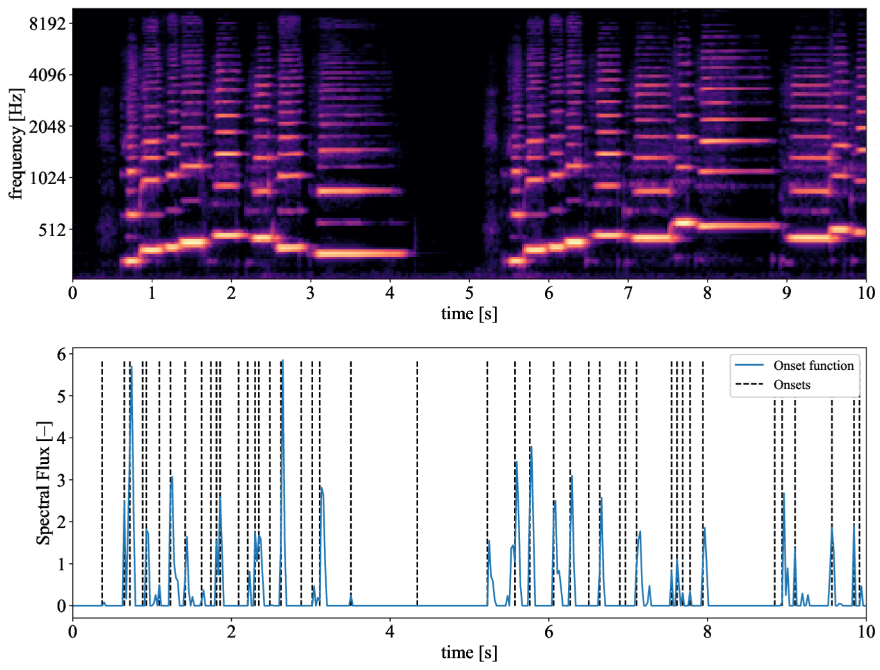


Fig. 3.3: Spectrogram and onset detection function for a solo clarinet recording.

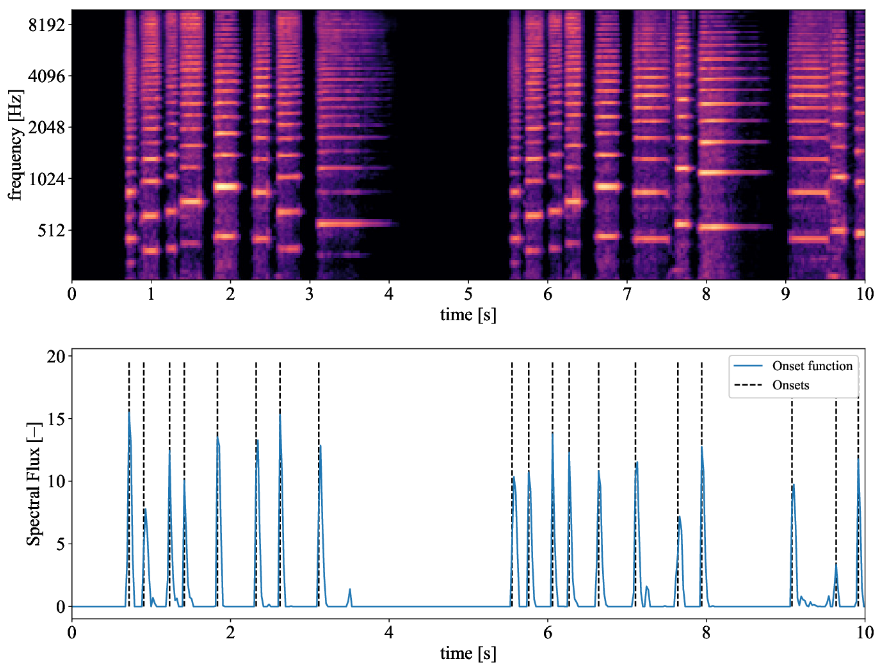


Fig. 3.4: Spectrogram and onset detection function for a solo clarinet recording with the TKEO applied.

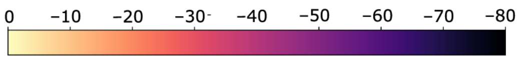


Fig. 3.5: Colorbar in dBFS units.

As we can see on the second spectrogram (Figure 3.4), the energy in spectra changed, frequencies do not correspond properly to the original signal and new tones are sharpened and much clearer. We give this example for a reason. Recording of a solo clarinet was the only audio track in which the accuracy of the onset detection function was improved. Adding TKEO into this conventional detection method lowered the general detection accuracy. It decreased the number of detected false positives and the true positives. The cause of this phenomenon is explained in the following Section 3.2.2.

3.2.2 TKEO Influence

We applied the proposed method with the TKEO included on more recordings and observed that in cases where the tones are fast (e.g., violin playing thirty-second notes) or the energy difference is very low, this method does not detect every onset properly. Adding the TKEO increased the detection tolerance of fast changes in the signal. This means that the operator added additional "latency" to the signal values. It also decreased this system's ability to capture low-energy spectral components. In general, fewer onsets were detected—only strong and more rhythmically important onsets remained. This is the advantage of the TKEO in the system. It suppresses less dominant spectral components and very fast tones, even though onset detectors are usually set to do the opposite.

Figures 3.6 and 3.7 show another analyzed track—a violin solo in a very fast tempo. There is a clear difference in spectrograms for the described detector and the same detection with the TKEO included. Most of the tones are quite visible in the spectrogram of the first figure. However, the system with the TKEO has its changes in the spectrum vaguer and blurry, which means that the onset function detected a lower number of onsets (especially between the 1st and the 4th second of this track). In this case, the conventional system detected more onsets correctly, but that still does not indicate that the estimation of GT would also be more accurate.

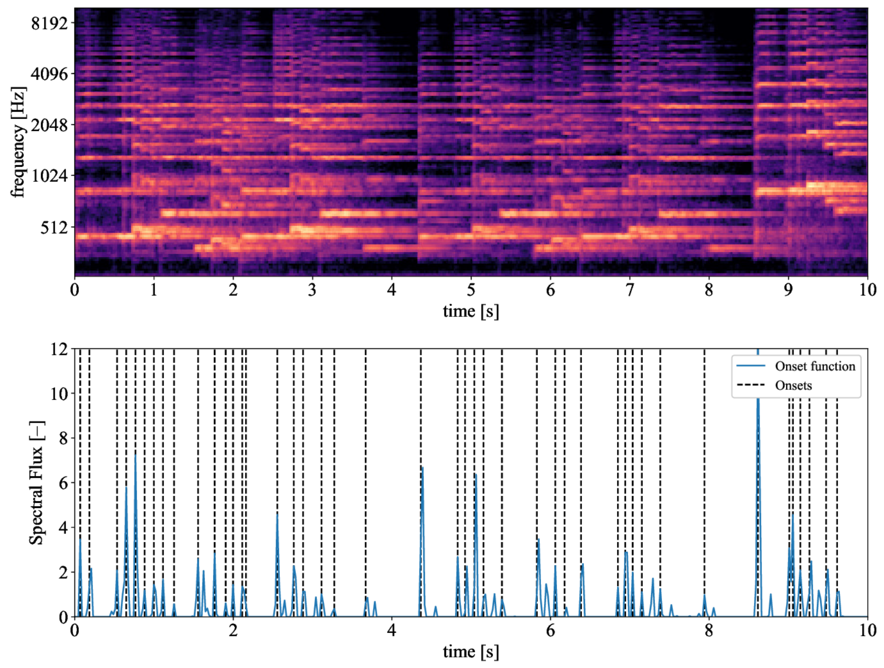


Fig. 3.6: Spectrogram and onset detection function for a solo violin recording.

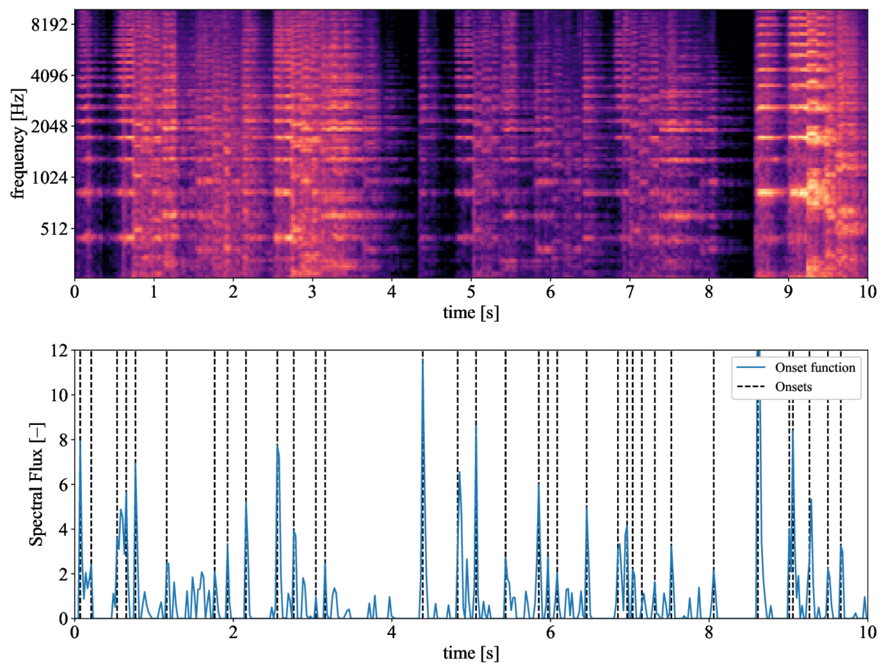


Fig. 3.7: Spectrogram and onset detection function for a solo violin recording using TKEO.

3.3 Tempo Representation

To create a tempo structure of given recordings, we need a representation of a tempo; in this case, how the density of onsets, or more precisely, the repetitiveness of significant onsets, is distributed. This can be done by several techniques; in our experiments, we focused on the method of dynamic beat tracking system proposed in [94]. This system first estimates onset positions in the ODF and picks the best beat candidates that follow specific rules (such as being within the minimal and maximal inter-beat-interval) within a pre-defined time interval—a parameter called default tempo. The default tempo is calculated automatically based on an autocorrelation function with respect to the standard 120 beats per minute (BPM) or set up manually based on prior information. The calculated beat positions can deviate from the default tempo in adjustable boundaries (depends on settings, e.g., Ellis reports approximately 10% [94]) utilizing the "tightness" parameter, which corresponds to the detection tolerance from the default tempo. It was set to 50 in all our experiments. Figure 3.8 shows how this system picks onset candidates from the onset curve and creates the beat positions by using periodicity information. Even though the selected beat tracking method may not produce robust beat estimates compared to today's ML beat detectors, we aim only to estimate the overall global tempo. Beat detectors are based on a calculation of beats in an audio signal and, therefore the metric structure from an elementary point of view. Usually, there is not enough information to divide beats into measures or bars without manual correction (or automatic downbeat detectors), but with proper segmentation, MIDI reference, and dynamic time warping (DTW) techniques, this is possible [105]. With automated systems, tempo and agogic changes can be retrieved faster; however, no detectors achieve consistent detections for all kinds of music. Musicologists could spend less time correcting calculated beat positions or creating manual annotations if they use tools to speed up the annotation process significantly. Therefore, we also consider the ABD parameter in the evaluation—the average deviation of reference beat positions to the calculated beat positions.

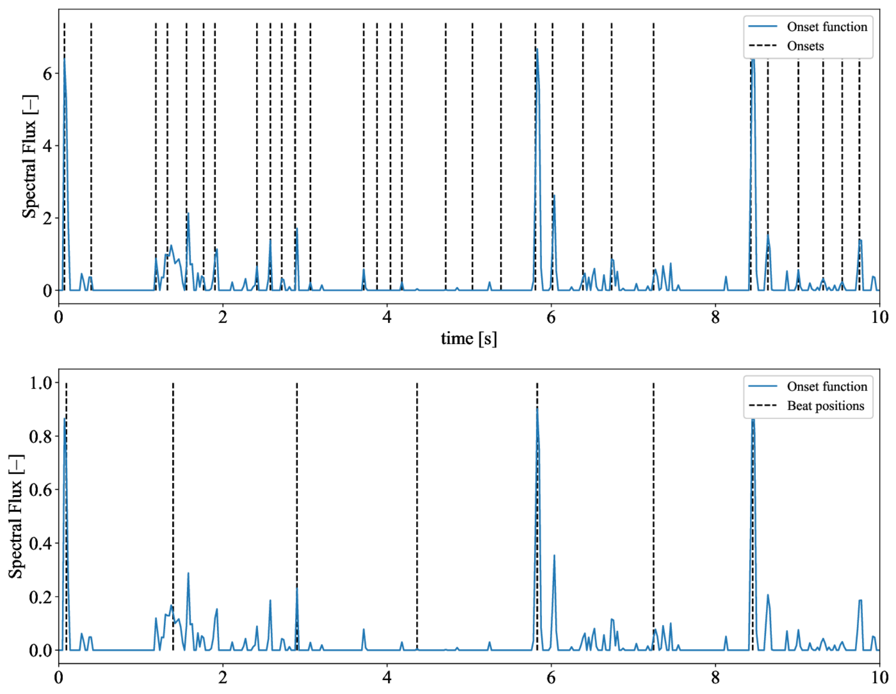


Fig. 3.8: Comparison of the onset and beat positions.

Figure 3.9 shows the estimated time positions of beats at the beginning of a string quartet segment. The system utilizes periodicity information to calculate beat positions even at places where no onsets are detected—in this specific part, a second violin and viola are playing very quietly (and no onsets are detected), and then a violin solo begins. There are strong onsets in the ODF between the sixth and tenth seconds of this track. Their periodicity information is then used to fill the gap in the silent part of this recording, which is one of the advantages of the dynamic programming beat tracker. In the postprocessing of beat activation functions from ML-based approaches, a Dynamic Bayesian Network (DBN) [106] is usually used to retrieve the final beat estimates and fill the silent parts. The limitation here is the default tempo—the algorithm searches for beat positions within a given interval, but there is no guarantee that true beat positions exist within specified limits (also concerning the tolerance parameter). The default tempo can be misleading if the recording is rhythmically unstable or the local tempo changes significantly.

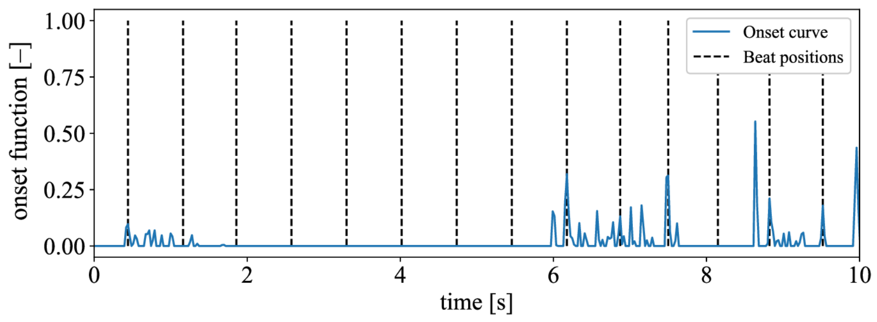


Fig. 3.9: Estimated beat positions by the beat detector based on the onset periodicity.

3.4 Dataset

First of all, we tested if the TKEO improves the estimation of the GT in general. We used the *SMC_MIREX* dataset [107], which consists of various recordings, from classical pieces to guitar solos. The recordings are sampled by 44.1 kHz. Their annotations contain manually corrected beat time positions, which will be used as a reference.

Music by string quartets is very specific because the tempo can be more or less stable, but the musical ornaments, intended gaps, fermatas, or other expressive musical attributes can be present. Every musician has her/his unique style of agogic performance. If we define meaningful musical parts by choosing important musical motifs, we can create segments that could be processed separately.

The second dataset consists of 33 different interpretations of *String Quartet No. 1 e minor "From My Life"*, composed by the Czech composer Bedřich Smetana. We also included two interpretations played by an orchestra. We divided the first movement into six segments of musical motifs in the view of the musical meaning. The first movement consists of an introduction (Beg), exposition (A), coda (B), development (C), recapitulation (D), and the last coda (E). We calculated the estimated average tempo (EAT) for every segment without any expressive elements and information about beat positions, using the physical length of the tracks and metric information in the corresponding music sheet. The EAT will be used as a reference tempo for setting up the default tempo parameter in the beat tracking system. The first page of the sheet music is provided as an example in Appendix B.1.

3.4.1 Application

Beat tracking systems are used in music analysis software for tempo, timbre, dynamics estimation, or other music analysis goals. An example of such freeware software is Sonic Visualiser [108]. Figure 3.10 shows an example of tempo analysis of the string quartet music from the second tested database. The first pane is the visualization of the audio wave, the second one is the spectrogram, and the last one is a layer of manually corrected beat positions. Beat positions were calculated automatically by the beat tracking system called BeatRoot [109] (Vamp plugin) and then corrected by trained ears. The green line shows how tempo evolves in time—if the audio track is locally slowing down or the tempo increases. The method which is proposed in this paper has not been developed as a Vamp plugin for Sonic Visualiser.

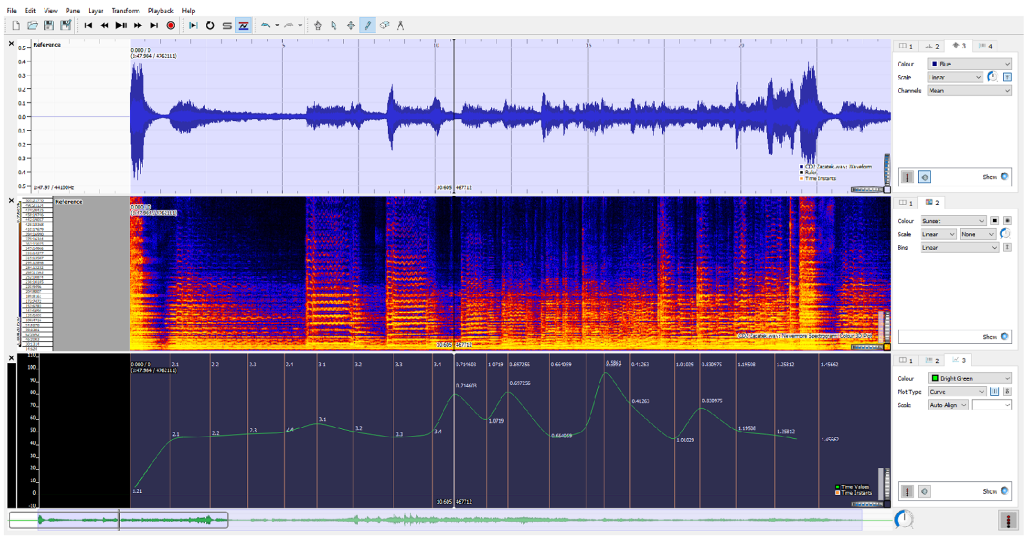


Fig. 3.10: Possible application of the beat tracking system.

Musicologists can then conclude from the measurement results. An automated beat tracking system can reduce the time of analysis significantly. For example, if we measure the EAT of the first motif of the second database for each recording, we get interesting results. One of the general assumptions is that presently, we usually play the same piece of classical music faster than we did before. Figure 3.11 shows that this assumption may not be correct. There is a trend (see the slope of the linear regression line based on the sum of squares)—older recordings are, on average, at a faster pace. We do not have enough audio recordings to declare it as a fact, but the tendency is there. However, when we plot the EAT of the entire first movement (Figure 3.12), the tempo decrease is not so evident. Each black dot represents one interpretation, and the blue line is a trend line. The sample from 1928 was an outlier; therefore, we did not consider it in the regression analysis.

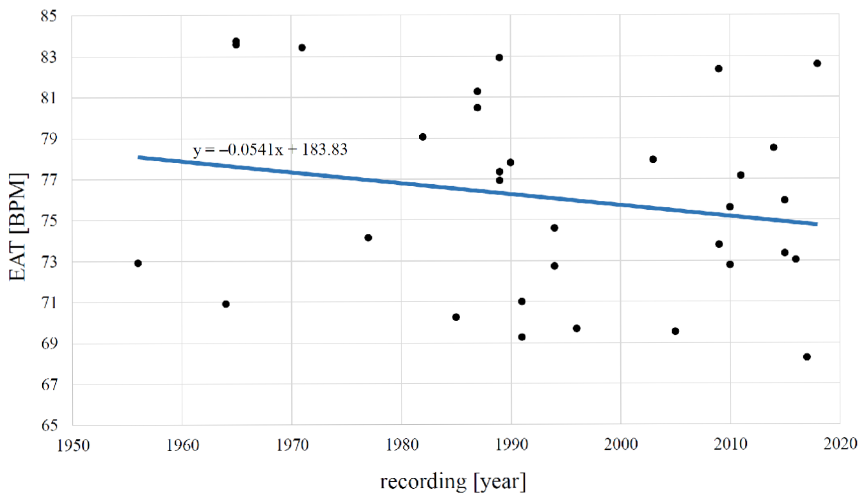


Fig. 3.11: Results of the EAT calculation for the first motif of the string quartet database.

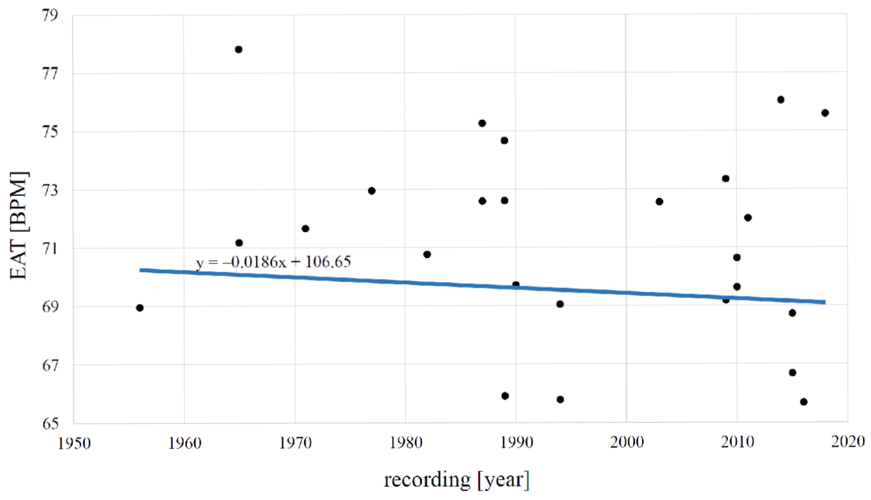


Fig. 3.12: Results of the EAT calculation for the entire first movement of the string quartet database.

3.5 System Evaluation

During the analysis, we first used the reference dataset to determine the GT and ABD estimation accuracy. We computed the GT of each track by the proposed beat tracking method using both the proposed onset detection function (DS, default system), and the same onset detection function with the TKEO (TS, system with the TKEO included). Then we compared the reference values (annotation of the dataset) of each tested track with values estimated by the DS and the TS. The reference tempo was obtained as the number 60 (BPM definition) divided by the median of time differences between consecutive beat time positions. Then we calculated the median (Me) and the mean value ($\bar{x}$) of time differences of consecutive beats in all recordings and also in which the average was less than 1 s. This represents the ABD of tracks that were close to the reference tempo (some recordings achieved more than ~20 BPM difference in the GT when tested; they were excluded for the extended ABD testing).

Next, we analysed the string quartet database. First, all 33 recordings were divided into six segments with a relatively steady tempo, and then all motifs were tested by the TS and the DS to estimate the GT. We computed the reference EAT of all segments of each interpretation (Table 3.1) by calculating the number of quarter notes (Table 3.2) and dividing them by the time length of each recording. The complete table is in Appendix B.1. Finally, the EAT and the computed GT were compared.

Tab. 3.1: The EAT of all motifs of the string quartet dataset.

track	Beg	A	B	C	D	E
CD01	80.61	69.37	34.41	88.56	55.60	74.50
CD02	77.80	69.03	44.14	81.84	59.52	72.43
CD03	77.93	73.19	41.60	87.09	62.36	79.14
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
CD33	76.92	63.24	42.05	74.62	56.26	68.31

All values are in BPM.

Tab. 3.2: Calculation of quarter notes in all motifs.

motif	Beg	A	B	C	D	E
measures	1–70	71–110	111–118	119–164	165–225	226–262
quarter notes	280	160	32	184	244	148

3.6 Results

Table 3.3 shows the GT estimation for the first 30 tracks of the first dataset. The complete table is shown in Appendix B.2. The average deviation from the reference tempo was 18.19 BPM (17.74%) for DS and 9.99 BPM (11.28%) for TS. We also applied the *t*-test (Paired Two Sample for Means) for each system. The P-value for the TS is 0.038 and 0.024 for the DS ($\alpha = 0.05$). Next, Table 3.4 presents the overall results of the GT estimation: median, mean, standard deviation, relative standard deviation, variance for each tested system, and the deviations from the reference values for all metrics. The mean value of the reference GT was 76.78 BPM, the average computed GT 88.97 BPM for DS and 83.75 BPM for TS.

Table 3.5 shows the mean value and the median of the ABD testing for all analyzed tracks. The average difference between consecutive beat time positions of the reference and the DS was 2.84 s and 2.30 s for TS. Table 3.6 shows the average of mean and median of time difference values of the recordings in which the ABD were less than 1 s. This means 9 recordings for the DS (30%) and 11 recordings for the TS (37% of the second dataset). The TS detected the right rhythmic pulse in more recordings than the DS. Average deviations from the reference beat positions were 0.39 s and 0.29 s for the TS and 0.95 s and 0.36 s for the DS, respectively.

Table 3.1 shows results based on the EAT of all motifs of our second database—33 different interpretations of *String Quartet No. 1 e minor "From My Life"*. Finally, Table 3.7 shows the difference between the estimated GT and the EAT for both proposed systems. The complete table is shown in Appendix B.3. The average deviation for the TS is 6.42 BPM and 6.59 BPM for the DS. Due to the nature of the results of the second dataset, no further statistical processing was used.

Tab. 3.3: Reference GT and computed GT of the *SMC_MIREX* dataset.

track no.	reference	TS	DS	TS (dev.)	DS (dev.)
1	48.15	47.85	47.85	0.30	0.30
2	66.99	73.83	73.83	6.84	6.84
3	68.00	95.70	95.70	27.70	27.70
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
30	63.36	63.02	63.02	0.34	0.34
mean	76.78	83.75	88.97	9.99	18.19
P-value		0.038	0.024		

TS—beat detection with the TKEO degradation; DS—default beat detection; dev.—deviation from the reference global tempo; $\alpha = 0.05$.

Tab. 3.4: Results of GT testing—the *SMC_MIREX* dataset.

type	Me	$\bar{x}$	sd	rsd	var
reference	77.11	76.78	33.01	0.43	1089.50
TS	82.05	83.75	37.30	0.45	1391.05
DS	76.07	88.97	41.05	0.46	1685.16
TS dev.	5.31	9.99	15.75	1.58	247.93
DS dev.	7.26	18.19	24.08	1.32	579.71

Me—median; $\bar{x}$—mean value; sd—standard deviation; rsd—relative standard deviation; var—variance.

Tab. 3.5: Results of the ABD testing for all recordings.

metrics	TS (s)	DS (s)
$\bar{x}$	2.30	2.84
Me	1.81	2.57
sd of the $\bar{x}$	1.90	2.17
sd of the Me	2.14	2.31

Tab. 3.6: Results of the ABD testing for recordings with the average ABD < 1 s.

	dev. < 1 s in the mean of TS				<1 s in the mean of DS			
	TS		DS		TS		DS	
	$\bar{x}$	Me	$\bar{x}$	Me	$\bar{x}$	Me	$\bar{x}$	Me
mean	0.39	0.38	0.95	0.70	0.29	0.12	0.36	0.22

Figure 3.13 shows differences between the reference GT and calculated GT of the TS and DS of the first database. The TS generally follows the reference tempo more accurately, mainly because it often determines the correct metric pulse. The DS shows greater local deviations of the GT from the tested tracks.

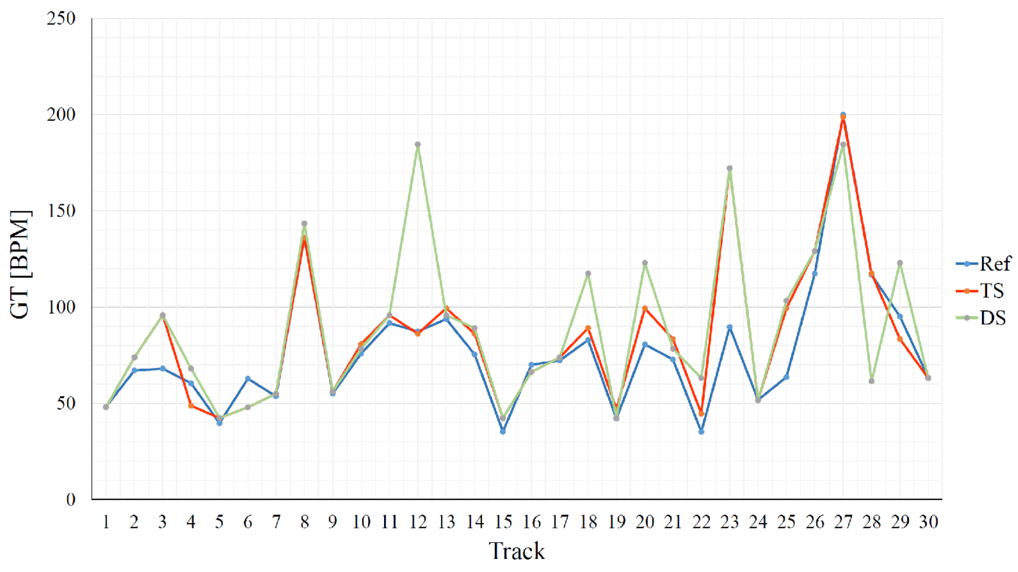


Fig. 3.13: Visualisation of the GT computation—Ref, TS and DS estimation.

Tab. 3.7: Differences between the estimated GT and the EAT for both systems.

	TS						DS					
Track	Beg	A	B	C	D	E	Beg	A	B	C	D	E
CD01	15.09	13.98	6.61	3.73	5.92	3.80	8.49	22.92	6.61	3.73	1.82	3.80
CD02	14.49	0.81	6.53	1.51	6.74	3.57	14.49	9.27	2.84	1.51	3.50	1.40
CD03	14.36	1.41	7.15	5.20	4.94	6.99	11.17	10.16	11.13	5.20	13.64	6.99
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
CD33	3.83	6.60	9.63	0.79	5.26	3.47	3.83	6.60	9.63	0.79	11.74	5.52
mean	7.56	6.57	8.13	1.78	9.01	5.49	6.92	7.17	8.71	1.78	9.58	5.38
result	6.42						6.59					

3.7 Discussion

The proposed method with the TKEO degradation provided some improvements to the reference *SMC_MIREX* dataset. The results suggest (Table 3.4) that the TKEO can help the proposed beat tracking system pick better onset candidates for the beat positions and slightly improve the GT estimation. The difference was about

8 BPM on average for all tested recordings of the reference dataset. However, many recordings reported the same estimated GT for both methods. We used the first dataset with manually corrected beat positions to determine the accuracy of both systems. We did not use F-measure, one of the standard metrics in a beat tracking task, but rather average differences between consecutive beats. This gives us an idea of how close the beat tracking was to the reference positions, which may provide more insight into the possibilities of automated annotations for music analysis. The system with the TKEO reported lower ABD for all settings used overall. The results suggest that the TKEO pre-processing improved the accuracy of the beat tracking system while reducing the onset detection accuracy. The only exception was the recording of the clarinet from the reference dataset.

Experimenting with the string quartet dataset, the results were again slightly in favor of the system with TKEO included. All 33 recordings of the second database were tested. The difference between the average deviation from the EAT of TS and DS was only 0.17 BPM. Both systems had very similar detection accuracy. We chose such string quartet music to see how the enhancement would deal with the difficult task of detecting tempo in music with weak onsets and complex metric and rhythmic structures.

The TKEO was used in the pre-processing stage to help the onset detection function find more relevant onsets and enhance the beat tracking system to choose better candidates for beat positions. It reduced the number of insignificant onsets detected. Onset detection accuracy has usually been reduced, but the final beat detection output may be more stable; the algorithm chooses from less and more important onsets. This is useful for analyzing tracks where we suspect a stable and non-agogic rhythm.

The limitation of this study is that the EAT in the string quartet dataset may serve as a reference value for the beat tracking system, but it is not the actual GT of a particular track since we cannot include any expressive elements in it. It does not provide any information about beat positions or local tempo changes. The same thing applies to the reference global tempo. In music performance analysis, we need to track all beat positions in the segment and compare them to the real beat positions. However, in this case, we analyzed relatively stable tracks with no abrupt tempo changes.

3.8 Conclusions

This study introduces an enhancement of the conventional beat tracking system by adding the TKEO into the pre-processing stage. It briefly describes the onset detection function and the beat tracking method with its possible application. The onset detection accuracy decreased in most analyzed tracks, but the accuracy of the GT and ABD detection increased.

The influence of the TKEO was tested on different recordings, and it was found that in the case of woodwind instruments, the TKEO increased the onset detection accuracy. In the future, we would like to focus on the possible applications of the TKEO on music recordings as it generally changes the magnitude of frequency components in a signal and acts as a filter.

The estimation of the GT was improved in the reference database. The average deviation from the reference GT in the reference database is 9.99 BPM (11.28%), which improves the conventional methodology (18.19 BPM, 17.74%). The P-values indicate that there is a clear difference between the proposed systems. Both systems were also tested on the string quartet database. In this case, however, the results are not convincing. The proposed TS will be further used in the subsequent music analysis of the string quartet database. The aim is to create an automated system for capturing beat positions that are as close as possible to the actual beat positions in the recordings, even for complex music such as string quartet. In this way, minimizing the time required for manual processing and labeling is possible. This study has a pilot character and provides some suggestions for improving the beat tracking system for music analysis.

4 TCN Beat Tracking

In this chapter, we introduce the conference article "Beat Tracking: Is 44.1 kHz Really Needed?" [5] that focuses on training multiple beat tracking detectors with various input sampling frequencies.

Beat tracking is essential in music information retrieval, with applications ranging from music analysis and automatic playlist generation to beat-synchronized effects. In recent years, deep learning methods, usually inspired by well-known architectures, outperformed other beat tracking algorithms. The current state-of-the-art offline beat tracking systems utilize temporal convolutional and recurrent networks. Most systems use an input sampling rate of 44.1 kHz. In this paper, we retrain multiple versions of state-of-the-art temporal convolutional networks with different input sampling rates while keeping the time resolution by changing the frame size parameter. Furthermore, we evaluate all models using standard metrics. As the main contribution, we show that decreasing the input audio recording sampling frequency up to 5 kHz preserves most of the accuracy and, in some cases, even slightly outperforms the standard approach.

4.1 Introduction

In MIR, one of the core tasks is beat tracking or beat detection. It aims at detecting "tactus" positions in an audio signal—described as "the most comfortable foot-tapping rate when unconsciously tapping to a piece of music" [110]. Early conventional approaches to beat tracking usually utilized a two-stage strategy. First, an onset detection function was computed from time-frequency representations, such as spectrograms or mel-spectrograms. Then, a post-processing phase with prior musical knowledge was implemented to determine which onsets might correspond to beats. Well-known examples of non-machine learning approaches are BeatRoot [109], dynamic programming-based method [94], and Predominant Local Pulse [89, 90].

Over the years, numerous beat tracking methods have been proposed, ranging from rule-based systems and probability methods to machine learning models. With the rise of deep learning techniques, beat tracking has significantly shifted toward data-driven deep neural networks. The increasing availability of data and their annotation[1] helped to outperform every conventional non-machine learning beat tracker (see MIREX results[2]). However, expressive music and genres with

[1]Beat annotation consists of discrete time points of beat occurrence.

[2]`https://www.music-ir.org/mirex/wiki/2019:Audio_Beat_Tracking` (accessed on 27 March 2023)

more complex metric and rhythmic structures are still considered highly challenging. The Recurrent Neural Networks (RNN) proved that data-driven approaches provide better results than conventional systems [111, 112]. Furthermore, the multi-model approach [95] was implemented to reflect different rhythms depending on the music style and genre. Selecting the best-performing model as the state-of-the-art is challenging due to the absence of a beat tracking competition (the last MIREX beat tracking competition ended in 2019, and further evaluation is based on the beat tracking community). However, many studies use n-fold cross-validation and similar datasets, achieving more than 90 % F-score (explained in Section 4.3.3) for non-classical and less expressive music.

The Temporal Convolutional Networks (TCNs), based on the original implementation of WaveNet [113], are one of the newest methods for beat tracking and usually achieve the highest F-score. The well-known examples are [114] and [110]. All mentioned models use time-frequency representation (modified spectrograms). Authors in [115] show an end-to-end approach using time-domain representation, feeding raw audio samples into the TCN, achieving similar results as state-of-the-art systems. It is also possible to train beat, downbeat, and tempo activation functions jointly [116, 110], solving more tasks with just one model. For more details about deep learning beat and downbeat tracking TCN models, we refer to [117].

In this paper, we implement multiple TCN beat tracking systems and evaluate their abilities to detect beats on standard datasets. We add skip connections to the networks as one of the possible network modifications and treat them as separate models. As the main contribution, we train five models on 44.1, 22.05, 11.025, and 5.5 kHz input sampling rates to demonstrate how much higher-frequency information is needed for the beat tracking task. We show that lower sampling rates slightly outperform the standard approach with 44.1 kHz input signal in most models. This may be useful for applying beat tracking systems in other MIR-related tasks, such as improving the synchronization accuracy when used jointly with Dynamic Time Warping methods [14] without the need for resampling. The results indicate that even a system trained on the 5.5 kHz input audio signal is comparable to the standard 44.1 kHz model. The higher frequency content seems redundant for the universal beat tracking task.

The rest of the paper is organized as follows. Section 4.2 describes the architecture, pre-processing, and models of TCN beat tracking systems. Section 4.3 explains the training and evaluation process. Finally, Section 4.4 shows the results followed by discussion and conclusions in Section 4.5.

4.2 Methods

4.2.1 Architecture

Temporal convolutional networks are a class of deep neural networks that have gained popularity in various time-series applications. TCNs consist of multiple layers of temporal convolutions and non-linear activations, allowing them to capture long-term dependencies in sequential data. The ability to model long-term dependencies makes TCNs an attractive option for beat tracking, as the tempo of a musical piece is inherently a temporal pattern, with the exception of expressive performances. For beat tracking, the input to the TCN is a sequence of audio features, usually modified spectrograms. The output of the TCN is a sequence of beat activations, representing the likelihood of a beat occurring at each time step. The beat activations are then post-processed to estimate the exact time positions of beats. We apply a post-processing method called Dynamic Bayesian Network (DBN) [106] as a standard approach to obtain a sequence of beats.

To adapt TCNs to beat tracking, researchers have proposed various modifications to the standard TCN architecture. One of the modifications is to use skip connections, allowing the network to bypass certain layers and directly propagate information from earlier to later ones. Skip connections have been shown to improve the training stability and convergence of TCNs. In our paper, we experiment with three slightly different versions of the TCN beat tracker and modify two of them with additional skip connections.

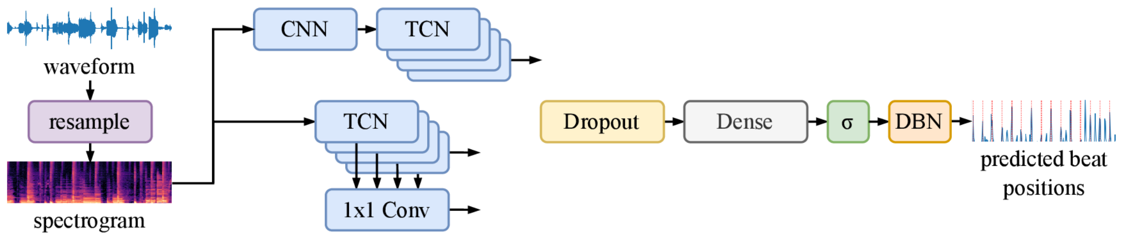


Fig. 4.1: High-level overview of different approaches to TCN beat tracking.

Figure 4.1 shows different variants of beat tracking neural networks. The first approach is to use a two-dimensional Convolutional Neural Network (CNN) to extract musically motivated features from the spectrograms and then use a sequence of TCN blocks to capture temporal information. In this work, we experiment with discarding the CNN and using only the TCN blocks to reduce the number of trainable parameters. We also utilize skip connections and combine the intermediate outputs of the TCN blocks using a 1×1 convolutional layer instead of taking only the output from the last TCN block. We evaluate all models using 44.1, 22.05, 11.025, and 5.5 kHz as input sampling rates.

4.2.2 Pre-processing

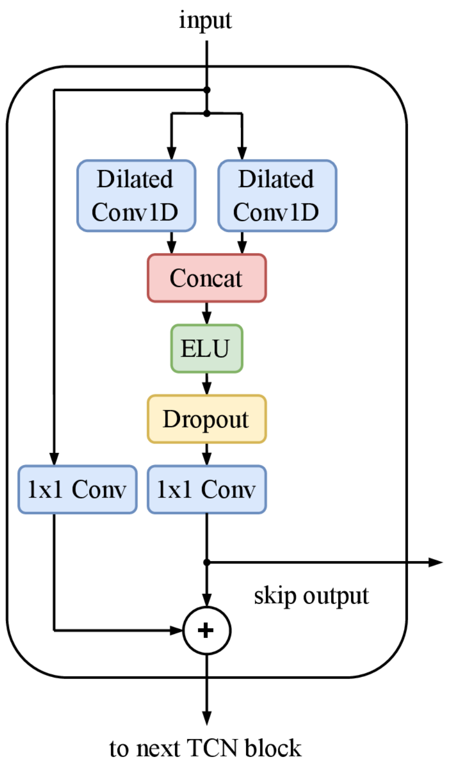
We use frame sizes 2048, 1024, 512, and 256 samples to maintain the same temporal context for 44.1, 22.05, 11.025, and 5.5 kHz sampling rates, respectively. However, the 5.5 kHz sampling rate is a rounded number (the correct rate would be 5 512.5, which is impractical). Therefore, this model does not exactly follow the sampling rate/frame size compromise. We apply the Short-Time Fourier Transform to the audio frames, followed by a filter bank to obtain magnitude spectrograms with logarithmically spaced frequency bins. We refer to [117] for a detailed description of the pre-processing step. We ensured that the fps = 100 stayed the same for each scenario. The time resolution corresponds to the output beat activation function.

4.2.3 Models

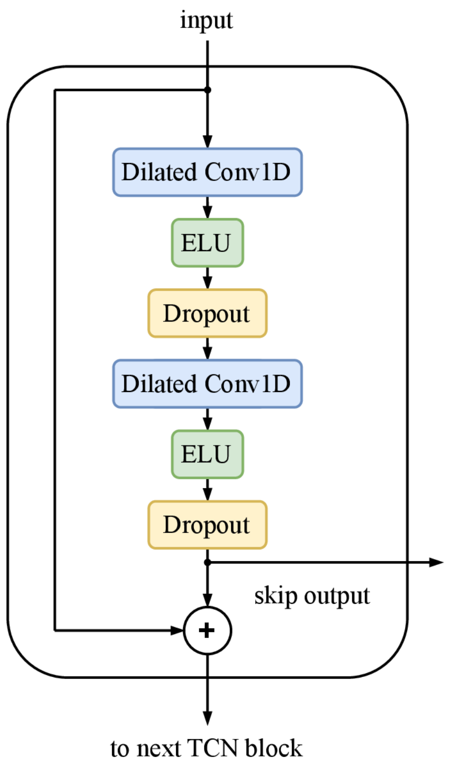
We use the model `bock_2020` from [110] as a baseline for our experiments. This model includes a CNN to extract relevant spectral features from the spectrograms, which are then fed as input to the first TCN block. The inner structure of the TCN block is shown in Figure 4.2a. The input gets first processed by two parallel dilated convolutional layers with different dilation rates. The output of the layers is then concatenated by the channel dimension, followed by an Exponential Linear Unit (ELU) activation function. The next block is a spatial dropout layer used only during training to prevent overfitting. We set the value of spatial dropout to 0.1 in all experiments. Then, a 1×1 convolutional layer is used to reduce the number of convolution channels in half. The TCN block also contains a residual connection with an additional 1×1 convolutional layer, which helps to retain information from previous TCN blocks. We also use a simplified TCN block described in [86] and implemented in `simple_tcn` and `tcn_dp`. The structure is depicted in Figure 4.2b. The models are listed below; each row represents a different architecture:

- `bock_2020_`x ,
- `simple_tcn_`x ,
- `simple_tcn_skip_`x,
- `tcn_dp_`x ,
- `tcn_dp_skip_`x .

To differentiate between various inputs of each model, we added a postfix: x stands for 44, 22, 11, or 5, which is equal to 44.1, 22.05, 11.025, and 5.5 kHz sampling rates, respectively.



(a) Diagram of a TCN block used in [110].



(b) Diagram of a TCN block presented in [86].

Fig. 4.2: Different proposed TCN blocks for a beat tracking task.

4.3 Experiments

4.3.1 Dataset

In our experiments, we use well-known datasets that have been used for beat tracking tasks for many years. We used the corrected annotations from S. Böck[3] with the exception of the Beatles dataset, in which all annotations were manually corrected to the corresponding ground-truth beat positions based on [86]. All datasets combined consist of 2 263 recordings with a total duration of around 26 hours and 175 127 ground-truth beat annotations. The list of datasets is described below:

- Ballroom [118] – excerpts around 30 s in length, dance music genres such as cha-cha, jive, quickstep, rumba, waltz, or tango,
- Hainsworth [119] – excerpts around 60 s in length, organized into six categories: rock/pop, dance, jazz, folk, classical, and choral music,
- GTZAN [120] – a large dataset containing 30 s excerpts and 10 different genres,

[3]`https://github.com/superbock/ISMIR2020` (accessed on 27 March 2023)

- SMC [121] – excerpts around 40 s in length, specifically selected to be challenging for the state-of-the-art beat tracking systems (for example, expressive performances, local tempo deviations, or complex music compositions),
- Beatles [122] – a collection of songs from Beatles with corrected annotations based on [110] and [86].

4.3.2 Training

First, we merge all datasets from Section 4.3.1 into one dataset and split it to train, test, and validation sets using the 80/10/10 strategy, respectively. Train and validation sets are shuffled for training, but the test set always contains the same recordings and annotation data.

We train each model on the training data while monitoring the performance on the validation set. We use the following settings: Adam optimizer, binary cross-entropy loss, and reduction of learning rate by a factor of 0.2 if the training does not improve for 10 epochs with the lower bound of learning rate set to 1×10^{-7}. Furthermore, early stopping is called if the change of validation loss is less than 1×10^{-4} for more than 20 epochs. The best checkpoint is then saved as the final model. Contrary to the original implementations, we use an augmentation inspired by [115]. During training, we shift the beat positions forward or back by a random amount between $\pm$ 70 ms. Table 4.1 shows the number of trainable parameters and training time for all models. The average training time of all proposed models combined was 50 minutes. The trainable parameters of the networks ranged from 48 481 to 71 521.

The model `bock_2020` derived from [110] was trained only on 44.1 and 22.05 kHz sampling rates, and without skipping modifications. Changing the network's input size was impossible without changing the inner structure—for example, convolution channels or the dilation factor. However, we decided to keep it in our experiments and show the difference between the original 44.1 and 22.05 kHz models.

The training and validation losses are shown in Figures 4.3 and 4.4, respectively. We only display one of the models (`tcn_dp_skip`) with all sampling rates for brevity.

4.3.3 Evaluation

We use standard F-score metrics on the test set to evaluate proposed models in terms of prediction accuracy. The F-score is a harmonic mean of precision and recall, based on true positives, false positives, and false negatives [123]. To decide if the target beat is within the range of ground-truth beat position, we use a window of length 70 ms, which is a standard value in the beat tracking community [78]. We

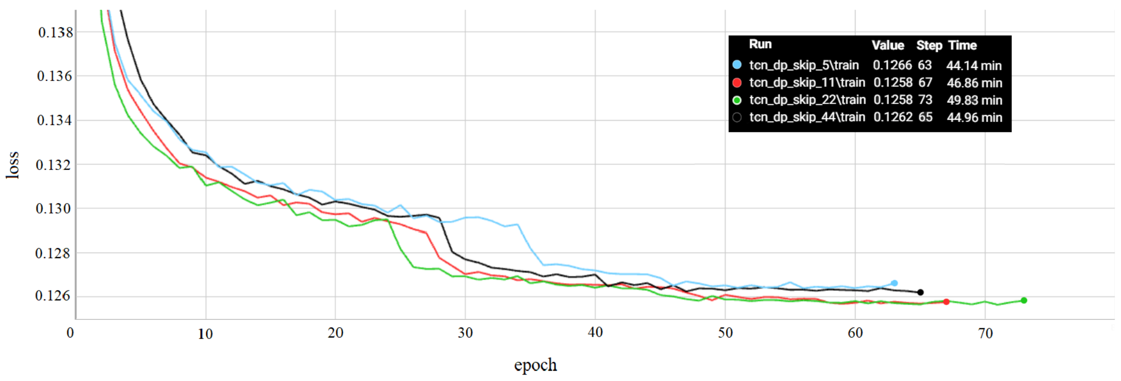
Tab. 4.1: The number of parameters, train time in seconds for each model, and the mean train time for each architecture.

model	params	train time [s]	mean [s]
`bock_2020_22`	65 941	3 286	3 099
`bock_2020_44`	65 941	2 911	
`simple_tcn_5`	64 701	3 308	3 249
`simple_tcn_11`	67 341	2 782	
`simple_tcn_22`	69 981	3 635	
`simple_tcn_44`	71 521	3 272	
`simple_tcn_skip_5`	64 483	3 355	3 340
`simple_tcn_skip_11`	67 123	3 422	
`simple_tcn_skip_22`	69 763	3 474	
`simple_tcn_skip_44`	71 303	3 110	
`tcn_dp_5`	48 481	2 374	2443
`tcn_dp_11`	49 921	2 180	
`tcn_dp_22`	51 361	2 570	
`tcn_dp_44`	52 201	2 648	
`tcn_dp_skip_5`	48 683	2 777	2 918
`tcn_dp_skip_11`	50 123	2 951	
`tcn_dp_skip_22`	51 563	3 117	
`tcn_dp_skip_44`	52 403	2 826	

compute additional metrics (Cemgil, P-score, Goto, and CMLc metrics) and refer to [122] for more details about their implementation.

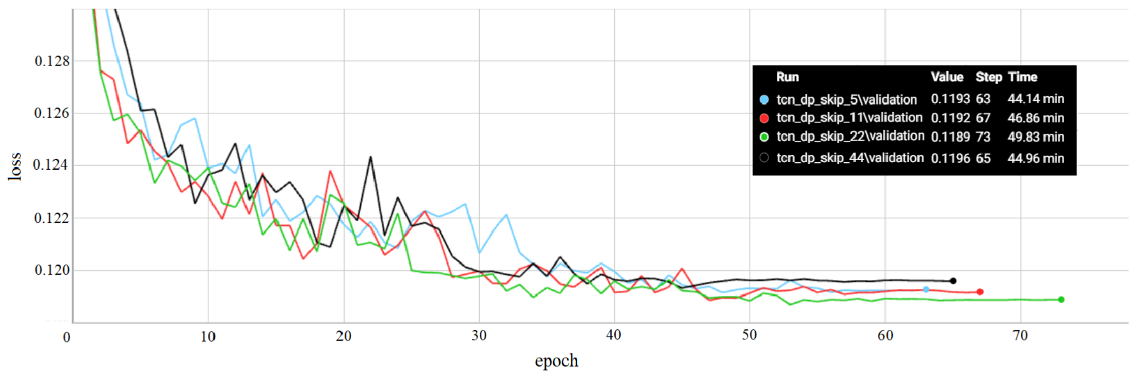
4.4 Results

We evaluated all models on the test set described in Section 4.3.1 and 4.3.2. Table 4.2 shows the F-score, Cemgil, P-score, Goto, and CMLc metrics for each architecture and sampling rate. Bold numbers indicate the best result for given metrics and architecture. The `bock_2020_22` model achieves the highest scores overall (F-score = 0.928, Cemgil = 0.829, P-score = 0.912, Goto = 0.828, and CMLc = 0.840), surpassing the 44.1 kHz version. Lower sampling rates slightly increase the models' accuracy. An exception is the `tcn_dp_44` model with F-score = 0.912 compared to the `tcn_dp_22` model with F-score = 0.900. The differences, however, are not significant. Furthermore, `tcn_dp_skip_22` is comparable to the state-of-the-art beat tracking model `bock_2020_44` with worse Cemgil and slightly better P-score and CMLc metrics. The difference between the training and validation process of the same architecture but varied input sampling rates is shown in Figures 4.3 and 4.4. There is no connection between training time and the input sampling frequency due to the early stopping mechanism.



Run	Value	Step	Time
● tcn_dp_skip_5\train	0.1266	63	44.14 min
● tcn_dp_skip_11\train	0.1258	67	46.86 min
● tcn_dp_skip_22\train	0.1258	73	49.83 min
○ tcn_dp_skip_44\train	0.1262	65	44.96 min

Fig. 4.3: Loss of the `tcn_dp_skip` model on the training data for each epoch.



Run	Value	Step	Time
● tcn_dp_skip_5\validation	0.1193	63	44.14 min
● tcn_dp_skip_11\validation	0.1192	67	46.86 min
● tcn_dp_skip_22\validation	0.1189	73	49.83 min
○ tcn_dp_skip_44\validation	0.1196	65	44.96 min

Fig. 4.4: Loss of the `tcn_dp_skip` model on the validation data for each epoch.

Tab. 4.2: Beat tracking evaluation of all models on the test set using standard metrics. Bold numbers indicate the best result for given metrics and architecture.

model	F-score	Cemgil	P-score	Goto	CMLc
`bock_2020_22`	**0.928**	**0.829**	**0.912**	**0.828**	**0.840**
`bock_2020_44`	0.925	0.815	0.904	0.806	0.821
`simple_tcn_5`	0.907	**0.799**	0.889	0.771	0.799
`simple_tcn_11`	0.900	0.773	0.878	0.744	0.778
`simple_tcn_22`	**0.917**	**0.799**	**0.903**	**0.789**	**0.823**
`simple_tcn_44`	0.907	0.765	0.887	0.767	0.798
`simple_tcn_skip_5`	0.907	0.772	0.890	**0.784**	0.810
`simple_tcn_skip_11`	**0.915**	**0.778**	**0.894**	0.771	0.801
`simple_tcn_skip_22`	0.907	0.767	0.890	0.775	0.807
`simple_tcn_skip_44`	0.909	0.773	0.893	**0.784**	**0.811**
`tcn_dp_5`	0.899	0.790	0.879	0.767	0.783
`tcn_dp_11`	0.885	0.804	0.863	0.740	0.763
`tcn_dp_22`	0.900	**0.805**	0.884	0.758	0.796
`tcn_dp_44`	**0.912**	**0.805**	**0.896**	**0.806**	**0.813**
`tcn_dp_skip_5`	0.905	0.751	0.890	0.771	0.799
`tcn_dp_skip_11`	0.918	0.770	0.900	0.784	0.809
`tcn_dp_skip_22`	**0.925**	**0.776**	**0.912**	**0.806**	**0.838**
`tcn_dp_skip_44`	0.909	0.764	0.895	0.775	0.806

4.5 Discussion and Conclusions

In this paper, we trained multiple beat tracking systems with slightly modified architectures on standard datasets and evaluated their performance. Using additional skip connections increased the metrics in most cases, except for `tcn_dp_44` and `simple_tcn_22` models. The well-known `bock_2020` system achieved the highest detection accuracy when trained on a 22.05 kHz audio input sampling rate, although its authors used 44.1 kHz. All networks except `tcn_dp` provided better results when trained on lower sampling rates. This may be thanks to, for example, redundant information in higher frequencies. In most music genres, the beat structure is defined by lower frequencies and specific pulsations. Even 5.5 kHz models show comparable performance, considering many instruments contain overtones and timbre above 2.5 kHz. It seems that 44.1 kHz might not be needed for the beat tracking task. For some applications, the lower input sampling rates may be beneficial, as most of the common music processing pipelines and extraction tools work with 22.05 kHz audio signals. In the future, we want to build on these experiments and release open-source models with different input sampling rates to provide more options for subsequent applications.

5 MPA-motivated Beat Tracking Evaluation

In this chapter, we present the conference article “The Application of Tempo Calculation for Musicological Purposes” [8] that focuses on comparing conventional and ML beat tracking systems for MPA-oriented tempo computation.

Beat tracking systems capture time positions of beats within digital recordings. They are used, for example, in streaming portals, but applications in musicological analysis are often neglected. In this article, two different methods of beat tracking systems are tested—conventional and state-of-the-art—on the specific motif of a string quartet music, which is one of the most complex tasks for beat detectors in general. The aim here is to determine which system is better for musicology purposes. This often involves determining not only the position of individual beats and estimating the tempo but also the accuracy of determining their number. Evaluation analysis may be suitable for comparing the accuracy of detectors, but may not necessarily reflect the requirements of musicological analysis. The results of selected detectors show that a system based on a recurrent neural network seems to be the most suitable.

5.1 Introduction

Beat tracking and rhythmic analysis are one of the key and most developed problems in the field of Music Information Retrieval (MIR). Algorithms determine the rhythmic or metric structure of a digital recording by specifically manipulating the audio signal and extracting valid information from it. Previously, the problem was grasped in various ways—the most successful algorithms were based mainly on the calculation of periodicity and the distribution of onsets over time. With the development and availability of artificial neural networks, virtually all MIR topics have been transformed and state-of-the-art algorithms have been replaced. Musicological analysis can deal, among other things, with the comparison of different interpretations or performances of the same composition. Here is the advantage of the MIR field, which provides the possibility of extracting the required information by machine, i.e. in much larger quantities, unified, faster, and perhaps even more objectively. The well-known cases of cooperation between MIR researchers and musicologists include [124] and [125]. In addition to the UK, Austria and Germany, for example, also participate in a cooperation [126]. Although these are two theoretically very different fields, they have many challenges in common. Estimation of rhythmic structure and tempo by automatic methods provides the possibility of complex musical analysis. But are conventional detection systems really suitable for musicological analysis?

5.2 Methods

The analysis consists of comparing and possibly modifying several beat tracking methods in an application for musicological analysis. This is specific for the type of music on which detection systems are usually not trained and tested and also for their focus on the same recordings, but different interpretations. As part of the development of new methods and the improvement of existing systems, for example at the MIREX competition [127], it is very rare to see testing exclusively on classical music or the same composition using different interpretations. The most commonly used datasets contain a large number of recordings of various genres and musical instruments. The system may then be able to generalize better, but its specialized application may fall behind.

5.2.1 Beat Tracking

Earlier detection systems used periodicity in the onset strength envelope by modifying the spectral difference or spectral flux. An example of an advanced system is the beat tracking included in the librosa module [91] for the Python language. Figure 5.1 describes the signal flow in a conventional system and a neural network system. The left part of the image corresponds to librosa processing. The system is hereinafter referred to as **lib**. The problem with the detector is generally the inability to adapt to a rapidly changing tempo. Abrupt changes in tempo and meter are not expected. In the basic setting, an average tempo of 120 beats per minute (BPM) is assumed, from which the adjustable tolerance is determined. Therefore, it is a question of to what extent systems with a similar architecture can adapt to the agogic and complex rhythmic structure of string quartets.

The second tested system is from the madmom module [82]. It uses a bidirectional recurrent neural network with Long Short-Term Memory (LSTM) cells, which are based on the determination of beat times according to a modified spectral envelope. The type of architecture is chosen deliberately here. By adding LSTM cells to the network, it is possible to store the time information that is theoretically necessary to determine the longer-term rhythmic structure, and thus the correct detection of musical beats. Finally, a Dynamic Bayesian Network (DBN) approximated by a Hidden Markov Model (HMM) to compute the probability of a beat within given frames (the neural network is trained on 100 frames per second resolution) is used. However, the question is how well the system can handle string quartets, as the network has not been trained for this style of music.

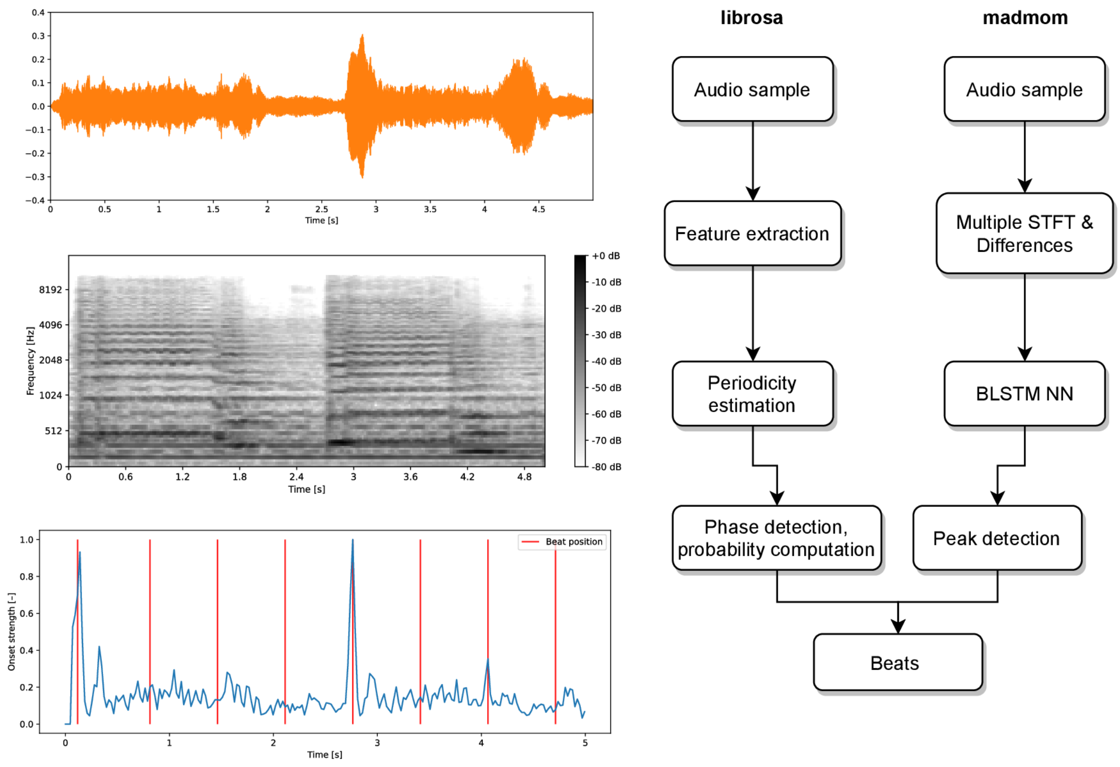


Fig. 5.1: Signal flow of two beat tracking systems.

5.2.2 Global Tempo

How can we compute the average tempo of recordings? Is it necessary to have a beat tracking system? For many applications of these algorithms (music recommendations, classification, genre recognition) within streaming portals, this is more or less the only available option. This procedure may not be necessary for musicological analysis. In a musicological analysis, we often work with one selected composition—then, the acquisition of recordings from different performers (i.e. different interpretations) takes place and the analysis is conducted. The advantage of comparing the same composition, but different musical performances, lies in the possibility of using music notation. Obtaining a notation of a given composition in the *.pdf* and *.xml* formats is typically not a problem (musicologists are usually familiar with notation software such as Sibelius or the open-source variant, MuseScore). Information about the number and absolute positions of the beat times (concerning the melodic and harmonic sequence) can be obtained from the notation. Thus, the global (average) tempo (GT) of a musical motif can simply be calculated as follows: $\mathrm{GT} = 60 \cdot N/d$, where $N =$ the number of beats and $d =$ the time duration of the motif. In this way, however, we do not obtain information about the time position of individual beats, but only about the average tempo of the analyzed section.

5.2.3 Dataset

For the analysis, a first motif (bars 4–10) of the *String Quartet No. 1 e minor "From My Life"* by Bedřich Smetana was used. Together with Prof. Spurny from Masaryk University, we have collected 33 different interpretations. This motif contains a total of 12 beats, which were manually annotated using Sonic Visualiser software to obtain ground truth data. In addition to GT, GTT (Ground Truth Tempo) is introduced. Manual ground truth annotations can be used to obtain GTT. First, the time difference d between successive beats was calculated, which was then averaged (ad) and the relationship GTT $= 60/\text{ad}$ was used. This value should theoretically be equal to GT, but it will be slightly different, as manual annotation and segmentation of digital recordings are not completely accurate. Accuracy can be determined by the difference between GT and GTT.

5.2.4 Approach

First, the motif was segmented from all available recordings of the database. GT, GTT, and their difference were calculated. Then, all recordings were analyzed by detection systems Lib and madmom. The parameters were first left in the default settings (**lib**), then other parameterizations were tested:

- **lib_t** (*t* stands for tuned): the original sampling frequency $f_s = 44100$ Hz was not changed and a different Mel spectrogram setting (138 mels), length of FFT (2048 samples), and hop factor (512 samples) was used.
- **lib_avgGT** (average Global Tempo): a specific parameter was selected that affects the expected start tempo (start_bpm). Thus, the system does not calculate the predominant tempo from the most frequently used value of 120 BPM but automatically calculates the position of the beats according to the most suitable candidate based on the selected tempo, including the adjustable tolerance (tightness).

The advantage of lib_avgGT is mainly the elimination of so-called octave tempos (120 BPM is the octave tempo of 60 BPM and so on). However, this parameter cannot be set separately in many cases, as we do not know the expected tempo. In a musicological analysis, however, we can know GT in advance, as we have recordings and sheet music available. Therefore, lib_avgGT uses the average GT of the entire tested database, 88 BPM. The last system is a detector using the previously mentioned bidirectional recurrent neural network.

5.3 Results

The results are attached in Table 5.1. All mentioned detectors were used for the musical motif, GT was calculated from the notation, and GTT from the ground truth annotation. The mean, median, and standard deviation were computed for all items. The absolute difference between GT and GTT is on average 0.318 BPM, which shows the high accuracy of manual segmentation and annotation of beat positions. The standard deviation, in this case, is 0.261. The lib and lib_t system show very inaccurate tempo detection, the average tempo value according to lib_avgGT is closest to GTT (the difference is 4.783 BPM, which can be considered a promising result). Madmom differs by 8.977 BPM.

Tab. 5.1: Tempo estimation of the string quartet motif.

track	GT	GTT	Δ	lib	lib_t	lib_avgGT	madmom
CD01	100.000	100.660	0.660	117.454	123.047	95.703	99.548
CD02	93.506	93.576	0.070	78.303	143.555	86.133	72.398
CD03	88.615	88.877	0.261	234.908	90.666	89.103	88.829
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
CD33	98.630	100.660	2.030	56.174	139.675	99.384	100.457
mean	88.129	88.447	0.778	130.943	119.298	**93.230**	79.470
median	87.681	88.539	0.487	93.994	120.185	**92.285**	80.932
std	7.592	7.853	0.751	79.510	13.101	**7.717**	15.721

Table 5.2 shows the mean and median number of beats determined for all tracks and systems. Here we can see the advantage of the madmom system. Even though the average tempo determined by lib_avgGT is closest to the reference, the median number of detected beats is 12 (11 for lib_avgGT). Besides, the number of tracks in which exactly 12 beats were detected is 25 for madmom and only 3 for lib_avgGT (out of a total of 33 recordings).

5.4 Discussion

Although lib_avgGT is the best system chosen according to the results of average tempo detection, the most suitable system for musicological analysis of the rhythmic structure and tempo estimation is madmom (RNN based). The accuracy of detectors is usually compared by F-score (F-measure) metrics, however, this is not the

Tab. 5.2: Number of beats detected for each track from the dataset.

track	lib	lib_t	lib_avgGT	madmom
CD01	13	14	11	12
CD02	7	18	10	9
CD03	28	12	11	12
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
CD33	5	18	11	12
mean	15.788	16.212	**11.636**	11.152
median	11	16	11	**12**
12 beats	1	2	3	**25**

most important factor for application in music analysis. Until the systems reach an F-score of about 0.95 and higher, a manual correction will always be needed for the analysis to be truly meaningful. The key factor here is the minimization of the researcher's time in correcting the actual time positions of the beats in the individual recordings. Of course, this article is limited to comparing only a few detectors, but librosa is considered a conventional system and madmom a state-of-the-art [127]. It is also a question of whether neural network-based systems should not be considered conventional today. Sonic Visualiser uses the BeatRoot system, which is similar to librosa but older and less accurate. Detectors based on neural networks are not accessible in this software, as they remain as testing tools in the development environment (Sonic Visualiser uses Vamp plugins), although they offer high potential for musicological analysis. Until they are implemented in a user-friendly environment, musicologists will not use them and their application will depend on the cooperation of musicologists and researchers in the field of MIR.

5.5 Conclusion

This article deals with the application of beat tracking systems for musicological analysis. It tests selected systems on recordings of string quartets, which are generally very challenging for detectors. The test dataset of recordings is well segmented and annotated, which is also confirmed by the additional calculation of the difference between GT and GTT. Madmom seems to be the most suitable system (of the selected detection systems), as it provides relatively high accuracy in determining the average tempo and also estimated the correct number of beats in 25 cases out of 33. The detector from the librosa module, despite the preset start tempo, was able to identify precisely 12 beats in only 3 cases. It should be noted that the system missed only once (i.e. detected 11 or 13 beats) in 21 cases. From a technical point of view, musicological analysis requires minimizing the time spent on manually editing the annotation and ground truth. Accurate determination of tempo, and time positions of beats, but also their number, is an important factor for testing the validity of detection systems for musicology analysis and their future use in analysis software.

6 Possibilities of Automated Annotation

This chapter is based on the conference article “Exploring the Possibilities of Automated Annotation of Classical Music with Abrupt Tempo Changes” [9].

In this paper, we introduce options for automatic measure detection based on synchronization, beat detection, and downbeat detection strategy. We evaluate the proposed methods on two motifs from the dataset of Leoš Janáček’s string quartet music. We use specific user-driven metrics to capture annotation efficiency simulating a scenario where a musicologist has to use the output of an automated system to create ground-truth annotations on given recordings. In the case of the first motif, synchronization outperformed other methods by detecting most of the measure positions correctly. This procedure was also the most suitable for the second motif—it achieved a low number of correct detections, but the vast majority of transferred time positions belonged within the outer tolerance window. Therefore, in most cases, only shifting operations were needed resulting in higher annotation efficiency. Results suggest that state-of-the-art downbeat tracking is not yet efficient for expressive music.

6.1 Introduction

Music Information Retrieval (MIR) is a well-established interdisciplinary area that combines technical approaches and methods with musical analysis. The MIR researchers deal with many music-driven tasks, such as automatic detection of musical features and high-level parameters, user-centric semantic retrieval, recommendation systems, or transcription of audio recordings into symbolic representations [128]. In this paper, we focus on the automatic identification or detection of measure (bar) positions in string quartet recordings, which is closely related to the challenges of Musical Performance Analysis (MPA).

Measures are musically meaningful segments with defined metric patterns. Regarding Western music notation, information about their exact position in a given musical hierarchy is automatically encoded in the corresponding score (sheet music). To obtain measure positions in structurally complex music such as string quartets, one needs to have a score available. Manual labeling and annotation is a time-consuming procedure but it is a common approach to obtaining ground-truth data. However, recent developments in machine learning methods may change this workflow.

One of the most established topics in MIR is beat tracking or beat detection[1]. A standard beat tracking system outputs a vector of time positions that correspond to individual beats in a given music recording. In our case, we want to obtain only the first beat of each measure—such detectors do not usually distinguish the beat index within measures. Therefore, downbeat tracking systems have been developed which, together with the time position of beats[2], also estimate their position in a musical structure. The second option is a strategy based on a synchronization procedure. The general goal of music synchronization is to establish an alignment between musically corresponding time positions (measures, in this case) of the same piece (e.g., audio-to-score or audio-to-audio alignment) [129].

In this paper, we focus on computer-generated annotations and test the state-of-the-art offline beat and downbeat tracking for measure detection on chamber music. We compare the detectors with the music synchronization technique and evaluate all methods by a user-driven metric.

6.2 Methods

6.2.1 Dataset

First, we introduce our dataset, which consists of two separate motifs from Janáček's *String Quartet No. 1 "Kreutzer Sonata", JW 7 No. 8* and *String Quartet No. 2 "Intimate Letters", JW 7 No. 13*, respectively. Figure 6.1 shows a score for the first motif. This motif contains 11 measures of the first movement. At the beginning, all strings except violoncello play *con sordino*[3] and the second violin uses *finger tremolo*[4] technique which may blur the starting point of the second bar. Furthermore, the overall dynamics is higher for the upbeat than for the downbeat. After *subito forte* (suddenly loud), the tempo changes rapidly with possible local deviations based on individual interpretation. The second motif contains 10 measures of the second movement. It is even more complex within the metric structure with many accents in the middle of measures. We selected these excerpts for their various tempo, challenging structure, and expressive nature. We gathered 17 different interpretations for each motif and carefully annotated all ground-truth measure positions. In our experiments, the first recording from both motifs by the Belcea Quartet (year of recording 2018) was selected as a reference. The remaining recordings were used for testing purposes.

[1]In the context of this paper, we use the terms beat tracking and beat detection interchangeably.
[2]The system outputs the probability of beats and downbeats separately.
[3]A technique that uses a "mute pad" to soften the produced sound.
[4]The player uses fingers to alternate rapidly between two notes.

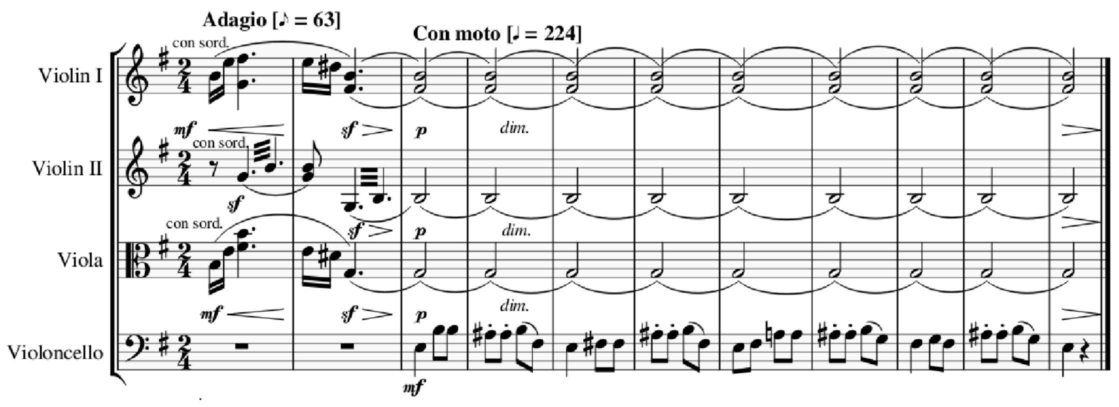


Fig. 6.1: The score for the first motif of our dataset.

6.2.2 Beat and Downbeat Detection

Beat tracking systems provide time positions of computed beats for any given music recording. In the case of neural network-based approaches, their output is usually an activation function—its value within a specified feature rate is related to the novelty function or confidence of beat occurrence. Then, peak-picking methods or probabilistic and statistical methods, such as conditional random fields or Dynamic Bayesian Networks (DBN), are often used.

In this paper, we use a beat detector based on the variant of Recurrent Neural Network (RNN) [111] in combination with DBN [106] and a downbeat detector also based on RNN [116] and DBN but with different settings and functionality. We kept the default settings for the DBN with a range of possible tempo detection between 55 and 215 BPM (beats per minute). This system will demonstrate the problematic part of beat tracking when applied to expressive chamber music.

The DBN estimator of downbeats outputs two vectors of data. The first one contains time positions of beats and the second their index within a measure—e.g., output vector B = [2.5, 3] shows the third beat of a measure in the time of 2.5 s. Thus, we have selected only those beats that correspond to the first position of each measure creating a downbeat sequence. Ideally, the output of this modified detector should produce only the beginning of each measure and follow the ground-truth data structure. We also added the prior knowledge (2 or 3 beats per bar) about the metric structure of selected motifs into the detector.

6.2.3 Synchronization Method

The second option to obtain time positions of measures is a synchronization procedure. This is a common approach in MPA due to its advantages. In our experiment, we use an alignment method based on a variant of Dynamic Time Warping

(DTW), called Memory-restricted Multiscale DTW (MrMsDTW) that is faster and may provide a better synchronization accuracy [45]. First, we compute chroma energy normalized statistic (CENS) features [130] of reference and target recording with a resolution of 50 features per second. The MrMsDTW is applied to compute a cost-minimizing alignment between both CENS matrices and the resulting warping path is limited to be strictly monotonic by postprocessing. The ground-truth annotations are then transferred from the reference to the target recording by the resulting warping path.

This strategy has an advantage over the automated detectors—there will be always the right number of measures detected. The question is whether chroma features contain enough information for alignment to work accurately e.g. in music structures, where there is almost no new information present, but measure number increases.

6.2.4 User-driven Metric

Each machine annotation of musical content usually ends with a certain number of mislabeled time positions. Either the desired time point may not appear in the machine annotation at all, or it is misplaced. In [131], the authors introduced the *annotation efficiency* (*ae*) metric, which is based on how much effort a user has to exert to manually correct detections by shifting, deleting, or inserting time positions. The insert and delete operations correspond to the counts of false negatives and false positives, respectively. The shifting should theoretically be counted twice, once as a false positive and the second time as a false negative. In practice, however, it is more sensible to count this operation separately and prioritize it over deletion and insertion, since it is the most common correction performed by the user.

The process of calculating the *ae* metric is as follows. First, an inner tolerance window of ± 70 ms is created around each ground truth annotation. Then, the true positives (unique detections), t^+, are counted. Detections that match ground truth annotations are removed from further calculations and incorrect detections are marked as candidates to be shifted or removed. For each remaining annotation, an outer tolerance window of ± 1 s is then created to search for the closest detection that does not match the ground truth. If there is a detection in this window, it is marked as a shift. After the analysis of unaccounted detections, the number of shifts s is calculated. The remaining annotations correspond to false negatives, f^-, with leftover detections marked for deletion and counted as false positives, f^+. The *ae* metric is defined by the following equation:

$$ae = t^+/(t^+ + s + f^+ + f^-). \tag{6.1}$$

6.3 Results

First, we transferred the ground-truth annotations based on the DTW alignment method, then calculated beats and downbeats as described in section 6.2. Figure 6.2 shows the user-driven metric computation and the pipeline with all possible operations for one of the recordings. Operations are marked with different colors to increase readability. We kept the same inner and outer tolerance window as the original beat tracking evaluation in [131].

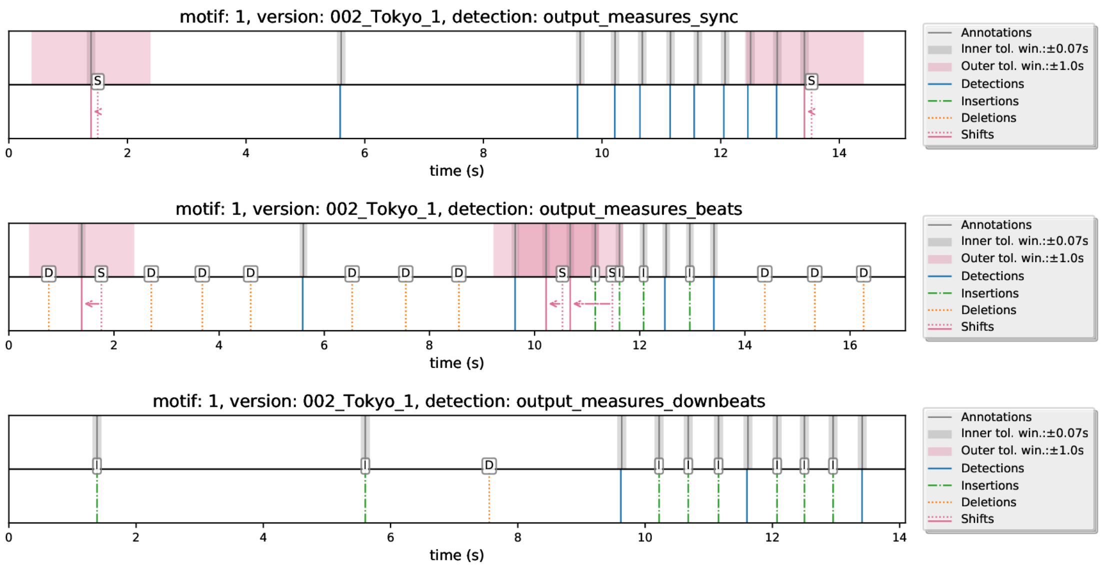


Fig. 6.2: The user-driven metric for synchronization, beat tracking, and downbeat tracking strategies; evaluation of the Tokyo Quartet recording, first motif.

In this case, the synchronization procedure outperforms all other methods. The final synchronized positions are not in the exact time positions, however, they mostly fit into the inner tolerance window. The value of beat confidence for the downbeat tracker was in the first motif too low—measures that have an ambiguous nature were not detected at all and measures of a faster-paced segment with an abrupt change of rhythmic structure were partially omitted. On the other hand, the RNN beat tracker detected many false positives. The DBN method fills the positions between confident output beats based on their past and future occurrence—this method can work well with small deviations of tempo but fails when the rhythmic and metric structure is unpredictable and highly changing.

Table 6.1 shows the sum of all operations and annotation efficiency, recall, and precision for both motifs and each method. Synchronization outperformed other methods for the first motif with 142 correct detections and only 36 additional operations. In the second scenario, however, the beat tracking captured the highest number of correct measure positions. Although the synchronization method achieved the lowest number of all corrections and the best annotation accuracy, recall and precision remained low. Recall and precision scores may give the impression that beat and downbeat detection are more suitable tools for the automatic detection of measure positions in a complex structure, but the number of deletion operations reveals that they are, in fact, counterproductive in this scenario. None of the proposed methods was successful considering only the second motif.

motif 1 (176 measures in total)								
method	$\sum$ det	$\sum$ ins	$\sum$del	$\sum$ shf	$\sum$ ops	*ae*	*R*	*P*
beat t.	70	42	167	64	273	0.208	0.375	0.225
downbeat t.	33	116	35	27	178	0.155	0.176	0.329
sync	**142**	2	2	32	**36**	**0.799**	**0.727**	**0.727**
motif 2 (160 measures in total)								
method	$\sum$ det	$\sum$ ins	$\sum$del	$\sum$ shf	$\sum$ ops	*ae*	*R*	*P*
beat t.	**67**	0	516	93	609	0.101	**0.356**	0.087
downbeat t.	38	61	54	61	176	0.196	0.194	**0.219**
sync	38	11	11	111	**133**	**0.224**	0.156	0.156

Tab. 6.1: The number of detections, insertions, deletions, shifts, and total corrections, annotation efficiency, recall, and precision for each motif and method.

6.4 Discussion

The synchronization procedure, even if it always detects the correct number of measures, relies only on chroma features, their resolution, and DTW accuracy. The ground-truth annotations may not be always precise—the resulting warping path can transfer reference time positions with some deviations. It depends, e.g., on the harmonic structure, occurrence of onsets, or the ADSR envelope of given instruments. If we tolerate larger deviance (such as 100 ms), almost all annotations will be transferred correctly.

The beat detector has shown an experimental role in illustrating the function of predicting the rhythmic structure and beat occurrence. In the second motif, it achieved the best recall and number of correct measure positions. However, it also detected too many false positives; that would be true even if ground-truth annotations were based on beat positions. The method of filling in beats, even in places where there is no underlying information, can work well in simpler musical structures without significant changes in rhythm and meter. Furthermore, detectors are usually trained on specific audio datasets, for which there are manual ground-truth annotations available—string quartet music is not among them.

The downbeat detector was not sensitive enough or predicted false beat indexes, although it contained prior knowledge about the musical structure (see section 6.2.2). Table 6.1 shows that so far, the only valid option for expressive string quartet music with many abrupt tempo changes, local tempo deviations, and weak onset and beat positions, is the synchronization strategy. Its accuracy can be improved by the choice of additional features for the alignment procedure. In this case, however, the ground-truth annotations are always needed.

6.5 Conclusion

In this contribution, we proposed and evaluated different methods of obtaining measure positions in string quartet music. We first created reference ground-truth data and then compared the synchronization method, beat tracking, and downbeat tracking based on a specific user-driven metric. This metric allows us to calculate the number of operations that one needs to make to obtain the ground-truth annotation of measure positions. We tested different strategies on two carefully selected string quartet motifs from Leoš Janáček's compositions. Both proposed segments are musically challenging, they contain many weak onset positions, ambiguous beats, and abrupt tempo and rhythm changes. Results suggest that the synchronization method is superior to all other possible options. Beat and downbeat tracking approaches are not yet efficient on very expressive pieces of classical music.

Part II: Analysis of Performance Differences

7 Feature Extraction Pipeline for MPA

This chapter is based on the conference article “Towards Automatic Measure-Wise Feature Extraction Pipeline for Music Performance Analysis” [10].

The task of obtaining ground-truth annotations is of fundamental importance for Music Performance Analysis. Measure positions could be used to navigate throughout the piece, indicate the tempo changes, or help with structure segmentation. In this paper, we introduce an automatic measure-wise feature extraction pipeline. We first annotate one interpretation of the string quartet music and use an audio synchronization strategy to transfer measure positions to all other recordings. We extract features related to tempo, dynamics, and timbre. We compute average values in each measure and propose measure-wise feature matrices. This procedure could be used for any number of recordings as long as at least one reference annotation is available. Finally, we create a binary label for each interpretation based on the Czech origin of performers as an experiment and evaluate the measure-wise tempo distribution.

7.1 Introduction

In Music Information Retrieval (MIR), there are still many challenging tasks. Methods of MIR have a significant impact on the interdisciplinary field of Music Performance Analysis (MPA) [24]. MPA research focuses, e.g., on the characteristics of interpretations and their differences. The goal of analysis always depends on the context. For example, beat detection is a well-known MIR challenge that arose originally from musicology. In the context of both MIR and MPA fields, one wants to obtain the time positions of beats within given music recordings. However, in MIR, beat detectors are usually evaluated by many metrics [78] to capture different aspects of the output system’s accuracy. On the other hand, for MPA researchers, the goal is to obtain all correct beat positions with 100% accuracy. Music experts and musicologists can use automated systems to create “candidates” that may or may not be beats. They need to edit these candidates manually in specialized software such as Sonic Visualiser [108]. Even a small error may impact the analysis. Higher F-score or metrical level-based metrics do not necessarily indicate a more suitable detector for performance analysis. There are, however, some exceptions, e.g., user-driven metric proposed in [131].

In this paper, we combine MIR techniques to obtain meaningful data for MPA purposes. We focus on the automatic extraction of features for further music performance analysis. Our goal is to automatically obtain specific temporal information and extract features in a measure-wise fashion. This is possible even for string quartets, thanks to the relatively high synchronization accuracy. We follow a specific pipeline for segment separation based on the annotation of a reference recording, audio-to-audio synchronization, and specific feature extraction. In the future, machine learning methods could be used in combination with proposed feature matrices for classification tasks (such as the origin of performers). However, measure positions would be needed in advance. In our experiments, we use the *String Quartet No. 12 in F major, Op. 96*, by Antonín Dvořák, to demonstrate the pipeline and possible evaluation for further music analysis. The rest of the paper is organized as follows: Section 7.2 describes the experiments and methods of annotation, audio-to-audio synchronization, and feature extraction. The results are presented in Section 7.3, followed by a conclusion in Section 7.4.

7.2 Experiments and Methods

7.2.1 Dataset

To demonstrate the proposed feature extraction pipeline and evaluation, we gathered 76 different recordings (interpretations) of the *String Quartet No. 12 in F major, Op. 96, 3rd movement*, composed by Antonín Dvořák. The sum of the duration of all interpretations for the 3rd movement is given in Table 7.1. The total duration of all recordings combined is roughly 5 hours. We also denote the Czech performers as label 1 and the rest as label 0 for further evaluation.

Tab. 7.1: Recording identifier, duration of each recording, and a label for Czech performers (label 1) and all others (label 0).

rec ID	duration [s]	label 1	label 0
001	225.57	False	True
002	216.93	False	True
003	248.34	True	False
...	...	...	...
076	249.12	False	True
total	17746.10	18	58

7.2.2 Annotation

In our experiments, we wanted to extract reliable features from all interpretations to evaluate all interpretations or compare their differences. We first used beat tracking and downbeat tracking approaches to find the measure (downbeat) positions, but the results were not sufficient. We manually annotated measure positions of one interpretation (chosen as a reference recording) to obtain ground-truth (GT) data. Time positions of measures may provide sufficient resolution and useful information for further evaluation [129]. They are also used in many MIR tasks, music analysis, or music navigation. If the goal of evaluation or required time resolution changes, one can annotate, e.g., beats instead.

One of the problems of MPA is the small number of recordings in datasets and thus the lack of generalizability [24]. Sometimes, only a few recordings are analyzed—manual labeling on a larger scale is very time-consuming and tedious. In our case, we annotate only one recording and automatically acquire annotations for all other recordings based on the audio-to-audio synchronization strategy. This way, we can use a large number of recordings. However, the chosen method for obtaining relevant data should always depend on the goal and context of the performance analysis.

7.2.3 Synchronization

To obtain measure positions for each interpretation, we resample all recordings to 22 050 Hz and compute the time alignment of the reference and all target recordings following the sync-toolbox pipeline[1] [56]. First, a variant of chroma vectors, also known as Chroma Energy Normalized Statistics (CENS) features [130], is computed. The tuning is estimated to shift features accordingly, and the Memory-restricted Multiscale DTW algorithm (MrMsDTW) [45] is applied to find the optimal alignment between both recordings. If the synchronization procedure is surpassed in the future, it will be possible to change or adjust the method and obtain more precise results. Measure positions are then transferred from the reference to the target recording based on the warping path.

A common problem when dealing with performance differences and analysis is the music structure ambiguity. In this step, we can automatically detect structure differences. If the reference and target recording have the same music structure and harmonic progression, the warping path would be more or less diagonal. Figure 7.1 shows the warping path of MrMsDTW for an exemplary case when the music structure differs. We can see the horizontal path indicating a repetition in the target

[1]Available: `https://github.com/meinardmueller/synctoolbox` (accessed on 21 April 2022)

recording. We can automatically discard such cases by calculating a difference between all synchronized measure positions (or the slope of a warping path). If the difference is larger or smaller than a certain threshold, we remove this recording from the feature extraction. In our experiments, we used two thresholds: $\tau_1 > 12$ s checks if there is an extra repetition at the beginning or inside the recording and $\tau_2 < 0.1$ s if there is a part missing. Considering the 3rd movement, however, all obtained recordings followed the same structure.

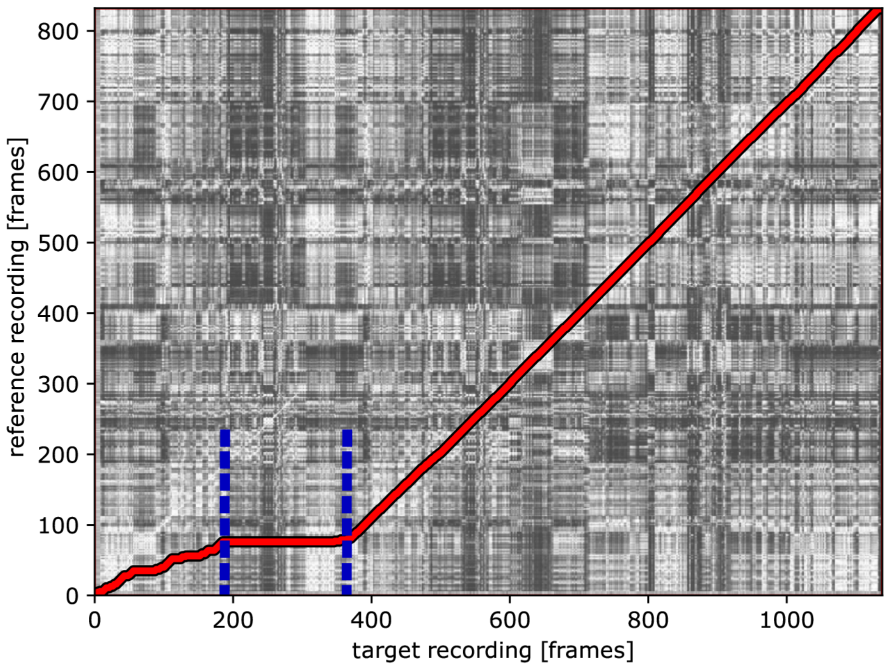


Fig. 7.1: An example of the cost matrix and alignment path of the reference and target recording. The horizontal path marked with blue dashed lines indicates repetition within the target recording.

The alignment method cannot, so far, replace the manual annotation process when dealing with time-precise annotations. The methods to obtain relevant data always depend on the goal of analysis, and they should be chosen accordingly. To demonstrate the synchronization accuracy, we manually annotated the second reference recording and computed the mean d_{mean} and median d_{med} difference between GT positions of the second reference and the transferred measure positions from the first reference recording (see Table 7.2). We found out that $d_{\text{mean}} = 40$ ms and $d_{\text{med}} = 19$ ms. The chosen alignment strategy provides relatively accurate results if our goal is a measure-wise resolution (the median duration of measures in the reference recordings equals 754 and 791 ms, respectively).

In the proposed pipeline, we do not extract features based on the warping path itself, e.g., as in [132]. We use the synchronized measure positions to divide audio recordings into small parts (that correspond to the duration of measures) and compute the features of each part separately.

Tab. 7.2: Two reference recordings used in the synchronization pipeline; mean and median duration of all GT measures; mean and median differences between GT and synchronized measures for both reference scenarios.

rec	mean dur	med dur	mean & med diff	mean & med diff
ref 1	832 ms	754 ms	reference	28 ms; 18 ms
ref 2	908 ms	791 ms	40 ms; 19 ms	reference

7.2.4 Features

Synchronized measure positions give us an overall time grid for feature extraction. They also segment recordings into logical musical structures that could be later selected and used for the evaluation. The music performance parameters can be divided into a few basic categories [24]:

- dynamics – how the loudness varies based on phrasing, accents, tension, or musical structure;
- timing (tempo) – rhythmic structure, micro-timing (onsets or beats), global tempo, or local tempo deviations;
- timbre – choice of instrumentation, instruments, playing techniques, and acoustic conditions;
- pitch – intonation, deviations from the score, and choice of playing techniques such as vibrato.

Most of the parameters cannot be unconditionally connected to the direct semantic level. For example, timbre is a very ambiguous parameter if the context, acoustic conditions, recording and encoding choices, or post-processing options are not taken into consideration. We first normalized all recordings to -26 LUFS (Loudness Unit Full Scale)[2] so that the overall dynamics could provide useful information when comparing extracted features of each measure. We selected three parameters to cover tempo, loudness, and timbre to some degree. We computed signal length (duration), Root Mean Square (RMS) value, and log mel spectrogram in a measure-wise fashion.

[2]The LUFS are described in EBU R 128 recommendation. Available: `https://tech.ebu.ch/docs/r/r128.pdf` (accessed on 21 April 2022)

To obtain tempo information, the source audio signal (each measure) is first converted to a sampling rate of 22 050 Hz. We derive tempo T simply as:

$$T = \frac{60 \cdot B}{d}, \tag{7.1}$$

where d is the duration of a given measure and $B = 3$. The whole 3rd movement is in $\frac{3}{4}$ time signature. Next, segmentation into frames of 2048 samples with a hop size of 512 samples is performed. We calculate the RMS value for each frame x according to the equation:

$$x_{\mathrm{RMS}} = \sqrt{\frac{1}{N} \sum_{n=0}^{N-1} x(n)^2}, \tag{7.2}$$

where N is the length of the audio frame in samples and n is the sample index. In the case of the log mel spectrogram, each signal frame (with the same parameters as for RMS) is first multiplied by a Hann window, and the real part of the complex spectrum is obtained using the Short-Time Fourier Transform. We chose to limit the frequencies: 65 Hz – 8 kHz (65 Hz roughly corresponds to the lowest tone C2 produced by a violoncello). Then, the power spectrum is calculated as a squared magnitude of the complex coefficients. The power spectrum is further converted into a mel-scaled power spectrum using a mel-filterbank of 64 filters. The number of bands is inspired by audio pattern recognition evaluation [133]. The final step is to convert the magnitudes to decibels by:

$$S_{\mathrm{dB}} = 10 \log_{10}(S), \tag{7.3}$$

where S is the mel-scaled power spectrum. All values were averaged within each measure. The final feature matrix structure for the reference recording is shown in Table 7.3.

Tab. 7.3: The feature matrix of the reference recording: measure index, duration in seconds, mean values of RMS, and mean magnitude of 64 mel bands in decibels.

measure	1	2	3	...	244
duration	0.830	0.647	0.824	...	3.784
rms	0.026	0.023	0.035	...	0.011
melbin1	−28.792	−23.068	−27.275	...	−16.921
melbin2	−25.918	−25.502	−23.861	...	−5.682
melbin3	−21.770	−24.522	−20.031	...	−5.270
...	...	...	...	...	...
melbin64	−52.002	−48.377	−47.645	...	−60.313

The matrix can be easily enhanced by a different choice of features; however, the number of measures always corresponds to the reference. In the last step, we created a binary label for each interpretation based on the available information about the origin of performers to demonstrate the evaluation (see Table 7.1). Although the labels are not distributed equally in the dataset (18 vs. 58), the motivation here is that Czech string quartets may play the music of a Czech composer slightly differently based on musical traditions and cultural predispositions. We do, however, understand that this categorization may be questionable.

7.3 Results

Figure 7.2 shows the mean tempo and RMS values for each measure. All 76 feature matrices are averaged into one representation. The red dotted lines in Figure 7.2a specify the negative peaks of the curve.

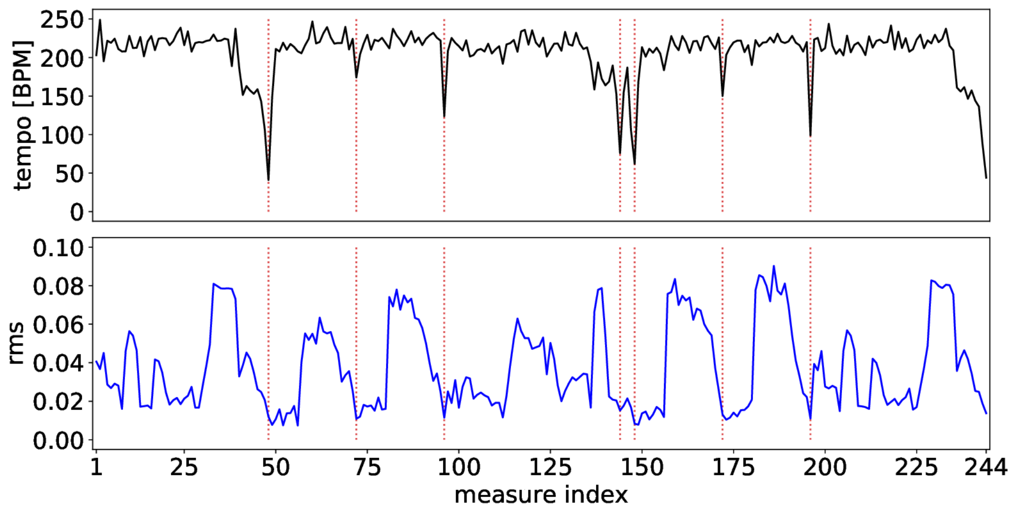


Fig. 7.2: Mean tempo curve and mean values of RMS for all recordings. The red dots indicate troughs or negative peaks of the tempo curve.

The overall tempo progression depends on the underlying musical structure. When there is a key or motif change, it is indicated by a *ritardando* (slowing down). These regions correspond to the red dotted lines (measures 48, 72, 96, 144, 148, 172, and 196). The RMS curve seems to follow this pattern with additional peaks throughout the piece. Figure 7.3 shows the final log filtered mel spectrogram for mean values of all feature matrices. This way, we can visualize average spectrum values for each measure within the whole dataset. This spectrogram could be further used as a tool for statistical timbre evaluation as it contains information from all available interpretations combined.

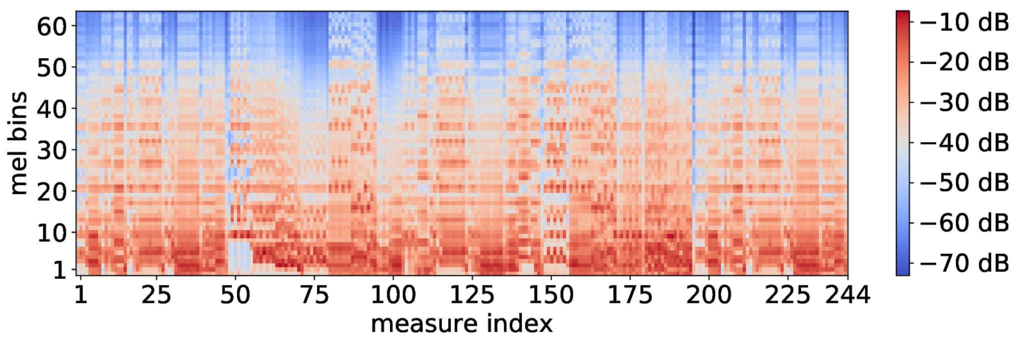


Fig. 7.3: The log filtered mel spectrogram: the x-axis indicates measures; the magnitude of each mel bin is converted to dB and averaged throughout the whole dataset.

Furthermore, the absolute tempo difference for each measure between Czech and other performances is given in Figure 7.4a. Values seem to be shifted toward positive numbers—positive difference means that Czech performers play the same segment at a higher tempo. On the other hand, the most prominent peaks of this curve correspond to negative values (measures 72, 146, 172, and 195). These positions also align with regions of the key or motif changes. Figure 7.4b shows the same scenario, but the labels were chosen randomly. Here, the range of deviations on the y-axis is much smaller, and the values seem to follow the normal distribution.

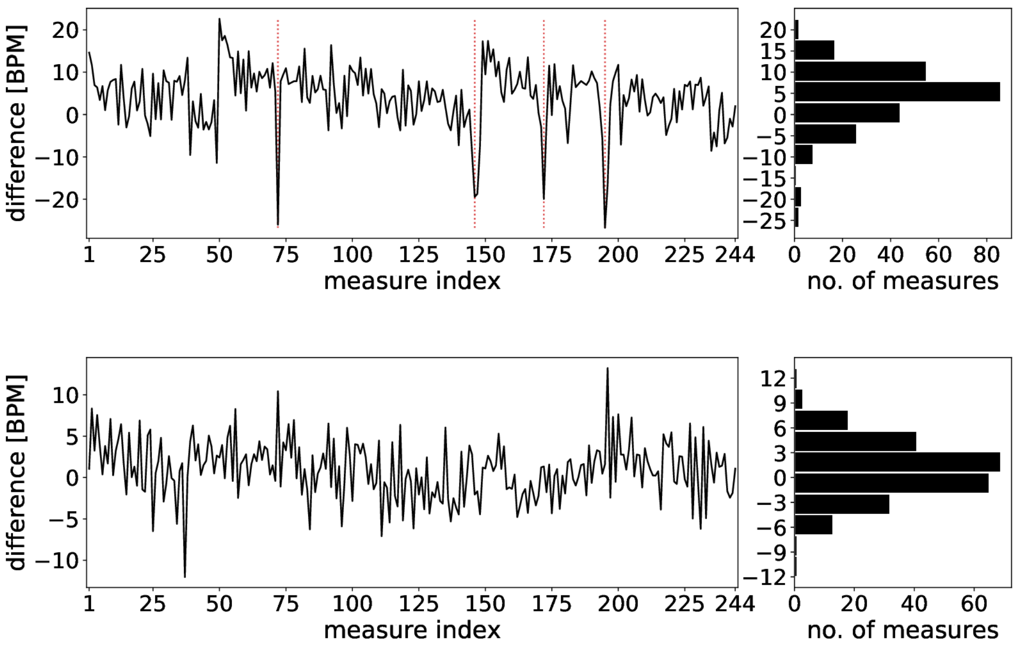


Fig. 7.4: From top to bottom: The absolute tempo difference between the Czech and non-Czech labels for each measure and a corresponding histogram; the same scenario but both labels were chosen randomly.

7.4 Conclusion

In this work, we have presented a specific pipeline of feature extraction for MPA purposes. The synchronized time positions based on reference measure annotations are first used to physically segment recordings. Then, the mean value of features is calculated measure-wise. We can use this method for a dataset of arbitrary length as long as the synchronization procedure is accurate relative to the nature of GT annotations. We gathered 76 interpretations of the 3rd movement of the *String Quartet No. 12* by Antonín Dvořák, extracted proposed feature matrices, and evaluated them based on the origin of performers. Results of experiments suggest that, in this case, there may be a difference between Czech and other interpretations. It seems that they tend to play faster but slow down more when there is a change of key or musical structure. In the future, we plan to use the proposed feature matrices for a specific MPA task—binary classification of the origin of performers using machine learning methods.

8 Classification of Interpretation Differences

This chapter is based on the journal article “Classification of Interpretation Differences in String Quartets Based on the Origin of Performers” [11].

Music Information Retrieval aims at extracting relevant features from music material, while Music Performance Analysis uses these features to perform semi-automated music analysis. Examples of interdisciplinary cooperation are, e.g., various classification tasks—from recognizing specific performances, musical structures, and composers to identifying music genres. However, some classification problems have not been addressed yet. In this paper, we focus on classifying string quartet music interpretations based on the origin of performers. Our dataset consists of string quartets from composers A. Dvořák, L. Janáček, and B. Smetana. After transferring timing information from reference recordings to all target recordings, we apply feature selection methods to rank the significance of features. As the main contribution, we show that there are indeed origin-based tempo differences, distinguishable by measure durations, by which performances may be identified. Furthermore, we train a machine learning classifier to predict the performers’ origin. We evaluate three different experimental scenarios and achieve higher classification accuracy compared to the baseline using synchronized measure positions.

8.1 Introduction

Music Information Retrieval (MIR) deals with extracting, processing, and organizing meaningful features from music material [128]. From the analysis of audio signals to symbolic representations and musical blueprints (score), MIR focuses on many challenging tasks such as content-based search, music tagging, automatic transcription, feature detection, music recommendation, and much more [30]. MIR methods significantly impact the Music Performance Analysis (MPA) field [24], providing more accurate detectors and possibilities for automated music analysis. In the case of classical music, a performance affects how listeners perceive a piece of music. Each interpretation may be special thanks to, e.g., modifying information from the score and converting various musical ideas into musical renditions [128]. The communication between members of ensembles also shapes a performance [134, 135, 136]. Classification tasks, such as the classification of music genres [137, 138], mood [139], music structures [140], or composers [141, 142], are examples of interdisciplinary approaches—a combination of MIR techniques with MPA, musicology, and music analysis.

Music-related classification problems are common [143, 138], but only the minimum deals with the classification of origin-based or music school-related differences of interpretations. In this paper, we combine MIR techniques with MPA goals. We focus on identifying the differences between interpretations of the same musical composition. In other words, we aim to create a classifier that could differentiate music performances based on the origin of a given composer. If it is possible to train a classifier, we can conclude that there are noticeable differences. To our best knowledge, there is only one study [144] with a similar goal using machine learning besides studies with a phylogenetic approach [140] or comparative music analysis [98, 145, 146]. However, there are many studies combining MIR and MPA disciplines and focusing on expressive performance [147, 148]. Machine learning models have been researched in the MPA community for, e.g., modeling nuances of dynamics and timing of expressive performances using inter-onset-intervals with Hidden Markov Models [149] and linear regression [150], or score following tasks [151]. Many other approaches to computational modeling of expressive performance also include neural networks [150, 152, 153].

We focus on string quartet music from Czech composers Antonín Dvořák, Leoš Janáček, and Bedřich Smetana. First, we collect a large dataset (compared to the average size of MPA datasets, see [24]) and label each recording to create two classes: Czech and non-Czech interpretations. The underlying hypothesis is that the Czech performers may play the piece differently, e.g., considering the same cultural background and tradition shared with the composers of the analyzed music. We can address this problem quantitatively thanks to the increasing number of available recordings and the accuracy of synchronization methods [56, 154]. We extract relevant timing-related features from all interpretations that may cover information about the expressiveness of a given performance and construct feature matrices to train and test a machine learning classifier. Figure 8.1 shows the overview of our classification approach.

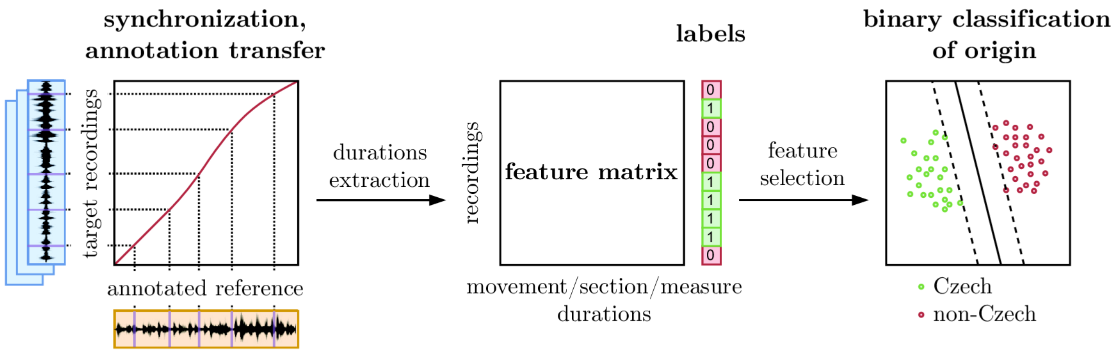


Fig. 8.1: Overview of the proposed classification strategy.

As this paper's main contribution, we show a general trend in the rhythmic conception (duratas) of given string quartets based on the proposed binary classes. Although various music schools, cultures, and traditions influence musicians, we can train a classifier to identify Czech and non-Czech interpretations of given string quartets with relatively high accuracy in most cases. To better understand the features and classification results (and why it is possible to train such a classifier), we split our experiments into three scenarios, each applying a different time resolution of features. Unlike the approach in [144], we use various string quartets, more recordings, and extract features based on a semi-automated approach instead of relying on automated systems with a possibility of significant misdetection. Furthermore, we support our feature selection with MPA principles (see Section 8.3.1) to focus only on timing parameters that may show the expressiveness of music performances (third scenario). We can achieve high classification accuracy if the selected features, derived from ground-truth (GT) data and a synchronization strategy, capture local tempo deviations. We understand the controversial nature of defining the "origin" of musicians and splitting our dataset into two binary classes; however, we want to show that the interpretation differences may be significant when using a machine learning method, even though they wouldn't be qualitatively noticeable by music experts. We do not claim that a difference in interpretation has any quality to it—we only show that there is a difference. To provide additional data, we share a GitHub repository[1].

The rest of the paper is organized as follows. Section 8.2 introduces the string quartet dataset, annotation, labeling process, and audio-to-audio synchronization and compares automated and semi-automated approaches for measure detection. Section 8.3 describes a feature selection, visualization method, dimensionality reduction, and design of experiments. The results are reported in Section 8.4, followed by a discussion in Section 8.5 and conclusions with prospects for future work in Section 8.6.

8.2 Methods

This section introduces our string quartet dataset, annotation process, and audio-to-audio synchronization strategy to obtain transferred measure positions. We show the validity of synchronization accuracy by comparing the automated downbeat tracking systems with the semi-automated synchronization procedure.

[1]github.com/xistva02/Classification-of-interpretation-differences (accessed on 10 March 2023)

8.2.1 Dataset

We collected string quartets of Antonín Dvořák, Leoš Janáček, and Bedřich Smetana from various sources, such as the Naxos Music Library, the Czech Museum of Music, and Masaryk University. Each composition is divided into four movements—in the following text, each movement is regarded as a separate recording. The composers, compositions, and movements (Roman numerals) are divided as follows.

- Antonín Dvořák:
  - String Quartet No. 12 in F major, Op. 96
    - I. *Allegro ma non troppo*
    - II. *Lento*
    - III. *Molto vivace*
    - IV. *Vivace ma non troppo*
  - String Quartet No. 13 in G major, Op. 106
    - I. *Allegro moderato*
    - II. *Adagio ma non troppo*
    - III. *Molto vivace*
    - IV. *Andante sostenuto*
  - String Quartet No. 14 in A♭ major, Op. 105
    - I. *Adagio ma non troppo*
    - II. *Molto vivace*
    - III. *Lento e molto cantabile*
    - IV. *Allegro non tanto*
- Leoš Janáček:
  - String Quartet No. 1, "Kreutzer Sonata", JW 7/8
    - I. *Adagio con moto*
    - II. *Con moto*
    - III. *Con moto – Vivace – Andante – Tempo I*
    - IV. *Con moto*
  - String Quartet No. 2, "Intimate Letters", JW 7/13
    - I. *Andante*
    - II. *Adagio*
    - III. *Moderato*
    - IV. *Allegro*
- Bedřich Smetana:
  - String Quartet No. 1 in E minor, "From My Life", JB 1:105
    - I. *Allegro vivo appassionato*
    - II. *Allegro moderato à la Polka*
    - III. *Largo sostenuto*

IV. *Vivace*

– String Quartet No. 2 in D minor, JB 1:124

I. *Allegro*

II. *Allegro moderato*

III. *Allegro non più moderato, ma agitato e con fuoco*

IV. *Presto*

For more details about compositions, we refer to International Music Score Library Project (IMSLP)[2]. Most versions are studio recordings, but we also keep the live versions. Table 8.1 shows the composers, musical compositions, the number of recordings, binary labels (classes) of the performer's origin (1 refers to the Czech class and 0 to the non-Czech class), and the total duration of all interpretations of the given composition combined. Our dataset consists of 1315 string quartet recordings with a total duration of roughly six days. We focused on the well-known string quartets of Czech composers, increasing the probability of gathering enough data for the proposed analysis.

Tab. 8.1: The original dataset of string quartets from Czech composers; composer (csr), composition (com), the number of different interpretations (recs), class 1 (Czech interpretation), class 0 (non-Czech interpretation), and total duration (dur) of all recordings in hh:mm:ss or dd:hh:mm:ss format.

csr		Dvořák		Janáček		Smetana		
com	No. 12	No. 13	No. 14	No. 1*	No .2	No. 1	No. 2	Σ
recs	304	100	92	264	280	171	104	1315
class 1	72	40	40	88	80	75	84	479
class 0	232	60	52	176	200	96	20	836
dur	32:34:28	15:58:55	12:22:27	19:52:32	30:08:06	20:36:51	8:01:24	05:19:34:43

* In this case, the number of recordings varies within movements.

We understand this labeling is problematic (performers may study abroad and be inspired by many composers, teachers, musicians, and interpretations). However, Czech musicians may play the string quartets of Czech composers differently, inheriting a specific style or carrying on the music tradition that led to the compositions in the first place. Such labeling could be later changed (such as Europe/rest of the world or Central Europe/Western Europe) with different aims of the analysis. As we show in this study, specific details in the tempo of measures may differentiate performers, perhaps even without their prior intention.

[2]https://imslp.org/ (accessed on 10 March 2023)

Based on the open-source policy, we would like to contribute with string quartet data to the performance datasets (see [155] and [24]). However, the vast majority of recordings are not under a CC license. Therefore, we share at least measure information (annotations) of each interpretation and composition in the GitHub repository.

8.2.2 Annotation

To characterize or evaluate differences in interpretations, we want to obtain or extract reliable timing information from each recording (see Section 8.3.1). First, we used automated methods with little success (see Section 8.2.4). We manually annotated one interpretation (chosen as a reference recording) per composition to obtain GT data and acquire annotations for all other interpretations based on the audio-to-audio synchronization strategy [56].

We considered the sequence of beats or measures as our timing parameter. Both can describe a given piece's local and global tempo and can be connected to the underlying score material. However, we chose measures to easily segment sections based on the score and reduce the time needed to annotate each reference recording. Time positions of measures may provide sufficient resolution and valuable information for further evaluation [129]. If the goal of analysis or required time resolution changes, one can annotate and synchronize, e.g., beats instead (see Section 8.2.3).

We annotated GT measure positions (obtaining reference measure positions) for each reference recording based on a corresponding score. Furthermore, we annotated sections—meaningful segments of each movement usually marked by numbers or letters. Table 8.2 shows the number of sections and measures for all compositions and movements. We did not annotate sections of Smetana's String Quartet No. 2 as they were not included in the score.

8.2.3 Synchronization

To obtain measure positions for each interpretation, we resample all recordings to 22 050 Hz and compute the time alignment of the reference and all target recordings following the sync-toolbox pipeline in [56]. First, a variant of chroma vectors, also known as Chroma Energy Normalized Statistics (CENS) features [130], is computed. The tuning is estimated to shift CENS accordingly, and the Memory-restricted Multiscale DTW algorithm (MrMsDTW) [45] is applied to find the optimal alignment between both chroma representations. Measure positions are then transferred from the reference to each target recording based on the warping path and final interpolation. Following this strategy, one can obtain any time-related annotation (onsets, Tab. 8.2: The number of sections and annotated measures for all recordings of our dataset; composers, compositions, and movements; x means that data are not available—either we did not obtain this information from a score or the chosen reference recording was different from the available score, so we excluded given recordings from the analysis.

composer	composition	mov	no of sections	no of measures
Dvořák	No. 12	mov1	19	239
		mov2	9	97
		mov3	13	244
		mov4	16	382
	No. 13	mov1	14	393
		mov2	10	202
		mov3	13	510
		mov4	12	563
	No. 14	mov1	11	204
		mov2	x	x
		mov3	7	102
		mov4	15	534
Janáček	No. 1	mov1	8	164
		mov2	14	236
		mov3	9	103
		mov4	16	189
	No. 2	mov1	17	314
		mov2	17	218
		mov3	15	216
		mov4	24	356
Smetana	No. 1	mov1	12	262
		mov2	12	250
		mov3	10	97
		mov4	18	285
	No. 2	mov1	x	140
		mov2	x	187
		mov3	x	76
		mov4	x	x

beats, measures, regions) if both reference and target recording follow the same harmonic and melodic structure and at least one set of GT data is available.

In the case of string quartets, there may be problems with repetitions and, e.g., codas. As a pre-processing step, we check the structure differences first. We compute anchor points (the first 10% and the last 90% of the duration of a given recording), test points projected on the warping path (approximately one point every two seconds), and connect the anchor points to form a line (see Figure 8.2). We consider only 10–90% of the warping path to avoid possible applause at the beginning or end of a recording. Furthermore, we compute the relative slope τ_r (the difference between the slope of the projected line and consecutive points on the warping path) and absolute slope τ_abs (the projected line is not taken into consideration). If $\tau_\mathrm{r} > 3$

or $\tau_{\mathrm{abs}} < 0.13$, we suspect a structural change in the musical content (see Figure 8.2). In other words, if the slope of the projected consecutive points is too steep or too flat, the target recording is not valid for further processing. For example, $\tau_{\mathrm{r}} = 3$ corresponds to the situation when the given time segment of the target recording is played three times faster than the reference recording. Based on our observations and dataset, it is unlikely for longer time segments even with expressive music such as string quartets. Following this strategy, we should automatically select all interpretations that follow the same score. The threshold values τ_{r} and τ_{abs} were set empirically. If both reference and target recordings are duplicates, the slope of all consecutive points on the warping path is ideally 1. We discarded all duplicates and proceeded only with recordings that followed the same structure as a reference recording. The final number of all interpretations for the classification is given in Appendix, Tables C.1, C.2, and C.3, depending on the composer.

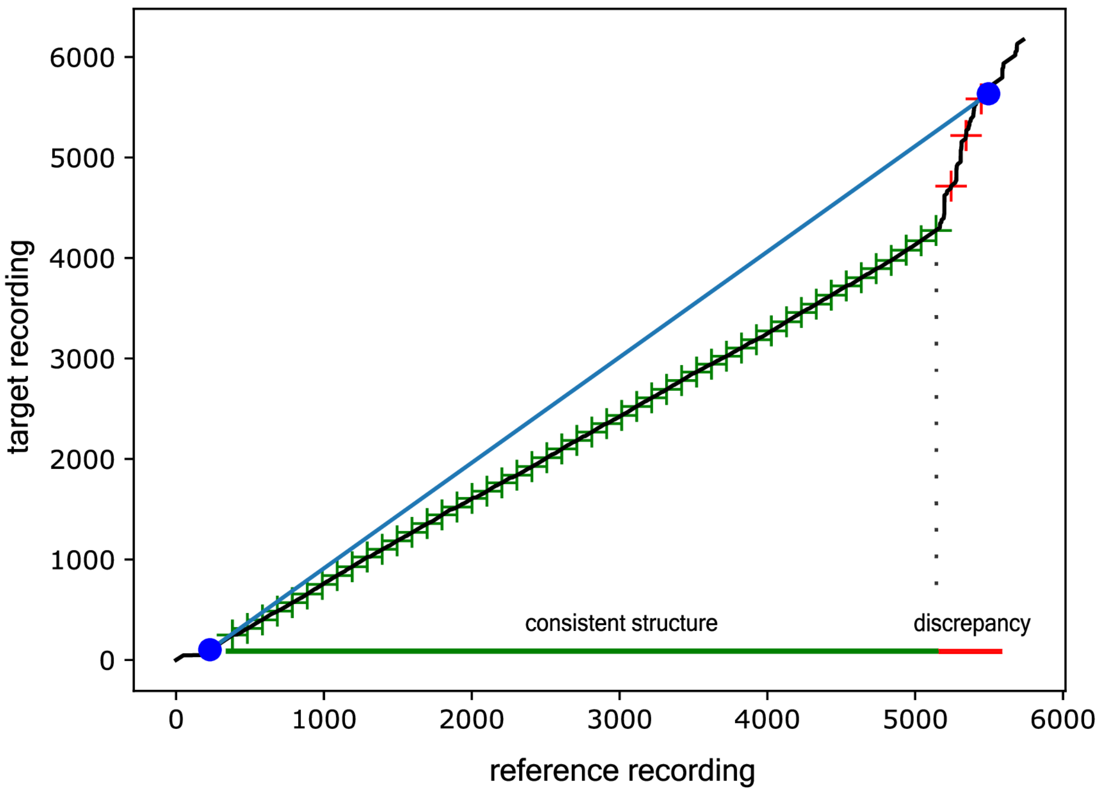


Fig. 8.2: An example of a warping path between a reference and a target recording; interpretations differ in the underlying musical structure (the target recording contains measures that are not included in the reference recording); blue dots correspond to the anchor points; the blue line shows the diagonal path between anchor points; green points (crosses) are projected on the warping path and are equally distributed; red points (crosses) indicate the region of dissimilarity as their $\tau_{\mathrm{r}} > 3$.

Interestingly, we encountered a situation where two recordings were duplicates even though they had different duration and audio qualities. One was the original copy from the phonograph recording; the second was a newer CD release. They differed in the source (database), metadata, duration, and thus global tempo, audio quality, and the presence of noise. Audio fingerprinting and image hashing methods would probably struggle with this case (their goal is slightly different), but the proposed synchronization technique detected the duplicates correctly. The limitation of this approach, and the reason why it is not commonly used on big datasets, is in its computational time, which grows with the number of input recordings (synchronization pairs) even with optimized DTW methods. The number of all combinations C is $C = n \cdot (n-1)/2$, where n is the number of recordings. Adding one track to the dataset requires running all possible combinations with a given recording again.

8.2.4 Validity of Synchronization Accuracy

In MPA, many timing parameters (onsets, beats, measures, and tempo) may be derived from GT annotations. In the case of classical music or string quartets, the automated systems (onset, beat, and downbeat trackers/detectors) still need to be improved for fully automated analysis. To demonstrate this, we apply a well-known RNN-based downbeat detector [116] and WaveBeat downbeat detector [115] to one of the reference recordings with GT data available and compare the results with the semi-automated synchronization approach. For this purpose, we manually annotated the second reference recording in the same way described in Section 8.2.2. We did not use the latest downbeat detector based on Temporal Convolutional Networks (TCN), introduced in [110], because the pre-trained neural network models are not publicly available.

Table 8.3 shows the results for both downbeat detectors and a synchronization strategy. In addition to classical scores for comparing the accuracy of detectors (F-measure, continuity-based evaluation scores CMLc, CMLt, AMLc, AMLt, and Information Gain (D) that represents the entropy of measure error histogram), we computed absolute mean (Δ_{mean}) and median (Δ_{med}) difference in seconds between GT positions of the first reference and the transferred measure positions from the second reference recording. To compute the F-measure, we used a window size of $\tau_w = 0.1$ (instead of default $\tau_w = 0.07$ for beat tracking tasks) to compensate for the nature of soft onsets produced by string instruments and a coarser time resolution of measures. For further details and information about metrics, we refer to [78, 122]. Δ_{mean} and Δ_{med} are computed only for the synchronization method as the number of references and estimated measures are always the same—condition, which cannot be satisfied using automated methods.

Tab. 8.3: The F-measure, continuity-based metrics, and information gain (D) of automated downbeat tracking methods (madmom and wavebeat) and semi-automated audio-to-audio synchronization strategy (sync) evaluated on the reference recordings of Dvořák's String Quartet No. 12, movement 3. Δ_{mean} and Δ_{med} (in seconds) are computed only for the synchronization method.

	F-measure	CMLc	CMLt	AMLc	AMLt	D	Δ_{mean}	Δ_{med}
madmom	0.337	0.000	0.000	0.154	0.285	0.158		
wavebeat	0.338	0.037	0.143	0.037	0.143	0.082		
sync	**0.927**	**0.290**	**0.963**	**0.290**	**0.963**	**0.426**	0.040	0.025

Results suggest that the synchronization approach is, as expected, more robust and reliable (F-measure = 0.927 and $\Delta_{\text{mean}} = 25$ ms) and, in contrast to the automated detectors, always outputs the correct number of measures. Unlike downbeat detectors, the evident and problematic limitation is the necessity of at least one manual reference annotation. The downbeat trackers are not trained on string quartets and expressive music in general. This problem is partly addressed in, e.g., [156] or [9], where the evaluation is based on user-driven metrics [131].

8.3 Feature Selection and Design

8.3.1 Features

There are many parameters that can characterize music performances. We can divide them into a few basic categories [24]:

- dynamics: how the loudness varies based on phrasing, accents, or structure,
- timing: rhythmic structure, micro-timing (onsets or beats), global tempo, or local tempo deviations,
- timbre: choice of instrumentation, instruments, playing techniques, and acoustic conditions,
- pitch: intonation, deviations from the score, unintentional intonation choices, and playing techniques such as vibrato.

Most parameters cannot be unconditionally connected to the direct semantic level. For example, timbre is a very ambiguous parameter if the context, acoustic conditions, recording and encoding choices, or post-processing options are not considered. Computing the dynamics can also be inaccurate as the original music carrier, quality, and post-processing choices (although this is not usually the case for classical music) may change even the relative proportions. Therefore, we focused solely on the timing parameter, which should not be affected by the abovementioned situations. However, an exception may be the inability to fit the interpretation into the older music medium (such as the maximum duration of 3.5 minutes on a 10-inch 78 RPM phonograph record). The oldest phonograph recordings from our dataset are from 1928 and 1929 (Ševčík-Lhotský and Czech Quartet, respectively), yet we do not consider this possibility in the analysis.

We construct a feature vector where each value represents the duration of consecutive movements, sections, or measures. By stacking these vectors vertically (each row represents features of a given recording), we obtain a feature matrix. In contrast to the approach in [10], where the feature matrices consisted of spectral parameters, dynamics, and timing properties for each synchronized measure of the piece, we focus only on differences in the duration of measures, musical sections, or entire movements to reduce the number of features. The examples of proposed feature matrices are shown in Section 8.3.4.

8.3.2 mRMR

To further preprocess our data, we use a technique called minimum-Redundancy Maximum-Relevance (mRMR), first introduced in [157] and later used in numerous studies [158, 159, 160]. This algorithm performs an efficient selection of the n most relevant features, decreasing the feature redundancy [161]. The first step of mRMR is to search for features satisfying the Maximal-Relevance criterion (8.1), which approximates Max-Dependency $D(S, c)$ with the mean value of all mutual information I values between individual feature x_i and class c:

$$\max D(S, c), \quad D = \frac{1}{|S|} \sum_{x_i \in S} I(x_i; c), \tag{8.1}$$

where S denotes the feature set to be selected. The second step is to deploy the minimum-Redundancy condition [157] as the features selected by the Maximum-Relevance could have a significant amount of redundancy. This condition is defined by:

$$\min R(S), \quad R = \frac{1}{|S|^2} \sum_{x_i, x_j \in S} I(x_i; x_j). \tag{8.2}$$

The mRMR criterion is the combination of the constraints mentioned above, and it is defined by the operator $\Phi(D, R)$, which integrates D and R. The simplest form to optimize D and R simultaneously is given by:

$$\max (D, R),\ \Phi = D - R. \tag{8.3}$$

In some cases (the second and third scenario, explained in Section 8.3.4), each recording consists of a different number of features, thus the variable length of the feature matrix. The utilization of mRMR allows us to uniform all feature matrices in length and to select the most significant features in terms of the difference between Czech and non-Czech classes. We use the implementation from the mRMR Python library in our experiments[3] and refer to [161] for more details about mRMR algorithm.

8.3.3 SVM

We build on a machine learning method called Support Vector Machines (SVM) to perform binary classification on our dataset. We use the LIBSVM implementation [162] of ν-Support Vector Classification (ν-SVC) [163] available via scikit-learn package[4]. The user-specified regularization parameter ν, similar to the standard C parameter used in C-SVC [164], represents an upper bound on the fraction of training errors and a lower bound of the fraction of support vectors. Therefore, a user specifies the ν, where $\nu \in (0, 1]$. In our case, we used $\nu = 0.5$. As described in [162], in a binary classification scenario, given training vectors $x_i \in \mathbb{R}^n$, $i = 1, ..., l$ and a vector $y \in \mathbb{R}^l$ such that $y_i \in \{1, -1\}$, the primal optimization problem is:

$$\begin{aligned} \min_{w,b,\xi,\rho} \quad & \frac{1}{2} w^T w - \nu\rho + \frac{1}{l} \sum_{i=1}^{l} \xi_i \\ \text{subject to} \quad & y_i(w^T \phi(x_i) + b) \geq \rho - \xi_i, \\ & \xi_i \geq 0, i = 1, \dots, l, \quad \rho \geq 0. \end{aligned} \tag{8.4}$$

The dual problem is:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^T Q \alpha \\ \text{subject to} \quad & 0 \leq \alpha_i \leq 1/l, \quad i = 1, \dots, l, \\ & e^T \alpha \geq \nu, \quad y^T \alpha = 0, \end{aligned} \tag{8.5}$$

where $Q_{ij} = y_i y_j K(x_i, x_j)$. The decision function of ν-SVC is defined by:

$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{l} y_i \alpha_i K(x_i, x) + b \right). \tag{8.6}$$

We also utilized the linear SVC during our experiments, but we found that the classification accuracy was slightly better when using ν-SVC. We used the Radial

[3]github.com/smazzanti/mrmr (accessed on 10 March 2023)

[4]scikit-learn.org/stable/modules/generated/sklearn.svm.NuSVC.html

Basis Function (RBF) as a kernel for all machine learning scenarios described in Section 8.3.4. A more detailed description of various SVM algorithms can be found in [163].

8.3.4 Design of Experiments

Bowen [98] points out the complex relationship between the choice of tempo and the composition duration. Generally, a slower chosen tempo at the beginning implies a longer duration of the entire piece and vice versa: a faster tempo shortens the composition. However, very often, this is not the case. We can look for more differences in the ratio between tempo and duration in a fragmented form. This allows a "relaxed" interpretation full of agogic changes and expressive caesuras. Demonstrable results are shown by the procedure in which the pace of shorter, meaningful sections, related to the whole duration, is calculated. The opposite method, which is based on measuring large parts or whole movements and calculating the average tempo of the composition, has no significant meaning because such a procedure "neutralizes" the particular characteristic of the interpretation. Considering the nature of our data and to address this problem, we decided to split the experiments into three scenarios.

Each scenario deploys a different feature matrix—they all contain timing information (see Section 8.3.1) but differ in resolution. We standardize all features to a mean of zero and a standard deviation of one (removing the mean and scale to unit variance). Then, the SVM classifier is deployed (see Section 8.3.3) to all matrices. Precision, recall, and F-measure (also called F-score) metrics are computed. Whole movements give the coarsest resolution, then sections, and finally, measures of a given piece. The description of scenarios with examples of feature matrices (corresponding tables) is as follows:

- First scenario: classification based on the duration of all 4 movements (Table 8.4).
- Second scenario: classification based on the duration of all sections (Table 8.5).
- Third scenario: classification based on the duration of the ten most relevant measures, selected by the mRMR method from all measures (Table 8.6).

Using mRMR in the first and second scenarios only ranks the relevance of given features but does not change the input for ν-SVC. The third scenario utilizes mRMR to select the first ten most important measures, which are further used as the input of ν-SVC. To compensate for an imbalanced dataset, we always randomly undersample the class with more recordings. Furthermore, we stratify the training and test subset so there is always the same number of recordings in both Czech and non-Czech classes. Training and testing data are split into 75/25 subsets and shuffled randomly. The SVM classifier (see Section 8.3.3) is used; precision, recall, and F-measure are computed on the testing subset. This procedure is repeated 1000× and a mean and a standard deviation (σ_F for F-measure, σ_P for precision, and σ_R for recall) are computed.

Tab. 8.4: Exemplary feature matrix of the first scenario; each row represents a set of features for a given recording; ID – identification of a performance/recording, mov1–mov4 – the duration of each movement in seconds; binary label based on the origin of a performer.

ID	mov1	mov2	mov3	mov4	label
002	559.52	428.62	213.51	306.37	0
003	620.81	420.10	240.55	325.27	1
004	559.21	470.88	205.29	335.96	1

Tab. 8.5: Exemplary feature matrix of the second scenario; each row represents a set of features for a given recording; ID – identification of a performance/recording, section1–section8 – the duration of each section in seconds; binary label based on the origin of a performer.

ID	section1	section2	section3	section4	···	section8	label
001	22.64	22.63	37.23	34.65	···	27.48	0
002	23.87	21.46	38.07	31.05	···	22.58	0
003	24.09	22.30	40.13	32.21	···	24.46	0

Tab. 8.6: Exemplary feature matrix of the third scenario; each row represents a set of features for a given recording; ID – identification of a performance/recording, measure1–measure239 – the duration of each measure in seconds; binary label based on the origin of a performer.

ID	measure1	measure2	measure3	···	measure239	label
001	4.12	1.91	2.54	···	3.09	0
002	1.87	2.01	2.02	···	2.81	0
003	2.24	1.97	2.26	···	3.51	1

The following example of the third scenario shows the workflow of processing.

- Feature matrix of size 27×10 (27 recordings, 10 most significant measures selected by mRMR), 15 recordings of class 1 (Czech), 12 of class 0 (non-Czech).
- All features are standardized by removing the mean and scaling to unit variance.
- To balance the dataset, 12 recordings of class 1 are randomly chosen, and the rest of class 1 is discarded in this run.
- Data is split into the training subset (75% of 24, hence 18 recordings) and the testing subset (25% of 20, hence 6 recordings).
- It is also ensured that the ratio of class 1 and 0 stays the same, if possible, for both training and testing subsets.
- The final training subset: 9 recordings of class 1 and 9 recordings of class 0.
- The final testing subset: 3 recordings of class 1 and 3 recordings of class 0.
- ν-SVC is used and evaluated in terms of F-measure, precision, and recall on the test subset.
- The whole run is repeated 1000×
- A mean and a standard deviation of all F-measure, precision, and recall values are computed.

Contrary to this example, the mRMR method shows the relevance of given features even in the first and second scenarios. However, we only use it to show the importance of given features for the upcoming classification. The computation of the F-measure differs from the one used in a synchronization (see Section 8.2.4); here, no window is used.

8.4 Results

This section reports the results of mRMR and classifications. We focus on identifying differences between Czech and non-Czech interpretations using string quartets of Czech composers and implementing a classifier that can successfully predict the binary classes on previously unseen data represented by test subsets. We did not use validation subsets as the number of items for both classes is usually low.

8.4.1 First Scenario

In this experiment, we use feature matrices based on the duration of all movements (Table 8.4). We used only those interpretations in which all four movements were well-synchronized with a reference recording (e.g. if movement 2 of one of the interpretations was discarded in a pre-processing step (see Section 8.2.3); we did not use any of the performance's movements). This decreased the number of items within both classes. Table 8.7 shows the result of the mRMR method. It ranks the significance of features; e.g., in the case of Dvořák's String Quartet No. 12, a feature containing the most relevant information (rank 1), given proposed classes, is the duration of movement 2.

Tab. 8.7: The relevance ranking of the movements as features used in the first scenario; each number represents a movement of given importance compared to other movements of the composition.

composer	Dvořák		Janáček		Smetana
composition	No.12	No.13	No.1	No.2	No.1
rank 1	2	4	4	3	1
rank 2	4	2	3	4	2
rank 3	1	3	2	1	3
rank 4	3	1	1	2	4

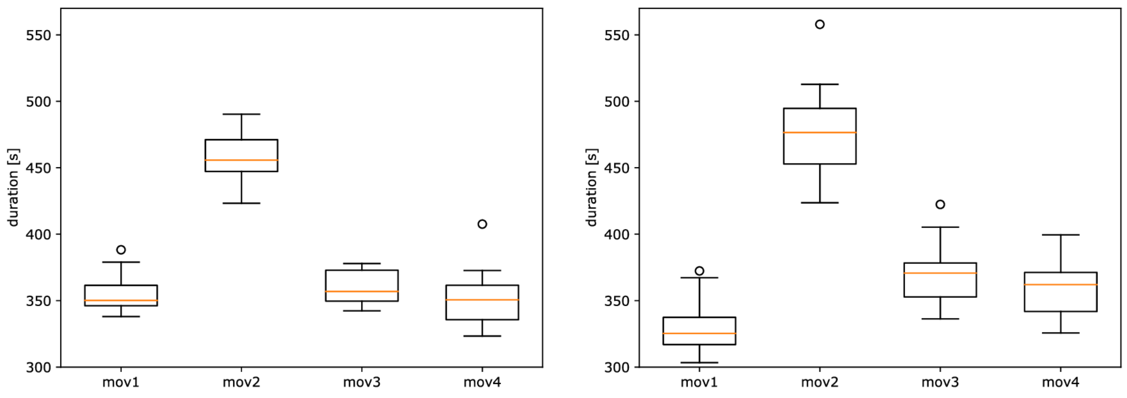
Table 8.8 shows the binary classification results. We report F-measure, precision, recall, and standard deviations of all metrics. In other scenarios, we show only the F-measure and its standard deviation. The prediction accuracy for Dvořák's string quartets is very low. With F-measures close to 0.50 and high deviations, it is very similar to random predictions. On the other hand, Janáček's String Quartet No. 2 seems to be the opposite—the F-measure = 0.87 with σ_F = 0.10. In this case, we can distinguish Czech and non-Czech interpretations with relatively high accuracy solely based on the duration of whole movements and their relationship. In the case of Smetana's String Quartet No. 1, F-measure = 0.70 with σ_F = 0.15.

Tab. 8.8: The F-measure, precision, recall, and corresponding standard deviations for the first scenario.

composer	composition	F-measure	precision	recall	σ_F	σ_P	σ_R
Dvořák	No. 12	0.47	0.50	0.50	0.23	0.28	0.21
	No .13	0.48	0.48	0.52	0.25	0.30	0.24
Janáček	No. 1	0.64	0.68	0.65	0.13	0.14	0.12
	No. 2	0.87	0.89	0.87	0.10	0.09	0.10
Smetana	No. 1	0.70	0.75	0.72	0.15	0.16	0.14

Figure 8.3 shows the statistics of the Czech and non-Czech classes for Janáček's String Quartet No. 2. To display the data distribution and statistical properties, we use boxplots—a box marks the second and third quartile; the whiskers are the first and the fourth quartile; a vertical line implies the median, and outliers are

presented as circles. The first movement of class 1 varies from 345 to roughly 360 s in contrast to class 0 with 320 to almost 340 s. The median of class 1 is, in this case, significantly higher, which is opposite to all other movements. The difference in the duration of the first movement is probably the main reason why the F-measure is high.



(a) The boxplot of class 1 (CZ).   (b) The boxplot of class 0 (non-CZ).

Fig. 8.3: The boxplots of the first scenario for Janáček's String Quartet No. 2 show both proposed classes' statistics and data distribution.

8.4.2 Second Scenario

In the second scenario, we construct feature matrices based on the duration of all sections (Table 4b) instead of movements, increasing the time resolution of features. Table 8.9 shows the application of the mRMR method, where the five most relevant sections are identified. The actual number of sections is shown in Table 8.2. For the sake of simplicity, we display only two compositions that achieved the highest accuracy in the classification task.

Tab. 8.9: The relevance ranking of the sections as features used in the second scenario; each number represents a section of given importance compared to other sections of the movement.

composer	Janáček				Smetana			
composition	No.2				No.1			
movement	mov1	mov2	mov3	mov4	mov1	mov2	mov3	mov4
rank 1	9	3	5	14	9	3	9	4
rank 2	12	14	11	9	1	12	4	16
rank 3	14	1	14	15	11	9	6	3
rank 4	7	4	4	19	4	4	1	18
rank 5	17	8	15	17	6	7	7	1

Table 8.10 presents the classification results. Here, we report the F-measure and its standard deviation. The trend is similar to the first scenario but with higher accuracy in most cases. Dvořák's String Quartet No. 14 shows the worst results (F-measure = 0.31 to 0.59) but consists of the least number of interpretations available. The standard deviation σ_{P} is high, overall. Janáček's String Quartet No. 2 (F-measure = 0.88 and $\sigma_{\mathrm{P}} = 0.09$ for the second movement) and Smetana's String Quartet No. 1 (F-measure = 0.77 and $\sigma_{\mathrm{P}} = 0.11$ for the first movement) provide interesting results. We now have information about the classification of each movement, which may show relationships within movements, e.g., overall, the second movement seems to provide a more accurate classification than the third movement.

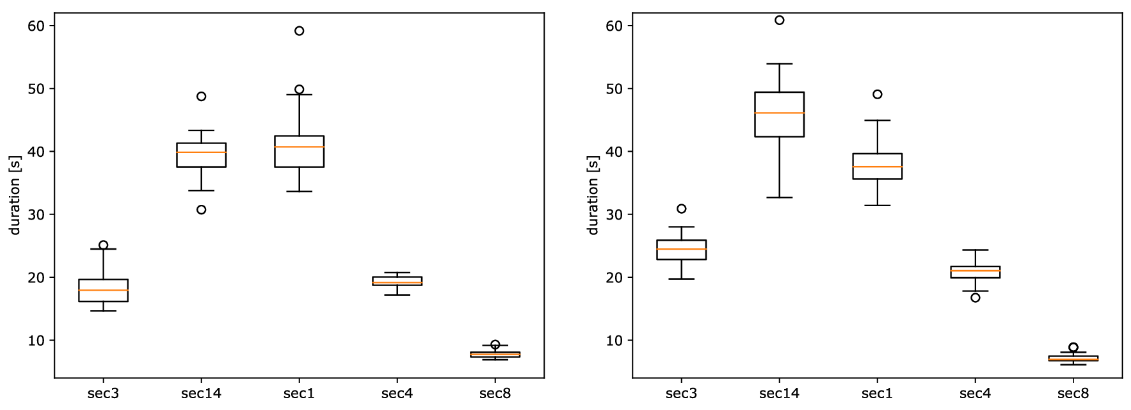
Tab. 8.10: The F-measure and its standard deviation for the second scenario; x represents data that were not available (see Table 8.2).

		F-measure				σ_{F}			
composer	composition	mov1	mov2	mov3	mov4	mov1	mov2	mov3	mov4
Dvořák	No. 12	0.57	0.69	0.57	0.69	0.21	0.15	0.16	0.16
	No. 13	0.61	0.72	0.70	0.47	0.20	0.20	0.24	0.24
	No. 14	0.54	x	0.59	0.31	0.20	x	0.23	0.21
Janáček	No. 1	0.56	0.62	0.53	0.66	0.15	0.14	0.14	0.13
	No. 2	0.84	0.88	0.77	0.85	0.12	0.09	0.12	0.12
Smetana	No. 1	0.77	0.74	0.69	0.69	0.11	0.15	0.16	0.15

Figure 8.4 shows the statistics of the Czech and non-Czech classes for Janáček's String Quartet No. 2, movement 2. The x-axis shows the first five sections chosen by the mRMR method. Here, we can notice more differences—the second and third quartiles of class 1 is below 20 s, while all data from class 0 are above 19.5 s. Section 3 corresponds to measures 34–44 marked in the score as *dolcissimo espressivo*, i.e., as sweet as possible and expressive. Czech performers seem to play this section statistically at a faster pace. Section 14 also shows a similar trend.

8.4.3 Third Scenario

In the third scenario, we use feature matrices based on synchronized measure positions. First, we apply mRMR to select the ten most relevant measures that are then used as input for the ν-SVC (see Table 8.12). With this information, we can identify measures according to which the Czech and non-Czech interpretations can be best distinguished. Increasing the time resolution of features (from movements and sections to measures) improved the recognition of interpretation differences between the proposed classes.



(a) The boxplot of class 1 (CZ). (b) The boxplot of class 0 (non-CZ).

Fig. 8.4: The boxplots of the second scenario for Janáček's String Quartet No. 2, movement 2, show both proposed classes' statistics and data distribution.

First, to create a baseline for the classifier, we select the binary labels randomly and use the proposed pipeline. Table 8.11 presents the results of the classification. Dvořák's String Quartet No. 13, movements 2, 3, and 4, show F-measure = 0.84, 0.86, 0.83 with $\sigma_\mathrm{P} = 0.13$, 0.15, and 0.15, respectively. Some compositions seem to be played differently enough that even two random classes are somewhat separable—all ensembles are, to some extent, distinct. We tried multiple randomly selected labels (different seeds) with similar results. We also tested the non-mRMR approach, where all measures are always used, but the classifier does not train, and the outputs are similar to random guesses.

Tab. 8.11: The F-measure and its standard deviation for the third scenario using random binary labels; x represents data that were not available (see Table 8.2).

		F-measure				σ_F			
composer	composition	mov1	mov2	mov3	mov4	mov1	mov2	mov3	mov4
Dvořák	No. 12	0.71	0.60	0.66	0.62	0.12	0.10	0.09	0.10
	No. 13	0.76	0.84	0.86	0.83	0.15	0.13	0.15	0.15
	No. 14	0.36	x	0.83	0.74	0.20	x	0.16	0.19
Janáček	No. 1	0.76	0.68	0.68	0.67	0.09	0.10	0.10	0.11
	No. 2	0.69	0.60	0.71	0.73	0.09	0.10	0.11	0.09
Smetana	No. 1	0.70	0.68	0.39	0.66	0.10	0.36	0.31	0.34
	No. 2	0.59	0.80	0.49	x	0.18	0.16	0.17	x

Table 8.13 provides the results of classification. Each combination of composer, composition, and movement shows high accuracy (except Dvořák's String Quartet No. 14 and Smetana's String Quartet No. 2, where the standard deviation is up

Tab. 8.12: The relevance ranking of the measures as features used in the third scenario; each number represents the measure of given importance compared to other measures of the movement.

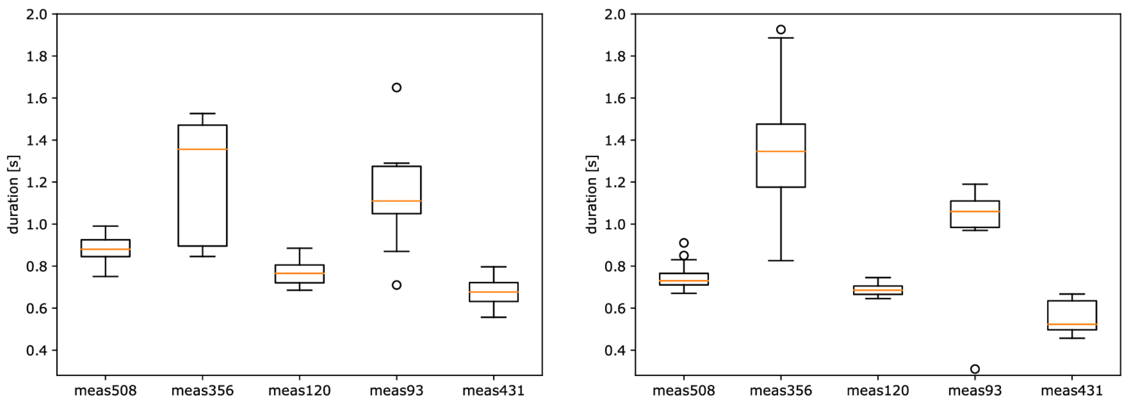
csr	Dvořák				Janáček				Smetana			
comp	No. 13				No. 2				No. 1			
mov	mov1	mov2	mov3	mov4	mov1	mov2	mov3	mov4	mov1	mov2	mov3	mov4
rank 1	132	71	508	207	52	42	76	235	224	52	51	30
rank 2	359	19	356	460	140	209	196	214	126	107	64	280
rank 3	388	70	120	468	166	41	77	30	76	220	62	62
rank 4	134	133	93	109	199	44	168	234	116	40	81	257
rank 5	342	140	431	355	233	39	52	119	25	166	45	276
rank 6	387	72	137	346	252	191	53	89	182	51	53	197
rank 7	139	1	95	37	97	43	144	194	118	65	41	281
rank 8	392	10	378	467	295	37	212	231	181	162	80	127
rank 9	128	61	228	167	86	171	127	148	26	57	52	34
rank 10	339	134	132	484	107	121	89	87	204	84	40	51

to 0.30). The F-measure of Dvořák's String Quartet No. 13, movements three and four, is 0.99 with $\sigma_\mathrm{P} = 0.05$. Furthermore, in the case of Janáček's String Quartet No. 2, the F-measure = 0.96 with $\sigma_\mathrm{P} = 0.06$ and F-measure = 0.94 with $\sigma_\mathrm{P} = 0.08$ for the first and third movement, respectively.

Tab. 8.13: The F-measure and its standard deviation for the third scenario; x represents data that were not available (see Table 8.2).

		F-measure				σ_F			
composer	composition	mov1	mov2	mov3	mov4	mov1	mov2	mov3	mov4
Dvořák	No. 12	0.76	0.78	0.76	0.81	0.16	0.14	0.14	0.12
	No. 13	0.87	0.88	0.99	0.99	0.14	0.13	0.05	0.05
	No. 14	0.76	x	0.74	0.77	0.18	x	0.18	0.21
Janáček	No. 1	0.82	0.76	0.75	0.86	0.11	0.13	0.12	0.10
	No. 2	0.96	0.91	0.94	0.88	0.06	0.09	0.08	0.10
Smetana	No. 1	0.84	0.90	0.82	0.89	0.09	0.10	0.13	0.10
	No. 2	0.70	0.88	0.86	x	0.30	0.18	0.21	x

When we increase the time resolution of features to individual measures, the difference between classes also increases. Figure 8.5 shows the statistics of the last scenario, Dvořák's String Quartet No. 13, movement 3. Results indicate that, on average, Czech performers play these measures at a lower tempo. Measures are around one second long, yet there are differences up to one second between interpretations. Interestingly, if we calculate the duration of measure 508, we can guess (e.g., more than 0.8 s) the Czech performers with relatively high accuracy. When all five proposed measures are combined, we can achieve up to 99% accuracy with a machine learning classifier (Table 8.13).



(a) The boxplot of class 1 (CZ). (b) The boxplot of class 0 (non-CZ).

Fig. 8.5: The boxplots of the third scenario for Dvořák's String Quartet No. 13, movement 3, show both proposed classes' statistics and data distribution.

8.5 Discussion

This study aims to train a machine learning classifier that predicts the performer's origin (Czech and non-Czech classes) of any interpretation given well-known string quartets of Czech composers. We propose feature matrices based on duration information, ignoring dynamics or timbre parameters as the acoustics, recording environment and equipment, instruments, and post-processing may make the input features of classification unreliable. Contrary to [144], we use only suitable timing information.

All features might describe specific qualities of a given performance, but in this paper, we choose only robust timing information for the origin classification. The duration of small time segments (such as measures) provides information about musical expressiveness and interpretive differences. If we choose larger segments, such as the duration of whole movements or sections composed of many measures, the significant differences and the accuracy of the potential classification decrease (compare Table 8.13 with Table 8.8 or 8.10). The exception is Janáček's String Quartet No. 2, where we achieved F-measure = 0.87. Converting the duration to tempo values does not affect the classifier; it might only serve as a more intuitive visualization. We chose measures for a few reasons: firstly, measures are well-defined by the corresponding score; secondly, they are easier to annotate manually than, e.g., beats; and thirdly, they can be used to segment recordings to sections or other logical structures (while ignoring the metrical structure of a given composition).

In Section 8.2.4, we show that automated downbeat tracking systems are not yet efficient for expressive string quartet music. Thus, the synchronization strategy (with available manual annotation) remains preferable. Feature selection explained in Section 8.3.2 helped the chosen classifier achieve higher accuracy while ranking the importance of features for a given task. This information can be further used for music analysis and a detailed comparison of differences. Using general structures such as measures has one more advantage—it allows us to generalize the classification pipeline to arbitrary music compositions, instruments, and genres.

The limitation of this study is the number of interpretations for given compositions. We have collected a large dataset of string quartet recordings, but only a portion of them was used (see Section 8.2.3) due to the different music structures. To balance the data, we stratify the training and test subsets in each classification run so there is always the same number of items in both classes. Considering compositions such as Janáček's String Quartet No. 2, Dvořák's String Quartet No. 13, or Smetana's String Quartet No. 1, the classifier provides promising results, confirming the original idea that proposed classes (Czech and non-Czech interpretations) are distinguishable (see Table 8.13). However, if we use random labels, binary classification based on the duration of specific measures (given by the composition and all available interpretations) already provides relatively high accuracy in some cases. This is expected, as the mRMR method chooses ten relevant features that distinguish these classes the most. If we do not implement a feature selection method, the classifier cannot be trained using the proposed strategy. When we use the CZ and non-CZ labels, the classification accuracy increases overall.

This study shows that origin-based differences in interpretations exist and are measurable. However, the proposed machine learning pipeline cannot be universally used—the reference measure positions are always needed for at least one recording of a given composition, and we train and test the classifier for each composition separately. So far, we cannot classify the origin of arbitrary recording without prior knowledge of the piece and other interpretations. In the future, we would like to test the strategy on string quartets from, e.g., Joseph Hayden or Ludwig van Beethoven with Austrian/German labels and provide a more detailed analysis of interpretation differences.

8.6 Conclusions

In this paper, we investigated the possibilities of string quartet interpretation classification based on performers' origin. We collected a large dataset of string quartets from Czech composers Dvořák, Janáček, and Smetana. We manually annotated ground-truth measure positions of reference recordings and applied a method of time alignment to transfer measure positions to all target recordings. Furthermore, we used measures to segment recordings into separate sections and split our experiments into three scenarios, each specified by different features. We trained and tested a machine learning classifier to distinguish Czech and non-Czech interpretations of string quartet pieces. We showed that it is possible to train such a classifier. The classifier achieved poor results when feature matrices contained the duration of whole movements, except for Janáček's String Quartet No. 2 with F-measure = 0.87. Increasing the time resolution of features, from movements to sections and measures, improved the prediction accuracy. For the third scenario, where measure positions were used, we achieved F-measure = 0.99 for Dvořák's String Quartet No. 13, movements 3 and 4, and up to 0.96 in the case of Janáček's String Quartet No. 2. Using proposed labels, the accuracy increased compared to the baseline with random labels, which already provided relatively high accuracy. It seems that interpretation-based differences are already distinguishable, in some cases, even in random subsets. In the future, we will experiment with other string quartet composers, use more labels, and further describe and explain the interpretation differences. We plan to experiment even with finer time resolution, such as beats, to train classifiers and identify differences in various interpretations.

9 Application of MIR Methods for Comparative MPA

This chapter is based on the conference article “Application of Computational Methods for Comparative Music Analysis” [6].

Music Performance Analysis can thrive from computational methods of Music Information Retrieval. Besides extracting and analyzing symbolic music data, performance analysis also focuses on retrieving performance parameters from digital audio recordings. On the other hand, the aim of the comparative performance analysis is often qualitative and stands on our perception and musical principles. In this paper, we utilize feature extraction strategies and comparative analysis, leveraging computational methods while focusing on the goals of musicology. We aim to provide insight into music performance data for subsequent case studies. As the main contribution of this paper, we present a specific combination of extraction methods for performance music analysis on the application level. Furthermore, we demonstrate an early version of open-source software that deploys the proposed strategy in a user-friendly web-based environment.

9.1 Introduction

In Music Information Retrieval (MIR), the researchers deal with many music-oriented challenges using methods from signal processing to machine learning and statistics [165, 128]. They focus on tasks such as low- and high-level feature detection, symbolic data representations, music recommendation, content-based search, automatic tagging, transcription, instrument separation, and many more [32, 30]. On the other hand, Music Performance Analysis (MPA) has traditionally been a peripheral topic for the MIR community [24]. Furthermore, the comparative part of MPA aims at multiple music interpretations, performances, or versions (in this paper, we use all three terms interchangeably) and compare their differences [146]. The research in computational musicology, which combines both fields, is usually performed on symbolic music representations such as music notation or midi data [166, 167]. However, we focus on audio data without symbolic transcriptions as this format is the most common (including studio and live versions).

The performance is essential in how listeners perceive a piece of music. One can analyze dynamics, tempo, and expressive performances on a macro scale (whole compositions) or a micro level (segments, motifs, or measures) using manual annotation or some form of feature extraction [146]. However, a combination on the application level, where the MIR-oriented feature extraction supplements the qualitative case studies, is relatively rare. The well-known examples are Mazurka project [124] or Sonic Visualiser software [108].

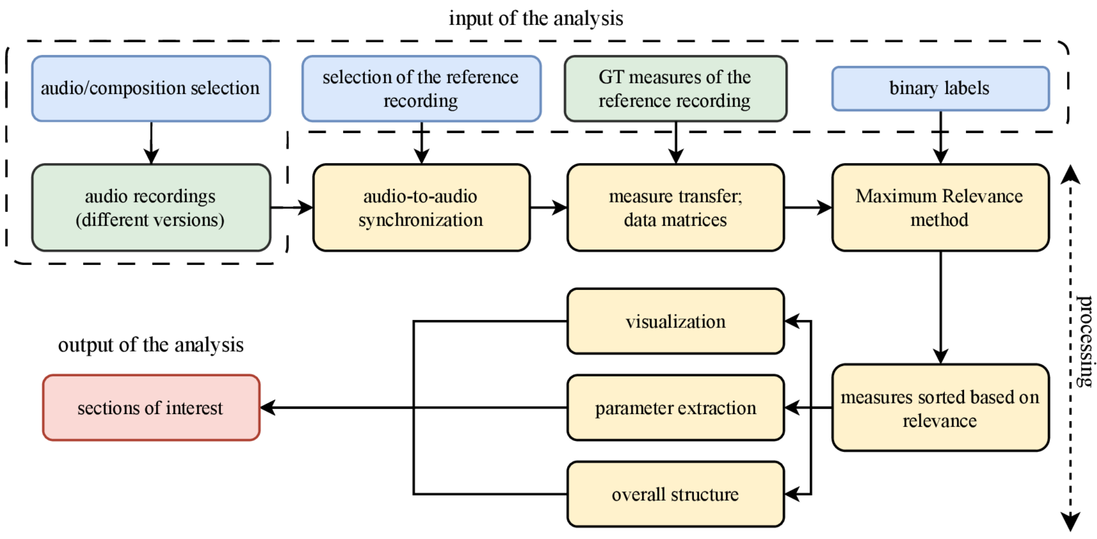

input of the analysis
audio/composition selection
selection of the reference recording
GT measures of the reference recording
binary labels
audio recordings (different versions)
audio-to-audio synchronization
measure transfer; data matrices
Maximum Relevance method
processing
output of the analysis
sections of interest
visualization
parameter extraction
overall structure
measures sorted based on relevance

Fig. 9.1: Pipeline of the proposed approach to obtain hints for the MPA studies based on the quantitative music analysis. Green boxes indicate input data (audio files, e.g., *.wav* and text files, e.g., *.csv*), blue boxes input choices, yellow boxes data processing, and the red box the desired output.

In this paper, we utilize MIR audio feature extraction and time-alignment methods to provide information for subsequent comparative MPA research. As the main contribution, we utilize a strategy to obtain meaningful cues to statistical differences between interpretations of the same musical piece. We first extract performance parameters, deploy a feature selection method used in machine learning, and apply it in a music analysis scenario. The proposed strategy produces specific cues about interpretations and provides insights for subsequent case studies. Finally, we present an early version of open-source software[1] that utilizes the proposed strategy and allows users to analyze multiple versions of the same piece, including MPA-related playback, ranking of feature relevance, and visualization. Similar synchronization features (score-to-audio), but with no MPA-driven functions, can be found in the Interpretation switcher [126] or Sync Player [168]. However, both are not publicly available. Furthermore, we refer to [169] for more information about similar interfaces.

The rest of the paper is organized as follows. Section 9.2 introduces audio-to-audio chroma synchronization, combined synchronization approach, and measure

[1]The software will be available at `https://github.com/stepanmk/memovision` (accessed on 7 July 2023)

transfer to obtain measure durations and detect different music structures. Section 9.3 describes the feature selection method and relevance rating. Section 9.4 shows the first snippets from our future software and a possible way of future development, followed by a discussion in Section 9.5 and conclusions in Section 9.6.

9.2 Extraction Pipeline

In a music performance analysis scenario, one could, for example, focus on the analysis of interpretation differences between Chopin Nocturnes given specific labels, such as the year of recording. The goal of the research may be: In which parts of the piece and how do the performers differ the most before and after 1970?

Typically, a musicologist would listen to the recordings, retrieve meaningful information (dynamics, timing, tempo, pitch deviations, etc.) from the selected sections (e.g., motifs or logical segments) and visualize the parameters of recordings based on the proposed labels. However, finding the areas with the highest parameter variance in a large-scale dataset by comparing the parameters of whole recordings may be tedious. Moreover, one needs to have information about segments, such as measures, to split the parameters accordingly because each interpretation usually varies in a local tempo and, therefore, in the timing of musical events. In this paper, we utilize a strategy to overcome the time-consuming and mostly manual approach and provide cues or highlight the sections of interest for the possible qualitative case studies. The strategy is shown in Figure 9.1 and is described in the following sections.

9.2.1 Chroma Synchronization

We deal with multiple versions of the same composition and, therefore, leverage a synchronization method, a well-known approach in MIR [129, 170]. Compared to other studies with score-to-audio synchronization, we do not use symbolic MIDI data—we utilize an audio-to-audio strategy. This design choice was inspired by our previous experiments with a combined synchronization approach (see Section 9.2.2).

First, we focus on obtaining measure positions for all composition recordings. Measures are essential segments of each piece. They are used in MIR and MPA for many tasks, such as cross-version analysis [171] or segmentation [172]. Instead of annotating each recording manually or automatically via a downbeat detector, we use the time alignment of one reference and all target recordings following the Sync toolbox pipeline [56]. It consists of modified chroma vectors [130], the estimation of tuning to shift the chroma features so that they match the same pitch classes, and the memory-restricted multiscale DTW algorithm (MrMsDTW) [45] to find the optimal time alignment between a reference chroma $X = (x_1, \dots, x_N)$ and a target chroma $Y = (y_1, \dots, y_M)$. Tuning estimation compensates for pitch deviation of older recordings or different tuning of instruments and increases the synchronization accuracy. The MrMsDTW method is faster compared to the standard DTW and produces an optimal warping path from cost matrix $\mathbf{C}_{\mathtt{CH}}(n, m) = c(x_n, y_m)$ of size $N \times M$, where c defines a local cost measure, $n \in \{1, \dots, N\}$, and $m \in \{1, \dots, M\}$. The warping path is then saved as in a binary .npy file for further processing.

Unlike the basic approach where annotations are created manually (as in Sonic Visualiser), we can use the underlying music structure (which should always be the same in our scenario) to obtain measure positions semi-automatically. If we use a score-to-audio strategy, we would need transcribed MIDI files instead of measure annotations, which may be a more time-consuming task for non-piano music.

Any time-related annotations (beats, measures, motifs, regions) can be obtained if reference and target recordings follow the same harmonic structure. However, the chroma synchronization method may struggle in sections where the harmonic information of an audio segment is homogeneous (see [154] or [14] for more details). For example, let us have sustained piano tones with no harmonic and melodic changes over two measures. There are no temporal or harmonic cues to detect the start position of the second measure because the cost matrix values in this section are similar. To overcome this problem and to make the method more robust, we use a combined synchronization approach.

9.2.2 Combined Approach

Authors in [154] proposed decaying locally adaptive normalized chroma-based onset features (DLNCO) and combined them with a chroma cost matrix $\mathbf{C}_{\mathtt{CH}}$ to increase the synchronization accuracy. This method works well with piano music containing strong transients at the attack phase of individual tones. However, we exploit the beat activation function from a neural network model to provide temporal clues. In recent work [14], the beat activation function combined with chroma features increased the accuracy of measure transfer in the case of more complex string quartet music. We build on this idea and use a simplified version of the state-of-the-art beat detection system based on a temporal convolutional network (TCN) [110, 114]. We trained a new model with nonstandard parameters—temporal resolution of 50 fps instead of 100 fps; time-frequency transformation is performed on 22 050 Hz recordings instead of 44 100 Hz unlike in, e.g., madmom module [82]—to match the synchronization pipeline without any further resampling of input audio recordings or chroma features. We use the combined cost matrix $\mathbf{C}_{\mathtt{CH+B}}$ consisting of equally weighted chroma cost matrix $\mathbf{C}_{\mathtt{CH}}$ with cosine distance and the beat cost matrix $\mathbf{C}_{\mathtt{B}}$ computed using the beat activation function (a function corresponding to the probability of beat events that yields values between 0 and 1) with Euclidean distance:

$$\mathbf{C}_{\mathtt{CH+B}} = 0.5 \cdot \mathbf{C}_{\mathtt{CH}} + 0.5 \cdot \mathbf{C}_{\mathtt{B}}. \tag{9.1}$$

We refer to our previous study [5] for more information on the TCN model description, to [117] for information about deep learning-based beat tracking, and to [154] and [14] for combined synchronization approach.

9.2.3 Structural Differences

Finally, we check the slope of the resulting warping path and compare it to the threshold. The value of the threshold is set experimentally. If the slope is too steep or flat, we can exclude the recording or its part from the analysis. This happens when there is an additional repetition or a missing section in the recording. Some studies deal with similar problems introducing matching technique [173], jump DTW [174], or Hierarchical DTW [175]. We build on a more straightforward approach without directly modifying core functions or a warping path of the MrMsDTW algorithm. As the last step, we transfer manually annotated measures of a reference recording to all other interpretations using linear interpolation. We end up with measure positions for all recordings and can segment each performance accordingly.

9.3 Feature Selection

We can use any number of target recordings and rely only on one set of annotations, thanks to the synchronization procedure. Theoretically, we could skip the synchronization part by utilizing a downbeat tracking system, but it generally does not achieve satisfactory results yet [110]. Following the strategy in [11], we compute a data matrix that contains measure durations expressed in seconds. The rows correspond to versions of a given composition, and the columns correspond to measure indices. We can label each recording to create binary classes based on the goal of analysis—for example, a label based on the year of recording (before 2000 vs after 2000), the origin of performers (or from different musical backgrounds and cultures), or simply the interpretation of one performer vs the others. The selection of a binary label is essential for the next phase of this pipeline.

Our final goal is to estimate the relevance of each measure regarding the proposed labels. To rank the relevance of measures and differentiate between labels, we use the Max-Relevance (MR) method. This algorithm efficiently ranks the relevance of features. It searches for features satisfying the maximal relevance criterion (9.2), which approximates maximum dependency $D(S, c)$ with the mean value of all mutual information I values between individual feature x_i and class c [157]:

$$\max D(S, c), \quad D = \frac{1}{|S|} \sum_{x_i \in S} I(x_i; c). \tag{9.2}$$

We use implementation of the Min-Redundancy Max-Relevance (mRMR) method[2] with a focus on the relevance only. The minimum redundancy criterion is essential when the features are further used, e.g., for a machine learning classifier [158]. However, unlike in our previous study with a binary classification of string quartets [11], we use only the relevance criterion and discard the redundancy computation. In the case of obtaining the most significant features to distinguish between two classes, redundancy would be counterproductive as it may remove measures that are correlated with previously selected ones but are still highly relevant. Following this approach, we compute the relevance of each measure regarding the difference between given labels and sort them in descending order.

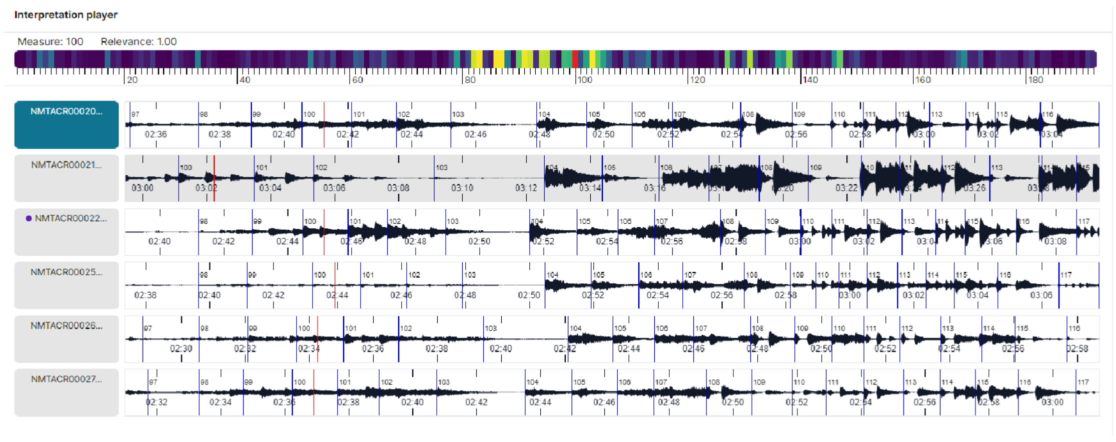


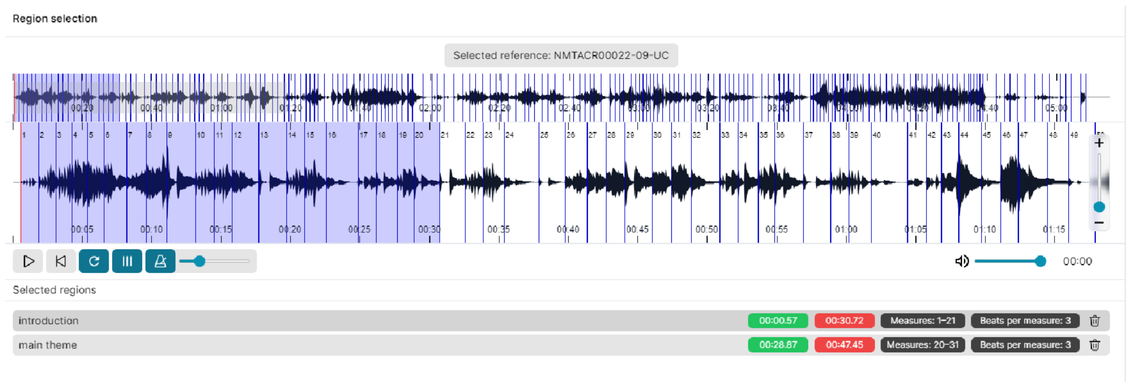
Fig. 9.2: Interpretation player – navigation, playback, measure visualization, and relevance of measures. The blue lines indicate measures. The red lines show the current position within a given recording.

9.4 Software

We work on software (web-based interface; JavaScript frontend with Python backend) for music analysis that utilizes the proposed strategy in a user-friendly and easy-to-use environment. We plan to make the software open-source and available online at the beginning of January 2024. Thanks to the Python language, it is easier

[2]see `https://github.com/smazzanti/mrmr` (accessed on 7 July 2023)

to follow state-of-the-art methods and add new features in the future. For comparison, Sonic Visualiser uses outdated Vamp plugins that are, to our knowledge, no longer actively developed.



introduction	00:00.57	00:30.72	Measures: 1–21	Beats per measure: 3
main theme	00:28.87	00:47.45	Measures: 20–31	Beats per measure: 3

Fig. 9.3: Region selector – playback, sonification, and region selection based on reference measures.

9.4.1 Demonstration

In this section, we would like to demonstrate the first outputs of the software. Users can upload recordings and measure annotations, select the reference recording, and start the extraction process. By default, the reference recording belongs to one class (label) and the rest to the second. A user can listen to the reference recording, sonify measure positions with a click, and create a region of interest based on the time or measures (see Figure 9.3). Then, it is possible to switch to the interpretation player tab (Figure 9.2). One can see the current time position of other recordings (vertical red lines) while listening to one selected recording, display the regions and individual measures and play them in the loop, switch between recordings, and display the results of the MR method. Finally, at the top, there is a color map indicating the relative relevance of each measure throughout the composition to spot the more relevant (and potentially interesting) sections. However, this applies only to the differences between selected binary labels (e.g., one interpretation vs. the rest).

9.4.2 Future Development

We can use the transferred measure positions to interpolate parameters and transform the time axis into the measure axis, which is more convenient for performance analysis. Instead of physical time, we use a relative axis and directly compare any sections of interpretations. The example of relative dynamics is shown in Figure 9.4.

In this case, the loudness is based on pyloudnorm module [176] and resampled to 20 values per measure. It is computed on normalized recordings via FFmpeg and EBU R 128 recommendation. A parameter derived directly from the synchronization process is the duration or tempo of each measure (Figure 9.5). We plan to make such visualizations in the software possible with any measure range of a given composition. The user can limit the x-axis, analyze and compare the parameters of selected sections, or play the interpretations while following parameter progression.

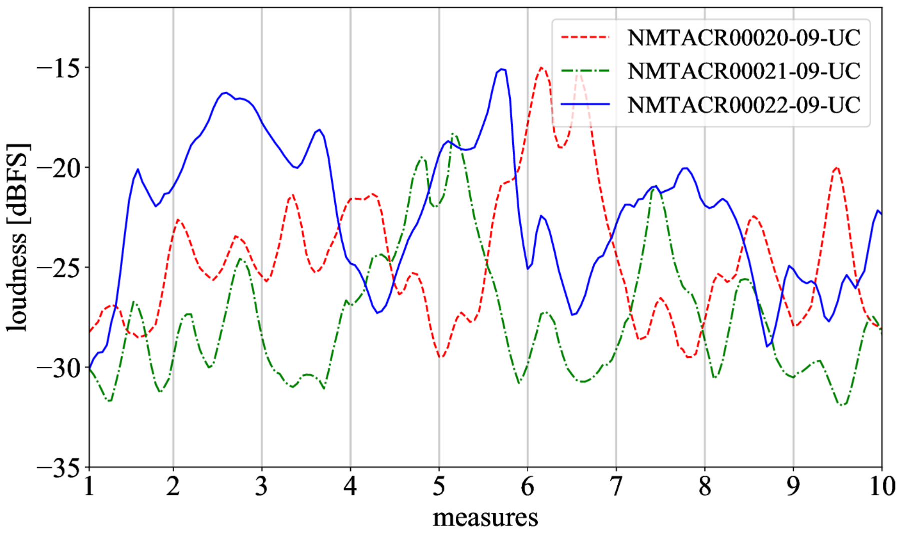


Fig. 9.4: Resampled loudness for the first nine measures of three example versions of the piano piece 'Little Onion' from Bedřich Smetana.

9.5 Discussion

If we follow the proposed strategy, we can obtain relatively precise measure positions for all interpretations. However, we need ground-truth measures for at least one performance. Transferring measures using conventional DTW alignment is known to the MPA community, but we incorporate temporal cues based on the TCN beat tracker to improve the measure-level synchronization. Furthermore, we address the problem of finding the most different measures or sections of compositions by applying a feature selection method on the duration of individual measures with given binary labels.

As the development of downbeat tracking systems continues, in the future, we may be able to skip the synchronization part altogether and obtain robust downbeat estimates automatically. In the performance analysis scenario, the number of all interpretations in the dataset increases the computational time of analysis. However,

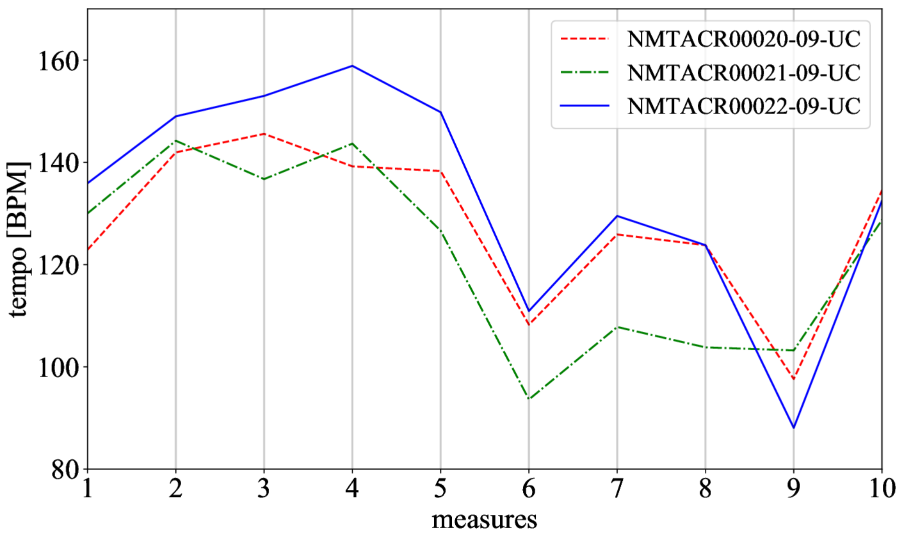


Fig. 9.5: Tempo of the first nine measures (three beats per measure) of three example versions of the piano piece 'Little Onion' from Bedřich Smetana. Lines between values are indicative—they connect the consecutive points directly and do not reflect the tempo deviations inside measures.

the dataset size is easily scalable and theoretically unlimited compared to a time-consuming and tedious manual approach to annotation.

One could use the output and hints from this strategy to perform case studies on specific recordings with any labels. For example, the proposed functions of the software could first indicate the relevant sections of the composition based on the predefined label. The user could then analyze the chosen sections by ear, compare extracted parameters, or use multiple visualization options. Furthermore, the software could output the data for each recording in IEEE 1599 format if required by musicologists. We followed a similar pipeline (without the web-based implementation and combined approach to synchronization) already in our pilot study on string quartets [11] from A. Dvořák, B. Smetana, and L. Janáček, where we trained a ν-support vector machine classifier on the output of the mRMR method.

The main limitation is the selected feature—duration of measures. Although measures are useful in many tasks, the only 'difference' between performances we compare using the MR method is the timing of measures. Other parameters utilize measure durations for their segmentation and visual comparison but not for relevance ranking. This may be impractical in many scenarios, as the interpretation has many more qualities and possible deviations, such as inter-beat and inter-onset timing, local tuning estimation, dynamics, or other parameters of expressivity that we do not include in a proposed data matrix. However, we plan to build on this pipeline and incorporate different time-related parameters, dynamics, or spectral properties into the data matrix and feature selection method to have more options for comparing interpretations and ranking the feature relevance.

9.6 Conclusion

In this paper, we proposed the extraction pipeline for quantitative music analysis. We utilized semi-automated synchronization, beat tracking, and a feature selection method to provide meaningful information about the differences between interpretations of the same musical piece. We build on the performance analysis scenario and speed up the tedious manual annotation or listening process of finding the interpretation deviations. However, the strategy still relies on one set of ground-truth measure positions. It is possible to sort measures or sections with the highest discriminatory power concerning the selected binary labels. For example, one can discover where a specific performance differs the most or where one group of performances (e.g., more interpretations of the same performer) differs from the second group. The strategy is highly variable, and the settings, such as the labels of the recordings, depend on the goal of the analysis. We hope the proposed pipeline will provide insight into comparative music performance analysis. We are working on software that utilizes this strategy in an easy-to-use and user-friendly environment to simplify extracting time-related parameters and to provide hints or cues for subsequent qualitative case studies.

9.7 Further Notes

The software is also described in the published methodology [16], including a case study on piano recordings from Bedřich Smetana and string quartets from Antonín Dvořák. To show more functions of this tool, we provide a brief demonstration in Appendix D.

Summary and Future Prospects

In this thesis, I helped to develop and present tools and computational resources for comparative music performance analysis. I utilized information retrieval methods to reflect the goals of musically-driven performance analysis and experimented with various string quartet and piano music datasets.

In Part I, I used several audio degradations to evaluate the robustness of onset detectors (Chapter 2) and reported differences between conventional and ML-based approaches when applying impulse response degradations, lossy compressions, or TKEO. I further extended the idea behind intentional audio degradation by analyzing the energy operator's influence in the beat tracking system pipeline (Chapter 3). I showed that reducing onset candidates may increase the tempo estimation accuracy, which is useful for MPA-related automated systems. Next, I trained multiple TCN beat tracking models (Chapter 4) on various sampling rates to provide insight into the training process and discussed the advantages and limitations of input rate reduction for further applications. I reported the results and shared all models online [18]. One of the models was later used in the synchronization pipeline of our software MemoVision [6, 7, 21], providing a computational improvement over standard models thanks to its modified sampling rate and temporal resolution. I also tackled the problems of annotation-related beat tracking evaluation (Chapter 5). Furthermore, I evaluated user-driven metrics for automated beat and downbeat detectors and semi-automated synchronization strategy in the case of complex string quartet music (Chapter 6). These studies have set the foundation and provided an evaluation of available methods for the analysis of performance differences [10, 11, 6, 7].

In Part II, I addressed the extraction of performance data from a dataset of string quartet recordings (Chapter 7). Based on the modified feature matrices that represent individual performance data while providing data unification, I trained a machine learning classifier to differentiate between Czech and non-Czech performances in the large corpora of string quartets from Anotnín Dvořák, Leoš Janáček, and Bedřich Smetana (Chapter 8). I used an extraction pipeline and feature selection methods to retrieve significant performance parameters for any selected groups of interpretations. We designed three scenarios with different settings and demonstrated the importance of higher temporal resolution when analyzing the expressivity of performances. Furthermore, I helped to develop the MemoVision software, which utilizes parameter extraction, modified synchronization, feature selection, and visualization in a user-friendly and easy-to-use web environment (Chapter 9). I report additional MPA experiments and present the user interface of MemoVision software in Appendix D.

The main goal of this thesis was to utilize computational methods of music processing and MIR to analyze and compare differences between various interpretations of the same musical piece. This was achieved by first evaluating various automatic MIR tasks, such as onset, beat, and downbeat detection under specific conditions, such as degradations, that may be present in MPA datasets. Some degradations may lead to higher detection accuracy while extending other limitations of automated methods. The results of experiments and evaluations support a semi-automated approach in the case of MPA case studies instead of fully automated detectors. Based on these conclusions, I derived a modified synchronization approach and combined it with parameter extraction and feature selection methods. Furthermore, I presented two large and unique datasets of string quartet and piano music by Czech composers. I built on MIR methods, utilized them in comparative music analysis scenarios, and helped implement a software that provides our findings to musicologists and researchers in an easy-to-use environment.

In the future, automatic detectors may fully cover the needs of computational musicology and music performance analysis fields. They have not yet achieved the desired accuracy; the semi-automated approach based on music synchronization is still preferable. However, despite its advantages, it is not often used in MPA studies. Most of the recording annotation for MPA has been derived manually, even though less time-consuming and tedious options may exist. The final goal of this research was to bring the gap between MIR and MPA a little closer by openly sharing and demonstrating the advantages of computational methods in the field of music analysis.

Bibliography

[1] M. Ištvánek, “The effect of audio degradation on onset detection systems,” *Elektrorevue*, vol. 24, no. 1, pp. 1–13, 2022. [Online]. Available: http://hdl.handle.net/11012/214075

[2] M. Ištvánek, Z. Smékal, L. Spurný, and J. Mekyska, “Enhancement of conventional beat tracking system using teager–kaiser energy operator,” *Applied Sciences*, vol. 10, no. 1, pp. 1–20, 2020.

[3] ——, “Vylepšení metody výpočtu globálního tempa v rámci beat tracking systému na základě teagerova-kaiserova energetického operátoru,” in *Proceedings of the conference STUDENT EEICT 2019*, Brno, Czech Republic, 2019, pp. 398–400.

[4] J. Mekyska, M. Ištvánek, L. Spurný, and Z. Smékal, “Enhancement of beat tracking in string quartet music analysis based on the teager-kaiser energy operator,” in *International Conference on Telecommunications and Signal Processing (TSP)*, Budapest, Hungary, 2019, pp. 351–354.

[5] M. Ištvánek and Š. Miklánek, “Beat tracking: Is 44.1 khz really needed?” in *Proceedings of the conference STUDENT EEICT 2023 Selected Papers*, Brno, Czech Republic, 2023, pp. 227–231.

[6] M. Ištvánek, Š. Miklánek, K. H. Mühlová, L. Spurný, and Z. Smékal, “Application of computational methods for comparative music analysis,” in *4th International Symposium on the Internet of Sounds*, 2023, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/10335098

[7] M. Ištvánek and Š. Miklánek, “Memovision: a tool for feature selection and visualization of performance data,” in *Extended Abstracts for the Late-Breaking Demo Session of the 24th International Society for Music Information Retrieval Conference (ISMIR)*, Milan, Italy, 2023, p. 3. [Online]. Available: https://ismir2023program.ismir.net/lbd_322.html

[8] M. Ištvánek, “The application of tempo calculation for musicological purposes,” in *Proceedings of the conference STUDENT EEICT 2021 Selected Papers*, Brno, Czech Republic, 2021, pp. 265–269.

[9] M. Ištvánek and Š. Miklánek, “Exploring the possibilities of automated annotation of classical music with abrupt tempo changes,” in *Proceedings of the conference STUDENT EEICT 2022 Selected Papers*, Brno, Czech Republic, 2022, pp. 286–290.

[10] ——, “Towards automatic measure-wise feature extraction pipeline for music performance analysis,” in *45th International Conference on Telecommunications and Signal Processing (TSP)*, 2022, pp. 192–195.

[11] M. Ištvánek, Š. Miklánek, and L. Spurný, “Classification of interpretation differences in string quartets based on the origin of performers,” *Applied Sciences*, vol. 13, no. 6, pp. 1–20, 2023.

[12] M. Ištvánek, M. Mejzr, M. Schüller, F. Šír, and G. Tyson, “Metodika digitalizace fonografickýchválečků: proces a postupy digitálního přepisu fonografických válečků na přístroji endpoint,” methodology, p. 52, 2022.

[13] M. Ištvánek and F. Šír, *Záchrana zvukového kulturního dědictví: aktuální situace, problémy, možnosti.* Brno: Littera, 2022, ch. Digitální formáty používané pro archivaci zvukového záznamu, pp. 44–54.

[14] Y. Özer, M. Ištvánek, V. Arifi-Müller, and M. Müller, “Using activation functions for improving measure-level audio synchronization,” in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Bengalúru, India, 2022, pp. 750–756.

[15] L. Spurný, M. Ištvánek, and J. Jiraský, “Paměť zvuku (smetana–dvořák–janáček). k metodě zpracování zvukových nahrávek pomocí nástroje memovision,” *Hudební věda*, vol. 55, no. 4, pp. 501–533, 2023. [Online]. Available: https://hudebniveda.cz/2023/04/article-03.html

[16] M. Ištvánek, K. H. Mühlová, L. Spurný, and M. Mejzr, “Metodika analýzy hudebně-interpretačního výkonu s podporou the memovision software,” certified methodology, p. 56, 2023.

[17] M. Ištvánek, “The effect of audio degradation on onset detection systems,” 2022, accessed on January 1, 2024. [Online]. Available: https://github.com/xistva02/The-Effect-of-Audio-Degradation-on-Onset-Detection-Systems

[18] ——, “Beat tracking tcn,” 2023, accessed on January 1, 2024. [Online]. Available: https://github.com/xistva02/BeatTrackingTCN

[19] ——, “Classification of interpretation differences,” 2023, accessed on January 1, 2024. [Online]. Available: https://github.com/xistva02/Classification-of-interpretation-differences

[20] ——, “Audio duplicate finder,” 2023, accessed on January 1, 2024. [Online]. Available: https://github.com/xistva02/AudioDuplicateFinder

[21] Štěpán Miklánek and M. Ištvánek, “Memovision,” 2023, accessed on January 1, 2024. [Online]. Available: https://github.com/stepanmk/memovision

[22] M. Müller, *Fundamentals of Music Processing – Using Python and Jupyter Notebooks*, 2nd ed. Springer Verlag, 2021.

[23] Z. Smékal, *Číslicové zpracování signálů.* Vysoké učení technické v Brně, 2012.

[24] A. Lerch, C. Arthur, A. Pati, and S. Gururani, “An interdisciplinary review of music performance analysis,” *Trans. Int. Soc. Music Inf. Retr.*, vol. 3, no. 1, pp. 221–245, 2021. [Online]. Available: https://doi.org/10.5334/tismir.53

[25] P. Hill, *From score to sound.* Cambridge University Press, 2002, pp. 129–143.

[26] E. Clarke, *Understanding the psychology of performance.* Cambridge University Press, 2002, pp. 59–72.

[27] V. Bergeron and D. M. Lopes, “Hearing and seeing musical expression,” *Philosophy and Phenomenological Research*, vol. 78, no. 1, pp. 1–16, 2009. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1933-1592.2008.00230.x

[28] F. Platz and R. Kopiez, “When the Eye Listens: A Meta-analysis of How Audio-visual Presentation Enhances the Appreciation of Music Performance,” *Music Perception*, vol. 30, no. 1, pp. 71–83, 09 2012. [Online]. Available: https://doi.org/10.1525/mp.2012.30.1.71

[29] C.-J. Tsay, “Sight over sound in the judgment of music performance,” *Proceedings of the National Academy of Sciences*, vol. 110, no. 36, pp. 14 580–14 585, 2013. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.1221454110

[30] M. Müller, B. Pardo, G. J. Mysore, and V. Välimäki, “Recent advances in music signal processing,” *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 17–19, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8588415/

[31] M. Kassler, “Toward musical information retrieval,” *Perspectives of New Music*, vol. 4, no. 2, pp. 59–67, 1966. [Online]. Available: http://www.jstor.org/stable/832213

[32] M. Müller, *Fundamentals of Music Processing – Audio, Analysis, Algorithms, Applications.* Springer Verlag, 2015.

[33] A. Domínguez, "Highlights in the history of the fourier transform [retrospectroscope]," *IEEE Pulse*, vol. 7, no. 1, pp. 53–61, 2016.

[34] E. C. Titchmarsh, "Hankel transforms," *Proc. Camb. Philos. Soc.*, vol. 21, pp. 463–473, 1923.

[35] M. Plancherel, "Sur la convergence et sur la sommation par les moyennes de cesàro de f (x)cos xydx. z a z lim "3 #"," *Math. Ann.*, vol. 76, pp. 315–326, 1915.

[36] P. Lejeune-Dirichlet, "Sur la convergence des séries trigonométriques qui servent à représenter une fonction arbitraire entre des limites données," *Journal für die reine und angewandte Mathematik*, vol. 4, pp. 157–169, 1829.

[37] J. W. Cooley and J. W. Tukey, "An algorithm for the machine computation of complex fourier series," *Mathematics of Computation*, vol. 19, pp. 297–301, 1965.

[38] D. Gabor, "Theory of communications," *J. Inst. Elec. Eng.*, vol. 93, no. 3, pp. 429–457, 1946.

[39] F. Korzeniowski and G. Widmer, "Feature learning for chord recognition: The deep chroma extractor," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, New York, USA, 2016.

[40] M. Müller and S. Ewert, "Chroma toolbox: Matlab implementations for extracting variants of chroma-based audio features," in *International Society for Music Information Retrieval Conference (ISMIR), Miami, United States*, 2011, pp. 215–220.

[41] M. Müller, H. Mattes, and F. Kurth, "An efficient multiscale approach to audio synchronization," in *International Society for Music Information Retrieval Conference (ISMIR), Victoria, Canada*, 2006, pp. 192–197.

[42] S. Dixon and G. Widmer, "Match: A music alignment tool chest," in *International Society for Music Information Retrieval Conference (ISMIR), London, GB*, 2005.

[43] C. Raphael, "A hybrid graphical model for aligning polyphonic audio with musical scores," in *International Society for Music Information Retrieval Conference (ISMIR), Barcelona, Spain*, 2004, pp. 387–394.

[44] F. Kurth, M. Müller, C. Fremerey, Y. ha Chang, and M. Clausen, "Automated synchronization of scanned sheet music with audio recordings," in *International Society for Music Information Retrieval Conference (ISMIR), Vienna, Austria*, 2007, pp. 261–266.

[45] T. Prätzlich, J. Driedger, and M. Müller, "Memory-restricted multiscale dynamic time warping," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2016), Shanghai, China*, 2016.

[46] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *KDD Workshop*, U. M. Fayyad and R. Uthurusamy, Eds. AAAI Press, 1994, pp. 359–370. [Online]. Available: http://dblp.uni-trier.de/db/conf/kdd/kdd94.html#BerndtC94

[47] R. Agrawal, "Towards context-aware neural performance-score synchronisation," 2022.

[48] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[49] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, no. 1, pp. 67–72, 1975.

[50] Z. Geler, V. Kurbalija, M. Ivanović, M. Radovanović, and W. Dai, "Dynamic time warping: Itakura vs sakoe-chiba," in *IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA)*, 2019, pp. 1–6.

[51] S. Salvador and P. K. Chan, "Fastdtw: Toward accurate dynamic time warping in linear time and space," in *KDD Workshop on Mining Temporal and Sequential Data*, 2004.

[52] R. Macrae and S. Dixon, "Accurate real-time windowed time warping." in *International Society for Music Information Retrieval Conference (ISMIR)*. ISMIR, 2018, pp. 423–428. [Online]. Available: https://doi.org/10.5281/zenodo.1416156

[53] M. Cuturi and M. Blondel, "Soft-dtw: A differentiable loss function for time-series," in *Proceedings of the 34th International Conference on Machine Learning*, ser. ICML'17, vol. 70. JMLR.org, 2017, pp. 894–903.

[54] J. Zeitler, S. Deniffel, M. Krause, and M. Müller, “Stabilizing training with soft dynamic time warping: A case study for pitch class estimation with weakly aligned targets,” in *International Society for Music Information Retrieval Conference (ISMIR)*, 2023, pp. 433–439.

[55] I. Bükey, J. Zhang, and T. Tsai, “Flexdtw: Dynamic time warping with flexible boundary conditions,” in *International Society for Music Information Retrieval Conference (ISMIR)*, 2023, pp. 733–740.

[56] M. Müller, Y. Özer, M. Krause, T. Prätzlich, and J. Driedger, “Sync toolbox: A python package for efficient, robust, and accurate music synchronization,” *J. Open Source Softw.*, vol. 6, no. 64, p. 3434, 2021. [Online]. Available: https://doi.org/10.21105/joss.03434

[57] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler, “A tutorial on onset detection in music signals,” *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 5, pp. 1035–1047, 2005.

[58] L. C. H. L. Tan, Y. Zhu and S. Rahardja, “Audio onset detection using energy-based and pitch-based processing,” in *IEEE International Symposium on Circuits and Systems*, Paris, France, 2010, pp. 3689–3692.

[59] T. Maka, “A comparative study of onset detection methods in the presence of background noise,” in *International Conference on Signals and Electronic Systems (ICSES 2016)*, Krakow, Poland, 2016, pp. 51–56.

[60] B. L. M. Gainza, E. Coyle, “Onset detection using comb filters,” in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, NY, 2005, pp. 263–266.

[61] M. Mauch and S. Ewert, “The audio degradation toolbox and its application to robustness evaluation,” in *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, Curitiba, Brazil, 2013, pp. 83–88.

[62] B. McFee, E. J. Humphrey, and J. P. Bello, “A software framework for musical data augmentation,” in *International Society for Music Information Retrieval Conference (ISMIR)*, 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:852445

[63] P. J. M. M. Mohri and E. Weinstein, “Robust music identification, detection, and analysis,” in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, Vienna, Austria, 2007, pp. 135–138.

[64] Y. L. W. Li and X. Xiangyang, "Robust audio identification for mp3 popular music," in *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2010)*, Geneva, Switzerland, 2010, pp. 627–634.

[65] J. Osmalskyj and J.-J. Embrechts, "Effects of acoustic degradations on cover song recognition," in *Proceesings of the 22nd International Congress on Acoustics*, Buenos Aires, Argentina, 2016.

[66] N. Collins, "Using a pitch detector for onset detection," in *Proceeding of the 6th International Conference on Music Information Retrieval (ISMIR)*, London, UK, 2005.

[67] W. Wang, Y. Luo, J. A. Chambers, and S. Sanei, "Non-negative matrix factorization for note onset detection of audio signals," in *IEEE Signal Processing Society Workshop on Machine Learning for Signal Processing*, Maynooth, Ireland, 2006, pp. 447–452.

[68] J. Schlüter and S. Böck, "Improved musical onset detection with convolutional neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2014)*, Florence, Italy, 2014, pp. 6979–6983.

[69] ——, "Musical onset detection with convolutional neural networks," in *International Workshop on Machine Learning and Music (MML 2013)*, Prague, Czech Republic, 2013.

[70] A. Roebel, C. Jacques, and A. Aknin, "Mirex 2018: Training cnn onset detectors with artificially augmented datasets," in *International Society for Music Information Retrieval Conference (ISMIR)*, Paris, France, 2018.

[71] E. Marchi, G. Ferroni, F. Eyben, L. Gabrielli, S. Squartini, and B. Schuller, "Multi-resolution linear prediction based features for audio onset detection with bidirectional lstm neural networks," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2014)*, Florence, Italy, 2014, pp. 2164–2168.

[72] F. Eyben, S. Bock, B. Schuller, and A. Graves, "Universal onset detection with bidirectional long short term memory neural networks," in *International Conference on Music Information Retrieval (ISMIR)*, Utrecht, Netherlands, 2010, pp. 589–594.

[73] S. Böck and G. Widmer, "Maximum filter vibrato suppression for onset detection," in *International Conference on Digital Audio Effects (DAFx-13)*, Maynooth, Ireland, 2013.

[74] NEMA, “Mirex 2018: Audio onset detection – mirex05 dataset,” https://nema.lis.illinois.edu/nema_out/mirex2018/results/aod/, accessed: 2023-11-23.

[75] S. Böck, F. Krebs, and M. Schedl, “Evaluating the online capabilities of onset detection methods,” in *International Society for Music Information Retrieval Conference (ISMIR)*, Porto, Portugal, 2012, pp. 49–54.

[76] D. Stowell and M. D. Plumbley, “Adaptive whitening for improved real-time audio onset detection,” in *International Computer Music Conference (ICMC 2007)*, Copenhagen, Denmark, 2007, pp. 312–319.

[77] E. Benetos and S. Dixon, “Polyphonic music transcription using note onset and offset detection,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2011)*, Prague, Czech Republic, 2011, pp. 37–40.

[78] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis, “mir_eval: A transparent implementation of common mir metrics,” in *International Conference on Music Information Retrieval (ISMIR)*, Taipei, Taiwan, 2014, pp. 367–372.

[79] F. I. for Integrated Circuits IIS, “What is mp3?” https://www.mp3-history.com/en/whatismp3.html, accessed: 2023-11-23.

[80] J. F. Kaiser, “On a simple algorithm to calculate the ’energy’ of a signal,” in *International Conference on Acoustics, Speech, and Signal Processing*, Albuquerque, NM, 1990, pp. 381–384.

[81] A.-O. Boudraa and F. Salzenstein, “Teager–kaiser energy methods for signal and image analysis: A review,” *Digital Signal Processing*, vol. 78, pp. 338–375, 2018.

[82] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, “madmom: a new python audio and music signal processing library,” in *ACM International Conference on multimedia*, Amsterdam, The Netherlands, 10 2016, pp. 1174–1178.

[83] S. Böck and G. Widmer, “Local group delay based vibrato and tremolo suppression for onset detection,” in *International Society for Music Information Retrieval Conference (ISMIR)*, Curitiba, Brazil, 2013.

[84] A. Holzapfel, Y. Stylianou, A. C. Gedik, and B. Bozkurt, “Three dimensions of pitched instrument onset detection,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1517–1527, 2010.

[85] B. McFee, J. W. Kim, M. Cartwright, J. Salamon, R. M. Bittner, and J. P. Bello, “Open-source practices for music signal processing research: Recommendations for transparent, sustainable, and reproducible audio research,” *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 128–137, 2019.

[86] T. Suchánek, “Detektor tempa hudebních nahrávek na bázi neuronové sítě,” Master Thesis, Brno University of Technology, Department of Telecommunications, 2021.

[87] S. D’Amario, H. Daffern, and F. Bailes, “A new method of onset and offset detection in ensemble singing,” *Log. Phon. Voc.*, vol. 44, pp. 143–158, 2019.

[88] P. Grosche and M. Müller, “Cyclic tempogram—a mid-level tempo representation for music signals,” in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2010)*, Dallas, TX, USA, 2010, pp. 14–19.

[89] ——, “Extracting predominant local pulse information from music recordings,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 6, pp. 1688–1701, 2011.

[90] ——, “Tempogram toolbox: Matlab tempo and pulse analysis of music recordings,” in *International Conference on Music Information Retrieval (ISMIR)*, Miami, FL, USA, 2011, pp. 24–28.

[91] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, “librosa: Audio and music signal analysis in python,” in *Proceedings of the 14th python in science conference*, 2015, pp. 18–25.

[92] O. Lartillot and P. Toiviainen, “A matlab toolbox for musical feature extraction from audio,” in *Conference of the Gesellschaft für Klassifikation e.V.*, Breisgau, Germany, 2007.

[93] J. Zapata, M. Davies, and E. Gómez, “Multi-feature beat tracking,” *IEEE Transactions of Audio, Speech and Language Processing*, vol. 22, pp. 816–825, 2014.

[94] D. P. Ellis, “Beat tracking by dynamic programming,” *Journal of New Music Research*, vol. 36, no. 1, pp. 51–60, 2007.

[95] S. Böck, F. Krebs, and G. Widmer, “A multi-model approach to beat tracking considering heterogeneous music styles,” in *International Society for Music Information Retrieval Conference (ISMIR)*, Taipei, Taiwan, 2014, pp. 27–31.

[96] A. Srinivasamurthy, “A data-driven bayesian approach to automatic rhythm analysis of indian art music,” Ph.D. dissertation, Pompeu Fabra University, Barcelona, Spain, 2016.

[97] N. Cook, *Beyond the Score: Music as Performance.* Oxford, United Kingdom: Oxford University Press, 2013.

[98] J. A. Bowen, “Tempo, duration, and flexibility: Techniques in the analysis of performance,” *Journal of Musicol. Res.*, vol. 16, no. 2, pp. 111–156, 1996. [Online]. Available: https://doi.org/10.1080/01411899608574728

[99] S. Solnik, P. DeVita, P. Rider, B. Long, and T. Hortobágyi, “Teager-kaiser operator improves the accuracy of emg onset detection independent of signal-to-noise ratio,” *Acta Bioeng. Biomech.*, vol. 10, no. 2, 2008.

[100] S. Kopparapu, M. Pandharipande, and G. Sita, “Music and vocal separation using multiband modulation based features,” in *Proceedings of the Symposium on Industrial Electronics and Applications (ISIEA 2010)*, Penang, Malaysia, 2010.

[101] S. Das and J. Hansen, “Detection of voice onset time (vot) for unvoiced stops (/p/, /t/, /k/) using the teager energy operator (teo) for automatic detection of accented english,” in *Proceedings of the 6th Nordic Signal Processing Symposium (NORSIG 2004)*, Espoo, Finland, 2004.

[102] R. Hamila, A. Lakhzouri, E. Lohan, and M. Renfors, “A highly efficient generalized teager-kaiser-based technique for los estimation in wcdma mobile positioning,” *EURASIP J. Appl. Signal Process.*, vol. 5, pp. 698–708, 2005.

[103] E. Kvedalen, “Signal processing using the teager energy operator and other nonlinear operators,” Master Thesis, University of Oslo, Oslo, Norway, 2003.

[104] D. Dimitriadis, A. Potamianos, and P. Maragos, “A comparison of the squared energy and teager-kaiser operators for short-term energy estimation in additive noise,” *IEEE Trans. Signal Process.*, vol. 57, pp. 2569–2581, 2009.

[105] V. Konz, “Automated methods for audio-based music analysis with applications to musicology,” Ph.D. dissertation, Saarland University, Saarbrücken, Germany, 2012.

[106] F. Krebs, S. Böck, and G. Widmer, “An efficient state-space model for joint tempo and meter tracking,” in *International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[107] A. Holzapfel, M. Davies, J. Zapata, J. Oliveira, and F. Gouyon, "Selective sampling for beat tracking evaluation," *IEEE Trans. Audio Speech Lang. Process.*, vol. 20, pp. 2539–2548, 2012.

[108] Q. M. U. o. L. Centre for Digital Music, "Sonic visualiser," https://www.sonicvisualiser.org/, (accessed on 29 November 2023).

[109] S. Dixon, "An interactive beat tracking and visualisation system," in *International Computer Music Conference (ICMC 2001)*, Havana, Cuba, 2001.

[110] S. Böck, J. S. Cardoso, and M. E. P. Davies, "Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[111] S. Böck and M. Schedl, "Enhanced beat tracking with context-aware neural networks," in *Proceedings of the 14th International Conference on Digital Audio Effects (DAFx-11)*, 2011.

[112] S. Böck, F. Krebs, and G. Widmer, "Accurate tempo estimation based on recurrent neural networks and resonating comb filters," in *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[113] A. van den Oord *et al.*, "WaveNet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499v2*, 2016.

[114] M. E. P. Davies and S. Böck, "Temporal convolutional networks for musical audio beat tracking," in *2019 27th European Signal Processing Conference (EUSIPCO)*, 2019.

[115] C. J. Steinmetz and J. D. Reiss, "Wavebeat: End-to-end beat and downbeat tracking in the time domain," in *151st Audio Engineering Society Convention*, 2021.

[116] S. Böck, F. Krebs, and G. Widmer, "Joint beat and downbeat tracking with recurrent neural networks," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[117] M. E. P. Davies, S. Böck, and M. Fuentes, "Tempo, beat and downbeat estimation," November 2021, (accessed on 27 March 2023).

[118] F. Krebs, S. Böck, and G. Widmer, "Rhythmic pattern modeling for beat and downbeat tracking in musical audio," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2013.

[119] S. W. Hainsworth and M. D. Macleod, “Particle filtering applied to musical tempo tracking,” *EURASIP Journal on Advances in Signal Processing*, vol. 15, 2004.

[120] G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.

[121] A. Holzapfel, M. E. P. Davies, J. R. Zapata, J. L. Oliveira, and F. Gouyon, “Selective sampling for beat tracking evaluation,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 9, pp. 2539–2548, 2012.

[122] M. E. Davies, N. Degara, and M. D. Plumbley, “Evaluation methods for musical audio beat tracking algorithms,” Queen Mary University of London, Centre for Digital Music, Tech. Rep., 2009.

[123] M. F. McKinney, D. Moelants, M. E. P. Davies, and A. Klapuri, “Evaluation of audio beat tracking and music tempo extraction algorithms,” *Journal of New Music Research*, vol. 36, no. 1, pp. 1–16, 2007.

[124] C. f. t. H. AHRC Research and A. of Recorded Music (CHARM), “Mazurka project,” http://www.mazurka.org.uk, accessed: 2023-07-11.

[125] D. o. C. S. UK: University of London, “Machine intelligence and music informatics research group,” http://mirg.city.ac.uk/projects, (accessed on 15 January 2021).

[126] D. Damm, C. Fremerey, F. Kurth, M. Müller, and M. Clausen, “Multimodal presentation and browsing of music,” in *ICMI ’08: Proceedings of the 10th international conference on Multimodal interfaces.* New York, NY, USA: Association for Computing Machinery, 2008, p. 205–208. [Online]. Available: https://doi.org/10.1145/1452392.1452436

[127] M. community, “2012:mirex home,” https://www.music-ir.org/mirex/wiki/2012:MIREX_Home, (accessed on 29 November 2023).

[128] M. Schedl, E. Gómez, and J. Urbano, “Music information retrieval: Recent developments and applications,” *Found. Trends Inf. Retr.*, vol. 8, no. 2-3, pp. 127–261, 2014. [Online]. Available: https://doi.org/10.1561/1500000042

[129] C. Weiß, V. Arifi-Müller, T. Prätzlich, R. Kleinertz, and M. Müller, “Analyzing measure annotations for western classical music recordings,” in *17th International Society for Music Information Retrieval Conference (ISMIR), New York City, NY, USA*, 2016.

[130] M. Müller, F. Kurth, and M. Clausen, “Audio matching via chroma-based statistical features,” in *International Conference on Music Information Retrieval (ISMIR), London, United Kingdom*, 2005.

[131] A. S. Pinto, I. Domingues, and M. E. P. Davies, “Shift if you can: Counting and visualising correction operations for beat tracking evaluation,” in *Extended Abstracts for the Late-Breaking Demo Session of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, Virtual, 2020.

[132] M. Müller, T. Prätzlich, and J. Driedger, “A cross-version approach for stabilizing tempo-based novelty detection,” in *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR)*, 2012, pp. 427–432.

[133] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, “Panns: Large-scale pretrained audio neural networks for audio pattern recognition,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[134] F. Seddon and M. Biasutti, “A comparison of modes of communication between members of a string quartet and a jazz sextet,” *Psychol. Music*, vol. 37, no. 4, pp. 395–415, 2009. [Online]. Available: https://doi.org/10.1177/0305735608100375

[135] L. Bishop, C. Cancino-Chacón, and W. Goebl, “Moving to communicate, moving to interact: Patterns of body motion in musical duo performance,” *Music Percept.*, vol. 37, no. 1, pp. 1–25, 2019. [Online]. Available: https://doi.org/10.1525/mp.2019.37.1.1

[136] P. Papiotis, M. Marchini, A. Perez-Carrillo, and E. Maestre, “Measuring ensemble interdependence in a string quartet through analysis of multidimensional performance data,” *Front. Psychol.*, vol. 5, p. 963, 2014. [Online]. Available: https://doi.org/10.3389/fpsyg.2014.00963

[137] G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” *IEEE Trans. Audio Speech Lang. Process.*, vol. 10, no. 5, pp. 293–302, 2002. [Online]. Available: https://doi.org/10.1109/TSA.2002.800560

[138] K. Seyerlehner, M. Schedl, T. Pohle, and P. Knees, “Using block-level features for genre classification, tag classification and music similarity estimation,” http://www.cp.jku.at/people/schedl/Research/Publications/pdf/MIREX_SSPK2_2010.pdf, (accessed on 6 November 2022).

[139] S. Mo and J. Niu, "A novel method based on ompgw method for feature extraction in automatic music mood classification," *IEEE Trans. Affect. Comput*, vol. 10, no. 3, pp. 313–324, 2017. [Online]. Available: https://doi.org/10.1109/TAFFC.2017.2724515

[140] E. Liebman, E. Ornoy, and B. Chor, "A phylogenetic approach to music performance analysis," *Journal of New Music Research*, vol. 41, no. 2, pp. 195–222, 2012. [Online]. Available: https://doi.org/10.1080/09298215.2012.668194

[141] R. Hillewaere, B. Manderick, and D. Conklin, "String quartet classification with monophonic models," in *International Society for Music Information Retrieval Conference (ISMIR), Utrecht, The Netherlands*, 2010.

[142] K. C. Kempfert and S. W. Wong, "Where does haydn end and mozart begin? composer classification of string quartets," *J. New Music Res.*, vol. 49, no. 5, pp. 457–476, 2020. [Online]. Available: https://doi.org/10.1080/09298215.2020.1814822

[143] A. Lykartsis and A. Lerch, "Beat histogram features for rhythm-based musical genre classification using multiple novelty functions," in *18th International Conference on Digital Audio Effects (DAFx-15), Trondheim, Norway*, 2015.

[144] T. Kiska, Z. Galáž, V. Zvončák, J. Mucha, J. Mekyska, and Z. Smékal, "Music information retrieval techniques for determining the place of origin of a music interpretation," in *10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), Moscow, Russia*, 5–9 November 2018.

[145] N. Cook, *Analysing performance and performing analysis.* Oxford, United Kingdom: Oxford University Press, 1999, ch. 11, pp. 239–261.

[146] C. S. Sapp, "Comparative analysis of multiple musical performances," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Vienna, Austria, 2007, pp. 497–500.

[147] C. E. Cancino-Chacón, M. Grachten, W. Goebl, and G. Widmer, "Computational models of expressive music performance: A comprehensive and critical review," *Frontiers Digit. Humanit.*, vol. 5, p. 25, 2018. [Online]. Available: https://doi.org/10.3389/fdigh.2018.00025

[148] C. E. Cancino-Chacón, T. Gadermaier, G. Widmer, and M. Grachten, "An evaluation of linear and non-linear models of expressive dynamics in classical

piano and symphonic music," *Machine Learning*, vol. 106, no. 6, pp. 887–909, 2017.

[149] C. E. C. Chacón, M. Bonev, A. Durand, M. Grachten, A. Arzt, L. Bishop, W. Goebl, and G. Widmer, "The accompanion v0.1: an expressive accompaniment system," in *In Late Breaking Demo, 18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017.

[150] G. Xia, Y. Wang, R. Dannenberg, and G. Gordon, "Spectral learning for expressive interactive ensemble music performance," in *International Society for Music Information Retrieval Conference (ISMIR), Malaga, Spain*, 2015, pp. 816–822.

[151] F. Henkel, S. Balke, M. Dorfer, and G. Widmer, "Score following as a multimodal reinforcement learning problem," *Transactions of the International Society for Music Information Retrieval*, vol. 2, no. 1, pp. 66–81, 2019.

[152] Cancino-Chacón, C. Eduardo, Grachten, and Maarten, "The basis mixer: A computational romantic pianist," in *Proceedings of the Late Breaking/ Demo Session, 17th International Society for Music Information Retrieval Conference (ISMIR)*, New York, NY, USA, 2016.

[153] J. Schlüter, "Deep learning for event detection, sequence labelling and similarity estimation in music signals," Ph.D. dissertation, Johannes Kepler University, 2017.

[154] S. Ewert, M. Müller, and P. Grosche, "High resolution audio synchronization using chroma onset features," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2009)*, Taipei, Taiwan, Apr. 2009, pp. 1869–1872.

[155] A. Lerch, "Software based extraction of objective parameters from music performances," Ph.D. dissertation, Technischen Universität Berlin, 2009.

[156] A. S. Pinto, S. Böck, J. S. Cardoso, and M. E. P. Davies, "User-driven fine-tuning for beat tracking," *Electronics*, vol. 10, no. 13, p. 1518, 2021. [Online]. Available: https://doi.org/10.3390/electronics10131518

[157] C. Ding and H. Peng, "Minimum redundancy feature selection from microarray gene expression data," *J. Bioinform. Comput. Biol*, vol. 3, no. 2, pp. 185–205, 2003. [Online]. Available: https://doi.org/10.1142/s0219720005001004

[158] Z. Zhao, R. Anand, and M. Wang, “Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform,” in *International Conference on Data Science and Advanced Analytics (DSAA 2019), Washington DC, USA*, 2019.

[159] P. Drotár, J. Mekyska, I. Rektorová, L. Masarová, Z. Smékal, and M. Faundez-Zanuy, “Analysis of in-air movement in handwriting: A novel marker for parkinson’s disease,” *Comput. Methods Programs Biomed.*, vol. 117, no. 3, pp. 405–411, 2014. [Online]. Available: https://doi.org/10.1016/j.cmpb.2014.08.007

[160] B.-Q. Li, L.-L. Hu, L. Chen, K.-Y. Feng, Y.-D. Cai, and K.-C. Chou, “Prediction of protein domain with mrmr feature selection and analysis,” *PloS One*, vol. 7, no. 6, 2012. [Online]. Available: https://doi.org/10.1371/journal.pone.0039308

[161] H. Peng, F. Long, and C. Ding, “Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, 2005. [Online]. Available: https://doi.org/10.1109/TPAMI.2005.159

[162] C.-C. Chang and C.-J. Lin, “Libsvm: A library for support vector machines,” *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, 2011. [Online]. Available: https://doi.org/10.1145/1961189.1961199

[163] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett, “New support vector algorithms,” *Neural Comput.*, vol. 12, no. 5, pp. 1207–1245, 2000. [Online]. Available: https://doi.org/10.1162/089976600300015565

[164] C. Cortes and V. Vapnik, “Support-vector networks,” *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995. [Online]. Available: https://doi.org/10.1007/BF00994018

[165] M. Müller, D. P. W. Ellis, A. Klapuri, and G. Richard, “Signal processing for music analysis,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 6, pp. 1088–1110, 2011.

[166] D. Meredith, *Computational Music Analysis.* Springer, 2016.

[167] C. McKay and I. Fujinaga, “jsymbolic: A feature extractor for midi files,” in *International Conference on Mathematics and Computing*, 2006. [Online]. Available: https://api.semanticscholar.org/CorpusID:14036386

[168] F. Kurth, M. Müller, D. Damm, C. Fremerey, A. Ribbrock, and M. Clausen, "Syncplayer - an advanced system for multimodal music access," in *International Conference on Music Information Retrieval (ISMIR), London, UK, 11-15 September 2005, Proceedings*, 2005, pp. 381–388. [Online]. Available: http://ismir2005.ismir.net/proceedings/1025.pdf

[169] M. Goto and R. B. Dannenberg, "Music interfaces based on automatic music signal analysis: New ways to create and listen to music," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 74–81, 2019.

[170] F. Zalkow, C. Weiß, T. Prätzlich, V. Arifi-Müller, and M. Müller, "A multi-version approach for transferring measure annotations between music recordings," in *AES Conference on Semantic Audio*, Erlangen, Germany, 2017. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=18772

[171] V. Konz, M. Müller, and R. Kleinertz, "A cross-version chord labelling approach for exploring harmonic structures—a case study on Beethoven's Appassionata," *Journal of New Music Research*, vol. 42, no. 1, pp. 61–77, 2013.

[172] C. Weiß, H. Schreiber, and M. Müller, "Local key estimation in music recordings: A case study across songs, versions, and annotators," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2919–2932, 2020.

[173] T. Prätzlich and M. Müller, "Freischütz Digital: A case study for reference-based audio segmentation of operas," in *International Society for Music Information Retrieval Conference (ISMIR)*, Curitiba, Brazil, 2013, pp. 589–594.

[174] C. Fremerey, M. Müller, and M. Clausen, "Handling repeats and jumps in score-performance synchronization," in *International Conference on Music Information Retrieval (ISMIR), Utrecht, Netherlands*, 2010, pp. 243–248.

[175] M. Shan and T. Tsai, "Improved handling of repeats and jumps in audio–sheet image synchronization," in *In Proc. of the 21st International Society for Music Information Retrieval Conference (ISMIR), Montréal, Canada*, 2020.

[176] C. J. Steinmetz and J. D. Reiss, "pyloudnorm: A simple yet flexible loudness meter in python," in *150th AES Convention*, 2021.

List of Appendices

A Audio Degradation for Onset Detection

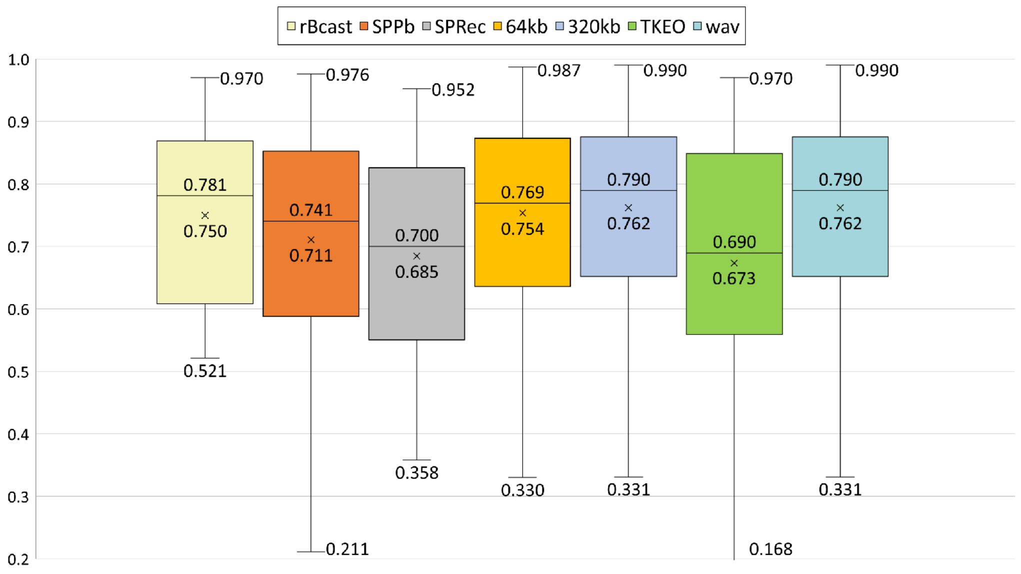


Fig. A.1: Box plot with F-score values (50 ms window) for all degradations; all detectors averaged.

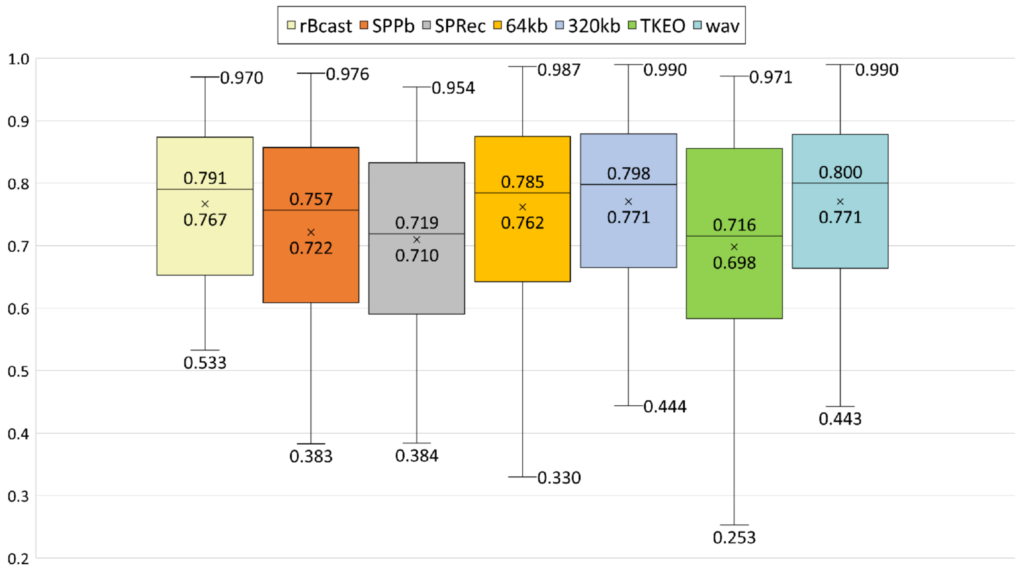


Fig. A.2: Box plot showing the F-score values (100 ms window) for all degradations; all detectors averaged.

Tab. A.1: The F-score for all categories, degradations, and systems; 50 ms window.

detector	cat	rBcast	SPPb	SPRec	64kb	320kb	TKEO	wav
CF	BS	0.727	**0.731**	0.583	0.686	0.689	0.607	0.689
	CM	0.799	0.780	0.730	0.799	**0.802**	0.752	**0.802**
	NPP	0.915	0.952	0.877	0.952	**0.954**	0.937	**0.954**
	PP	0.843	0.854	0.809	**0.861**	0.859	0.859	0.859
	vocal	0.568	0.560	0.427	0.568	**0.570**	0.399	**0.570**
	WI	0.710	0.727	0.642	0.757	0.766	0.638	**0.768**
Lib	BS	0.597	0.626	0.564	0.646	**0.648**	0.573	**0.648**
	CM	0.551	0.764	0.710	0.773	**0.777**	0.746	**0.777**
	NPP	0.884	0.964	0.915	0.962	0.960	**0.970**	0.960
	PP	0.841	0.830	0.820	**0.846**	0.845	0.845	0.845
	vocal	**0.525**	0.414	0.358	0.433	0.417	0.356	0.416
	WI	**0.612**	0.595	0.543	0.595	0.602	0.597	0.601
CNN	BS	0.803	0.743	0.677	0.765	0.806	0.643	**0.807**
	CM	0.869	0.781	0.793	0.866	**0.879**	0.736	**0.879**
	NPP	0.970	0.976	0.952	0.987	**0.990**	0.920	**0.990**
	PP	0.911	0.894	0.891	0.929	**0.937**	0.886	**0.937**
	vocal	0.714	0.566	0.468	0.638	0.739	0.399	**0.740**
	WI	0.841	0.738	0.742	0.841	**0.871**	0.745	0.870
RNN	BS	**0.641**	0.377	0.514	0.554	0.573	0.481	0.573
	CM	**0.763**	0.492	0.690	0.751	0.756	0.631	0.756
	NPP	0.943	0.895	0.920	0.962	**0.963**	0.895	**0.963**
	PP	0.869	0.812	0.843	0.872	**0.874**	0.815	**0.874**
	vocal	**0.521**	0.211	0.380	0.330	0.331	0.168	0.331
	WI	0.533	0.450	0.643	0.630	**0.653**	0.586	**0.653**
SF	BS	**0.708**	0.704	0.553	0.695	0.696	0.517	0.695
	CM	0.845	0.830	0.753	0.844	**0.846**	0.767	**0.846**
	NPP	0.910	0.942	**0.946**	0.944	0.942	0.931	0.942
	PP	0.845	0.852	0.819	**0.876**	0.873	0.832	0.873
	vocal	0.568	**0.605**	0.389	0.584	0.583	0.389	0.583
	WI	**0.662**	0.657	0.585	0.660	0.657	0.577	0.657

Tab. A.2: The F-score for all categories, degradations, and systems; 100 ms window.

sys	cat	rBcast	SPPb	SPRec	64kb	320kb	TKEO	wav
CF	BS	0.757	**0.761**	0.637	0.698	0.703	0.647	0.704
	CM	**0.817**	0.794	0.771	0.812	0.816	0.773	0.816
	NPP	0.918	0.953	0.880	0.953	**0.956**	0.937	**0.956**
	PP	0.849	0.860	0.818	0.865	0.864	**0.867**	0.864
	vocal	0.585	0.580	0.471	**0.593**	0.590	0.447	0.590
	WI	0.735	0.753	0.687	0.786	0.790	0.669	**0.792**
Lib	BS	0.608	0.641	0.595	**0.673**	0.665	0.621	0.664
	CM	0.556	0.778	0.734	0.783	**0.787**	0.763	**0.787**
	NPP	0.884	0.964	0.917	0.963	0.961	**0.971**	0.961
	PP	0.844	0.834	0.824	0.848	0.847	**0.852**	0.847
	vocal	**0.556**	0.434	0.384	0.458	0.444	0.399	0.443
	WI	**0.636**	0.618	0.578	0.608	0.620	0.622	0.619
CNN	BS	**0.812**	0.745	0.704	0.765	0.806	0.680	0.808
	CM	0.874	0.785	0.810	0.869	**0.882**	0.751	0.881
	NPP	0.970	0.976	0.954	0.987	**0.990**	0.920	**0.990**
	PP	0.913	0.895	0.895	0.931	**0.938**	0.893	**0.938**
	vocal	0.717	0.566	0.518	0.638	0.739	0.459	**0.740**
	WI	0.852	0.741	0.783	0.848	**0.878**	0.764	0.877
RNN	BS	**0.658**	0.383	0.535	0.558	0.577	0.538	0.576
	CM	**0.769**	0.496	0.703	0.754	0.759	0.654	0.759
	NPP	0.945	0.895	0.920	0.962	**0.963**	0.896	**0.963**
	PP	0.874	0.817	0.848	0.873	0.875	0.822	**0.876**
	vocal	**0.533**	0.218	0.425	0.330	0.334	0.253	0.334
	WI	**0.716**	0.466	0.680	0.644	0.664	0.619	0.664
SF	BS	**0.732**	0.724	0.596	0.703	0.706	0.548	0.705
	CM	**0.857**	0.844	0.787	0.853	0.856	0.792	0.856
	NPP	0.910	0.944	**0.947**	0.944	0.944	0.931	0.944
	PP	0.852	0.856	0.828	**0.880**	0.877	0.840	0.877
	vocal	0.587	**0.635**	0.426	0.591	0.603	0.416	0.603
	WI	0.687	**0.691**	0.633	0.689	0.684	0.595	0.684

B MPA-oriented GT Computation

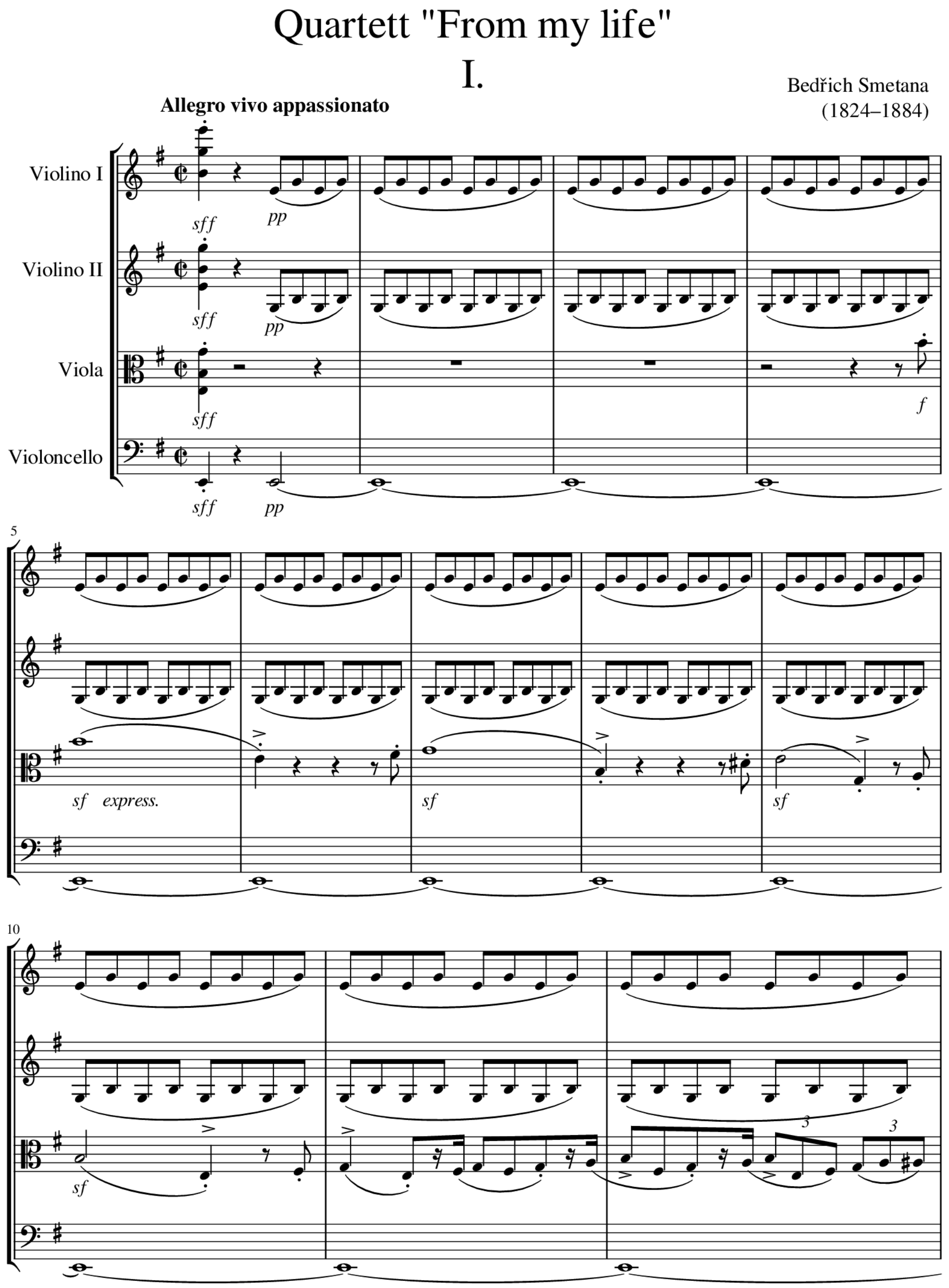


Fig. B.1: The beginning of the first movement of Smetana's String Quartet No. 1.

Tab. B.1: The EAT of all motifs of the second database.

Track	Beg	A	B	C	D	E
CD01	80.61	69.37	34.41	88.56	55.60	74.50
CD02	77.80	69.03	44.14	81.84	59.52	72.43
CD03	77.93	73.19	41.60	87.09	62.36	79.14
CD04	80.48	75.08	48.98	80.37	69.14	80.29
CD05	69.54	66.78	32.83	80.31	54.60	71.15
CD06	72.74	72.14	42.29	74.41	61.98	76.16
CD07	75.94	65.71	39.33	83.06	58.04	78.17
CD08	69.69	66.42	34.47	79.98	53.66	75.13
CD09	83.43	72.48	40.13	83.70	60.57	72.67
CD10	82.92	72.18	40.70	88.56	61.31	74.12
CD11	70.92	63.49	43.34	73.65	56.77	67.07
CD12	82.35	70.91	48.36	83.76	63.02	77.76
CD13	69.27	61.71	45.28	79.08	54.63	70.93
CD14	81.28	69.06	46.33	88.24	61.20	77.35
CD15	74.60	68.23	30.19	85.12	54.28	69.92
CD16	79.06	68.87	39.59	87.81	59.46	76.42
CD17	71.01	58.38	37.35	75.82	51.37	68.52
CD18	72.79	71.59	51.06	75.13	64.15	70.81
CD19	74.14	73.17	56.74	80.47	69.02	73.39
CD20	77.35	74.48	51.75	82.27	68.16	80.73
CD21	77.16	71.72	47.76	82.49	64.75	73.03
CD22	73.36	62.91	42.74	81.54	55.77	74.87
CD23	73.04	65.89	34.78	80.50	53.31	77.35
CD24	78.50	78.14	58.36	79.81	70.06	80.80
CD25	75.61	72.73	44.04	80.70	62.30	73.63
CD26	83.75	77.92	46.27	93.48	66.58	84.73
CD27	82.60	76.43	49.74	83.26	68.00	76.29
CD28	72.92	65.80	48.48	80.62	62.24	71.73
CD29	70.25	63.09	37.87	74.09	55.33	58.12
CD30	68.26	65.35	38.17	70.01	56.31	67.07
CD31	73.78	71.06	43.15	79.71	59.63	75.56
CD32	83.58	72.07	40.00	82.14	60.55	71.04
CD33	76.92	63.24	42.05	74.62	56.26	68.31

All values are in BPM.

Tab. B.2: Reference GT and computed GT of the reference database.

Track No.	Reference	TS	DS	TS Dev.	DS Dev.
1	48.15	47.85	47.85	0.30	0.30
2	66.99	73.83	73.83	6.84	6.84
3	68.00	95.70	95.70	27.70	27.70
4	60.41	48.75	68.00	11.66	7.59
5	39.71	42.36	42.36	2.65	2.65
6	62.76	47.85	47.85	14.91	14.91
7	53.67	54.98	54.98	1.31	1.31
8	136.05	136.00	143.55	0.05	7.50
9	55.15	56.17	56.17	1.02	1.02
10	75.86	80.75	78.30	4.89	2.44
11	91.63	95.70	95.70	4.07	4.07
12	87.27	86.13	184.57	1.14	97.30
13	93.75	99.38	95.70	5.63	1.95
14	75.38	86.13	89.10	10.75	13.72
15	35.34	42.36	42.36	7.02	7.02
16	70.01	66.26	66.26	3.75	3.75
17	72.20	73.83	73.83	1.63	1.63
18	82.87	89.10	117.45	6.23	34.58
19	41.99	46.98	42.36	4.99	0.37
20	80.65	99.38	123.05	18.73	42.40
21	72.73	83.35	78.30	10.62	5.57
22	35.09	44.55	63.02	9.46	27.93
23	89.71	172.27	172.27	82.56	82.56
24	51.81	51.68	51.68	0.13	0.13
25	63.56	99.38	103.36	35.82	39.80
26	117.46	129.20	129.20	11.74	11.74
27	200.00	198.77	184.57	1.23	15.43
28	116.73	117.45	61.52	0.72	55.21
29	95.09	83.35	123.05	11.74	27.96
30	63.36	63.02	63.02	0.34	0.34
Average	76.78	83.75	88.97	9.99	18.19
P-value		0.038	0.024		

TS—System with the TKEO; DS—Default system without the TKEO; Dev.—deviation from the reference global tempo; P—*p*-value for the *t*-Test (Paired Two Sample for Means), $\alpha = 0.05$. All values are in BPM.

Tab. B.3: Differences between the estimated GT and the EAT for both systems.

	TS						DS					
Track	Beg	A	B	C	D	E	Beg	A	B	C	D	E
CD01	15.09	13.98	6.61	3.73	5.92	3.80	8.49	22.92	6.61	3.73	1.82	3.80
CD02	14.49	0.81	6.53	1.51	6.74	3.57	14.49	9.27	2.84	1.51	3.50	1.40
CD03	14.36	1.41	7.15	5.20	4.94	6.99	11.17	10.16	11.13	5.20	13.64	6.99
CD04	0.27	0.92	34.37	0.38	17.16	3.06	0.27	3.22	31.77	0.38	9.16	3.06
CD05	4.29	3.06	4.62	0.44	10.00	7.15	8.76	11.52	10.24	3.04	21.40	4.89
CD06	3.26	6.16	8.38	1.59	4.28	4.59	3.26	6.16	5.56	1.59	14.02	4.59
CD07	7.41	8.12	8.42	3.07	9.96	2.58	7.41	6.07	14.50	3.07	15.79	7.96
CD08	11.06	9.58	7.21	3.37	10.94	5.62	11.06	9.58	6.55	0.77	10.94	5.62
CD09	8.86	5.82	3.67	2.43	15.43	3.33	8.86	5.82	3.22	2.43	15.43	5.63
CD10	12.78	6.12	15.47	0.54	12.52	4.18	9.37	3.82	10.98	0.54	3.89	6.63
CD11	7.38	4.51	16.75	2.35	7.83	2.77	5.08	0.47	16.75	2.35	0.65	0.93
CD12	1.00	6.31	7.81	3.01	3.24	2.99	1.00	7.39	5.47	0.41	1.58	5.59
CD13	9.03	14.29	10.89	1.67	5.46	2.90	6.73	19.04	8.55	1.67	15.21	2.90
CD14	11.01	11.69	8.65	0.86	8.64	3.40	11.01	6.94	8.65	4.05	0.32	8.78
CD15	11.53	12.52	8.38	1.01	5.81	10.83	8.75	12.52	7.81	1.01	19.55	19.18
CD16	13.23	7.13	12.09	1.29	10.38	2.59	13.23	9.43	5.74	1.29	8.54	1.88
CD17	7.29	13.40	5.72	2.48	4.80	5.31	9.74	0.96	9.63	0.18	2.46	9.78
CD18	5.51	6.71	10.46	0.87	1.13	21.48	3.21	6.71	0.62	0.87	4.06	0.97
CD19	1.86	10.18	4.01	0.28	11.73	4.91	1.86	7.58	13.10	0.28	9.28	4.91
CD20	3.40	8.87	0.98	1.08	3.56	0.02	3.40	6.27	0.98	1.08	0.16	11.56
CD21	6.19	4.28	2.91	0.86	11.25	7.72	8.97	6.58	10.97	0.86	0.15	7.72
CD22	7.39	3.35	11.09	1.81	14.07	11.26	12.77	1.69	9.99	1.81	5.75	3.43
CD23	16.06	3.95	10.55	2.85	2.86	3.40	16.06	10.11	5.59	2.85	24.99	3.40
CD24	0.20	5.21	1.73	0.94	10.69	8.30	2.25	2.61	0.37	0.94	10.69	2.55
CD25	2.69	8.02	0.51	2.65	16.00	2.37	2.69	10.62	4.71	2.65	9.57	2.37
CD26	8.54	6.14	11.15	2.22	16.77	7.56	5.35	6.14	11.15	2.22	14.17	4.37
CD27	3.53	6.92	5.24	0.09	15.35	0.29	3.53	6.92	4.09	0.09	12.75	4.46
CD28	5.38	10.20	2.19	2.73	7.60	9.02	0.91	1.20	3.20	2.73	7.60	11.62
CD29	8.05	1.51	8.27	4.21	10.93	0.61	8.05	1.57	19.55	4.21	7.69	0.70
CD30	7.74	2.33	7.16	0.17	2.48	22.03	0.26	5.26	6.38	2.01	9.95	6.76
CD31	4.52	2.77	6.54	1.04	10.21	0.44	4.52	2.77	14.27	1.04	14.20	0.44
CD32	12.12	3.93	3.07	1.21	13.28	2.79	12.12	8.68	6.98	1.21	15.45	7.26
CD33	3.83	6.60	9.63	0.79	5.26	3.47	3.83	6.60	9.63	0.79	11.74	5.52
Average	7.56	6.57	8.13	1.78	9.01	5.49	6.92	7.17	8.71	1.78	9.58	5.38
Result			6.42						6.59			

All values are in BPM—Beats Per Minute.

C Classification of Interpretation Differences

Tab. C.1: Dvořák's subset.

composer	composition	movement	no of recs	class 1	class 0
Dvořák	No. 12	mov1	51	11	40
		mov2	73	18	55
		mov3	72	17	55
		mov4	75	17	58
		Σ	**271**	**63**	**208**
	No. 13	mov1	25	10	15
		mov2	25	10	15
		mov3	22	8	14
		mov4	22	8	14
		Σ	**94**	**36**	**58**
	No. 14	mov1	22	10	12
		mov2	7	2	5
		mov3	23	10	13
		mov4	21	8	13
		Σ	**73**	**30**	**43**

Tab. C.2: Janáček's subset.

composer	composition	movement	no of recs	class 1	class 0
Janáček	No. 1	mov1	65	22	43
		mov2	66	22	44
		mov3	66	22	44
		mov4	66	22	44
		Σ	**263**	**88**	**175**
	No. 2	mov1	67	18	49
		mov2	66	19	47
		mov3	60	20	40
		mov4	69	19	50
		Σ	**262**	**76**	**186**

Tab. C.3: Smetana's subset.

composer	composition	movement	no of recs	class 1	class 0
Smetana	No. 1	mov1	60	27	33
		mov2	36	16	20
		mov3	35	15	20
		mov4	33	16	17
		Σ	**164**	**74**	**90**
	No. 2	mov1	26	21	5
		mov2	26	21	5
		mov3	26	21	5
		mov4	23	19	4
		Σ	**101**	**82**	**19**

D Additional Experiments

In this chapter, I summarize the previous work by presenting the utilization of proposed methods within the MemoVision software we created in the project TA ČR TL05000527, Memories of Sound. The software was presented at the International Symposium on the Internet of Sounds [6] and ISMIR conferences [7], respectively. Furthermore, we created a certified methodology called "The Methodology of Music Performance Analysis Using The MemoVision Software" [16]. It aims to provide related MPA studies, explain the software's core functions, and demonstrate them on piano and string quartet recordings. In the following sections, I outline the connection of MIR methods with the goals of MPA using the software, which is also one of the main contributions to the MIR and MPA communities. I use the user interface and visualization features to present a small case study and the possibilities of data-driven semi-automated performance analysis.

D.1 Software

This chapter provides a brief user-case study on piano recordings from the Czech composer Bedřich Smetana, *Neighbor's Dance* to demonstrate its functionalities. The software interface in [6] and [7] is slightly outdated compared to its current state (January 7, 2024). The following sections present options and possibilities for comparative performance analysis using MemoVision software. Note that all figures come from the software as screenshots and lack vector graphics quality.

D.2 Data Preparation

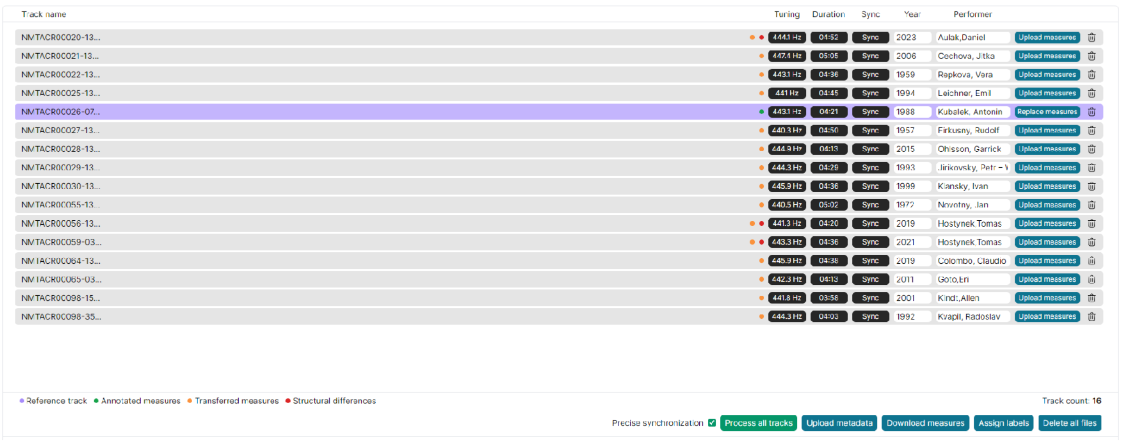
The recordings and metadata for the TL05000527 project and this case study were obtained from a variety of sources [16]:

- home funds of National Museum (NM), Prague,
- digital archive of Supraphon a.s.,
- acquisition of original audio carriers (gramophone records, CDs),
- private archives of collectors,
- recordings in open-access music databases,
- recordings available on streaming platforms.

Then, a unique identifier was attached to a given audio carrier or digital recording. It combines an alphabetical and numerical description and provides information about the digitization carrier, digitization institution (or under which project the digitization is taking place), what is the unique position of the given medium in a specific record series, and what is the order of the recording within the medium/carriers. More information is provided in the corresponding methodology article [16] or in our previous methodology on digitizing the phonographic recordings that follows a similar strategy [12]. A total of 308 recordings of Smetana's "Czech Dances II" were obtained, with interpretations by Czech pianists predominating in the total corpus of data (218 in total). In this chapter, we use 17 recordings of the *Neighbor's Dance* composition.

D.3 Data Upload and Cleaning

After logging in and creating a session, one can first upload all files, select the reference recording (NMTACR00026-07-UC, Firkušný Rudolf, 1957) for the synchronization process, and upload measure annotation (Figure D.1). Furthermore, one can assign a label to each recording to create various binary classes for further analysis. Here, I chose non-Czech – Czech (`noncz_cz`), male – female (`male_female`), less than 1960 – more than 1960, year of recording (`lt1960_mt1960`), and less than 2000 – more than 2000 (`lt2000_mt2000`) groups or classes (Figure D.2). The binary label for a given class is shown in Figure D.3.



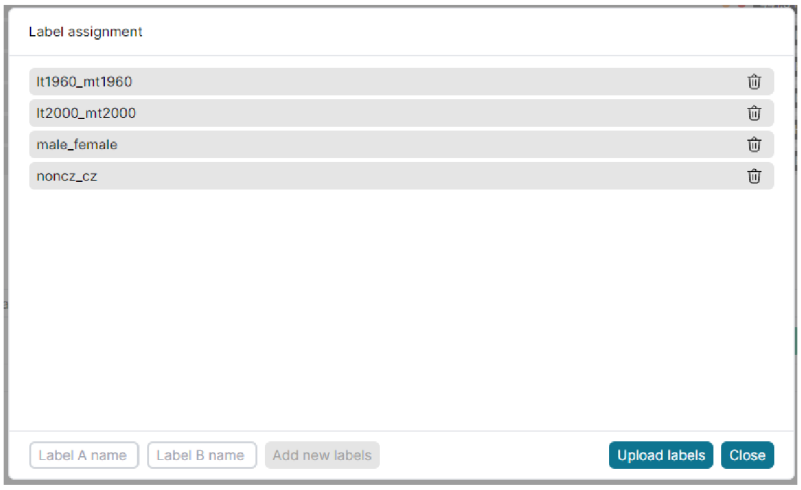
Track name		Tuning	Duration	Sync	Year	Performer		
NMTACR00020-13...	● ●	444.1 Hz	04:52	Sync	2023	Aulak,Dariel	Upload measures	🗑
NMTACR00021-13...	●	447.4 Hz	05:05	Sync	2006	Cechova, Jitka	Upload measures	🗑
NMTACR00022-13...	●	443.1 Hz	04:36	Sync	1959	Repkova, Vera	Upload measures	🗑
NMTACR00025-13...	●	441 Hz	04:45	Sync	1994	Leichner, Emil	Upload measures	🗑
NMTACR00026-07...	●	443.1 Hz	04:21	Sync	1988	Kubalek, Antonin	Replace measures	🗑
NMTACR00027-13...	●	440.3 Hz	04:50	Sync	1957	Firkusny, Rudolf	Upload measures	🗑
NMTACR00028-13...	●	444.9 Hz	04:13	Sync	2015	Ohlsson, Garrick	Upload measures	🗑
NMTACR00029-13...	●	444.3 Hz	04:29	Sync	1993	Jirikovsky, Petr – V	Upload measures	🗑
NMTACR00030-13...	●	445.9 Hz	04:36	Sync	1999	Klansky, Ivan	Upload measures	🗑
NMTACR00055-13...	●	440.5 Hz	05:02	Sync	1972	Novotny, Jan	Upload measures	🗑
NMTACR00056-13...	● ●	441.3 Hz	04:20	Sync	2019	Hostynek,Tomas	Upload measures	🗑
NMTACR00059-03...	● ●	443.3 Hz	04:36	Sync	2021	Hostynek,Tomas	Upload measures	🗑
NMTACR00064-13...	●	445.9 Hz	04:38	Sync	2019	Colombo, Claudio	Upload measures	🗑
NMTACR00065-03...	●	442.3 Hz	04:13	Sync	2011	Goto,Eri	Upload measures	🗑
NMTACR00098-15...	●	441.8 Hz	03:58	Sync	2001	Kindt,Allen	Upload measures	🗑
NMTACR00098-35...	●	444.3 Hz	04:03	Sync	1992	Kvapil, Radoslav	Upload measures	🗑

● Reference track ● Annotated measures ● Transferred measures ● Structural differences

Track count: **16**

Precise synchronization ✓

Fig. D.1: The first module of the software; upload of recordings, metadata, and labels.



Label assignment

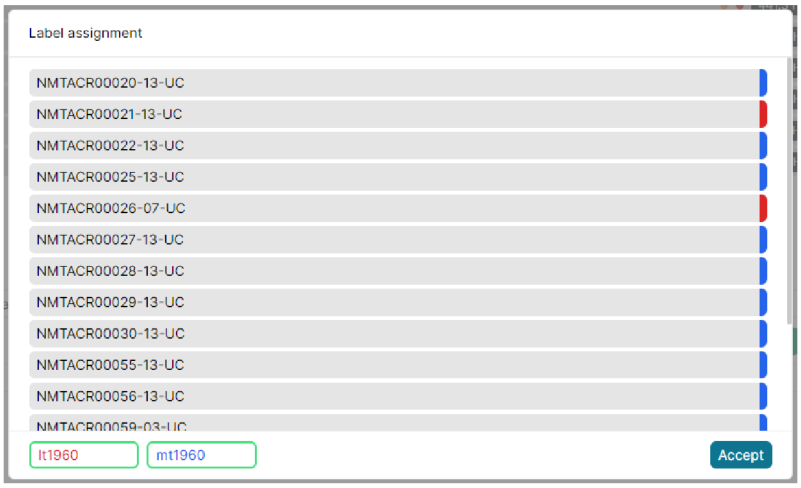
lt1960_mt1960

lt2000_mt2000

male_female

noncz_cz

Fig. D.2: List of provided binary labels.



Label assignment

NMTACR00020-13-UC

NMTACR00021-13-UC

NMTACR00022-13-UC

NMTACR00025-13-UC

NMTACR00026-07-UC

NMTACR00027-13-UC

NMTACR00028-13-UC

NMTACR00029-13-UC

NMTACR00030-13-UC

NMTACR00055-13-UC

NMTACR00056-13-UC

NMTACR00059-03-UC

Fig. D.3: Assigning of binary labels; less or more than 1960 (year of recording).

After the synchronization process, the software highlighted several recordings in which the synchronization might not have worked properly. After a manual check, I found out that one of the recordings (NMTACR00061-02-UC, Kindt Allen, 2001) does not follow the same structure, and discarded this recording from further visualizations and analysis. Furthermore, to compute the measure-wise tempo, I added the time signature information for the whole composition—in this case $\frac{3}{4}$ on its whole duration (Figure D.4).

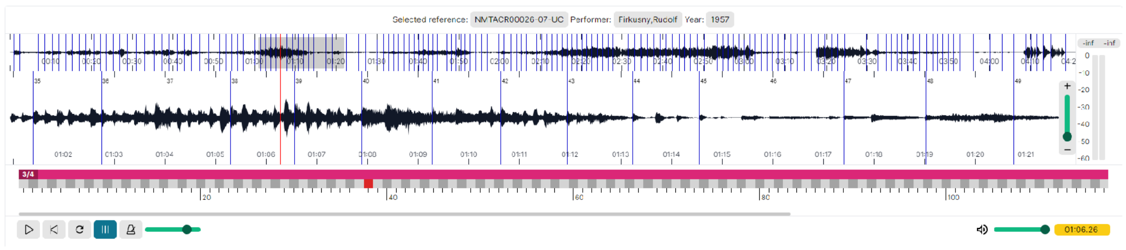

Selected reference: NMTACR00026-07-UC Performer: Firkusny,Rudolf Year: 1957
3/4
01:06.26

Fig. D.4: The reference recording in the second module of the software; adding time signatures.

D.4 Relevance

In the next module, *Interpretation player*, one can playback recordings or their sections, navigate using arbitrary measure ranges thanks to the time-aligned performances, and compare differences. I can visualize regions in which the synchronization might have been incorrect (Figure D.5), but in this case, the recordings followed the same harmonic and melodic structure and substantially varied in timing.

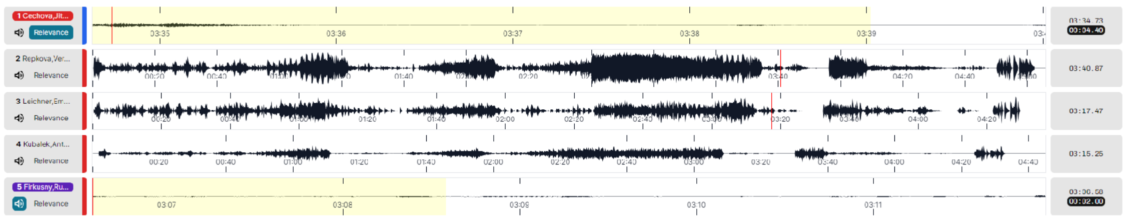


Fig. D.5: Highlighting of possibly different harmonic structures.

By clicking on the *Relevance* button of each recording, I can compare the relevance in duration of measures of individual performance against the rest of the dataset. Figure D.6 shows the most different measure (comparing the duration of measures) for the first recording of the dataset. The blue highlights measure 103. Note that all other presented performances play the same measure faster. The list of most relevant measures for this combination is shown in Figure D.7.

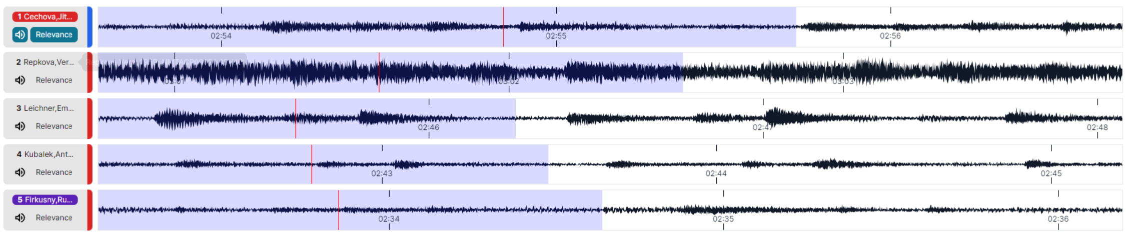
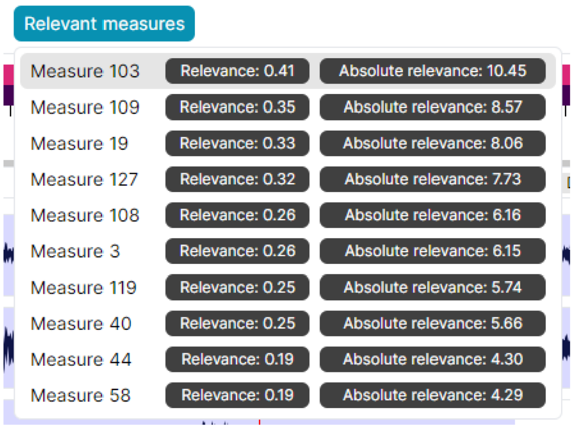


Fig. D.6: Duration of measure 103 for 5 different performances.



Relevant measures

Measure 103	Relevance: 0.41	Absolute relevance: 10.45
Measure 109	Relevance: 0.35	Absolute relevance: 8.57
Measure 19	Relevance: 0.33	Absolute relevance: 8.06
Measure 127	Relevance: 0.32	Absolute relevance: 7.73
Measure 108	Relevance: 0.26	Absolute relevance: 6.16
Measure 3	Relevance: 0.26	Absolute relevance: 6.15
Measure 119	Relevance: 0.25	Absolute relevance: 5.74
Measure 40	Relevance: 0.25	Absolute relevance: 5.66
Measure 44	Relevance: 0.19	Absolute relevance: 4.30
Measure 58	Relevance: 0.19	Absolute relevance: 4.29

Fig. D.7: The descending list of the most relevant measures when comparing a recording from Jitka Čechová with the rest of the dataset.

Figure D.8 shows the highest relevance measures when `noncz_cz` labels were selected. The maximum relevance is around 0.35, indicating a difference. When I select `lt1960_mt1960` labels, I can see higher differences based on the brighter color of measures 138 and 139 (Figure D.9).

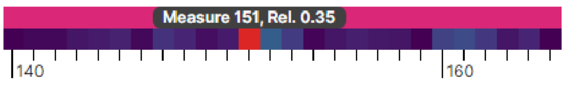


Fig. D.8: The relevance around measure 151 for `noncz_cz` labels.

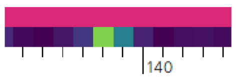


Fig. D.9: The relevance around measures 138 and 139 for `lt1960_mt1960` labels.

Figure D.10 presents waveform visualizations from *Interpretation player* of measures 138–140 (blue highlight); recordings marked as red belong to the `lt1960` class and blue to the `mt1960` class.

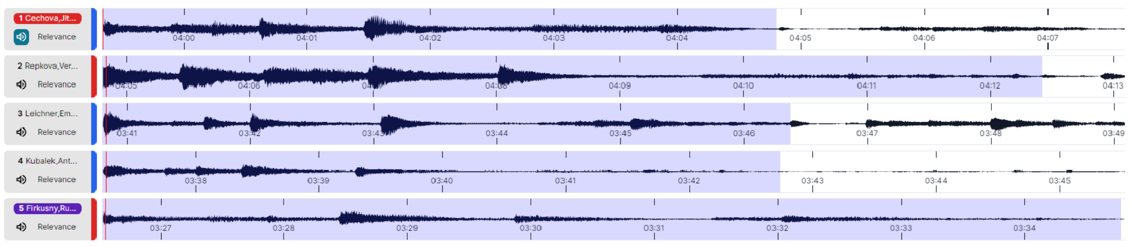


Fig. D.10: The duration of measures 138 and 140 for `lt1960_mt1960` labels.

D.5 Visualization

In the last module, one can visualize performance parameters and compare them. Figure D.11 shows waveforms, RMS, loudness, and beat tracking activation functions for the whole duration of two recordings from the dataset.

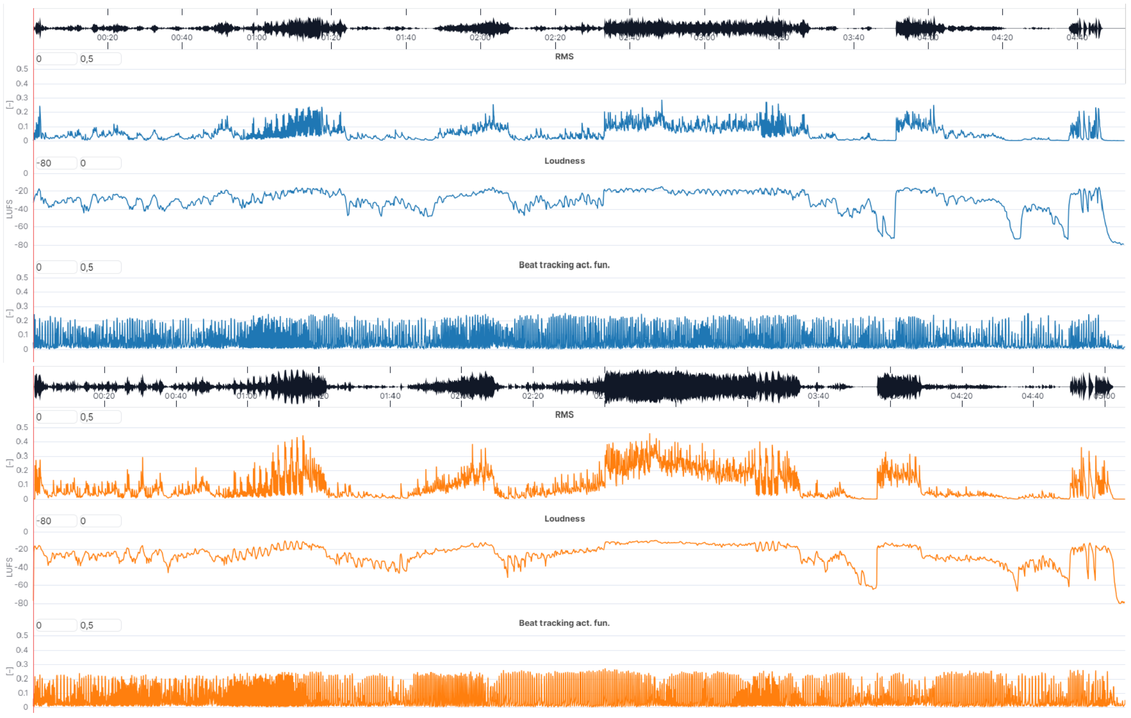


Fig. D.11: Waveforms, RMS, loudness, and beat tracking activation functions for two piano recordings.

I can see the variations in dynamics, differences between RMS and loudness curves, and overall quality (its sharpness or bluntness) of beat activation functions. I can change the limits of the time axis, zoom into measures 138–140, and compare parameters, such as dynamics values, for individual performances (Figure D.12).

One can visualize onset, beat, and downbeat activation functions from neural networks and compare them while playing back the audio. For instance, one may find relationships between acoustic parameters and the robustness of activation functions (Figure D.13). Next, I can compare and analyze the loudness and tempo of individual performances, but instead of using the time axis, resampling all parameters to the same measure axis. Figure D.1 shows a comparison of three performances and Figure D.15 all available interpretations.

In the current version of the software, one can use a pre-trained chord recognition model to segment recordings based on the harmonic structure (Figure D.16). However, this feature is experimental, as the detection accuracy varies.

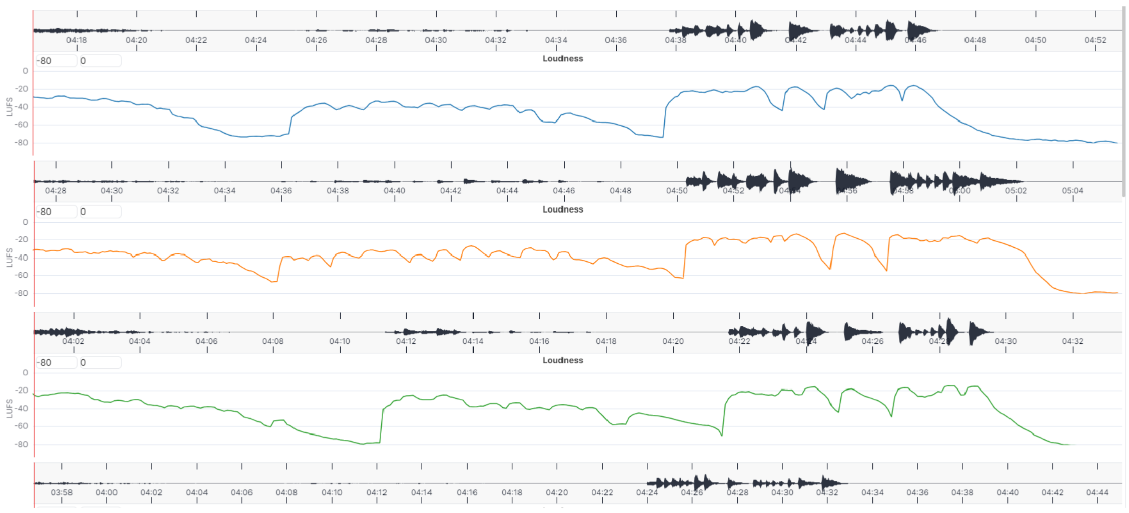


Fig. D.12: Loudness curves for three performances.

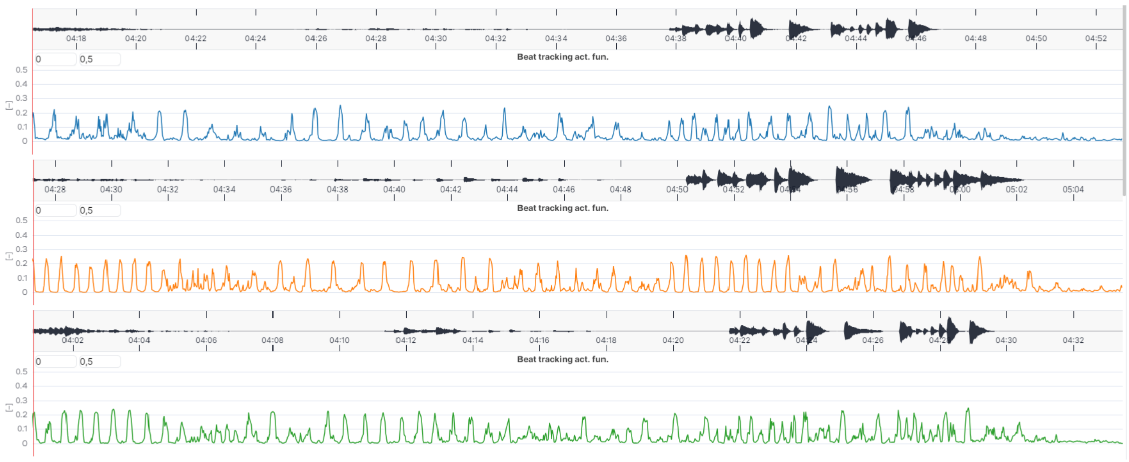


Fig. D.13: Beat activation functions for three performances.

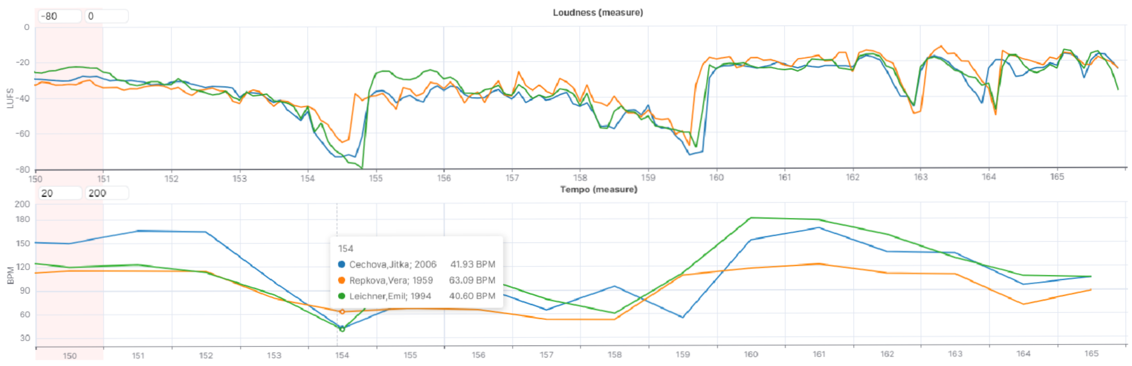


Fig. D.14: Loudness and tempo for three performances.

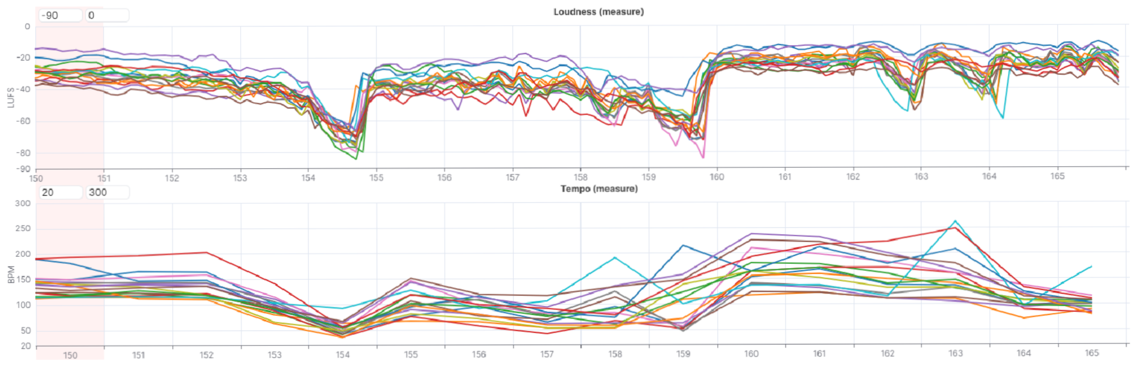


Fig. D.15: Loudness and tempo for all performances.

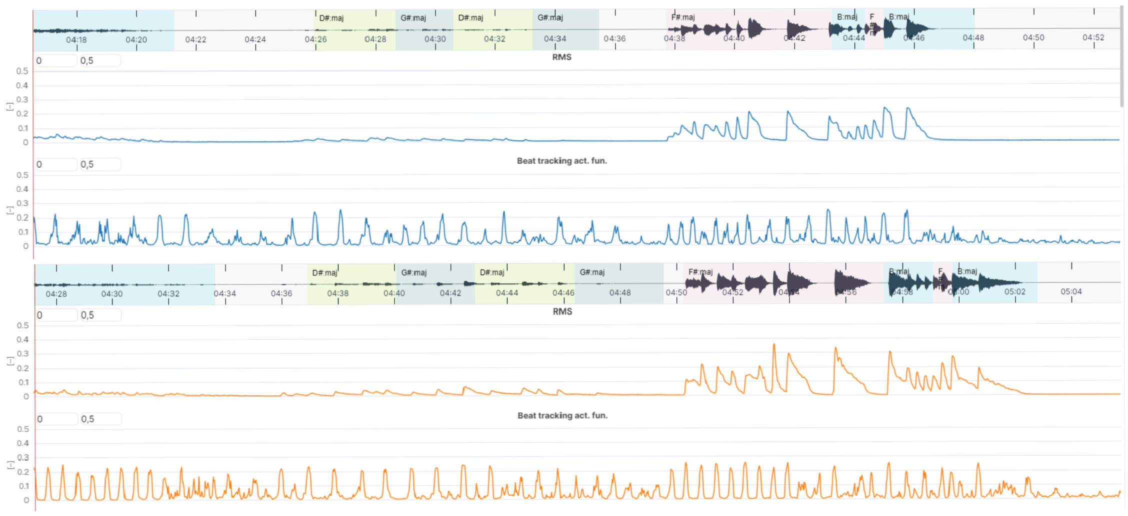


Fig. D.16: The RMS, beat tracking activation function, and chord detection.

Furthermore, I can compute the mean values of all parameters based on the labels/classes. Figure D.17 shows the mean of `noncz_cz` labels and Figure D.18 `male_female` labels. Figure D.19 displays the whole recordings based on the mean of `lt1960_mt1960` classes.

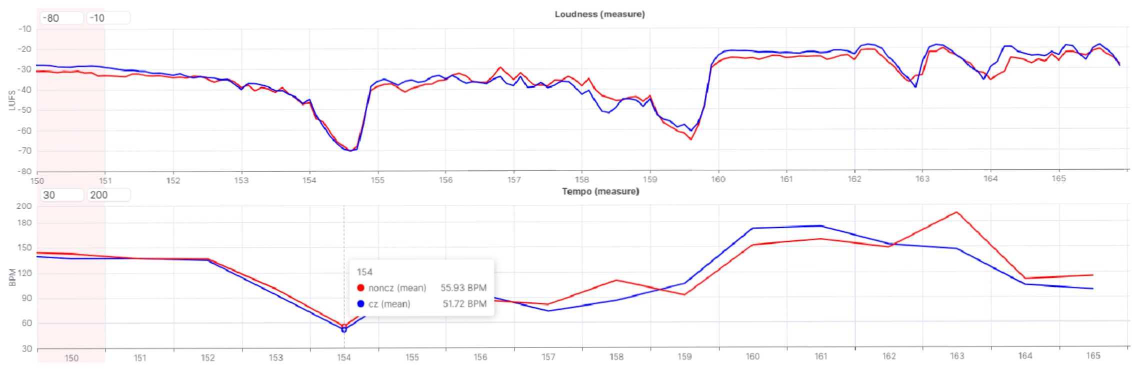


Fig. D.17: Mean loudness and tempo for `noncz_cz` labels.

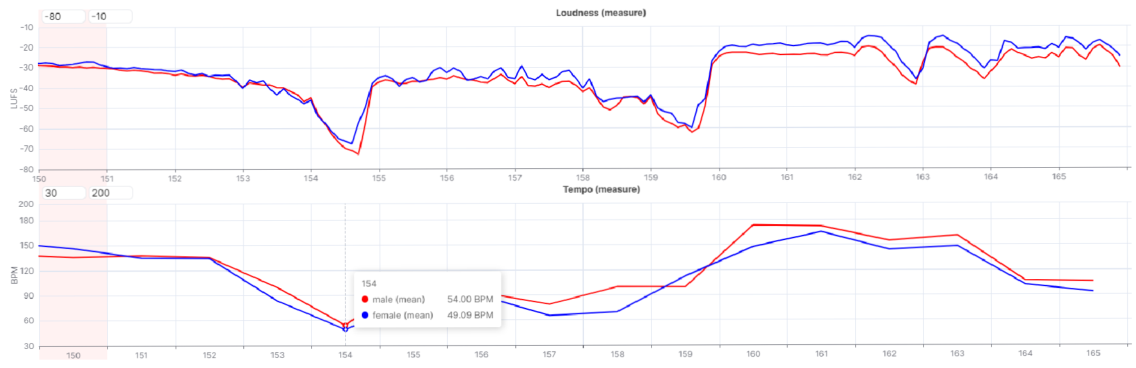


Fig. D.18: Mean loudness and tempo for `male_female` labels.

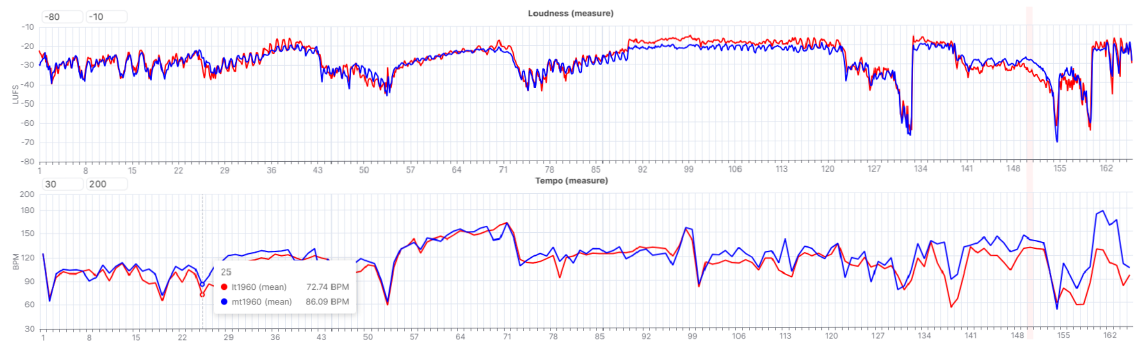


Fig. D.19: Mean loudness and tempo for the whole recordings; `lt1960_mt1960` labels.

Finally, Figures D.20 and D.21 show a scatter plot with `noncz_cz` labels for the whole composition and `male_female` labels for measures 150–166, respectively. This visualization is available only if a user provides years of recording as metadata in the first module. Each class is separated by color (blue or red), including their respective regression lines. Again, one can select any range of measures and labels or remove outliers.

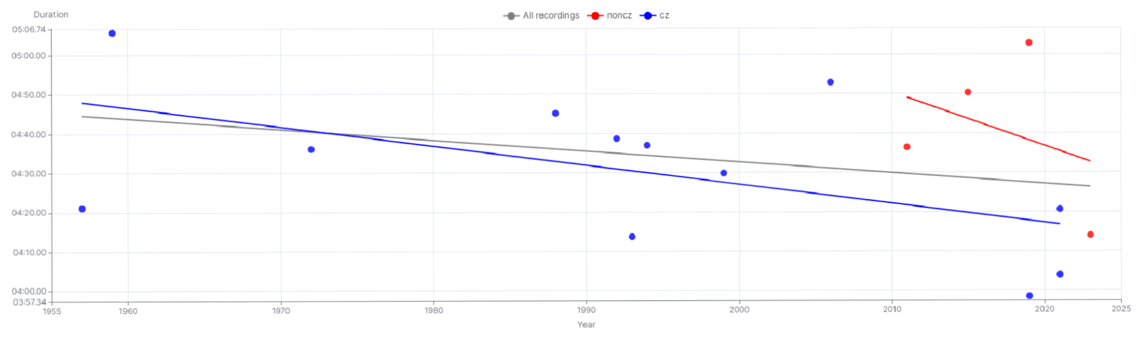


Fig. D.20: Scatter plot for `noncz_cz` labels; whole composition.

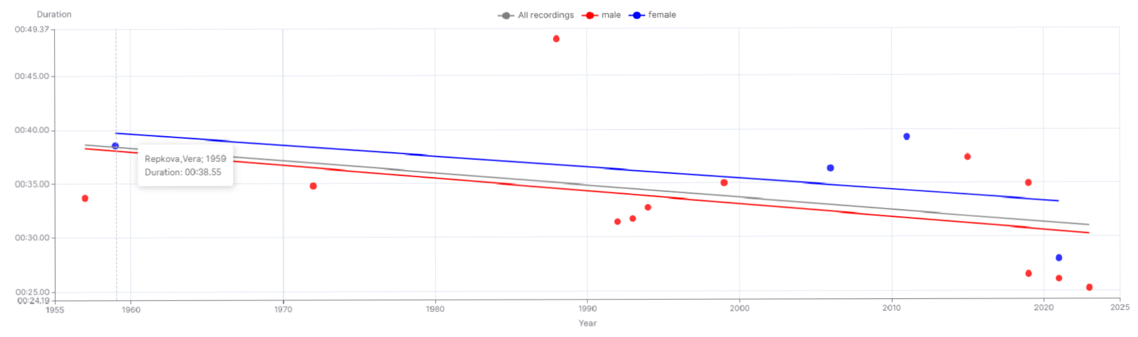


Fig. D.21: Scatter plot for `male_female` labels; ending of the composition.