Multi-scale computational rhythm analysis : a framework for sections, downbeats, beats, and microtiming

Magdalena Fuentes

université PARIS-SACLAY

NNT : 2019SACLS404

Thèse de doctorat

CentraleSupélec

TELECOM Paris

UNIVERSITÉ PARIS SUD
Comprendre le monde, construire l'avenir

Multi-Scale Computational Rhythm Analysis

A Framework for Sections, Downbeats, Beats, and Microtiming

Thèse de doctorat de l'Université Paris-Saclay
préparée à l'Université Paris-Sud

Ecole doctorale n°580 Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Traitement du Signal et des Images

Thèse présentée et soutenue à Palaiseau, le 12 novembre 2019, par

Magdalena Fuentes

Composition du Jury :

Gilles DUC *Professeur, L2S-CentraleSupélec, France*	Président du jury
Andre HOLZAPFEL *Maître de conférences, MID-KTH, Suède*	Rapporteur
Matthew DAVIES *Chargé de Recherche, CTM-INESC TEC, Portugal*	Rapporteur
Elaine CHEW *Chargée de Recherche, CNRS-STMS, IRCAM, France*	Examinatrice
Simon DIXON *Professeur, CDM-Queen Mary, Royaume Uni*	Examinateur
Marie JEHEL *Chargée de Recherche, CNRS-L2S, France*	Directrice de thèse
Slim ESSID *Professeur, Télécom ParisTech, France*	Co-directeur de thèse
Juan Pablo BELLO *Professeur, New York University, Etàts-Unis*	Co-directeur de thèse

To my family.

Acknowledgements

It is surprising how fast these last three years have passed, and how intense and amazing they have been. I would like to express my deep gratitude to my PhD supervisors Marie, Slim and Juan for their mentoring, support, constructive feedback and understanding during this time. Also for giving me the opportunity of meeting awesome people, and growing professionally and personally in a really motivating environment.

I am endlessly grateful to my collaborators, who enriched my research by sharing their expertise, creativity and curiosity. A especial thanks to Brian McFee, for teaching me so much in the short period we worked together, and mentoring me by example. Thanks to Martín Rocamora, who put me in the track of MIR for the first time some years ago, and has always helped me move forward. Without his support and advice who knows where I would have ended up, surely not where I did. Thanks to my dear fellow Lucas S. Maia for awesome discussions and great work together, and for being such a super office mate along with Laurent Benaroya and Mauricio V. M. Costa, who made me look forward to going to the office every day. Thanks to Luiz W. P. Biscainho, for his excellent advice and his kindness.

Thanks to all the members of the LTCI lab at Télécom Paris and of MARL at New York University, for creating a great environment and being so welcoming. Also, thanks to Justin Salamon and Srinkath Cherla for their invaluable advice as mentors aside from my PhD project, it was always fun and useful to talk to you.

I would not have gotten to this point without the support and help of my family and friends. Thanks to Rachel for a great time together, working, chatting or simply laughing. I wholeheartedly thank my flatmates and friends Guillaume and Rafa for many fun dinners and their constant support and advice, and to Umut, Ozan and Juanjo for our interesting discussions, musical moments and Friday drinks. Thanks to Pascal and Julien, for introducing me to climbing outdoors, and spending nights between mosquitos and cliffs. Thanks to Lara Raad, for being caring and fun and for helping me so much when I just arrived in Paris.

Thanks to the Cabot family, for all the fun, pizzas and beers every time I visit them. Thanks to Nico and Lara, for their support and generosity all these years. Thanks to my always encouraging friends Flo, Anita, Nati and Negro, for bearing with me during stressful periods and always cheering and backing me up. Thanks to my mum, my sister Lu and my granny Chula, who kept me going through the most difficult parts and always made me feel better. Thanks to Rob, my enthusiastic beloved partner of adventures and everyday life, who I am immensely grateful to have met, and with whom I am excited to share the next chapter of our lives.

Abstract

Computational rhythm analysis deals with extracting and processing meaningful rhythmical information from musical audio. It proves to be a highly complex task, since dealing with real audio recordings requires the ability to handle its acoustic and semantic complexity at multiple levels of representation. Existing methods for rhythmic analysis typically focus on one of those levels, failing to exploit music's rich structure and compromising the musical consistency of automatic estimations.

In this work, we propose novel approaches for leveraging multi-scale information for computational rhythm analysis. Our models account for interrelated dependencies that musical audio naturally conveys, allowing the interplay between different time scales and accounting for music coherence across them. In particular, we conduct a systematic analysis of downbeat tracking systems, leading to convolutional-recurrent architectures that exploit short and long term acoustic modeling; we introduce a skip-chain conditional random field model for downbeat tracking designed to take advantage of music structure information (i.e. music sections repetitions) in a unified framework; and we propose a language model for joint tracking of beats and micro-timing in Afro-Latin American music.

Our methods are systematically evaluated on a diverse group of datasets, ranging from Western music to more culturally specific genres, and compared to state-of-the-art systems and simpler variations. The overall results show that our models for downbeat tracking perform on par with the state of the art, while being more musically consistent. Moreover, our model for the joint estimation of beats and microtiming takes further steps towards more interpretable systems. The methods presented here offer novel and more holistic alternatives for computational rhythm analysis, towards a more comprehensive automatic analysis of music.

Resumé

La perception de la structure joue un rôle fondamental dans la manière dont les auditeurs perçoivent la musique. Sa structure riche et interdépendante distingue la musique des autres phénomènes auditifs, tels que la parole et l'environnement sonore. Les formes musicales telles que la mélodie ou l'harmonie sont interprétées dans le contexte de la mémoire à court et à long terme, et les structures musicales sont mieux perçues en présence de hiérarchies. Le rythme est essentiel à la perception de la structure, en tant que dimension fondamentale de la musique.

Le rythme musical est une organisation d'événements sonores qui appartiennent à différentes échelles temporelles et interagissent dans des hiérarchies, parfois organisées en motifs, certains événements étant synchrones, certains séquentiels. Cette organisation complexe est hautement structurée et axée sur les événements. L'analyse du rythme musical implique donc d'identifier et de caractériser de tels événements.

La modélisation computationnelle du rythme a pour objet l'extraction et le traitement d'informations rythmiques à partir d'un signal audio de musique. Cela s'avère être une tâche extrêmement complexe car, pour traiter un enregistrement audio réel, il faut pouvoir gérer sa complexité acoustique et sémantique à plusieurs niveaux de représentation. Les méthodes d'analyse rythmique existantes se concentrent généralement sur l'un de ces aspects à la fois et n'exploitent pas la richesse de la structure musicale, ce qui compromet la cohérence musicale des estimations automatiques.

Dans ce travail, nous proposons de nouvelles approches tirant parti des informations multi-échelles pour l'analyse automatique du rythme. Nos modèles prennent en compte des interdépendances intrinsèques aux signaux audio de musique, en permettant ainsi l'interaction entre différentes échelles de temps et en assurant la cohérence musicale entre elles. En particulier, nous effectuons une analyse systématique des systèmes de l'état de l'art pour la détection des premiers temps, ce qui nous conduit à nous tourner vers des architectures convolutionnelles et récurrentes qui exploitent la modélisation acoustique à court et long terme; nous introduisons un modèle de champ aléatoire conditionnel à saut de chaîne pour la détection des premiers temps. Ce système est conçu pour tirer parti des informations de structure musicale (c'est-à-dire des répétitions de sections musicales) dans un cadre unifié. Nous proposons également un modèle linguistique pour la détection conjointe des temps et du micro-timing dans la musique afro-latino-américaine.

Le travail présenté dans cette thèse se situe à l'intersection de l'apprentissage automatique, du traitement du signal audio et de la musicologie. Nous explorons la combinaison d'approches d'apprentissage en profondeur et de cadres probabilistes graphiques pour l'analyse automatique du rythme à partir de signaux audio musicaux. La thèse met l'accent sur le développement de modèles multi-échelles pour l'étude du rythme musical. Nous fournissons des méthodologies adaptables à d'autres genres musicaux ou scénarios.

Nos méthodes sont systématiquement évaluées sur diverses bases de données, allant de la musique occidentale à des genres plus spécifiques culturellement, et comparés à des systèmes de l'état de l'art, ainsi qu'à des variantes plus simples. Les résultats globaux montrent que nos modèles d'estimation des premiers temps sont aussi performants que ceux de l'état de l'art, tout en étant plus cohérents sur le plan musical. De plus, notre modèle d'estimation conjointe des temps et du microtiming représente une avancée vers des systèmes plus interprétables. Les méthodes présentées ici offrent des alternatives nouvelles et plus holistiques pour l'analyse numérique du rythme, ouvrant des perspectives vers une analyse automatique plus complète de la musique.

Cette thèse étant alignée sur des stratégies de recherche ouvertes et reproductibles, les idées, modèles, codes et données produits dans le cadre de cette thèse seront partagés avec la communauté sous licence ouverte, dans la mesure du possible.

Contents

List of Figures

List of Tables

List of symbols

Notation

$\mathbf{X}$	matrix or tensor
$\mathbf{x}$	vector or sequence of scalars
x	sequence of vectors
x	scalar

Acronyms

sr	Sampling rate
AC	Autocorrelation
BPM	Bar pointer model
bpm	Beat per minute
CH	Chroma
CNNs	Convolutional neural networks
CQT	Constant-Q transform
CRFs	Conditional random fields
CSD	Complex spectral difference
DBNs	Dynamic Bayesian networks
DNNs	Deep neural networks
EF	Energy flux
F0	Fundamental frequency
GMMs	Gaussian mixture models

GMs	Graphical models
HF	Harmonic feature
HMMs	Hidden Markov models
IT	Information technology
LCCRFs	Linear-chain conditional random fields
LSTMs	Long short-term memory networks
MAF	Mel auditory feature
MCQT	Melodic constant-Q transform
MFCCs	Mel frequency cepstral coefficents
MIR	Music Information Research/Retrieval
MIREX	Music Information Retrieval Evaluation EXchange
ML	Machine learning
MLPs	Multi-layer perceptrons
ODF	Onset detection function
PGM or GM	Probabilistic graphical models
PSF	Phase slope function
RNNs	Recurrent neural networks
SB	Spectral balance
SC	Spectral coefficients
SCCRFs	Skip-chain conditional random fields
SF	Spectral flux
SSM	Self-similarity matrix
STFT	Short-time Fourier transform
BN	Batch normalization
LBP	Loopy Belief Propagation

Chapter 1

Introduction

Music has always played an important role in our life, our cultural identity and social exchanges. Playing and listening to music has been a part of human activities across cultures through history, being a fundamental form of human expression. Understanding music and the way we relate with it sheds a light on a big part of our human nature.

Our way of interacting with music has changed dramatically in the last twenty years. As part of an information revolution that includes other modalities such as text, images and video, the availability and consumption of music has increased exponentially. Since it has become digital, music is easy to share and access, and online catalogues are bigger, more diverse and multicultural than they have ever been. This huge amount of available information has posed many interesting challenges to *information technology* (IT) researchers, one of the main ones being how to perform meaningful retrievals among millions of examples.

Considerable efforts have been made in the IT community towards facilitating the access of information and enriching the way we interact with it. That is also the case for the *Music Information Research* community (MIR).[1] MIR is an information technology community whose focus is dealing with music information in its various forms (e.g. metadata, semantic, acoustic). The MIR community develops tools for the retrieval, analysis, generation and synthesis of music [153, 131, 49]. It is a highly interdisciplinary field of research that brings together people with different backgrounds such as computer science or musicology, with the aim of developing tools and concepts to better understand music and how technology can help to effectively interact with it. The MIR field is now at the intersection between signal processing, machine learning, music theory, music perception and musicology [131], which provides rich and flexible foundations to address many interesting music technology research questions. The MIR community plays a key role in the development of automatic tools to make sense of the increasing amount of available musical data.

Music brings together perception and cognition processes, and carries evolutionary and cultural footprints. As a consequence, the automatic analysis and estimation of music con-

[1] Also known as *Music Information Retrieval*.

tent proves to be a highly complex and interesting task. There is great variability in audio signals, due to the many sound production modes and the wide range of possible combinations between the various acoustic events. The many different instrument *timbres* and the huge variety of audio effects and post-production, make music signals extremely rich and complex from a physical point of view. Besides, music audio signals are also complex from a semantic point of view, since they convey multi-faceted and strongly interrelated pieces of information. The multiple dimensions of music such as harmony, rhythm and melody interact with each other in an intricate manner, in such a way that it is usually necessary for a human to receive training in order to be able to understand it, and even to be aware of the cultural context in which the music was produced. Dealing with real audio recordings thus requires the ability to handle both uncertainty and a complex relational structure at multiple levels of representation.

Rhythm in particular, constitutes a foundation for music across many cultures, its understanding being fundamental for the understanding of music. The rhythmic dimension of a music piece consists of events organized in a hierarchical manner among distinct temporal granularities that interact with each other. This inherent structure, makes musical rhythm a challenging and interesting subject of study, as it clearly reflects the multi-scale nature of music. Rhythm itself can be studied from distinct viewpoints at different time-scales [6].

This dissertation is concerned with the problem of automatic musical rhythm analysis considering the interplay of different hierarchical levels of representation at different temporal scales. We present in the following the context and motivation of this dissertation, and formulate the research questions pursued in this work.

1.1 Research context

Given the intricate nature of music, a common approach in the MIR community is to tackle its multiple dimensions, e.g. harmony or rhythm, separately, and subdivide them into well defined tasks [153], as it is clear from the *Music Information Retrieval EXchange* (MIREX) campaigns. This approach leads to the development of effective systems for retrieving music objects such as chords or beats in an isolated fashion, but there is still much room to explore more holistic approaches that consider musics complex interrelationships. Besides the question of whether these holistic approaches can improve performance over their isolated counterparts, there is also the question of whether they result in more descriptive models for the automatic content of music. For instance, in many music genres—e.g. Latin-American Afro-rooted music traditions—the sole estimation of beats or downbeats, though important for their study, is insufficient to illustrate the complexity of their rhythm [64, 87]. Other information related to timing between articulation of events is as important as beats, and leads to a better understanding of these rhythms.

In recent years there have been increasing attempts to close the gap on this task division [134, 165, 78, 9, 15, 186, 176]. This perspective on how to formulate MIR problems poses questions about the musical coherence of the models, and since it requires the definition of more integral problems, it usually leads to richer analysis. In line with such approaches and despite recent efforts, the interplay of different temporal analysis scales in music, in particular in musical rhythm, is still understudied.

Besides, a consistent challenge in the MIR community, and so in the automatic analysis of rhythm and the development of holistic approaches, is the availability of properly annotated data. Within the current context of IT the availability of curated data has become ever more critical, since its accessibility and diversity are essential to building and testing machine learning based approaches. Furthermore, the creation of music datasets entails various challenges, ranging from musical audio copyrights to expensive annotations which require music expertise. Considerable attempts have been made recently in the MIR community to provide datasets to tackle a diversity of problems [10, 119, 183, 157]. Aligned with these efforts, the STAREL project[2] (*Statistical Relational Learning for Music Information Extraction and Expressiveness Studies from Audio Recordings*), aims as part of its goals to generate music corpora useful for rhythm analysis and develop models for the study of expressiveness in Afro-rooted Latin American music, in particular for Brazilian *Samba* and Uruguayan *Candombe*, genres which are great exponents of rich rhythmic structure. The goals of this project are aligned with the interests of this dissertation, and thus part of the research presented here was carried out in the frame of this collaboration.

1.2 Motivation and scope of this dissertation

The perception of structure plays a fundamental role on how listeners experience music. Its rich and interrelated structure distinguishes music from other auditory phenomena, such as speech and environmental sound [117]. Musical forms like melody or harmony are interpreted in the context of short and long-term memory [112], and musical structures are better perceived in the presence of hierarchies [42]. Rhythm is essential to the perception of structure, being a foundational dimension of music.

Musical rhythm is an organization of sound events which belong to different temporal scales and interact in hierarchies, sometimes organized in patterns, some events being synchronous, some being sequential. This complex organization is highly structured, and event-oriented, so the analysis of musical rhythm implies identifying and characterizing such events.

Among the different hierarchical layers in the rhythmic structure of music, the metrical scale plays a fundamental role in many music traditions across cultures, since it acts as a

[2] http://www.smt.ufrj.br/~starel/

pivot to the other temporal scales and hierarchical layers. Many musical objects in music are synchronized at the meter level, such as chord changes [132] or rhythmic patterns [95], and others are significantly longer than the meter but their beginnings and endings are also ruled by downbeat positions, as is the case of music segments (e.g. chorus, verse). This makes the metrical level a promising start for looking at these interactions and modelling them.

Even though all the mentioned dimensions of music and its hierarchical parts can be studied in isolation, given the significant interplay between each other, more holistic approaches which consider these variate aspects jointly should be further explored. Repetitions and patterns are ubiquitous in different temporal scales. Repeated structural parts of a musical piece rule the musical objects they comprise, being also essentially repeated. At the same time, segmenting a song into its structural parts provides very useful tools for an enriched interaction with music, such as easily navigating or studying the structural composition of a music piece. Such segmentation of music into boundaries, can benefit from looking to finer temporal levels such as the metrical structure, since most boundaries in a music piece are aligned with the beginning of the metrical cycle. Like this example, interrelating temporal scales such as beat and onsets could provide information about groove, since this information tells how the events are being articulated with respect to more regular, steady levels, and this organization can be particular to a certain style or performer.

Automatic tools developed for rhythm analysis are also useful in many other applications, such as music editing and processing, enhanced navigation of music collections, music creation tools such as mixing or human-computer performances, enriched music listening, or tools for musicological studies on expressiveness and style.

The work presented in this dissertation is at the intersection of machine learning, audio signal processing and musicology. We explore the combination of deep learning approaches and graphical probabilistic frameworks for the automatic analysis of rhythm from musical audio signals. The emphasis of the thesis is on developing multi-scale models for the study of musical rhythm. We provide methodologies which are adaptable to other music genres or scenarios. Our research goals are summarized as follows:

1. Identify relevant multi-level temporal interactions with potential impact on automatic rhythm analysis systems.

2. Perform a systematic analysis of state-of-the-art deep learning based downbeat tracking systems and its common variations.

3. Develop a novel state-of-the-art model for downbeat tracking exploiting multi-scale information, which will serve as a baseline for further studies into the interrelationship with other musical scales.

4. Further extend the downbeat tracking system to integrate knowledge from coarse temporal scales, and assess its impact on performance and musical coherence.

5. Propose and implement a model towards the study of expressiveness in music performances.

This thesis is also concerned with the problem of availability of annotated data and reproducibility in MIR. Our objectives in this regard are listed below:

6. Contributing to the conceptualization, annotation and release of datasets for the automatic analysis of rhythm in culturally-specific music.

7. Promoting initiatives for centralized and controlled usage of datasets to increase reproducibility.

This thesis is aligned with open and reproducible research strategies, so the ideas, models, code and data produced within the context of this thesis will be shared with the community under open licenses whenever possible.

1.3 Dissertation overview

This dissertation is structured in three parts: *Part I: Background*, *Part II: Datasets and tools* and *Part III: Proposed approaches*. Part I offers a review of the methods used in Part III, while Part II presents the datasets used in this work, as well as the datasets and tools developed during this dissertation. Within each part, chapters can be read independently. Figure 1.1 contextualizes the content and contributions of this work. The rest of the dissertation is organized as follows.

Part I: Background

Chapter 2 - Musical-theory and technical background. This chapter provides an overview of the background material necessary for *Part II*. It introduces relevant musical concepts and a basic introduction to rhythm analysis. It provides a non-exhaustive overview of the technical concepts and methods that will help the reader to better understand the contributions of this thesis.

Chapter 3 - Previous work. This chapter is devoted to presenting a brief review of the previous approaches on the relevant tasks addressed in this dissertation, and discussing their limitations.

Part II: Datasets and tools

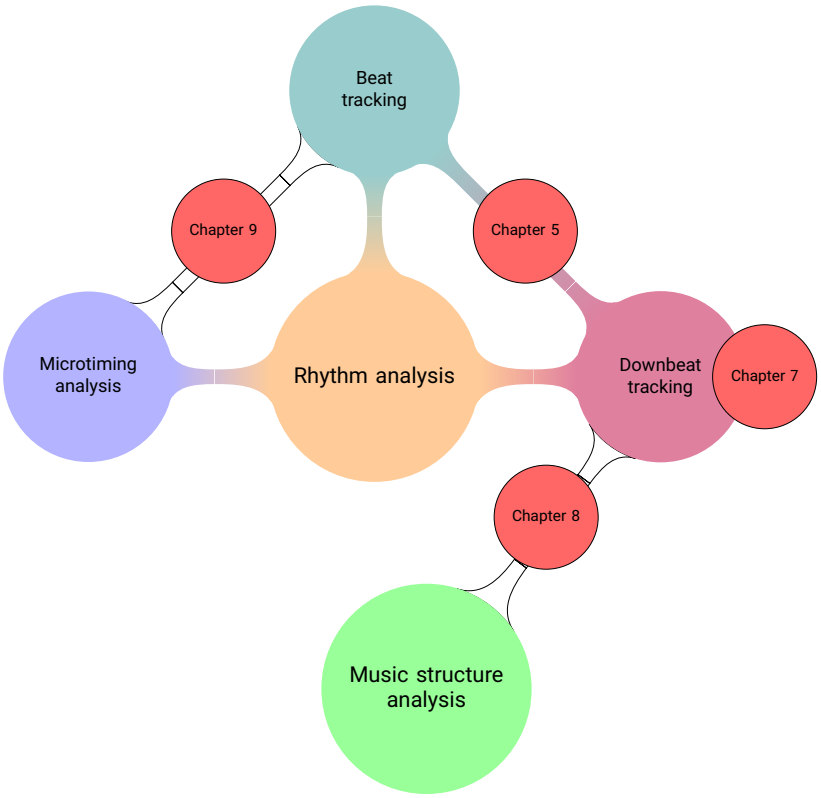


Figure 1.1 The methods developed in this dissertation are in the line of multi-scale rhythm analysis: our models establish a link between downbeats and sections, beats and microtiming. The datasets developed in collaboration with the STAREL project provide annotated beat and downbeat data. the explored links are indicated in white in the figure, and the red circles represent the chapters dedicated to each contribution.

Chapter 4 - Datasets and evaluation metrics. This chapter introduces the datasets and evaluation metrics used in following chapters.

Chapter 5 - Datasets of Brazilian music for computational rhythm analysis. This chapter includes details about the production of two datasets of Brazilian Afro-rooted Latin American music traditions carried out in this dissertation in collaboration with the STAREL project.

Chapter 6 - Common tools for MIR datasets: mirdata. This chapter presents a set of software tools to increase reproducibility in the use of datasets in the MIR community carried out in collaboration with the MIR research team at Spotify.

Part III: Proposed approaches

Chapter 7 - Analysis of deep learning systems for downbeat tracking. This chapter presents a systematic evaluation of common variations in state-of-the-art systems for downbeat tracking and introduces a novel convolutional-recurrent architecture for the task, which exploits information at multiple temporal scales and

performs as well as the state-of-the-art. It also presents a discussion on shortcomings of these systems in relation to the coherence of their estimations in coarser scales, and discusses future steps in this direction.

Chapter 8 - Structure informed downbeat tracking. This chapter presents a novel and flexible method for the inclusion of structure information for the task of downbeat tracking, presents a discussion on its advantages and shortcomings, and proposes future directions within this problem definition.

Chapter 9 - Beat and microtiming tracking. This chapter introduces an original formulation of microtiming profiles common to two distinct Latin-American music genres —candombe and samba—, it discusses its potential, and introduces a method for the joint estimation of beats and microtiming profiles, addressing its limitations and promising aspects.

Chapter 10 - Conclusions. The document ends with the main conclusions of this work, including directions for future research.

1.4 Contributions summary

The main idea pursued in this dissertation is to explore the interplay between music attributes at different temporal scales as part of models for the automatic analysis of rhythm. Besides, we contributed in the development of tools and datasets for the MIR community. The main contributions are listed below:

1. **Chapter 5:** two datasets of Afro-rooted Brazilian music for computational rhythm analysis, work performed in collaboration with the STAREL project and led by Lucas S. Maia. The work of this chapter led to the following publications:
   - L.S. Maia, M. Fuentes, L. W. P. Biscainho, M. Rocamora, S. Essid. (2019). SAMBASET: A Dataset of Historical Samba de Enredo Recordings for Computational Music Analysis. *In Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*.
   - L.S. Maia, P. D. de Tomaz Jr., M. Fuentes, M. Rocamora, L. W. P. Biscainho, M. V. M. Da Costa, S. Cohen (2018). A Novel Dataset of Brazilian Rhythmic Instruments and Some Experiments in Computational Rhythm Analysis. *In Proceedings of the AES Latin American Conference (AES LAC)*.
2. **Chapter 6:** a software package of common tools and loaders for MIR datasets, work carried out in equal contribution with Rachel M. Bittner and in collaboration with the MIR team at Spotify. The work of this chapter led to the following publication:

- R. M. Bittner, M. Fuentes, D. Rubinstein, A. Jansson, K. Choi, T. Kell (2019). mirdata: Software for Reproducible Usage of Datasets *In Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*.

3. **Chapter 7:** an analysis of commonly adopted design choices in state-of-the-art deep learning systems for downbeat tracking. A novel system for downbeat tracking that incorporates short and long temporal context using convolutional-recurrent neural networks. The work of this chapter led to the following publication:

    - M. Fuentes, B. McFee, H. C. Crayencour, S. Essid, J. P. Bello. (2018). Analysis of Common Design Choices in Deep Learning Systems for Downbeat Tracking. *In Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*.

4. **Chapter 8:** a model for downbeat tracking that handles structural segment repetition information, accounts for musical consistency and for rare music variations. The work of this chapter led to the following publication:

    - M. Fuentes, B. McFee, H. C. Crayencour, S. Essid, J. P. Bello. (2019). A Music Structure Informed Downbeat Tracking System Using Skip-Chain Conditional Random Fields and Deep Learning. *In Proceedings of the 44th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

5. **Chapter 9:** a study and conceptualization of microtiming profiles in time-keeper instruments in Afro-Latin American music. A model for the joint tracking of beats and microtiming profiles for this specific case. The work of this chapter led to the following publication:

    - M. Fuentes, L.S. Maia, L. W. P. Biscainho, M. Rocamora, H. C. Crayencour, S. Essid, J. P. Bello. (2019). Tracking Beats and Microtiming in Afro-Latin American Music Using Conditional Random Fields and Deep Learning. *In Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*.

Part I: Background

Chapter 2

Musical-theory and technical background

Summary

In this chapter we discuss relevant notions related to this dissertation. We introduce musical concepts that are central to the work presented here, in order to clarify the terminology used in following chapters. Subsequently, we present an overview of the main techniques and motivate their use in the context of multi-scale rhythm analysis. The next chapters rely on the content of this chapter and only focus on the extensions and contributions developed in this thesis.

Given the hierarchical and sequential structure of rhythm, we address the problem of rhythm analysis as a structure prediction problem over time sequences. For that reason, in this dissertation we apply and extend models that are able to handle and/or learn structured information over time. We chose to combine deep *neural networks* and *graphical models* due to their flexibility in learning complex structures and allowing for including expert knowledge in the models. In the following, we present a brief introduction to these models as well as relevant music concepts to the better understanding of the content of the rest of this dissertation.

2.1 Definition of relevant musical concepts

The definitions presented in this section are simplified and only intend to disambiguate the terminology we use in further chapters. The datasets used in this dissertation comprise music from different genres and cultures (see Chapter 4), hence we instantiate general def-

initions that hold in most cases, without the intention that these definitions represent in detail all music phenomena. Further details are provided within the chapters when needed.

All the concepts explained here are relevant to this dissertation either because we seek to develop automatic tools for their analysis, as for the case of downbeats, beats and microtiming; or because we make use of information related to them to inform other tasks, as it is the case of music structure and onsets. It is important to note that as it is the case with many other music concepts, the definition of meter, beat, microtiming, etc. carries a strong cultural bias.

Metrical structure

Metrical structure is a hierarchical organisation of pulsations in levels of different time span. Pulsations are a periodic grouping of events which in most of the cases match the articulation of a note or percussion event, but also can match silent positions. In the context of the music genres addressed in this dissertation, the metrical structure is commonly divided in the tatum, beat and downbeat levels, as shown in Figure 2.1, all comprising regular intervals of equal duration. We will refer to this structure within a bar as the meter.

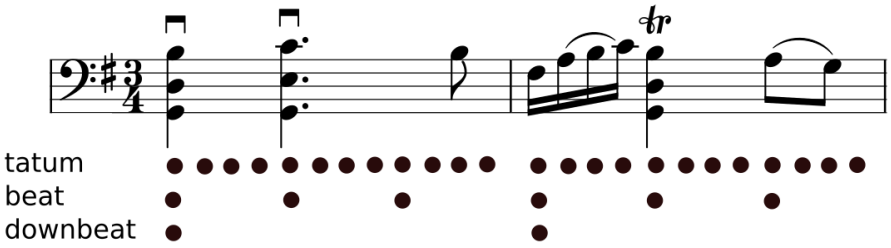


Figure 2.1 Illustration of what we refer to as metrical structure. Extract of *Sarabande, Suite Nº 1 for violoncello, BWV 1007*.

Beats

The most salient or predominant pulsation is the *beat*, which usually matches the foot tapping a person does when listening to a piece of music. Beats are fundamental to the perception of timing in music. It is key for musicians and dancers to synchronize with the music, and so for human-computer interaction in the context of music.

Tempo

Tempo refers to the frequency of the predominant pulsation of a piece. Its value is usually indicated in *beats per minute* (bpm), and it can be constant over the piece or it can change over time.

Downbeats

Among different music styles, beats of different accentuations are usually grouped in *bars*. This grouping organizes the musical events in a useful way for musicians, composers and dancers, since many music themes are related in one way or another to the duration or the beginning of a bar. The first beat of a bar is the *downbeat*. The number of grouped beats and tatums inside a bar is defined, at least in the context of this dissertation, by the meter or *time signature* of the piece.

Tatums

The *tatum* level corresponds to the fastest perceived pulsation in the metrical structure. As an example, in Figure 2.1 tatums correspond to the sixteenth note level.

Onsets

In the context of music, an *onset* is usually defined as the time instant of the *detectable* start of a melodic note, a harmonic change or percussion stroke [5]. The position of onsets are informative of where the events are actually articulated independently of the metrical levels. Onsets carry fundamental information about the expressive nature of a music performance, and are invaluable material for the study of rhythmic patterns.

Microtiming

In some cases, the articulated events in music—or onsets—present small-scale deviations with respect to the underlying metrical structure. This phenomenon is usually called *microtiming*, or *microtiming deviations*.[1] We refer to time intervals smaller than the temporal unit where events usually occur as small-scale deviations. As an example, a microtiming deviation in the excerpt of Figure 2.1 will be smaller than the duration of a sixteenth note, but microtiming deviations can appear at different metrical levels depending on the music genre. In some cases, it can be helpful to study microtiming in the context of rhythmic patterns [87], as the small-deviations of the articulated notes with respect to an expected—usually written—pattern.

[1] In this dissertation we will use both nomenclatures interchangeably.

Music sections

In a diversity of music genres such as pop, rock, jazz or classical music it is common to partition a musical piece in time into a small number of segments that correspond to structurally meaningful regions of the performance, which are repeated in different moments along the piece (such as intro, verses and chorus), as shown in Figure 2.2. The segmentation of a piece in this coarse set of segments is a highly subjective task [117], but in principle two segments will be labelled as the same if they convey *essentially* the same harmonic, semantic and rhythmic information, with some reasonable variations.

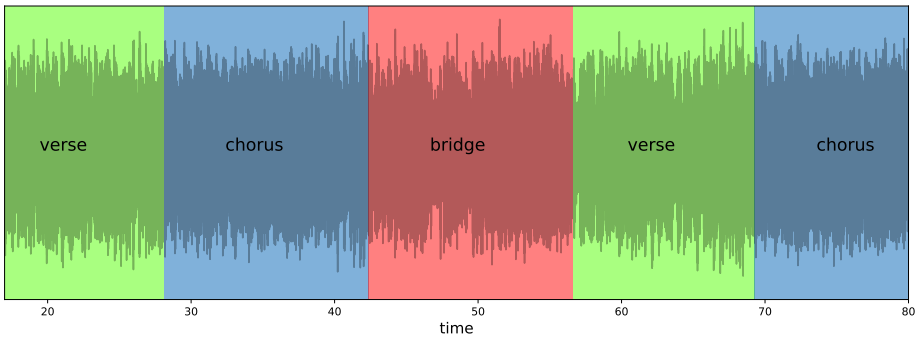


Figure 2.2 Illustration of music sections. Extract of *It Won't Be Long, The Beatles*.

2.2 Deep Learning Models

Motivated by the huge success of deep neural networks (DNNs) in Computer Vision, and due to recent advances that allow for faster training and scalability, DNNs have been widely used in many other domains, in particular in audio related tasks [61]. The inclusion of these models in MIR tasks has meant a considerable improvement in the performance of automatic systems, in particular downbeat tracking ones, as can be seen from the MIREX campaigns [84]. Moreover, the use of deep learning models presents other advantages over traditional machine learning methods used in MIR, i.e. they are flexible and adaptable across tasks. As an example, convolutional neural network based models from Computer Vision were adapted for onset detection [154], and then for segment boundary detection [175]. Furthermore, DNNs reduce —or allow to remove completely— the stage of hand-crafted feature design, by including the feature learning as part of the learning process.

However, the use of supervised deep learning models presents some disadvantages, one of the main ones being their dependence on annotated data. Annotated data is an important bottleneck in MIR especially due to copyright issues, and because annotating a musical piece requires expert knowledge and is thus expensive. Besides, solutions obtained in a data-driven fashion suffer from bias depending on the dataset used, a problem that also

occurs in other learning-based approaches. In this dissertation, we chose to work with deep learning models because of their flexibility, and to build upon existing research: the state-of-the-art models in meter tracking are deep learning-based and have shown the suitability of DNNs for the task.

In this section we provide a quick overview of relevant models used in our work, with the intention of motivating the use of these models in the context of this dissertation. A more detailed analysis, in particular about the use of DNNs in audio processing and MIR related tasks, can be found in the comprehensive studies proposed by McFee [113], Purwins et al. [143] and Choi et al. [29].

In general terms, a deep neural network consists of a composition of non-linear functions that acts as a function approximator $F_\omega : \mathbf{X} \rightarrow \mathbf{Y}$, for given input and output data $\mathbf{X}$ and $\mathbf{Y}$. The network is parametrized by its weights ω, whose values are optimized so the estimated output $\hat{\mathbf{Y}} = F_\omega(\mathbf{X})$ approximates the desired output $\mathbf{Y}$ given an input $\mathbf{X}$.

2.2.1 Multi-layer perceptrons

Multi-layer perceptrons (MLPs) are the simple and basic modules of DNNs. They are also known as *fully-connected layers* or *dense layers*, and consist of a sequence of layers, each defined by an affine transformation composed with a non-linearity:

$$\mathbf{y} = f(\mathbf{W}^T\mathbf{x} + \mathbf{b}), \tag{2.1}$$

where $\mathbf{x} \in \mathbb{R}^{d_{in}}$ is the input, $\mathbf{y} \in \mathbb{R}^{d_{out}}$ is the output, $\mathbf{b} \in \mathbb{R}^{d_{out}}$ is called the *bias vector* and $\mathbf{W} \in \mathbb{R}^{d_{in} \times d_{out}}$ is the weight matrix.[2] $f()$ is a non-linear activation function, which allows the model to learn non-linear representations. These layers are usually used to map the input to another space where hopefully the problem (e.g. classification or regression) can be solved more easily. However, by definition, this type of layer is not shift or scale invariant, meaning that when using this type of network for audio tasks, any small temporal or frequency shift needs dedicated parameters to be modelled, becoming very expensive and inconvenient when it comes to modelling music.

MLPs have been mainly used in early works before convolutional neural networks (CNNs) and recurrent neural networks (RNNs) became popular [29], and are now used in combination with those layers, as explained further in Section 2.2.4.

2.2.2 Convolutional networks

The main idea behind CNNs is to convolve their input with learnable kernels. They are suitable to problems that have two characteristics [113]:

[2] Note that for multi-dimensional inputs, e.g. $\mathbf{X} \in \mathbb{R}^{d_1 \times d_2}$, the input is flattened so $\mathbf{x} \in \mathbb{R}^d$ with $d = d_1 \times d_2$.

- statistically meaningful information tends to concentrate locally (e.g. within a window around an event),
- shift-invariance (e.g. in time or frequency) can be used to reduce model complexity by reusing kernels' weights with multiple inputs.

CNNs can be designed to perform either 1-d or 2-d convolutions, or a combination of both. In the context of audio, in general 1-d convolutions are used in the temporal domain, whereas 2-d convolutions are usually applied to exploit time-frequency related information. We will focus on models that perform 2-d convolutions, and we will omit the 2-d indication for readability in the following chapters. The output of a convolutional layer is usually called *feature map*. In the context of audio applications, it is common to use CNN architectures combining convolutional and pooling layers. Pooling layers are used to down-sample feature maps between convolutional layers, so that deeper layers integrate larger extents of data. A widely used pooling operator in the context of audio is *max-pooling*, which samples —usually— non-overlapping patches by keeping the biggest value in that region.

A convolutional layer is given by Equation 2.2:

$$\mathbf{Y}^j = f(\sum_{k=0}^{K-1} \mathbf{W}^{kj} * \mathbf{X}^k + \mathbf{b}^j), \tag{2.2}$$

where all $\mathbf{Y}^j$, $\mathbf{W}^{jk}$, and $\mathbf{X}^k$ are 2-d, $\mathbf{b}$ is the bias vector, j indicates the j-th output channel, and k indicates the k-th input channel. The input is a tensor $\mathbf{X} \in \mathbb{R}^{T \times F \times d}$, where T and F refer to the temporal and spatial—usually frequency—axes, and d denotes a non-convolutional dimension or *channel*. In audio applications d usually equals one. Note that while $\mathbf{Y}$ and $\mathbf{X}$ are 3-d arrays (with axes for height, width and channel), $\mathbf{W}$ is a 4-d array, so $\mathbf{W} \in \mathbb{R}^{h \times l \times d_{in} \times d_{out}}$, h and l being the dimensions of the convolution, and the 3rd and 4th dimensions account for the relation between input and output channels.

2.2.3 Recurrent networks

Unlike CNNs which are effective at modelling fixed-length local interactions, *recurrent neural networks* (RNNs) are good in modelling variable-length long-term interactions. RNNs exploit recurrent connections since they are formulated as [61]:

$$\mathbf{y}_t = f_y(\mathbf{W}_y \mathbf{h}_t + \mathbf{b}_y), \tag{2.3a}$$

$$\mathbf{h}_t = f_h(\mathbf{W}_h \mathbf{x}_t + \mathbf{U} \mathbf{h}_{t-1} + \mathbf{b}_h), \tag{2.3b}$$

where $\mathbf{h}_t$ is a hidden *state vector* that stores information at time t, f_y and f_h are the non-linearities of the output and hidden state respectively, and $\mathbf{W}_y, \mathbf{W}_h$ and $\mathbf{U}$ are matrices of

trainable weights. An RNN integrates information over time up to time step t to estimate the state vector $\mathbf{h}_t$, being suitable to model sequential data. Note that learning the weights $\mathbf{W}_y, \mathbf{W}_h$ and $\mathbf{U}$ in a RNN is challenging given the dependency of the gradient on the entire state sequence [113]. In practice, *back-propagation through time* is used [179], which consists in unrolling Equation 2.3b up to k time steps and applying standard back propagation. Given the accumulative effect of applying $\mathbf{U}$ when unrolling Equation 2.3b, the gradient values tend to either vanish or explode if k is too big, a problem known as the *vanishing and exploding gradient problem*. For that reason, in practice the value of k is limited to account for relatively short sequences.

The most commonly used variations of RNNs, that were designed to mitigate the vanishing/exploding problem of the gradient, include the addition of *gates* that control the flow of information through the network. The most popular ones in MIR applications are *long-short memory units* (LSTMs) [75] and *gated recurrent units* (GRUs) [27]. We will focus here on GRUs, which we use in our experiments in the following chapters, and mention LSTMs only to draw differences between the two neural networks.

Gated recurrent units

In a GRU layer, the *gate* variables $\mathbf{r}_t$ and $\mathbf{u}_t$ —named as *reset* and *update* vectors— control the updates to the state vector $\mathbf{h}_t$, which is a combination of the previous state $\mathbf{h}_{t-1}$ and a proposed next state $\hat{\mathbf{h}}_t$. The equations that rule these updates are given by:

$$\mathbf{r}_t = f_g(\mathbf{W}_r \mathbf{x}_t + \mathbf{U}_r \mathbf{h}_{t-1} + \mathbf{b}_r), \tag{2.4a}$$

$$\mathbf{u}_t = f_g(\mathbf{W}_u \mathbf{x}_t + \mathbf{U}_u \mathbf{h}_{t-1} + \mathbf{b}_u), \tag{2.4b}$$

$$\hat{\mathbf{h}}_t = f_h(\mathbf{W}_h \mathbf{x}_t + \mathbf{U}_h(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_h), \tag{2.4c}$$

$$\mathbf{h}_t = \mathbf{u}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{u}_t) \odot \hat{\mathbf{h}}_t; \tag{2.4d}$$

$\odot$ indicates the element-wise Hadamard product, f_g is the activation applied to the reset and update vectors, and f_h is the output activation. $\mathbf{W}_r, \mathbf{W}_u, \mathbf{W}_h \in \mathbb{R}^{d_{i-1} \times d_i}$ are the input weights, $\mathbf{U}_r, \mathbf{U}_u, \mathbf{U}_h \in \mathbb{R}^{d_i \times d_i}$ are the recurrent weights and $\mathbf{b}_r, \mathbf{b}_u, \mathbf{b}_h \in \mathbb{R}^{d_i}$ are the biases. The activation functions f_g and f_h are typically sigmoid and tanh, since saturating functions help to avoid exploding gradients in recurrent networks.

The GRU operates as follows: when $\mathbf{u}_t$ is close to 1, the previous observation $\mathbf{h}_{t-1}$ dominates in Equation 2.4d. When $\mathbf{u}_t$ gets close to 0, depending on the value of $\mathbf{r}_t$, either a new state is updated with the standard recurrent equation by $\hat{\mathbf{h}}_t = f(\mathbf{W}_h \mathbf{x}_t + \mathbf{v}_h \mathbf{h}_{t-1} + \mathbf{b}_h)$, if $\mathbf{r}_t = 1$, or the state is "reset" as if the $\mathbf{x}_t$ was the first observation in the sequence by $\hat{\mathbf{h}}_t = f(\mathbf{W}_h \mathbf{x}_t + \mathbf{b}_h)$.
The reset variables allow GRUs to successfully model long-term interactions, and perform

comparably to LSTMs, but GRUs are simpler since LSTMs have three gate vectors and one extra *memory* gate. Empirical studies show that both networks perform comparably while GRUs are faster to train [67, 86], so in this dissertation we will use GRUs for the study of long-term dependencies over time.

Bi-directional models

GRUs and RNNs in general are designed to integrate information in one direction, e.g. in an audio application they integrate information forward in time. However, it can be beneficial to integrate information in both directions, and so has been the case for neural networks in audio applications such as beat tracking [16] or environmental sound detection [135]. A bi-directional recurrent neural network (Bi-RNN) [155] in the context of audio consists of two RNNs running in opposite time directions with their hidden vectors $\mathbf{h}_t^f$ and $\mathbf{h}_t^b$ being concatenated, so the output h_t at time t has information about the entire sequence. Unless the application is online, Bi-RNNs are usually preferred due to better performance [113].

2.2.4 Hybrid architectures

As mentioned before, MLPs are now usually being used in combination with CNNs, which are able to overcome the lack of shift and scale invariance MLPs suffer. At the same time, MLPs offer a simple alternative for mapping representations from a big-dimensional space to a smaller one, suitable for classification problems.

Finally, hybrid architectures that integrate convolutional and recurrent networks have recently become popular and have proven to be effective in audio applications, especially in MIR [160, 114]. They integrate local feature learning with global feature integration, a playground between time scales that is in particular interesting in the scope of this dissertation. In the following chapters, and considering what was exposed in this section, we explore hybrid architectures for the task of rhythm analysis.

2.2.5 Learning and optimization

To optimize the parameters ω, a variant of gradient descent is usually exploited. A *loss function* $J(\omega)$ measures the difference between the predicted and desired outputs $\hat{\mathbf{Y}}$ and $\mathbf{Y}$, so the main idea behind the optimization process is to iteratively update the weights ω so the loss function decreases, that is:

$$\omega \leftarrow \omega - \eta \nabla_\omega J(\omega) \tag{2.5}$$

Where η is the *learning rate* which controls how much to update the values of ω at each iteration. Because the DNN consists of a composition of functions, the gradient of $J(\omega)$,

$\nabla\omega J(\omega)$, is obtained via the chain rule, a process known as *back propagation*. In the last four years, many software packages that implement automatic differentiation tools and various versions of gradient descent were released, [30, 1, 171], reducing considerably the time needed for the implementation of such models.

Since computing the gradient over a large training set is very expensive both in memory and computational complexity, a widely adopted variant of gradient descent is *Stochastic Gradient Descent* (SGD) [19], which approximates the gradient at each step on a mini-batch of training samples, B, considerably smaller than the training set. There are other variants of SGD such as *momentum* methods or *adaptive update* schemes that accelerate convergence dramatically, by re-using information of previous gradients (momentum) and reducing the dependence on η. From 2017, the most popular method for optimizing DNNs has been the adaptive method ADAM [89], which is the one we use in our work.

Another common practice in the optimization of DNNs is to use *early stopping* as regularization [162], which means to stop training if the training –or validation– loss is not improving after a certain amount of iterations. Finally, *batch normalization* (BN) [81] is widely used in practice as well, and consists of scaling the data by estimating its statistics during training, which usually leads to better performance and faster convergence.

2.2.6 Activation functions

The expressive power of DNNs is in great extent due to the use of non-linearities $f()$ in the model. The type of non-linearity used depends on whether it is an internal layer or the output layer. Many different options have been explored in the literature for intermediate-layer non-linearities —usually named *transfer functions*, the two main groups being saturating or non-saturating functions (e.g. *tanh* or *sigmoid* for saturated, because they saturate in 0 and 1, and *rectified linear units* (ReLUs) [124] for non-saturating ones). Usually non-saturating activations are preferred in practice for being simpler to train and increasing training speed [113].

2.3 Probabilistic graphical models

Probabilistic graphical models (PGM or GM) are a set of probabilistic models that express conditional dependencies between random variables as a graph. This graph can be *directed*, carrying a causal interpretation, or *undirected*, where there are no causal influences represented. Figure 2.3 illustrates these two distinct types of graphs. In directed graphs, the concept of *parent nodes* refers to nodes that precede topologically the others.

In the context of classification problems, the objective is to assign classes to observed entities. Two approaches commonly used in the context of sequential data classification are *generative* and *discriminative* models. Generative models are concerned with modelling

the joint probability $P(\mathbf{x}, \mathbf{y})$ given the input and output sequences **x** and **y**. They are generative in the sense that they describe how the output probabilistically generates the input, and the main advantage of this approach is that it is possible to generate samples from it (i.e. to generate synthetic data that can be useful in many applications). The main disadvantage of generative models is that to use them in classification tasks, where the ultimate goal is to obtain the sequence that maximizes $P(\mathbf{y}|\mathbf{x})$ (the most probable output given the input), one needs to model the likelihood $P(\mathbf{x}|\mathbf{y})$. Modelling $P(\mathbf{x}|\mathbf{y})$ can be very difficult when data involves very complex interrelations, but also simplifying them or ignoring such dependencies can impact the performance of the model [170]. In applications where generating data is not intended, it is more efficient to use *discriminative* models, which directly model the conditional probability between inputs and outputs $P(\mathbf{y}|\mathbf{x})$. The main advantage of these models is that relations that only involve **x** play no role in the modelling, usually leading to compact models with simpler structure than generative models.

Musical audio signals are very rich and complex from a physical point of view, due to the complexity of sound production modes and the wide range of possible combinations between the various acoustic events. Besides, they are also intricate from a semantic point of view. Music audio signals have multi-faceted, hierarchical and strongly interrelated information (e.g. melody, harmony, and rhythm). Probabilistic graphical models have been explored across different MIR tasks given their capacity to deal with structure in a flexible manner. In the following we introduce the main models exploited in the literature and their motivation, instantiating relevant works.

2.3.1 Hidden Markov Models

Hidden Markov models (HMMs) [144] are the most common graphical models used for music processing [134], in particular in rhythm analysis [51, 165] and widely used in the context of speech analysis and sequential problems in general. HMMs are generative models, so they compute the joint probability $P(\mathbf{x}, \mathbf{y})$ between a sequence of T *hidden states* **y** and a sequence of *observations* **x**. A HMM makes two important independence assumptions:

1. each observation x_t depends only on the current state y_t,
2. each state y_t depends only on its immediate predecessor y_{t-1}, which is called the Markovian assumption.[3]

The joint probability of the state sequence **y** and the observation sequence **x** in an HMM factorizes as a product of conditional probabilities, given by the parent node in the direct graph (see Figure 2.3) as:

[3] There is an extension of this case called *k-order HMMs*, in which the dependence of the current state y_t is extended up to k states in the past.

$$P(\mathbf{y}, \mathbf{x}) = P(y_1) \prod_{t=2}^{T} P(y_t|y_{t-1}) P(x_t|y_t), \tag{2.6}$$

where $P(y_t|y_{t-1})$ is the *transition probability*, $P(x_t|y_t)$ is the *observation probability*, and $P(y_1)$ is the distribution over initial states.

Rhythm analysis problems such as beat or downbeat tracking can be seen as sequence labelling problems, so that given a sequence of observations, the objective is to assign a pre-defined class to each one of these events. In the context of HMMs, this translates to finding the maximum likelihood sequence $\mathbf{x}^*$ that maximizes $P(\mathbf{x}|\mathbf{y})$, that is:

$$\mathbf{x}^* = \arg\max_x P(\mathbf{x}|\mathbf{y}) \tag{2.7}$$

The most common algorithm used to solve Equation 2.7 is the *Viterbi* algorithm [144].

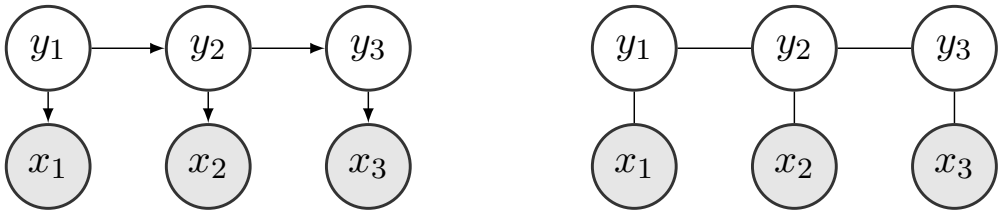


Figure 2.3 Example of a *directed* graph (left) and an *undirected* (right). In directed graphs *parent nodes* are those that precede topologically the others (e.g. y_1 is parent node of x_1).

2.3.2 Dynamic Bayesian Networks

Dynamic Bayesian Networks (DBNs) [123] are a generalization of HMMs. DBNs are Bayesian Networks that relate variables to each other over adjacent time steps. Like HMMs, they represent a set of random variables and their conditional dependencies with a directed acyclic graph, but they are more general than HMMs since they allow one to model multiple hidden states. Given the hidden and observed sequences of variables **y** and **x** of length T, the joint probability of the hidden and observed variables factorizes as:

$$P(\mathbf{y}, \mathbf{x}) = P(\mathbf{y}_1) \prod_{t=2}^{T} P(\mathbf{y}_t|\mathbf{y}_{t-1}) P(\mathbf{x}_t|\mathbf{y}_t), \tag{2.8}$$

with $P(\mathbf{y}_t|\mathbf{y}_{t-1})$, $P(\mathbf{x}_t|\mathbf{y}_t)$ and $P(\mathbf{y}_1)$ the transition probability, observation probability and initial states distribution as an HMM, but over a *set* of hidden variables. The initial state distribution is usually set to a uniform initialization in practice.

DBNs provide an effective framework to represent hierarchical relations in sequential data, as it is the case of musical rhythm. Probably the most successful example is the *bar pointer model* (BPM) [181], which has been proposed by Whiteley et al. for the task of meter

analysis and has been further extended in recent years [95, 79, 93, 166, 165]. Part of the work of this dissertation is based on the BPM, so a short overview of it and one of its relevant variants is presented in the following. Given that there are multiple versions of this model, we chose the variant presented in [165] for this explanation.

2.3.3 The bar pointer model

The BPM describes the dynamics of a hypothetical pointer that indicates the position within a bar, and progresses at the speed of the tempo of the piece, until the end of the bar where it resets its value to track the next bar. A key assumption in this model is that there is an underlying bar-length rhythmic pattern that depends on the style of the music piece, which is used to track the position of the pointer.

The effectiveness of this model relies on its flexibility, since it accounts for different metrical structures, tempos and rhythmic patterns, allowing its application in different music genres ranging from Indian music to Ballroom dances [166, 79].

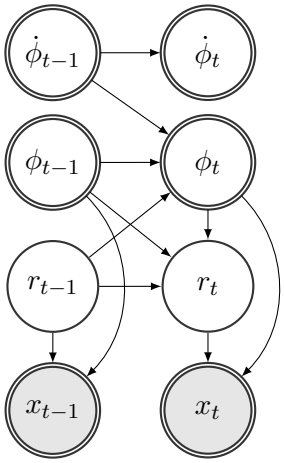


Figure 2.4 Simple version of a BPM, model-A in [165]. Double circles denote continuous variables, and simple circles discrete variables. The gray nodes are observed, and the white nodes represent the hidden variables.

Hidden states

The hidden state $\mathbf{y}_t$ represents the position of the hypothetical pointer at each time frame t, and is given by $\mathbf{y}_t = [\phi_t \ \dot{\phi}_t \ r_t]$, where each variable describes the position inside the bar, the instantaneous tempo, and the rhythmic pattern respectively.

- $r_t \in r_1 \dots r_R$ is a rhythmic pattern indicator that can be used to differentiate between R different rhythmic patterns, which can be known *a priori* or learned.

- The variable $\phi_t \in [0, M_{r_t})$ is the current position in a bar, and M_{r_t} is the length of a bar related to the considered rhythmic patterns. Different rhythmic patterns associated to different time signatures or meter cycles will have different number of discrete positions. Common practice is to fix the length of one time signature or meter cycle and scale the rest accordingly.
- $\dot{\phi}_t \in [\dot{\phi}_{min}, \dot{\phi}_{max}]$ is the instantaneous tempo (denoting the rate at which the bar pointer traverses a bar). $\dot{\phi}_t$ is given by the number of bar positions per frame. The tempo limits are assumed to depend on the rhythmic pattern state.

Transition Model

Due to the conditional independence relations shown in Figure 2.4, the transition model factorizes as:

$$P(\mathbf{y}_t|\mathbf{y}_{t-1}) = P(\phi_t|\phi_{t-1}, \dot{\phi}_{t-1}, r_{t-1}) \times P(\dot{\phi}_t|\dot{\phi}_{t-1}) \times P(r_t|r_{t-1}, \phi_{t-1}, \phi_t), \tag{2.9}$$

where the three factors are defined as:

- $P(\phi_t|\phi_{t-1}, \dot{\phi}_{t-1}, r_{t-1}) = 1_\phi$, where 1_ϕ is an indicator function that equals one if $\phi_t = (\phi_{t-1} + \dot{\phi}_{t-1}) \bmod M_{r_t}$ and 0 otherwise.
- The tempo transition from one frame to the next is assumed to follow a normal distribution and is given by: $P(\dot{\phi}_t|\dot{\phi}_{t-1}) \propto \mathcal{N}(\dot{\phi}_{t-1}, \sigma^2_{\dot{\phi}}) \times 1_{\dot{\phi}}$, where $\sigma_{\dot{\phi}}$ is the standard deviation of the tempo transition model and $1_{\dot{\phi}}$ is an indicator function that equals one if $\dot{\phi}_t \in [\dot{\phi}_{min}, \dot{\phi}_{max}]$, and 0 otherwise.
- $P(r_t|r_{t-1}, \phi_{t-1}, \phi_t) = \begin{cases} \mathcal{A}(r_t, r_{t-1}) & \text{if } \phi_t < \phi_{t-1}, \\ 1_r & \text{otherwise,} \end{cases}$
  where $\mathcal{A}(i, j)$ is the transition probability from r_i to r_j, and 1_r is an indicator function that equals one when $r_t = r_{t-1}$ and 0 otherwise.

Observation model

The observation model $P(\mathbf{x}_t|\mathbf{y}_t) = P(\mathbf{x}_t|\phi_t, r_t)$ proposed in [165] is given by learned features using *Gaussian Mixture Models* (GMMs) with two components. Among the variations of the BPM, other observation models have been proposed using RNNs [94, 15].

Inference

The inference of a model with continuous variables such as the one of Figure 2.4 is usually done approximately, where *sequential Monte Carlo (SMC)* algorithms have been explored in

the context of rhythm analysis [165, 98]. It is also common to discretize the variables $\dot{\phi}_t$ and ϕ_t, and then perform the inference using Viterbi.

2.3.4 Conditional Random Fields

Conditional Random Fields (CRFs) are a particular case of undirected PGMs. Unlike generative models such as *HMMs*, which model the joint probability of the input and output $P(\mathbf{x}, \mathbf{y})$, CRFs model the conditional probability of the output given the input $P(\mathbf{y}|\mathbf{x})$. CRFs can be defined in any undirected graph, making them suitable to diverse problems where structured prediction is needed across various fields, including text processing [158, 159], computer vision [72, 99] and combined applications of NLP and computer vision [189].

The problems addressed in this dissertation are sequential, i.e. the variables involved have a strong dependency over time. Equation 2.10 presents a generic CRF model for sequential problems, where Ψ_k are called *potentials*, which act in a similar way to transition and observation matrices in HMMs and DBNs, expressing relations between $\mathbf{x}$ and $\mathbf{y}$. k is a feature index, that exploits a particular relation between input and output. The term $Z(\mathbf{x})$, called the *partition function*, acts as a normalization term to ensure that the expression in Equation 2.10 is a properly defined probability. In this section, we focus on linear-chain and skip-chain CRFs, which are two variants of sequential CRF models suitable for rhythm analysis, because of their flexibility and their discriminative nature.

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{T} \prod_{k=1}^{K} \Psi_k(\mathbf{x}, \mathbf{y}, t) \tag{2.10}$$

$$Z(\mathbf{x}) = \sum_{y} \prod_{t=1}^{T} \prod_{k=1}^{K} \Psi_k(\mathbf{x}, \mathbf{y}, t) \tag{2.11}$$

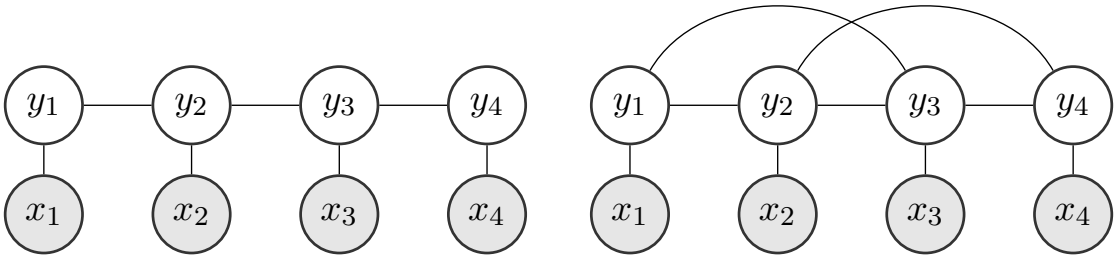


Figure 2.5 Example of a linear-chain CRF (left) and a skip-chain CRF (right).

Linear-chain Conditional Random Fields

Linear-chain CRFs (LCCRFs) restrict the model of Equation 2.10 to a Markov chain, that is, the output variable at time t depends only on the previous output variable at time $t-1$. Another

common constraint usually imposed in LCCRFs is to restrict the dependence on the current input x_t instead of the whole input sequence **x**, resulting in the following model:

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})}\Phi(x_1, y_1)\prod_{t=1}^{T}\Psi(y_{t-1}, y_t)\Phi(x_t, y_t). \tag{2.12}$$

These simplifications, which make the LCCRF model very similar to the ones presented in Sections 2.3.2 and 2.3.1, are often adopted in practice for complexity reasons [170], though there exist some exceptions [57]. The potentials Ψ_k of Equation 2.10 become simply Ψ and Φ in Equation 2.12, which are the *transition* and *observation* potentials respectively. The transition potential models interactions between consecutive output labels, whereas the observation potential establishes the relation between the input x_t and the output y_t. Note that potentials in CRF models do not need to be proper probabilities, given the normalization term $Z(\mathbf{x})$. The inference in LCCRFs is done to find the most probable sequence $\mathbf{y}^*$ so:

$$\mathbf{y}^* = \arg\max_y P(\mathbf{y}|\mathbf{x}), \tag{2.13}$$

and it is usually computed using the Viterbi algorithm, as in the case of HMMs [169].

Skip-chain Conditional Random Fields

Skip-chain CRFs (SCCRFs) are similar to the LCCRFs, but they also model relations between output variables y_u and y_v that are not consecutive in time. This is done by incorporating connections between distant nodes, called *skip connections*. Figure 2.5 illustrates an example of an LCCRF in contrast to an SCCRF for clarity. The conditional probability in the case of an SCCRF is given by:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})}\Phi(x_1, y_1)\prod_{t=1}^{T}\psi_t(y_t, y_{t-1})\phi(y_t, x_t)\prod_{(u,v)\in\mathcal{I}}\psi_{uv}(y_u, y_v). \tag{2.14}$$

Because the loops in the SCCRF can be long and overlapping, exact inference is intractable [169]. For this reason, in this work we perform loopy-belief propagation (LBP) for inference [32] when using SCCRFs. LBP is an algorithm based on message updates between nodes, as illustrated in Figure 2.6.

Although the algorithm is not exact and it is not guaranteed to converge if the model is not a tree, it has been shown to be empirically successful in a wide variety of domains such as text processing, vision, and error-correcting codes [184]. In the LBP algorithm, each node i sends a message to its neighbours, where neighbours are directly connected nodes in the graph no matter how distant those nodes are in time. The message from node i to node j is given by

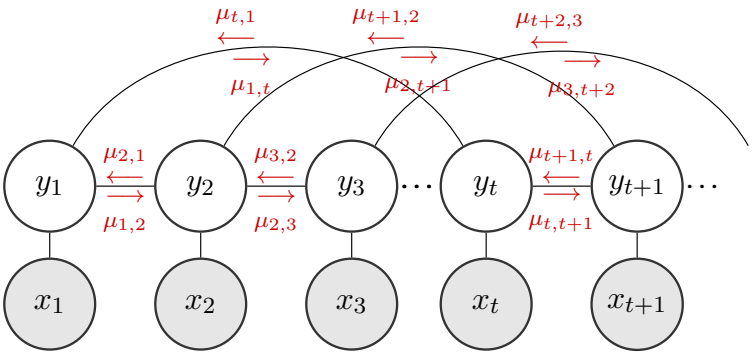


Figure 2.6 The inference is performed using a message passing algorithm: loopy belief propagation. Each node sends information to its neighbours.

$$\mu_{ij}(y_j) = \max_{y_i} \phi_i(y_i, x_i)\Psi_{ij}(y_i, y_j) \prod_{k \in N(i)\backslash j} \mu_{ki}(y_i), \tag{2.15}$$

where $N(i)\backslash j$ indicates the neighbours of node i except node j, and $\Psi_{ij} = \psi_t$ if $|i-j| = 1$ and $\Psi_{ij} = \psi_{uv}$ otherwise. This message exchange continues until the messages converge. Once convergence is achieved the belief of each node is computed as

$$b_i(y_i) = \phi_i(y_i, x_i) \prod_{j \in N(i)} \mu_{ij}(y_i), \tag{2.16}$$

and final inference is performed by $\mathbf{y}^* = \arg\max_y b(\mathbf{y})$.

Chapter 3

Previous work

Summary

In this chapter we provide a broad overview of relevant previous work to contextualize the contributions of this dissertation. We introduce the MIR tasks that are discussed in the different chapters of this document, providing a survey of the state of the art and its limitations.

Musical rhythm is organized into hierarchical levels which interact with each other. Each temporal scale in this hierarchical and interrelated structure is informative of a different music phenomena. The division of a music piece in coarse segments that repeat themselves, relates to the music genre of the piece or the composers' style, while finer scales will inform about the articulation of events and the expressiveness of the performance. The bar, which is at a middle-scale in this hierarchy, acts as a pivot with respect to other temporal scales: music segments usually start and end on a downbeat, and beats are grouped and accentuated in function of their position inside the bar. This places the downbeat in a privileged position in the rhythmic hierarchical structure, so we will study its interrelation in a multi-scale fashion in Chapters 7 and 8.

Meaningful musical events can also be grouped in scales finer than the bar. Rhythmic patterns are sometimes limited to the beat duration [125], and so the beat level can also encode information about repetitions and structure at a finer scale. This relation between the beat level and the articulation of events within it is of particular interest in Afro-rooted Latin American music, since it carries information about the genres' idiosyncrasy [64]. Moreover, this interrelationship becomes particularly interesting when looking at an even finer scale where small temporal deviations —microtiming— bring information about expressiveness. How beat-rate or how rhythmic patterns and deviations within them evolve over time are meaningful interrelationships that will be studied in Chapter 9.

The rhythm analysis tasks which have received more attention in recent years have been meter tracking and it sub-tasks, beat and downbeat tracking. Table 3.1 presents a broad

summary of the recent approaches in the literature. Automatic downbeat tracking aims to determine the first beat of each bar, being a key component for the study of the hierarchical metrical structure. It is an important task in Music Information Retrieval (MIR) that represents a useful input for several applications, such as automatic music transcription [109], computational musicology [70], structural segmentation [105] and rhythm similarity [137]. Microtiming analysis has also received considerable attention, though recently to a lesser extent than the meter tracking tasks. In the following we provide an overview of relevant works on these tasks to contextualize the work presented in this dissertation. Specific references to related work are made on each chapter separately for clarity, and this survey aims to give a broader contextualization.

3.1 Beat and downbeat tracking

As illustrated in Figure 3.1, recent beat and downbeat tracking approaches[1] usually consist of three main stages:

1. A first stage of *low-level feature* computation or *feature extraction*, where feature vectors that represent the content of musical audio are extracted from the raw audio signal.

2. A second step that usually consists of a stage of feature learning, whose outcome is an activation function that indicates the most likely candidates for beats and/or downbeats among the input audio observations.

3. Finally, a post-processing stage is often used, usually consisting of a probabilistic graphical model which encodes some relevant musical rules to select the final beat/-downbeat candidates.

Figure 3.1 General pipeline commonly used for beat and/or downbeat tracking systems.

Different alternatives were proposed for the distinct stages among beat and downbeat tracking systems. Here we give an overview of the main ideas presented in the literature.

[1] We refer to approaches proposed from 2010 onwards.

3.1.1 Downbeat tracking

Feature extraction

In the stage of feature extraction, a usually adopted approach is to exploit music knowledge for feature design using signal processing techniques. Three main categories of musically inspired features were largely explored: chroma (CH) [76, 132, 140, 88, 93] —used to reflect the harmonic content of the signal—, onset detection function (ODF) [76, 187, 50] or spectral flux (SF) [95, 88, 79] —as event-oriented indicators— and timbre inspired features [76, 168, 53] such as spectral coefficients or MFCCs. The feature extraction is usually based on a single feature [132, 140, 187, 79, 92, 94, 165, 13], with some exceptions exploiting more than one music property at the same time [50–52, 93], which results in systems robust to different music genres [52]. Recently, approaches based on deep learning exploring combinations of logarithmic spectrograms with different resolutions showed to perform competently [15].

Downbeat likelihood estimation

The objective of this stage is to map the input representation into a downbeat likelihood that indicates which are the most likely downbeat candidates given the observed features. There are two main groups of approaches in this respect: the first one uses heuristics to perform the mapping, while the second group exploits machine learning approaches. The latter group is the most popular one in the literature in the last years and also the state of the art.

The estimation of a downbeat likelihood with heuristics is performed differently depending on the features used. For instance, while using harmonic features it is usual to measure the distance between a given template and the computed features [140, 129]. Within the machine-learning group of approaches, there are broadly two subgroups: a first one that exploits 'traditional' machine learning techniques and a second one with focus on deep learning models.

Machine learning systems often focus on recognizing rhythm patterns in data, for instance by using *Gaussian Mixture Models* (GMM) and k-means [165–167, 98, 95, 79, 129]. This usually requires making some assumptions of style or genre, and distinctive rhythmic patterns are required for this kind of model to be effective. Deep learning approaches propose an alternative to such limitations given their capacity to learn rich and complex function mappings, and systems exploiting DNNs have became the state of the art in recent years [84]. Many different architectures have been explored, ranging from MLPs [50], CNNs [54, 52, 51, 78], RNNs [98, 13, 186], Bi-LSTMs [15], Bi-GRUs [93] and recently CRNNs [176].

Many approaches exploited RNNs or their variants (LSTMs, GRUs) given their suitability to process sequential data (see Section 2.2.3). In theory, recurrent architectures are flexible in terms of the temporal context they can model, which makes them appealing for music

applications. In practice there are some limitations on the amount of context they can effectively learn [66], and although it is clear that they can learn close metrical levels such as beats and downbeats [15], is not clear if they can successfully learn interrelationships between farther temporal scales in music.

Systems based on CNNs make the most of the capacity of such networks to learn invariant properties of the data while needing fewer parameters than other DNNs such as MLPs and being easier to train. Also, convolutions are suitable for retrieving changes in the input representations, which, as pointed out before, are indicators of downbeat positions (e.g. changes in harmonic content or spectral energy). Besides, CNNs have shown to be good high-level feature extractors in music [52], and are able to express complex relations. The main disadvantage of CNNs is their lack of long-term context and the requisite of having fixed-length inputs, which restrict the musical context and interplay with temporal scales that could improve their performance. This can be improved by combining CNNs with RNNs [176],[2] which we explore in Chapter 7.

Downbeat sequence extraction

The aim of this stage is to obtain the final downbeat sequence by selecting the most likely candidates in the downbeat likelihood given some model or criteria. As it can be seen from Table 3.1, probabilistic graphical models are the preferred option. This might be due to two main reasons: PGMs offer a flexible framework to incorporate music knowledge and then exploit interrelated structure [132, 140], and the BPM [181] (see Section 2.3.3) stands as a very effective and adaptable model for meter tracking, being popular for downbeat tracking.

The different post-processing variations with PGMs include multi-task approaches, where multiple solutions were proposed to perform joint beat and downbeat tracking using HMMs [140, 88, 129] or DBNs [15, 95, 79], and the joint estimation of chords and downbeats [132]. Besides, the BPM and similar models have proved to be adaptable to cultural-aware systems in diverse music cultures [168, 129, 79], being for instance adaptable to track longer meter cycles and different meters than the widely explored 3/4 and 4/4 [166, 167]. Finally, further efforts have been made in improving the BPM in practice, by reducing computational cost via an efficient state-space definition [94] or proposing *sequential Monte Carlo* methods (also called *particle filters*) for inference [165, 98]. The main limitation of the current proposals of the BPM is that all consider relatively short-term dependencies, with focus on beat and downbeat and ignoring other interrelations at other time scales. We will explore some ideas on how to address these limitations in Chapters 8 and 9.

[2] The architecture based on CNNs and RNNs presented in Vogl et al. 2017 [176] was developed in parallel to the one presented in Chapter 7.

3.1.2 Beat tracking

Feature extraction

Similarly to downbeat tracking, systems for beat tracking often rely on music knowledge and signal processing techniques for the feature design. In this case, the main features exploited are those related to event-oriented indicators, which assumes that changes in the spectral energy relate to beat positions. The principal features exploited (see Table 3.1) are SF [41, 95, 79, 98, 129, 165] and ODF in general [57]. [3] As in the case of downbeat tracking, some approaches exploit the use of multiple features inspired on different music attributes [187, 88], such as energy flux (EF), complex spectral difference (CSD), a harmonic feature (HF), mel auditory feature (MAF) and phase slope function (PSF). Finally, a few systems exploit the use of logarithmically scaled STFTs [15, 92].

Beat likelihood estimation

Again as in downbeat tracking systems, most beat tracking methods use machine learning to map the input features into beat likelihoods. The two main methodologies used are GMMs [95, 79, 98, 165–167] and DNNs [92, 13, 15, 78]. The state-of-the-art systems are those who exploit DNNs and Mel log STFTs, as the multi-model system of [13] and the joint beat and downbeat tracker in [15].

Beat sequence extraction

Starting from 2010, most systems for beat tracking (as shown in Table 3.1) use probabilistic graphical models as processing stages. The most popular approach is, as in the case of downbeat tracking, the BPM [95, 13, 79, 92, 98, 165, 166, 15, 93, 167]. The second most popular method is HMMs [140, 41, 88, 187, 129], with CRFs being rarely used [57]. The reason why probabilistic graphical models are so popular in the context of beat tracking is because they are flexible models that allow to include rules such as that beat periods change smoothly over time in most cases, which helps improving estimated likelihoods.

3.2 Microtiming analysis

Microtiming has been studied in the context of Music Information Retrieval (MIR) for many years [8, 64, 37]. Besides the interest in characterizing microtiming for musicological studies, it is important for music synthesis applications, since it is a central component for “humanizing” computer generated performances [73]. Depending on the musical context, mi-

[3] In some cases the ODF is based on a SF. We will disambiguate this in case is relevant to the discussion of this manuscript.

crotiming can take the form of tempo variations, like *rubato* or *accelerando*, or small-scale deviations of events with respect to an underlying regular metrical grid [87]. Therefore, in order to study microtiming deviations one has to know the expected position of the events within the metrical grid and the actual articulated positions—which can be inferred from information related to the onset times, the tempo and/or the beats.

Methods for the analysis of microtiming deviations can be split in two groups depending on the main techniques they exploit: methods based on patterns, and methods based on autocorrelation. In the following we present a brief review of those methods, mentioning methods applied to the analysis of microtiming in the context of Afro-Latin American music to contextualize the work in Chapter 9.

3.2.1 Methods based on grouping/statistics

The methods for microtiming analysis based on patterns [64, 125, 87, 126] usually exploit a first stage of feature extraction using ODF or SF, and a second stage of grouping patterns derived from these feature vectors, considering a temporal context corresponding to one beat, one or two bars, depending on the music genre in consideration. Then the beat or bar is quantized into an arbitrary number of time steps that makes sense for the music under study—i.e. that will allow to observe the phenomenon under study. For clustering the different patterns, k-means is usually employed. The main idea of grouping and comparing clusters is to characterize a recording by the main rhythmic patterns, while preserving the microtiming deviations that appear in the performance.

Some works based on this methodology were concerned with Brazilian rhythms. That's the case of [64, 125], which studied microtiming deviations in *samba* related genres—i.e. *samba de roda*, *samba carioca*, *samba de enredo*, *partido-alto*, *samba de roda baiano*. In these works, either an external beat tracking algorithm or manually annotated beats (and downbeats) are used to segment meaningful patterns. Those patterns are obtained using features such as complex spectral difference, onset detection function [64] or loudness curves based on auditory models [125]. Relevant observations were made in those works regarding the nature of these Brazilian *samba* related genres: 1) most patterns present local maxima at the sixteenth note level, indicating that those genres have a big amount of rhythmic patterns articulating that temporal level; 2) third and fourth sixteenth notes in these rhythmic patterns tend to be articulated ahead of their correspondent synchronous positions—i.e. positions given an equal subdivision of the beat interval; and 3) accents are usually located in the second and fourth sixteenth note, illustrating the contrametric tendency—i.e. to not follow beat or metric positions—of these music traditions.

In this line, some works were carried out concerning other Latin-American rhythms such as Malian jembe [141] and Uruguayan *candombe* [87]. In the case of *candombe*, though there is no clustering stage with k-means, the analysis is performed considering beat-length pat-

terns, grouping them, and studying the resulting statistics. Similar trends to *samba* genres are found. In particular, [87] reported that the third and fourth sixteenth notes in many rhythmic patterns concerned to the sixteenth note level are played ahead of their synchronous positions. Even though there is a clear similarity to Brazilian *samba* rhythm, to the best of our knowledge there are no comparisons in the literature between these rhythms. We address this in Chapter 9.

3.2.2 Methods based on autocorrelation

Dittmar et al. address microtiming deviations in jazz [45, 44], and Marchand et al. [107] determine the "level of swing" of diverse genres. These works are concerned with simpler rhythmic patterns—consisting of two eight-notes—and the authors use the autocorrelation computed from an ODF as feature. By looking at the autocorrelation one can obtain information about the relative position of events in a straightforward manner, similarly as it is done by clustering different pattern occurrences as in 3.2.1. The authors of these works propose interesting representations of microtiming deviations such as the *swing-ratio* [107] or the *swingogram* [44], which intend to bring information about the performance other than meter and tempo. This kind of effort in the literature helps closing the gap between what characterizes a musical performance and what automatic-analysis tools can retrieve. However, there is room to further develop these methods, e.g. by considering different rhythmic patterns or developing fully-automatic methods.

Table 3.1 Broad overview of recent methods for beat (*b*) and downbeat (*db*) tracking. The variants adopted at the different stages of the pipeline are indicated: input features (*features*), downbeat likelihood estimation (*likelihood*) and post-processing (*post-proc*). The different variations of the BPM simple indicated as (BPM).

authors	task		approach		
	b	db	features	likelihood	post-proc
Peeters (2010) [140]	✓	✓	CH + SB	template	HMMs
Papadopoulos (2010) [132]		✓	CH	template	HMMs
Degara (2011) [41]	✓		complex SF	comb filterbank	HMMs
Khadkevich (2012) [88]	✓	✓	SF + CH	multiple-pass decoding	HMMs
Hockman (2012) [76]		✓	CH + ODF + SC	—	SVR
Krebs (2013) [95]	✓	✓	SF	GMMs	DBN (BPM)
Zapata (2014) [187]	✓		EF+SF+CSD+HF MAF + phase slope	—	HMMs
Böck (2014) [13]	✓		mel log STFT	RNNs	DBN (BPM)
Durand (2014) [53]		✓	CH F0 + MFCCs	—	SVM
Holzapfel (2014) [79]	✓	✓	2D-SF	GMMs	DBN (BPM)
Srinivasamurthy (2014) [168]	✓	✓	tempogram + ODF	SSM	peak-pick
Korzeniowski (2014) [92]	✓		logSTFT	Bi-LSTMs	DBN (BPM)
Krebs (2015) [94]	✓	✓	mel log STFT + 2D-SF	GMMs + RNNs	DBN (BPM)
Krebs (2015) [98]	✓	✓	log SF	GMMs	DBN (BPM)
Srinivasamurthy (2015) [165]	✓	✓	2D-SF	GMMs	DBN (BPM)
Durand (2015) [50]		✓	LFS+ODF+MFCC+CH	DNNs	average
Fillon (2015) [57]	✓		ODF + tempogram	—	CRF
Nunes (2015) [129]	✓	✓	SF + AC + STFT	template	HMMs
Srinivasamurthy (2016) [166]	✓	✓	2D-SF	GMMs	DBN (BPM)
Böck (2016) [15]	✓	✓	mel log STFT	Bi-LSTMs	DBN (BPM)
Durand (2016) [51]		✓	3D-ODF+CQT+CH	CNNs	average
Krebs (2016) [93]		✓	CH+ODF	Bi-GRUs	DBN (BPM)
Holzapfel (2016) [78]	✓	✓	mel log STFT	CNN	DBN (BPM)
Durand (2016) [54]		✓	LFS+ODF+MCQT+CH	CNNs	CRF
Durand (2017) [52]		✓	LFS+ODF+MCQT+CH	CNNs	HMMs
Srinivasamurthy (2017) [167]	✓	✓	2D-SF	GMMs	DBN (BPM)
Vogl (2017) [176]	✓		logSTFT	CRNNs	peak-pick
Cheng (2018) [25]	✓		mel log STFT	RNNs	DBN (BPM)
Zahray (2019) [186]	✓	✓	mel log STFT + CH	RNNs	DBN (BPM)

Part II: Datasets and tools

Chapter 4

Datasets and evaluation metrics

Summary

In this chapter we describe the datasets and evaluation metrics used in the following chapters of this dissertation. We present an overview of their different characteristics as well as some remarks on how they were used in the context of this work. In the following chapters, minimum information about the datasets is provided.

In order to evaluate the methods presented in this dissertation, we use a total of 10 datasets covering different music genres. Tables 4.1 and 4.2 present an overview of these datasets and their annotations. The datasets cover a wide range of Western music genres, as well as cultural-specific genres. In particular, they include pop, rock, jazz, Western classical music, ballroom dances, Uruguayan candombe and Brazilian samba (the last two being Afro-rooted Latin American music traditions). The main task addressed in this dissertation is downbeat tracking, but we also addressed the relation of downbeats with music structure and micro-timing with beats, so we also needed datasets that provided suitable annotations for those tasks. The variety of annotated data of Tables 4.1 and 4.2 allows us to tackle interesting musical questions and evaluate our algorithms properly. In the following, we provide a short description of each dataset we used as well as some comments on modifications we introduced in some cases to adapt this data to our research questions.

4.1 Datastes

RWC Jazz

This dataset was introduced by Goto et al. [62] and is part of the *Real World Computing Music Database*[1], which comprises multiple annotated datasets of different genres, and it was

[1] https://staff.aist.go.jp/m.goto/RWC-MDB/#rwc_mdb_subwg.

Table 4.1 Datasets used in this dissertation.

name	# files	genre	duration
Ikala [24]	252	various genres	2h 6min
Salami [163]	1359	various genres	105h 31min
RWC Jazz [62]	50	jazz	3h 44min
RWC Pop [62]	100	pop	6h 47min
Beatles [108]	179	pop	8h 01min
Ballroom [65]	698	ballroom dance	6h 4min
Hainsworth [69]	222	various genres	3h 19min
Klapuri [137]	320	various genres	5h 09min
Robbie Williams [43]	65	pop	4h 31min
Rock [39]	200	rock	12h 53min
Candombe [149]	35	candombe	2h
BRID [Chapter 5]	367	samba and others	2h 57min

Table 4.2 Overview of annotated beats and downbeats available in each dataset (except Salami and Ikala).

dataset	beat	downbeat
RWC Jazz	18922	5500
RWC Pop	43322	10842
Beatles	52729	13938
Ballroom	44603	12221
Hainsworth	22339	6180
Klapuri	31834	8128
Robbie Williams	25754	6603
Rock	84275	21474
Candombe	18800	4700
BRID	17566	8884

facilitated by the Japan's National Institute of Advanced Industrial Science and Technology (AIST). This particular dataset consists of 50 music pieces belonging to the genre of jazz, with durations ranging from 2 to 7 minutes. Metadata information about the pieces can be found in https://staff.aist.go.jp/m.goto/RWC-MDB/rwc-mdb-j.html. They are characterized by having multiple instrument solos and being only instrumental pieces.

RWC Pop

This dataset also makes part of the RWC Music Database [62]. It contains 80 music pieces of Japanese pop from the 90's, and 20 pieces of American pop from the 80's. The duration of the pieces ranges from 3 to 6 minutes.

Beatles

The Beatles dataset was introduced by Mauch et al. [108], consists of 12 CDs of the band The Beatles, for a total of 179 songs. The main style of this dataset is pop. This dataset comprises songs with time signature changes and music variations, making it interesting for meter tracking. Besides, the dataset has annotations of music segments (i.e. *verse*, *chorus*). Figure 4.1 shows the distribution of annotations per segment class in this dataset. We grouped 9 classes with less than 50 occurrences in bigger groups that made semantic sense (e.g. *verse_instrumental* to *verse*) to simplify the annotations for the algorithm in Chapter 8. We end up with a total of 1628 segment annotations.

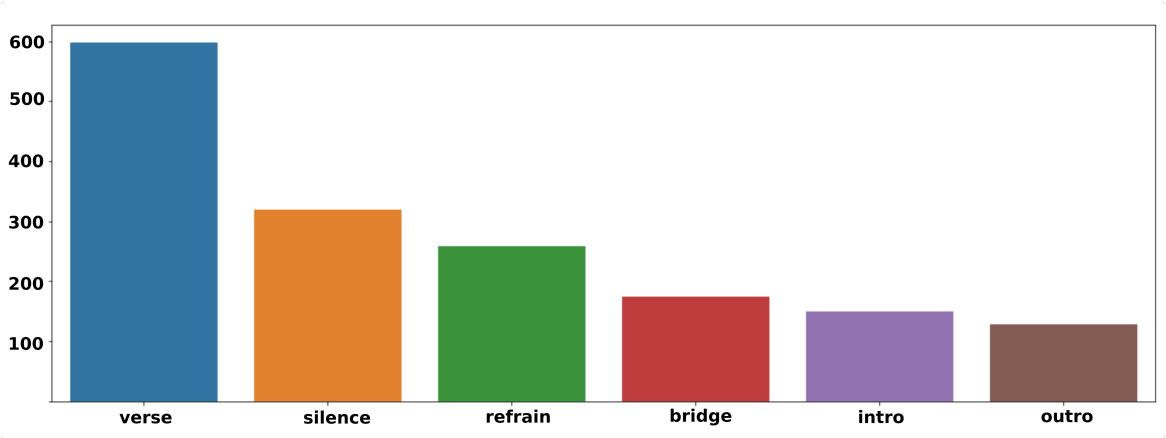


Figure 4.1 Distribution of segment classes in the Beatles dataset.

Hainsworth

This dataset was introduced by Hainsworth et al. [69]. It comprises 222 excerpts of diverse music genres, namely pop, rock, electronic music, folk, classical music, choir music and jazz. The duration of the excerpts is 1 minute approximately.

Klapuri

This dataset was produced by Paulus et al. [137] and by Klapuri et al. [90]. It consists of 474 pieces of music covering different music genres. The track durations are one minute in

mean, and the genres this dataset covers are mainly classical music, electronic music, hip hop, jazz, blues, rock, pop, soul and funk. A remark on this dataset is that the annotations for beat and downbeat were produced separately, and they do not always match. Among the 474 tracks, 320 have available downbeat annotations, which we use here.

Robbie Williams

This dataset consists of 65 tracks from the first 5 albums of the singer Robbie Williams. It was introduced by Di Giorgi et al. [43]. It comprises mainly pop and rock songs. The downbeat annotations of this dataset were provided by Böck et al. [15]

Ballroom

This dataset was created by Dixon et al. [48], and it consists of 698 tracks of ballroom dance music. There are pieces from 8 different styles of ballroom music: cha-cha, jive, quickstep, rumba, samba, tango, Viennese waltz and slow waltz. The mean duration of the tracks is 30 seconds. The downbeat annotations in this dataset were provided by Krebs et al. [95].

Candombe

This dataset consists of pieces of percussive ensemble music known as Candombe, a tradition of African roots developed in Uruguay. The data consists of audio and video recordings of ensembles of three and four drums, and associated data including meter and onset annotations. It contains tracks with isolated strokes, solo performances and drum ensembles. The dataset comprises 51 takes, for a total duration of nearly an hour and a half.

Ikala

The iKala dataset [24] is commonly used for melody estimation, vocal activity detection, and source separation. It contains isolated vocals and instruments (provided as left and right channels of a stereo audio file), along with vocal pitch contour annotations and lyrics. It contains 252 tracks of 30 seconds sampled from iKala songs.

Salami

The Salami dataset [163] is a popular dataset used for music structural segmentation. It consists of 1359 tracks across a wide variety of genres, namely classical, jazz, popular, world, among others. Each track has annotations of ‘coarse’ and ‘fine’ segments, and among the annotated files, a subset of 884 tracks was annotated by two distinct annotators.

4.2 Evaluation metrics

4.2.1 F-measure

Several evaluation measures for beat/downbeat tracking have been proposed in previous works [36, 35]. The most widely adopted measure is the *F-measure* or F1 score [132, 140, 53, 50, 15, 176, 93, 52, 54], so it is the metric most used in this work to compare to previous works. We evaluate the proposed systems by means of the F-measure implemented in the *mir_eval* library [145]. The F-measure is computed as:

$$F = \frac{2PR}{P+R} \tag{4.1}$$

where P denotes *precison* and R *recall*, defined as:

- *precision*: the ratio between the events that were correctly estimated and the total estimated ones;
- *recall*: the ratio between correctly estimated events and the total annotated events (that should have been retrieved).

The range of this measure is from 0% to 100%. Following previous works [15, 52, 93] we define as corrected estimated beats/downbeats those that are inside a tolerance window of 70 ms around the annotated event. In the case that we use another tolerance, as in Chapter 9, we indicate it explicitly.

Continuity-based measures

In continuity-based metrics, an estimated beat is considered correct if it is within a small tolerance around an annotation, the previous estimation has also been deemed correct, and the inter-beat interval is consistent with the inter-annotation interval within another tolerance—both generally set to 17.5% of the inter-annotation interval. CMLt ("correct metrical level") is the ratio between correct and annotated beats. The AMLt ("allowed metrical level") accepts phase errors of half a beat period or octave errors in estimation. We use these metrics jointly with F-measure and Information gain in Chapter 5 to evaluate an ensemble of beat trackers, since further information about them would be useful to discriminate between them.

Information-Gain

Defined as the Kullback-Leibler divergence between the observed beat error histogram (considering the timing errors of all estimated beats within a beat-length window around the annotations) and a uniform one (accounting for a pair of unrelated beat sequences), it spans the range $[0, \log_2(K)]$ bits, where K is the number of bins in the histograms (usually 40).

Chapter 5

Datasets of Brazilian music for computational rhythm analysis

Summary

In this chapter we describe two datasets developed in the context of this dissertation in collaboration with the STAREL project. The first one (BRID), consists of a copyright-free dataset and contains short solo- and multiple-instrument tracks of different Brazilian rhythmic styles. The second one (SAMBASET), is a dataset of Brazilian *samba* music that contains over 40 hours of historical and modern *samba de enredo* commercial recordings. To the best of our knowledge, these are the first datasets of this genre dedicated to computational rhythm analysis. In the following we describe the data collections and show preliminary experiments to illustrate the datasets' value.

5.1 Motivation

Within the Music Information Retrieval (MIR) community, there are several music audio databases available for tasks such as genre classification, tempo and beat tracking, chord recognition and music structure analysis [121]. These corpora are composed of either synthetic or real audio items, which in turn might have been either recorded for research purposes or sampled from commercial recordings [139]. Most available databases are composed of Western music, thus a large portion of MIR research is focused on music genres such as pop, rock, jazz or classical. This compromises the ability of MIR systems to deal with culturally-specific music [156], and limits the research scope of MIR methods.

Some datasets attempt to be universal and to cover a large number of music styles, but end up sacrificing the very representation of what they are trying to portray. This is the case, for example, of the well-known Ballroom and Extended Ballroom datasets, whose "Samba" class contains a mixture of songs of different origins, of which only a few examples

correspond to Brazilian rhythms. In other datasets, music from non-Western traditions is given generic labels as “Latin”, or “World”. This underscores the importance of the study of non-Western traditions found throughout the multicultural world we live in.

Recent efforts, such as the CompMusic project [156], help increasing the diversity of available datasets and step towards considering culturally-specific cases of study. However, Latin-American music genres are still under-represented, which is unfortunate given their rich and peculiar rhythms, which are appealing for computational rhythm studies.

In this chapter we present two novel datasets of Brazilian music genres for computational rhythm analysis: the Brazilian Rhythmic Instruments Dataset (BRID) and Sambaset. The BRID is copyright-free and contains short solo- and multiple-instrument tracks in different Brazilian rhythmic styles. We include a set of experiments on beat and downbeat tracking, rhythmic pattern recognition, and microtiming analysis performed on *samba* samples from the BRID to illustrate the need of such a dataset. Besides, we illustrate how some specificities of Brazilian rhythms may mislead state-of-the-art methods for computational analysis originally tailored for traditional Western music, and thus call for the development of more robust MIR tools. On the other hand, SAMBASET, is—to the best of our knowledge—the first large dataset of annotated *samba de enredo* recordings. We describe the dataset contents detailing the beat and downbeat annotation process, and we highlight possible musicological uses of this dataset.

5.2 Related work

5.2.1 Culturally-specific datasets

One of the biggest projects for the creation of datasets devoted to non-Western music traditions is CompMusic [156]. CompMusic focuses on five particular music cultures: Arab-Andalusian, Beijing Opera, Turkish-makam, Hindustani, and Carnatic. Within this project and for the different datasets, annotations for several tasks are provided, including melody (e.g. singer tonics, pitch contours), rhythm and structure (e.g. tala cycles), scores (e.g. for percussion patterns), and lyrics.

There are also some datasets of Latin-American music with annotated data suitable for MIR research. For instance, the dataset released in [129] comprises annotated audio recordings of Uruguayan *candombe* drumming, suited for beat/downbeat tracking. Aimed at music genre classification, the Latin Music Database [161] has Brazilian rhythms—*axé*, *forró*, *gaúcha*, *pagode*, and *sertaneja*—and music from other traditions: *bachata*, *bolero*, *merengue*, *salsa*, and *tango*. Closer to the genres of the datasets described in this chapter, there exist two datasets exclusively focused on Brazilian music: the first one, intended for music genre classification, is the Brazilian Music Dataset [164], which includes *forró*, rock and *repente*. The second one, the MPB (Brazilian popular music) dataset, contains *brega*, *sertanejo*, and

disco music genres, and was meant for beat/downbeat tracking and rhythmic pattern analysis.

5.2.2 Brazilian samba

Samba plays a special role in Brazil's image overseas. Every year, the country receives millions of tourists for Carnival activities in cities such as Salvador and Rio de Janeiro. Being Brazil's quintessential rhythm, *samba*'s development is closely related to that of Brazil itself.

Samba's roots can be traced back to dance and religious practices from the Afro-Brazilian diaspora [3, 74] and, as Araujo [3] points out, to the accommodation efforts made by people of African descent to maintain their heritage and cultural identity despite slavery and persecution. In many of these cultural practices, participants would form a *roda* (circle) and accompany one or more dancers (positioned at the center of the *roda*) by clapping, singing, and occasionally playing instruments [3, 152]. These traditions gave origin to different cultural manifestations, collectively associated with the term *samba*,[1] for example: *coco*, *samba de roda*, *partido-alto*, *samba de terreiro*, *pagode*, among others. In the post-Abolition period, *samba* overcame prohibition to become Brazil's national rhythm.

In the 1930s, the genre evolved to the rhythmic framework that still defines it today—generally characterized by duple meter (i.e., binary division of the periodically perceived pulsations) and strong syncopation. However, the idea of syncope—momentary contradiction of the prevailing meter or pulse [146]—can only be adequately applied to "Western" music, creating a fundamental problem in music traditions where this disruption of the pulse is the norm, and not the exception. That is why some authors prefer to resort to the concepts of commetricity and contrametricity [152], which indicate respectively when the surface rhythm confirms or contradicts the underlying meter, a terminology more commonly used in African music studies [91, 4, 152]. Therefore, it is more appropriate to say that *samba* presents a strong tendency towards contrametricity.

Later developments in *samba* led to, arguably, the most internationally famous of all its subgenres, the *samba de enredo*. These are *sambas* subject to an *enredo* (plot) composed in the context of an *escola de samba*[2], and presented at parades in an organized competition annually held along a so-called Sambadrome during Carnival. At the core of every *escola de samba* lies the *bateria* (percussion ensemble). During a performance, the rhythmic aura of a *bateria* is created by the superposition of several cyclical individual parts, assigned to each multi-piece instrument set, similarly to what is observed in percussion ensemble

[1] Possibly a variation of *semba*, word for a kind of circle dance practice in the Angolan Kimbundu language [152].

[2] A popular association for the practice of *samba*. *Samba schools* are usually strongly connected to a specific community, where their social events take place and to whom they provide several social services. The climactic event for *samba schools* is the annual carnival parade, when imbued with communal effort they compete for the title.

practices throughout sub-Saharan Africa [3]. The *bateria* sets the mood of *samba*, but recent studies have observed an increase in the average tempo of *bateria* performances, an effect attributable to stricter parading time constraints [82, 142, 33].

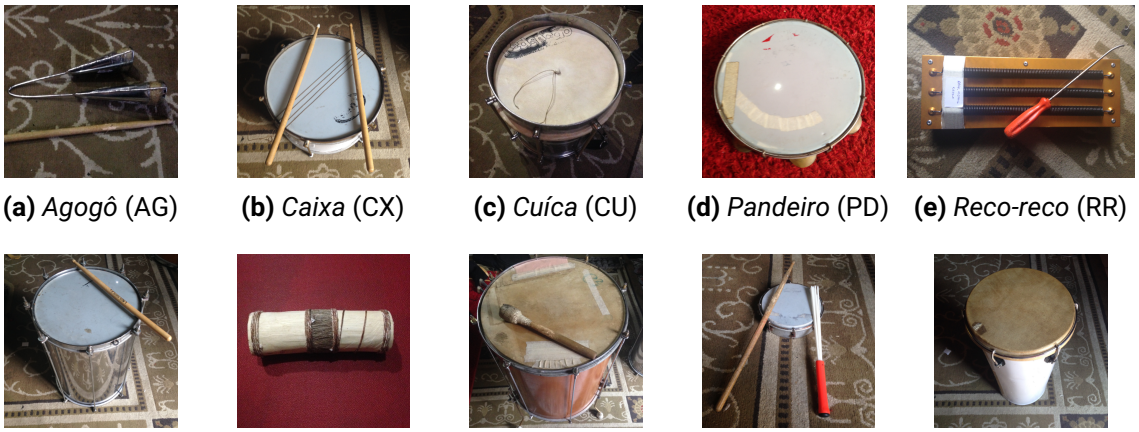
5.3 BRID

5.3.1 Dataset overview

The BRID was originally developed in the context of sound source separation [40], but its applicability can be extended to other areas, computational rhythm analysis in particular. The dataset contains 367 short tracks of around 30s on average, totaling 2 hrs 57min. The tracks consist of recordings of solo or multiple instruments, playing characteristic Brazilian rhythms.

Instruments and Rhythms

The recorded instruments were selected among the most representative ones in Brazilian music, more specifically in *samba* music. Ten different instrument classes were chosen: *agogô*, *caixa* (snare drum), *cuíca*, *pandeiro* (frame drum), *reco-reco*, *repique*, shaker, *surdo*, *tamborim* and *tantã*. To provide a variety of sounds, both membranophones and idiophones were featured. Also, whenever possible, instruments were varied in shape, size, material (e.g., leather or synthetic drumhead), pitch/tuning (e.g., in a *samba school*, *surdos* are usually tuned in three different pitch ranges) and in the way they were struck (e.g., with the hand, or with a wooden or a plastic stick), spanning a total of 32 variations. For example, the dataset features two *caixa* variations (12” in diameter with either 4 or 6 snare wires), six *pandeiro* variations (either 10”, 11” or 12” in diameter with a leather or nylon drumhead) and three *tamborim* variations (one with a leather head struck with a wooden stick, and another



(a) *Agogô* (AG) **(b)** *Caixa* (CX) **(c)** *Cuíca* (CU) **(d)** *Pandeiro* (PD) **(e)** *Reco-reco* (RR)

(f) *Repique* (RP) **(g)** Shaker (SK) **(h)** *Surdo* (SU) **(i)** *Tamborim* (TB) **(j)** *Tantã* (TT)

Figure 5.1 Instrument classes in BRID.

one with a nylon head struck with either a wooden or a plastic stick[3]). Figure 5.1 shows the instrument classes considered.

The recordings present instruments played in different Brazilian rhythmic styles. Although *samba* and two sub-genres (*samba-enredo* and *partido alto*) have been favored, BRID also features *marcha*, *capoeira*, and a few tracks of *baião* and *maxixe* styles. The number of tracks per rhythm is summarized in Tables 5.1 and 5.2.

Table 5.1 Tempi/number of solo tracks per rhythm.

Rhythm	Tempo (bpm)	*# Tracks*
Samba (SA)	80	54
Partido alto (PA)	100	55
Samba-enredo (SE)	130	60
Marcha (MA)	120	27
Capoeira (CA)	65	12
Samba - virada (VSA)	75 or 80	3
Partido alto - virada (VPA)	75 or 100	36
Samba-enredo - virada (VSE)	130	17
Marcha - virada (VMA)	120	8
Other (OT)	-	2

Table 5.2 Number of multi-instrument tracks per rhythm.

Rhythm	*# Tracks*
Samba (SA)	41
Partido alto (PA)	28
Samba-enredo (SE)	21
Marcha (MA)	3
Capoeira (CA)	-

All featured rhythms are in duple meter. *Samba* is specially known for this type of bar division and for the accentuation of the second beat [60]. Only combinations of instruments and rhythms that are traditionally seen in Brazilian music were considered, to provide a faithful representation of each rhythm.

Dataset Recording

All the recordings were made in a professional recording studio in Manaus, Brazil, between October and December of 2015. The recording room has rectangular shape with dimensions of 4.3 m $\times$ 3.4 m $\times$ 2.3 m and is acoustically treated with a combination of wood and acoustic foam.

Both microphone model and positioning were optimized to translate the sound of each instrument as naturally as possible in the recording, considering the instrument size and the room acoustics. Most instruments were recorded with dynamic microphones within a

[3] A leather-head *tamborim* is not played with a plastic drum stick.

distance of around 20 cm. The digital files were recorded with a sampling rate of 44.1 kHz and 16-bit resolution.

There are two groups of tracks in the dataset. The first one consists of instruments recorded solo, with the musicians performing in various Brazilian styles following a metronome track. Three musicians were recorded separately, each playing around 90 different instrument–rhythm combinations. For each instrument class, there is at least one track that consists of a *virada* of one of the main rhythms.[4] These are free improvisation patterns (still subject to the metronome track), which are very common in *rodas de samba*.[5] It is worth mentioning that the musicians brought their own instruments for the recording sessions. Although the general characteristics of each instrument are the same, e.g., size and type of material, subtle differences in construction bring additional timbre variability to the dataset.

The second set of tracks of the dataset gathers group performances, with the musicians playing together different rhythmic styles without a metronome reference. The instruments were individually captured with directional microphones, which were strategically positioned to minimize sound bleed, and two condenser microphones in omni polar pattern captured the overall sound in the room. The performances were designed to emulate typical arrangements of each style. Following this procedure, 19 recordings were made with four musicians, 29 with three musicians, and 45 with two musicians playing at a time.

Track Labeling

Each audio track is given a unique filename, which starts with a four-digit number between brackets—a global identification number `[GID#]`, sequential for the entire dataset.

In solo track (`S`) filenames, the `GID#` is followed by four groups of characters, whose format is either `SW-XXX-YY-ZZ` or `SW-XXX-YY-VZZ`, where `W` is the number for the musician playing in the track, `XXX` specifies the instrument class and variation being played, `YY` consists of a counter for tracks with the same pair musician–instrument, and `ZZ` (or `VZZ`) indicates the rhythmic style (or a *virada* for that style).

For acoustic mixture tracks (`M`), the `GID#` is followed by three groups of characters, whose format is `MW-YY-ZZ`. Here, `W` indicates the number of instruments recorded in the track, `YY` is the counter for a given `MX` prefix, and `ZZ` means the same as in the case of solo tracks. The unique identifier for each instrument class and for each rhythm can be found in Figure 5.1, and in Tables 5.1 and 5.2, respectively.

Two samples from the dataset: file `[0192] S2-PD3-01-SA.wav` contains a solo recording of a *pandeiro* (variation 3: 11"; leather head) being played by musician #2 in a *samba* style; and file `[0010] M4-10-SE.wav` is a *samba-enredo* track performed by four musicians.

[4] Except for shaker tracks.

[5] A small and informal gathering to play and dance to *samba* music. It is a communal practice highly characterized by improvisation where musicians and dancers interact and learn with one another.

Table 5.3 Selected *solos*.

Filename	Instrument
[0218] S2-TB3-01-SE	*Tamborim*
[0229] S2-CX2-02-PA	*Caixa*
[0258] S2-SK2-02-PA	Shaker
[0280] S2-SU2-05-SE	*Surdo*

Table 5.4 Selected *mixtures*.

Filename	Instruments
[0013] M4-13-SE	*Cuíca, caixa, tamborim, surdo*
[0039] M3-20-SE	*Caixa, tamborim, tantã*
[0047] M3-28-SE	*Caixa, surdo, surdo*
[0051] M2-03-SA	*Tantã, surdo*

5.3.2 Experiments and discussion

Some experiments on computational analysis of musical rhythm are discussed in this section in order to show the usefulness of the dataset. First, beat and downbeat tracking tasks are addressed using a representative set of selected audio files exhibiting different characteristics. The aim of these experiments is to show the challenges inherent in *samba* music that arise when tackling these tasks, for this we use well known state-of-the-art systems to retrieve beat and downbeats and discuss the obtained results. Finally, based on the beat and downbeat annotations, we study the different rhythmic patterns present in a given audio recording.

Beat and Downbeat Tracking

Three state-of-the-art systems for beat and downbeat tracking are adopted in this work, namely MMB14 [13], JB16 [15], and K16 [93]. We used the implementations available in the madmom package.[6] JB16 and K16, which are the joint beat and downbeat tracker introduced by Böck et al. in 2016 and the downbeat tracker introduced by Krebs et al. 2016, will be introduced and explained in Chapter 7 which is concerned with state-of-the-art downbeat tracking systems, so we refer the reader to Section 7.2 for further information about these methods. In this section we use these methods only to motivate the usability of the dataset. We will then briefly explain MMB14, only used here, in the following. We refer as MMB14 to the model presented by Böck et al. [13] in 2014, which consists of multiple recurrent neural networks which are specialized to different music styles for the task of beat tracking. Each recurrent network consists of a concatenation of three Bi-Directional Long-Short Term Memory (Bi-LSTMs) hidden layers with 25 units per layer. The system chooses the most appropriate beat activation function for the given input signal by comparing the respective

[6] Madmom package version 0.16 [12].

Table 5.5 Beat F-measure scores for solos.

Model \ Track	0218	0229	0258	0280
MMB14 [13]	0.00	0.00	1.00	0.96
JB16 [15]	0.00	0.00	1.00	1.00

activation functions with a reference network trained in all music styles. Finally, tempo and beat phase are determined using a DBN.

A set of 8 audio files, representative of the content of the dataset, was selected for the experiments. It comprises 4 •solos and 4 *mixtures*, involving different rhythms (*samba*, *samba-enredo* and *partido alto*) and different instruments, as shown in Tables 5.3 and 5.4. We report the F-measure score (F) with a window of 70 ms.

Analysis of the selected *solos*

Many instruments, such as *tamborim* or *shaker*, have a rhythmic pattern that repeats itself within each beat (see Figure 5.2). Hence, it is very difficult (even for an expert) to establish the location of downbeats from a *solo* track without any further references. For this reason, only beat tracking is tackled in this section.

The beat positions for each *solo* track are estimated using the MMB14 [13] and JB16 [15] algorithms. The obtained results are presented in Table 5.5. A two-bar length excerpt of each audio file is shown in Figure 5.2, which depicts the annotated beat positions and the beat estimations for each algorithm. The annotations also indicate beat number in a 2/4 meter (i.e. 1 and 2). The rhythmic patterns in music notation are also provided.

The performance of both algorithms is very similar: they miss the phase of the beat in two of the files (0218 and 0229) and correctly track the other two (0258 and 0280). A detailed inspection of Figure 5.2 makes it clear that the troublesome rhythmic patterns, i.e. those of the *tamborim* and *caixa*, have strong phenomenological accents displaced with respect to the metric structure. Conversely, the pattern of the shaker accentuates every beat. In the case of the *surdo*, there are actually several different rhythmic patterns played, but most of the time the second beat of the bar is strongly articulated. This distinctive trait of *samba* rhythm, while advantageous for beat tracking, proved to be very challenging for downbeat estimation, as shown next.

Analysis of the selected *mixtures*

As for the mixtures, both beats and downbeats are tracked. The beats are estimated using the MMB14 [13] and JB16 [15], while the downbeats are estimated with K16 [93] and JB16 [15]. Since all the mixtures are in 2/4, we set the search-space of the DBN for downbeat tracking to bar lengths of 2 and 4 beats, both yielding the same results. Tables 5.6 and 5.7 show the beat and downbeat tracking results, respectively.

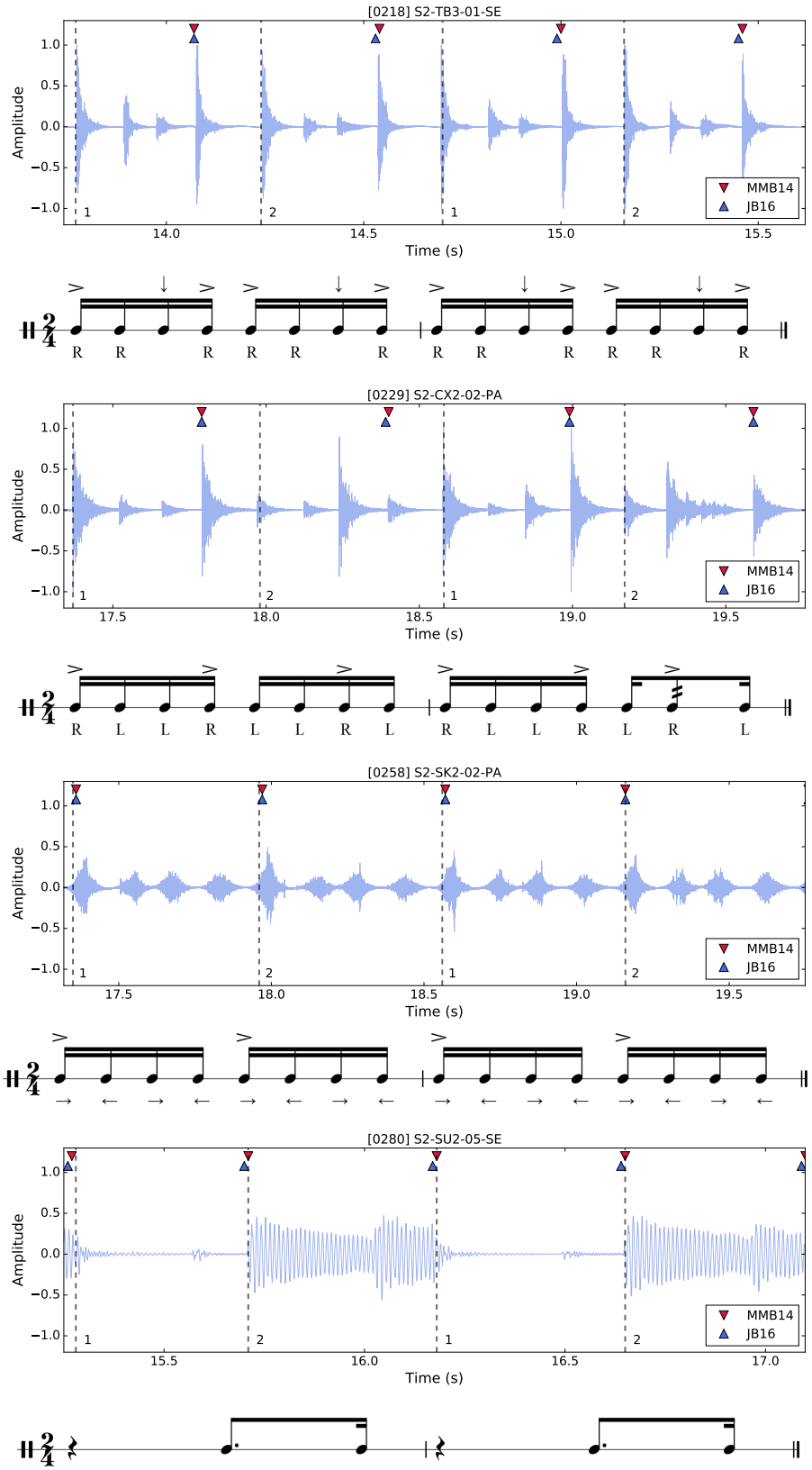


Figure 5.2 Beat tracking for the selected *solo* track examples. From top to bottom: *tamborim*, *caixa*, shaker and *surdo*. The waveform plots show two bars of the rhythmic patterns, with vertical lines indicating annotated beats. The estimated beats are depicted with markers. Rhythmic patterns are schematically represented in music notation. R and L indicate right and left hand respectively, the symbol '>' refers to an accent, and '↓' implies to turn the *tamborim* upside down and execute the strike backwards. The '→' and '←' indicate the movement of the *shacker*, forwards and backwards respectively.

Table 5.6 Beat F-measure scores for mixtures.

Model \ Track	0013	0039	0047	0051
MMB14 [13]	0.99	0.98	0.98	0.56
JB16 [15]	1.00	1.00	1.00	0.40

Table 5.7 Downbeat F-measure scores for mixtures.

Model \ Track	0013	0039	0047	0051
K16 [93]	0.00	0.00	0.00	0.00
JB16 [15]	0.00	0.00	0.00	0.00

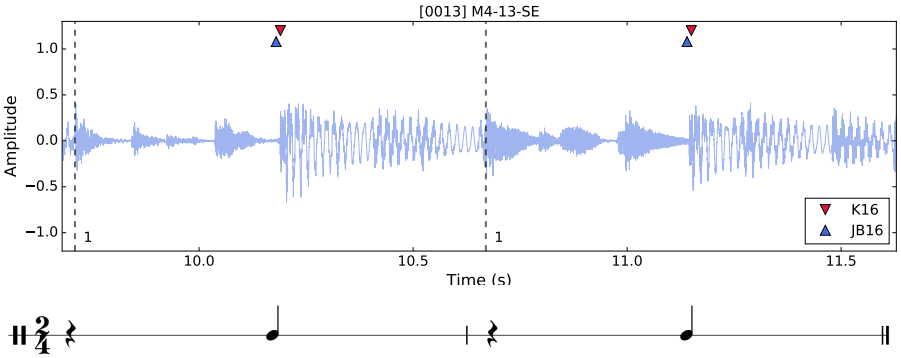


Figure 5.3 Downbeat tracking for one of the selected *mixture* track examples. The waveform plot shows two bars, with vertical lines indicating the annotated downbeats. The estimated downbeats are indicated by markers. The music notation shows the *surdo* onsets at the second beat of each bar, which is troublesome for downbeat detection.

Whereas beat tracking in the mixtures is not problematic (except for file 0051, in which half of the estimates are out of phase probably due to the anacrusis at the beginning), downbeat tracking is very challenging. Both algorithms fail to correctly track the downbeats for all the recordings. The downbeat estimates tend to follow the second beat, suggesting that the *surdo* is misleading the algorithms, as shown in Figure 5.3.

Pattern Extraction and Microtiming

In this experiment, we combine the knowledge of beat and downbeat locations (automatically annotated and manually corrected) with an onset detection function [46] in order to show the type of study of rhythmic patterns that could be carried out in this dataset. To that end, we follow the approach presented by Rocamora et al. in [148]. Firstly, the spectral flux is computed using the mel-STFT, for 40-ms windows with hop size of 2 ms. This onset function is then normalized by the 8-norm in a sliding window of length equal to half of the average bar length. Finally, the normalized onset function is time-quantized considering an isochronous grid at the sixteenth level anchored at downbeat pulses (assuming that in *samba* the lowest pulsation level can usually be found at the sixteenth-note level), and by taking the maximum value in the grid we obtain an 8-dimensional feature vector per bar. Vectors from adjacent bars are grouped as to form larger rhythmic patterns—here called cycles

(e.g., a 16-dimensional vector for 2-bar patterns). In each feature vector, articulated grid instants which translate to high onset function values are represented by a high value (close to 1.0, darker in the value of a cell in Figure 5.4). The feature vectors can then be displayed as rows of a cycle-length feature map [148], where a full track is explained in terms of the rhythmic cycles it contains. Furthermore, by clustering the feature vectors using K-means, one can recover 2-bar rhythmic patterns that are where cluster centroids act as templates for the patterns.

One such map can be seen in Figure 5.4 for an *agogô* recording in *samba-enredo* rhythm (file 0251). In this map, there are two different 2-bar length patterns that can be readily identified and that agree with the analysis of the recording by an expert musician. The bottom scores represent the patterns identified by the It is possible to see that, in the case of *samba* music, strong pulses are not always found at the start of a bar. In fact, for each rhythmic cycle (feature vector) in this example, the second beat of both bars, the first beat of the second bar and the last tatum are the ones competing for the role of strongest pulse.

Microtiming properties can also be studied by analyzing, for example, deviations of the articulated notes from their expected positions in the isochronous grid at each cycle. Figure 5.5 shows the calculated deviations (in percentages, relative to the average tatum period) for each point in the isochronous grid for the two patterns found in the *agogô* recording. As downbeats determine the grid, no deviation is shown at these points. On average, the second beat of the first bar falls on the grid, whereas at the second bar it is slightly delayed. This is probably due to the patterns themselves, in which the musician has to strike two or three notes in rapid succession. All other points in the metrical grid are continuously played ahead of time, with the third tatum of the first bar showing the highest deviation (almost 23 ms ahead of position at a tempo of 130 bpm). Gouyon reported a similar tendency to play ahead of the quantized grid in *samba de roda* recordings in [64].

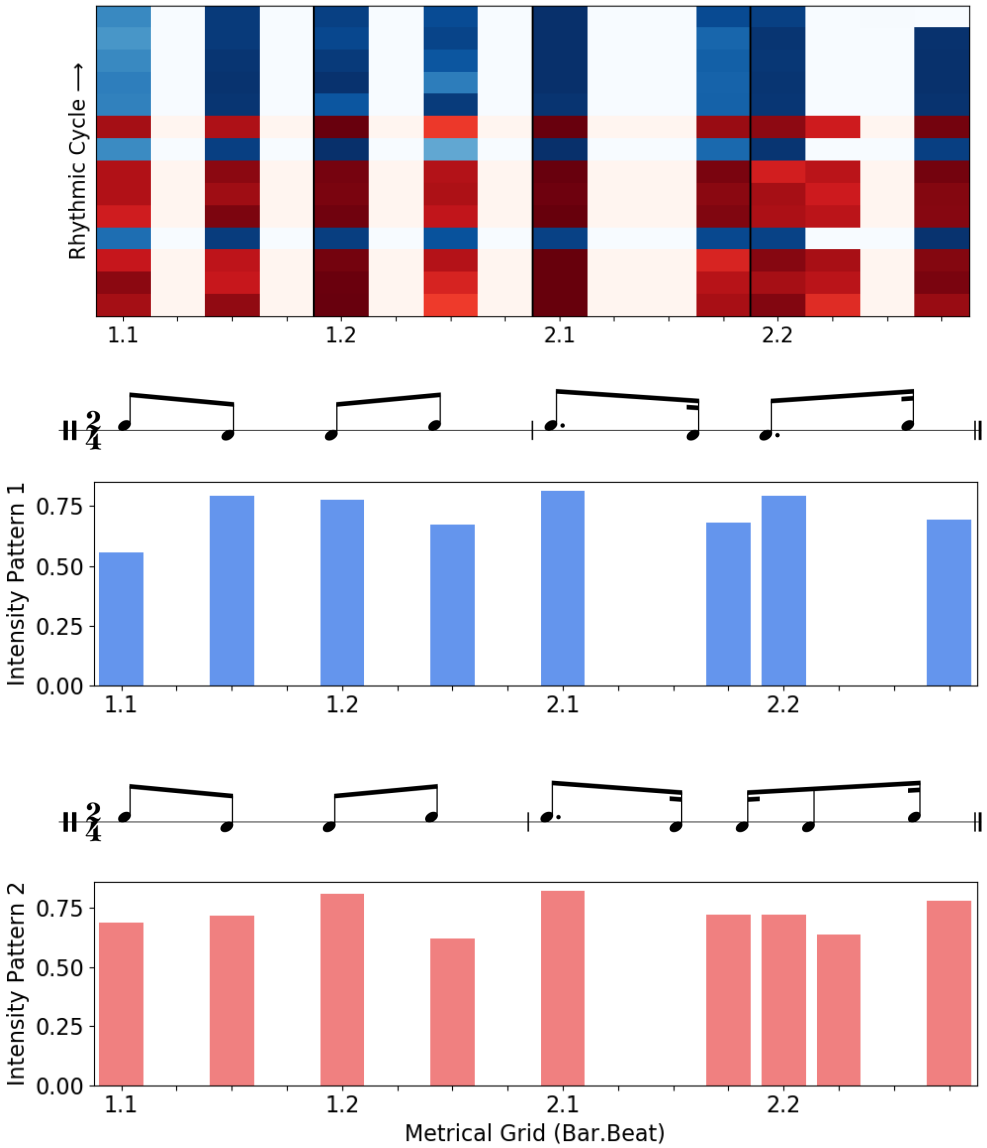


Figure 5.4 Feature map for the *agogô* recording. Two patterns (blue and red, transcribed as lines 1 and 2 above, respectively) are evident from this feature map.

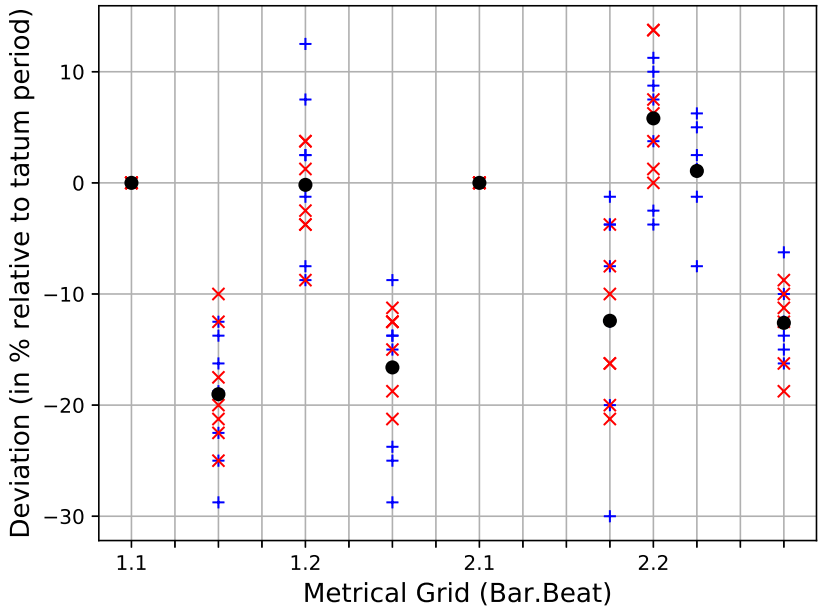


Figure 5.5 Deviations from the isochronous grid in the *agogô* recording. The different patterns are identified following the color convention of Figure 5.4.

5.4 SAMBASET

5.4.1 Dataset Overview

Sambas de enredo are well documented in the phonographic industry. Apart from historical collections, since 1968 the yearly *sambas de enredo* that competing *escolas de samba* will perform at the carnival parade have been professionally recorded and marketed. Initially available as LP records, these official compilations began to appear as CDs in 1990. Since then, the amount of musicians (instrumentalists and choir) in each track has only increased.

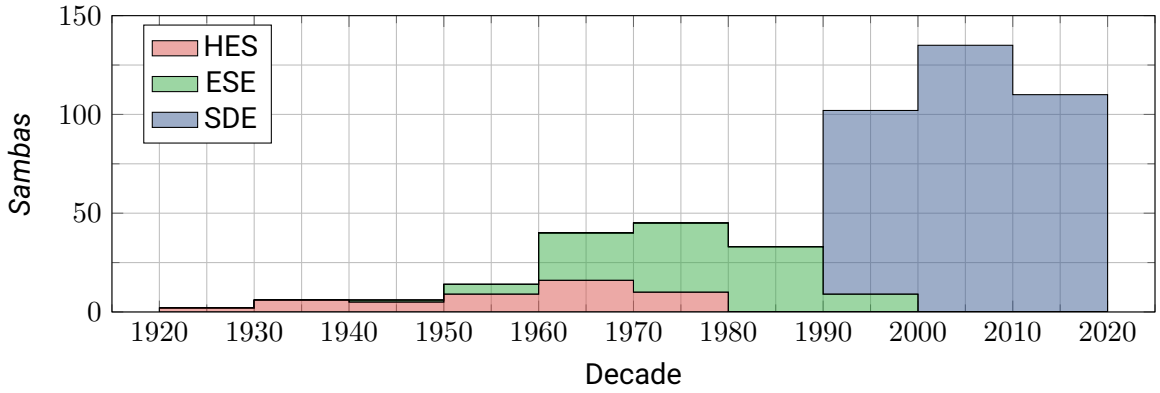


Figure 5.6 Recordings per decade of first performance. HES, ESE and SDE refer to the different collections included in this dataset: História das Ecolas de Samba, Escolas de Samba and Sambas de Enredo.

Currently comprised of audio recordings, annotations and metadata, SAMBASET covers different eras, from later renditions of old classics to the most recent *sambas de enredo* just out of the Sambadrome. Figure 5.6 indicates the distribution of *sambas* w.r.t. the year they were first performed (typically, the parading year). Three major collections make up the dataset; in chronological order:

- *História das Escolas de Samba* (HES): a collection of historical *sambas*, composed between 1928 and 1974, from four major *escolas de samba*, arranged and interpreted by the instrumentalists of each *escola*. Recorded in 1974, published in four LPs by Discos Marcus Pereira (redistributed as CDs in 2011), their 48 tracks also include a few *sambas de quadra/de terreiro* and *partidos-altos*.
- *Escolas de Samba – Enredos* (ESE): a collection of historical *sambas*, composed between 1949 and 1993, from ten traditional *escolas de samba* in the voices of many idols from *samba*'s history, accompanied by a selected ensemble of instrumentalists and choir. There are a total of 100 tracks recorded and released in 1993 by Sony Music. This collection includes a couple of tracks from different sub-genres (*samba de terreiro* and *samba-exaltação*).
- *Sambas de Enredo* (SDE): official compilations of *sambas de enredo* recorded by members of the top *escolas* from Rio de Janeiro, for each carnival parade between 1994 and 2018. The 25 CDs gather 338 tracks, published by RCA/BMG/Sony BMG (1994–2006) and by Universal Music (after 2007), with one *samba de enredo* per track.

Table 5.8 gives the number of tracks for each *escola de samba* featured in the dataset, by genre. In total, there are 493 recorded sambas in 486 audio tracks,[7] resulting in over 40 hrs 30 min of content. All files are stereo with a sampling rate of 44.1 kHz and 16-bit resolution. Not only do the three different collections allow for the coverage of different time periods, but they also have distinct sonorous characteristics. In HES, tracks feature only a few musicians playing very naturally and with great expression, as if they were in a *roda*. For several tracks in the official compilation (SDE), on the other hand, more than fifty instrumentalists play simultaneously while a choir of around the same size accompanies the main singer. Finally, ESE presents smaller ensembles and less expressiveness.

Metadata and Annotations

Metadata for albums and tracks were carefully curated and organized in an XML file. The information therein described was primarily obtained from CD booklets and later cross-

[6] Some imbalance can be observed in the distributions of both genres and *escolas*. However, ST/SQ and OT tracks were only kept for the completeness of the dataset in regard to the CD collections, and the *escolas*' playing styles are not so heterogeneous as to make this imbalance critical.

[7] Some tracks in the ESE collection contain more than one *samba*.

	Genres			
Escola	SE	ST/SQ	OT	Total
Mangueira	45	3	1	49
Portela	41	5	2	48
Salgueiro	42	5	-	47
Império Serrano	31	5	-	36
Mocidade	35	-	1	36
Beija-Flor	34	1	-	35
Imperatriz	35	-	-	35
Vila Isabel	33	-	-	33
União da Ilha	27	-	-	27
Grande Rio	25	-	-	25
Unidos da Tijuca	24	-	-	24
Viradouro	18	-	-	18
Estácio	16	-	-	16
Porto Da Pedra	15	-	-	15
Caprichosos	12	-	-	12
São Clemente	12	-	-	12
Tradição	11	-	-	11
Other *escolas* (7)	14	-	-	14
Total	470	19	4	**493**

Table 5.8 Number of recordings in *SAMBASET* separated by *escolas* and by genres: *samba de enredo* (SE), *samba de terreiro/samba de quadra* (ST/SQ), and others (OT).[8]

checked with both the *União Brasileira de Compositores*[9] (lit. Brazilian Union of Composers, UBC) and the *Instituto Memória Musical Brasileira*[10] (Brazilian Musical Memory Institute, IMMuB). Whenever corresponding information was available, data was also checked against online database services such as FreeDB, MusicBrainz or Discogs. Finally, we consulted *samba*-oriented forums and websites for additional, conflicting or missing information.

All XML tags can be seen in Figure 5.7. While most of these labels are straightforward (e.g. title, composer), some require further clarification. First, the album_code refers to a unique code given to each album in the dataset. Albums from the HES and ESE collections were sequentially numbered, i.e., they are referred to by the codes HES1 to HES4 and ESE1 to ESE10, respectively. For SDE, albums were specified via the publishing year, which is also present in the album's title (i.e., SDE1994–SDE2018). The track_number is used with the album_code to name all audio files (e.g. the metadata in Figure 5.7 corresponds to file HES1.06.wav). Track's start_time and end_time indicate the time each *samba* starts and ends, respectively. This is invaluable since many *samba de enredo* recordings are preceded by a short introductory speech or song motivating the performance, or succeeded by a

[9] http://www.ubc.org.br/
[10] https://immub.org/

```
<metadata dataset="SAMBASET"
          curator="John Doe"
          version="0.0.1">
...
  <album title="História das Escolas de Samba – Mangueira"
         arranger="Cartola"
         producer="J. C. Botezelli"
         instrumentalists="Various Artists"
         record_label="Discos Marcus Pereira"
         year_published="2011"
         length="00:29:48"
         total_tracks="12"
         album_code="HES1"
         barcode="7892141643634">
    ...
    <track track_number="6"
           title="Vale Do São Francisco"
           artist="Cartola"
           composer="Cartola and Carlos Cachaça"
           year_recorded="1974"
           year_first_performed="1948"
           genre="samba de enredo"
           length="02:49.226"
           samplerate="44100"
           bpm="78.4"
           start_time="00:07.895"
           end_time="02:49.226"
           checksum="d16974f135f0c374677c0e0db101cfea"/>
    ...
  </album>
...
</metadata>
```

Figure 5.7 Metadata file excerpt.

"farewell" shout after the music has already stopped. The checksum attributes were filled with the MD5 hash of the track's WAV file, to allow the verification of audio data integrity. Finally, mean bpm values were estimated from the beat annotations described in the following.

As of the writing of this dissertation, SAMBASET has annotations of beat and downbeat produced according to a semiautomatic procedure, after the results of the experiment described in Section 5.4.2. First, automatically-generated beat annotations were obtained for all audio files using the DBNBeatTracker system, which is available in the `madmom` package [12]—deemed a good candidate for providing reliable beat estimations (cf. Section 5.4.2). In a second step, these estimations were checked and manually corrected by one of the authors, who addressed eventual phase errors, and missing/extra beats. Since *samba de enredo* is always in duple meter, downbeats could be manually selected during this second

phase. This two-step procedure greatly reduced the amount of manual work necessary to annotate beats and downbeats for this entire dataset.

5.4.2 Experiments and discussion

Analysis of beat trackers' performance

In this section, we provide a performance analysis of different state-of-the-art beat tracking systems, which were applied on a subset of short (30-second) excerpts from SAMBASET. Samples were selected according to a criterion based on the mean mutual agreement between beat estimation sequences generated by the algorithms under analysis, inspired by the approach of Holzapfel et al. [77]. This subset was manually annotated using Sonic Visualiser [22] by an expert with an engineering background, knowledgeable of audio technologies, and with many years of experience as a practicing musician, in particular of *samba*. Estimations were evaluated against this ground truth using three different types of metrics.

Algorithms Considered

Fourteen algorithms (seen in Table 5.9) were used for sample selection and performance evaluation. We replaced eight of the algorithms originally featured in Holzapfel et al. [77] with six other algorithms released in following years, most notably those provided in the `madmom` package [12].

The algorithms are implemented in different programming languages and, in a few cases, require different operating systems. We used the Python implementations of AUB (version 0.4.9), ELL (provided by the librosa package [118] version 0.6.2), DEG and MFT (Essentia package [17] version 2.1-beta5-dev), BO1, BO2, and BO3 (madmom package [12] version 0.16.1); and the available releases of DIX (in Java, version 0.5.8), DAV (Vamp plugin in conjunction with the Sonic Annotator [21]), IB1 and IB2 (version 1.0 binaries). Finally, the C++ implementation of KLA was kindly provided by the author.

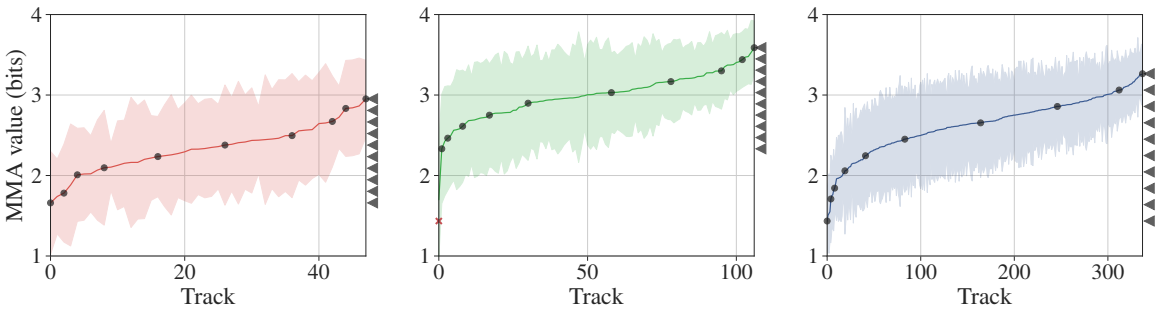
Selection of Ground Truth Excerpts

In [77], Holzapfel et al. presented a method for selecting challenging music examples for the beat tracking task without ground truth annotations. To do so, they first calculate the mean mutual agreement (MMA) between sequences estimated by a group of state-of-the-art beat trackers. The mutual agreement between two estimated beat sequences $\{i, j\}$ output by different beat tracking systems is given by the Information Gain in bits:

$$\mathrm{MA}_{i,j} = \mathrm{InfGain}(i, j), \quad i \neq j. \tag{5.1}$$

Beat Tracker	CMLt (%)	AMLt (%)	F-meas. (%)	Inf. Gain (bits)
Aubio (AUB) [20]	59.4	65.6	61.9	2.30
BayesBeat-HMM (KR1) [96, 97]	42.7	65.6	67.6	2.27
BayesBeat-PF (KR2) [96, 98]	47.6	52.9	58.0	2.25
*BeatRoot (DIX) [47]	79.4	82.8	86.4	3.15
Davies (DAV) [38]	97.2	97.2	97.5	3.66
*Degara (DEG) [41]	88.3	91.2	89.7	3.40
*Ellis (ELL) [55]	76.9	76.9	78.7	3.35
IBT causal (IB1) [130]	83.4	83.4	86.2	2.45
*IBT non-causal (IB2) [130]	51.1	90.8	80.0	2.49
*Klapuri (KLA) [90]	61.3	63.7	63.1	3.09
BeatTracker (BO1) [16, 14]	98.1	98.1	98.6	3.78
DBNBeatTracker (BO2) [13, 97]	**99.5**	**99.5**	**99.5**	**3.80**
DBNDownBeatTracker (BO3) [15]	94.0	97.3	97.1	3.68
MultiFeature (MFT) [187]	86.4	86.4	86.8	3.55
Mean	76.1	82.2	82.2	3.09

Table 5.9 Ground truth performance of each beat tracking algorithm on the audio excerpts of SAMBASET. The best performance for each metric is highlighted in bold. The five-member committee proposed in [77] is indicated by an asterisk.



(a) HES collection **(b)** ESE collection **(c)** SDE collection

Figure 5.8 Collections sorted by MMA mean (solid line), with standard deviation (shaded region). Annotated samples (solid circles) were chosen as the closest to ten evenly spaced MMA values (solid triangles). One sample was treated as an outlier (cross) in (b).

For a committee of N beat trackers, they then calculate the $N(N-1)/2$ different mutual agreements and average them all to obtain the MMA. The researchers show how a low mean mutual agreement coincides with perceptual and musical properties that make tapping difficult for humans. They build a challenging dataset by selecting samples with MMA < 1 bit, given a committee of five beat trackers.

Later, in [188], they calculate the MaxMA, i.e., the algorithm whose output presents the maximum mutual agreement with the rest of the committee, showing that it provides the most reliable estimation for that given music example. They conduct subjective listening tests to determine a perceptual threshold for acceptable quality of this chosen output. This

threshold is found to be 1.5 bits, for the same committee of five beat trackers. Their results also show a correlation between the test ratings and the MMA.

In this work, we followed a similar approach to select samples of various difficulties to the state-of-the-art algorithms. As mentioned in 5.4.2, we collected implementations of 14 beat tracking systems, removing some (the unavailable ones) featured in the original work [77], and adding others that were presented after its publication. We then extracted 30-second excerpts from all the different *sambas de enredo* in SAMBASET, and computed the MMA between estimations yielded by the beat trackers for all these 493 files. Figure 5.8 presents the ordered MMA values for excerpts from the three collections (HES, ESE, and SDE).

For each collection, using the curve obtained from its excerpts, we determined P evenly spaced MMA values (including the maximum and minimum points), and selected the excerpts closest to each of these values to annotate. The reasons for this procedure are twofold: first, by selecting the same number of samples from each collection, we compensate for the large imbalance between them (e.g. in SDE there are nearly seven times more excerpts than in HES), while ensuring that their unique characteristics will be equally represented in the subset (recalling the variability in HES is much higher than that in ESE, or SDE); second, we guarantee that the algorithms will be compared within a group of samples to which they share different levels of consensus (and that would possibly provide a human annotator with a gamut of challenges). In total, thirty files were manually annotated ($P = 10$, i.e., ten from each collection), totalling just over 1 900 beats. It should be noted that a moderate number of annotated samples is sufficient, since we are dealing with a single music genre, which considerably limits the range of variations between them.

We see in Figure 5.8 that, in general, the fourteen beat tracking systems show more agreement in estimations for tracks in the ESE collection, followed by those in SDE, with HES in last. In fact, for over 50% of the tracks in ESE, the algorithms presented MMA > 3 bits, against slightly under 12% for SDE tracks and 0% in HES tracks in the same conditions. Considering an MMA > 2.5 bits, those percentages grow to 95%, 70% and 23%, respectively. This agrees with the authors' overall impression that the HES collection is the most "flavorful", whereas ESE is less expressive.

Regarding the test with the sampled subset, Table 5.9 gives the accuracy values for all algorithms, averaged over the thirty samples. Seven beat trackers perform better than the mean in all metrics, some of them outperforming the others by a large margin. In the end, the four best algorithms for our dataset are BO2, BO1, DAV, and BO3.

For the sake of comparison, we also evaluated the 493 excerpts with the five-member committee proposed in [77] and used in [188]: for 98.6% of the set the committee shows MMA > 1.5 bits; a single excerpt has MMA < 1 bit. This indicates that, overall, SAMBASET excerpts are not very challenging to the algorithms in this committee, which would provide a good number of acceptable estimations. Indeed, the good results shown by commit-

tee members in the ground truth performance suggests likewise. This analysis of state-of-the-art algorithms indicates a safe approach to semiautomatically annotating beats in this dataset.

Musicological insights

Here we investigate the evolution of average tempo in *samba de enredo* recordings across the years as represented in the SDE collection. For each excerpt, we compute the average tempo in beats per minute (bpm) as the inverse of the mean inter-beat interval, using the automatically detected beats. Figure 5.9 shows the average tempo for every track in SDE, plotted against the release year.

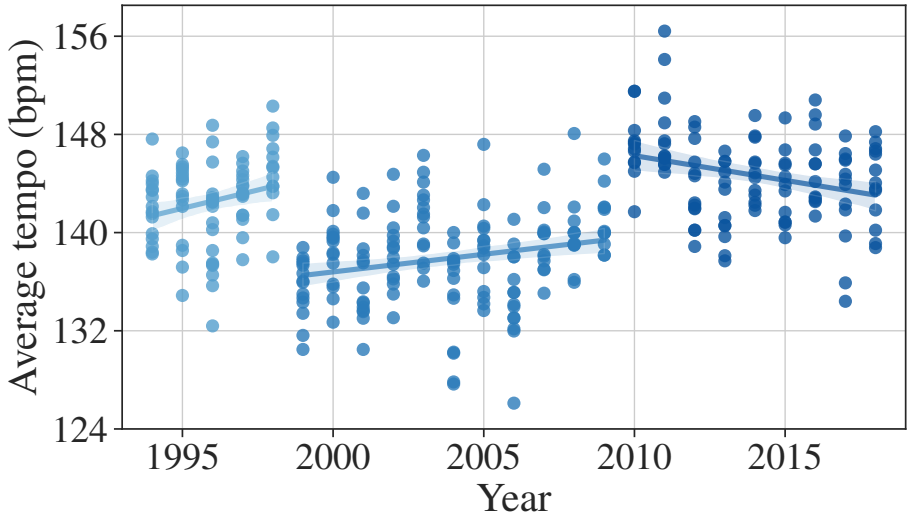


Figure 5.9 Variation of average tempo across the SDE collection with trend lines for three distinct regions and respective confidence intervals (shaded areas).

Although no clear trend is apparent from the whole data, we can readily verify the existence of local trends in three different regions of the graph. The first region accounts for the years of 1994 through 1998, and corresponds to the end of an era of "live" recordings in the *Teatro de Lona* (Barra da Tijuca), a large circus-like tent. As Moehn reveals on his essay "The Disc is not the Avenue" [120], by then the recordings were being made with a large number of musicians from each *escola* (around sixty) as well as large choirs from the respective community.

A radical change took place in the production of the 1999 disc: the entire process was moved to the studio and the number of *escola* members was reduced, not only to cut costs, but also to regain control over the sound organization [120]. Producers wanted the disc to sound "clear" and, thus, constrained the creative liberties of the *bateria*'s directors (e.g. they were not allowed to choose the tempo of the performance or to follow certain musical con-

ventions that are common in a live performance). This was an attempt to recover the disc's marketability (sales had been dropping in previous years), despite distancing it from the actual phenomenon of the *samba de enredo* [120]. In 2010, "live" recordings were resumed, this time in the *Cidade do Samba* (Gambôa). Producers retreated in their interference on the soundscape creation, and the *escolas* were able to reclaim the final saying in some aspects of the recording, such as the tempo. The larger space provided by the *Cidade do Samba* also lead to an increase in the number of musicians taking part in the recordings: more than 8000 for the 2014 CD against 1500 in the 1998 recording [120].

Therefore, the first and third regions of Figure 5.9 more closely represent actual *samba de enredo* performances. In particular, notice that the average bpm in the third region is above the averages in the other two regions. This can be seen as a direct translation to the digital media of the decisions to accelerate the live performances (and the marching pace), so that the *escolas* satisfy changes in parading time limits, as reported by many specialists [82, 142, 33].

Challenges

It would be very interesting to enrich this dataset with other types of annotations. In particular, one could think of generating ground truth for section boundaries (e.g. verses and the two different choruses that are very common in *samba de enredo* compositions), chord annotations (for instruments such as the *cavaquinho*), and instrument activity. As in the case of the CompMusic project [156], pitch contour annotations for soloist voices could be produced, as well as time-aligned lyrics and percussion transcriptions.

With these annotations, SAMBASET can provide many challenges to state-of-the-art algorithms in different MIR tasks. Tracks (especially in SDE) contain a plethora of simultaneous sounds of different qualities and textures, e.g. harmonic and percussive instruments, soloists and choirs. These could pose hard problems to vocal F0 or chord estimation systems. Also, singing voice annotation and lyrics could allow the study of soloist's interpretation as to phrasing, preferred ornaments, or characteristic syncopation; along with the metadata provided, it would be possible for example, to work on singer classification.

5.5 Conclusion

In this chapter, we presented BRID and SAMBASET, two datasets of Brazilian music for computational rhythm analysis developed during this dissertation in a collaboration with the STAREL project. The first dataset is a copyright-free dataset that provides interesting paths in the modelling of microtiming properties for *samba* and its sub-genres, beat and downbeat tracking, and rhythmic pattern analysis.[11]

[11] The dataset contents are available at http://www02.smt.ufrj.br/~starel/datasets/brid.html

The second one, is a large *samba de enredo* dataset with rich metadata, beat and downbeat annotations. We provided a detailed overview of its contents[12]and reported a study on the performance of a comprehensive group of beat trackers over the set. We also motivated one musicological use of the dataset, i.e., the study of changes in *samba de enredo*'s rhythmic properties across several years.

We expect that both BRID and SAMBASET allow for technical improvements in traditional MIR tasks via new perspectives on problem solving that arise from contemplating cultures different from those to which we are accustomed.

[12]Visit http://www.smt.ufrj.br/~starel/sambaset/ for more information.

Chapter 6

Common tools for MIR: *mirdata*

Summary

In this chapter we introduce an open source library called `mirdata`. This library was developed in collaboration and equal contribution with Rachel M. Bittner, and with the help of the MIR team at Spotify. This library aims to increase reproducibility in the use of datasets in their current distribution modes. In particular, it contains tools which: (1) validate if the user's data (e.g. audio, annotations) is consistent with a canonical version of the dataset; (2) load annotations in a consistent manner; (3) download or give instructions for obtaining data; and (4) make it easy to perform track metadata-specific analysis. In the following we briefly describe the library content and present experiments to motivate its need.

6.1 Motivation

Music Information Retrieval (MIR) systems are often software or algorithms which are evaluated and compared based on their performance according to appropriate metrics on chosen datasets. These systems are becoming increasingly complex; reproducing systems presented in academic publications requires access to the software and data [116]. As outlined in [116], some of the common elements of an MIR system are (1) Data (Audio and Annotations) (2) Codecs and Parsing (3) Modeling and (4) Evaluation. The reproducibility of each of these elements poses challenges, but efforts are being made to reduce potential inconsistencies.

For evaluation, different implementations of evaluation metrics can result in substantially different results, motivating the need for `mir_eval`—a common and transparent evaluation software [145]. For modeling, slightly different implementations of the same algorithm can result in very different results [116]. Recently, this has been mitigated by the availability of software with tools for popular MIR tasks. Some examples are `librosa` [118] and `essentia` [17]—tools for MIR related signal processing, simple models and commonly used algorithms; Scikit-Learn [138]—tools for training simple machine learning algorithms; and

`madmom` [12]—deep learning and machine learning models for common MIR tasks such as chord recognition, beat and downbeat tracking.

It is very difficult to get licenses to distribute music recordings openly. As a result, the majority of datasets available do not have freely available audio files. The exchange of this data is often done manually, which can result in varying data versions. When working with pairs of audio and annotation files, it is important that the audio files used are the same files that were used to create the annotations. When audio files are released separately from annotations, unknown differences between the original and other versions of the audio can create reproducibility issues. Websites such as Zenodo[1] and Figshare[2] provide permanent hosting and versioning of datasets, increasing reproducibility, but many datasets used in MIR are not available on such websites, and (often unknown) differences in data can adversely affect downstream performance.

Additionally, the annotations that come with each of these datasets exist in a huge variety of formats. Among these formats, some provide very complete information, e.g. in the form of a JAMS file [80, 115], while others lack crucial information needed to accurately use the data, such as the time stamps associated with different observations. Most of the time, researchers write their own code for parsing the specific annotation files they use for a particular dataset. This is both inefficient and error-prone; what was found for evaluation and modeling is also true for data parsing: small differences in annotation loading code can result in huge differences in results downstream. Finally, the pairing of audio and annotation files is often done manually each time, usually by matching on filename substrings. In addition to being cumbersome, this can also lead to mismatched audio and annotation files.

To summarize, obtaining datasets and writing code to process them is both time consuming and error-prone. As a result, researchers are less inclined to use multiple datasets for their task, and instead develop or test their models on single datasets, reducing the reliability of their results [173]. We believe that the two current biggest blockers of reproducibility in the MIR community are (1) the lack of open datasets, resulting in a lack of transparency as to the consistency of data across publications and (2) the lack of an open library for consistently loading annotations in various formats in common datasets.

In this chapter, we introduce an open source library, `mirdata`, which provides tools for using common MIR datasets. This library aims to be a useful tool for researchers, which will increase reproducibility, and facilitate and encourage the use of several datasets for evaluation. In particular, it contains tools for loading dataset-specific annotations in a consistent manner, validating if a dataset copy the user has is consistent with a canonical version of the dataset, downloading datasets, and linking audio and annotation files along with track level metadata. We demonstrate the need for this tool by highlighting inconsistencies in

[1] https://zenodo.org
[2] https://figshare.org

common practices when loading annotations and in the data itself (annotations and audio) for three popular MIR datasets, namely iKala, Salami, and the Beatles dataset.

The library is publicly available on Github at github.com/mir-dataset-loaders/mirdata.

6.2 Related Work

Data utility libraries exist for other fields such as text, video and image analysis [138, 1, 30, 136] which allow a user to download a dataset and load it into memory for ease of use in experimentation and consistent results. TensorFlow [1], a deep learning framework, includes a variety of datasets covering images, text, language translation and video[3]. In order to ensure the integrity of the data, TensorFlow hard codes expected file sizes and SHA256 checksums of each file in their library, as well as paths of the data included with the dataset. If the expected values do not match what is downloaded, the local dataset is not considered valid and is not available for use in the library.

Scikit-Learn [138], a popular machine learning library for Python, includes their own set of dataset loading utilities[4]. Some small datasets, known as "toy datasets", are included directly in the library. Larger datasets, known as "real-world datasets" are downloaded and stored in a "data home" directory on the local machine. Like TensorFlow, Scikit-Learn checks for dataset integrity based on a SHA256 checksum, but only checks the downloaded zip or tar file itself. This approach requires that users download the entirety of the Scikit-Learn library in order to use the dataset loaders.

MLDatasets[5] acts as a specification for how datasets should be managed. DataDeps [180] specifies dataset metadata that conforms to a management system. This method allows for multiple people to maintain their own dataset repositories or for a single organization to set up multiple, independent libraries, one for each dataset.

There are few examples of dataset utility libraries for music. However, the python library Nussl [106] for source separation contains some dataset utilities. Unlike the other libraries previously mentioned, `Nussl` does not include any utilities for dataset retrieval and expects the user to have the datasets locally on the machine before use. However, it includes expected checksums for the dataset audio and logic to check for validity and existence of the dataset as well as simple utilities for loading the data.

Software also exists for loading particular types of (music-centric) annotations, including `pretti_midi`[6] for MIDI data, JAMS [80] for data released in JAMS format, and Music21[7] for MIDI and MusicXML data. When annotations are released in these formats, custom load-

[3] www.tensorflow.org/datasets/datasets
[4] https://scikit-learn.org/stable/datasets
[5] https://github.com/JuliaML/MLDatasets.jl
[6] github.com/craffel/pretty_midi
[7] http://web.mit.edu/music21/doc/index.html

ing code is less necessary. However, many annotations are released in other formats and require custom loading code.

Audio files in MIR datasets

Datasets in MIR suffer from a unique constraint: most music is protected under copyright. Datasets which are built on copyrighted materials are not typically available for open download. There are several common levels of access for the audio files for different MIR datasets:

1. Open Access
2. Restricted Access (e.g. password protected)
3. "Do it yourself" Access (e.g. YouTube links)
4. No Access

We surveyed 128 MIR datasets from the "Audio Content Analysis" website [8] in April 2019 and determined their access levels. By our estimate, 80 were "open access", 19 were "restricted access", 15 were "DIY" Access, 14 were "no access", meaning that 22.8% of the total list is not openly available. These limited access datasets include historically popular datasets such as RWC [62], AudioSet [59], CAL10k [172], the Beatles dataset [71], iKala [24], the Million Song Dataset [7], and Salami [163].

The more restrictive the access level, the more room there is for "dataset telephone"; when it is difficult to access a particular dataset from a common repository, researchers may share their personal copies with each other, which may contain perturbations from when they first received it. Additionally, since the audio and annotations are sometimes released separately, if the audio is incorrect, the annotations will not correspond to the audio files, resulting in inconsistencies during model development and evaluation. As a result, researchers are performing experiments and computing metrics on datasets they believe are the same as other's versions, but may be quite different in reality.

In an ideal case, the audio files used for a dataset should be the same as those used to create the annotation files. There are a number of popular datasets for which the audio is difficult to obtain. For example, the 7-Digital preview clips of the Million Song Dataset [7] have often been used for music classification tasks. While the clips were previously available through an API, it has since been shut down and the clips are no longer available. In the Beatles dataset [71], audio is not released, but instead, catalog numbers and release years of the albums used are provided to prevent differences in audio versions used. Regardless, we found that different versions of the dataset have been used by researchers (see Section 6.3). In the case of AudioSet [59], audio is provided in the form of YouTube

[8] https://www.audiocontentanalysis.org/data-sets/

video identifiers, adding a new challenge in data reproducibility. However, the availability of the linked YouTube videos changes over time, and accessibility varies by country.

6.3 Experiments - Why a Common Tool is Needed

Differences in audio or annotations, or in the code used to load data into memory can have a huge impact on downstream results. In this section, we examine the effect of real differences we found on evaluation metrics in instances of three popular MIR datasets. The methods and metrics used for this experiments, e.g. for melody extraction, will not be further explained in this document, since a detailed explanation is out of the scope of this thesis.

The Beatles Dataset

The Beatles dataset [71] contains annotations for beats, downbeats, sections, and chords for the nearly entire Beatles' collection. However, as the audio is copyrighted, only the annotations are released as part of the dataset. The researchers are asked to use their personal copy of the Beatles' catalog and match the audio files with the annotations.

The annotations were created using a particular version of the audio, and they may not correspond well with other versions. For this dataset in particular, it is quite easy to end up with different versions of the same Beatles track, since there are several releases of every album, including remastered versions.

To evaluate these potential differences, we first compared checksums across four different researcher's copies of the audio files corresponding to the Beatles dataset. Out of the four versions, three had identical checksums, while one had invalid checksums on every single audio file, indicating that the audio is different between the two versions. Upon further examination of the differences, we found inconsistencies in the number of channels, the duration, and the average RMS of the audio files between the two versions.

The differences go even further than channels, duration and volume. In Figure 6.1, we first normalize a pair of audio files to have the same peak level and compute the absolute difference in their spectrograms. In the low frequencies, in particular, there are major differences between the frequency content of the two versions, despite sounding similar.

Next, we ran a chord recognition algorithm [114] on the two different versions of the audio collection, and computed the standard chord recognition metrics as implemented in `mir_eval` using the dataset's (public) reference chord annotations. The differences in the metrics are shown in Figure 6.2. While only one of these metrics had statistically significantly different results ("overseg", according to a paired t-test), we see that the same chord recognition algorithm produces results which are different enough to affect the metrics.

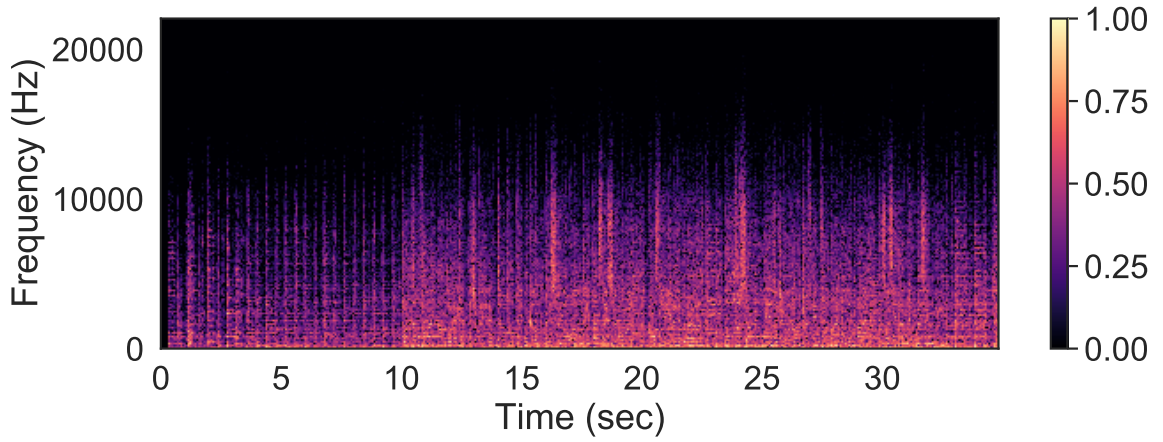


Figure 6.1 Normalized absolute difference between two spectrograms of the first 30 seconds of "Across the Universe" computed on audio files from two versions of the Beatles dataset audio.

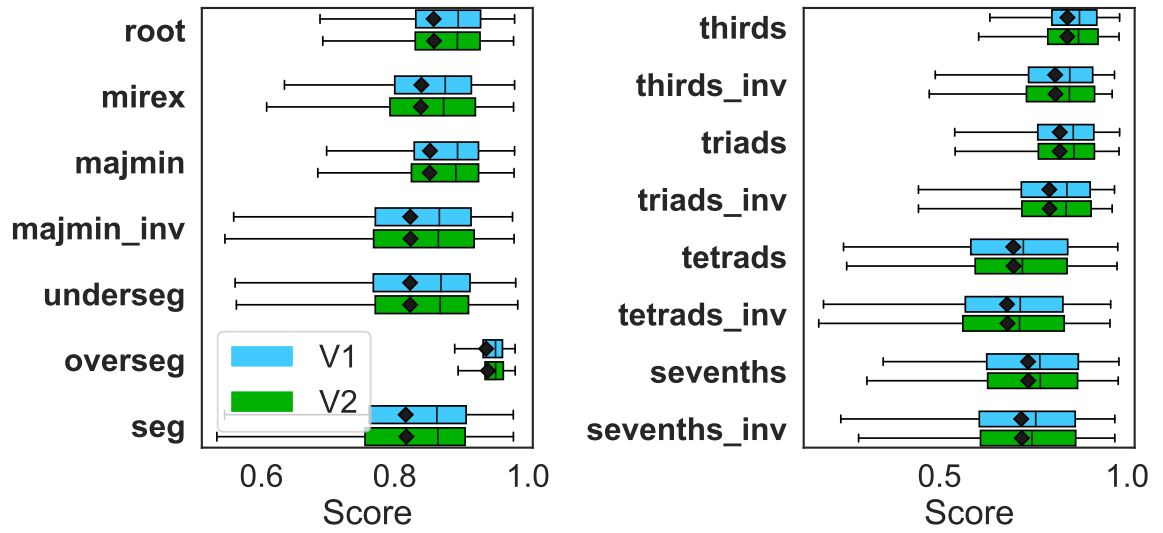


Figure 6.2 Chord metrics for chord estimates computed on two different versions of the Beatles audio, compared with the Beatles dataset reference chord annotations.

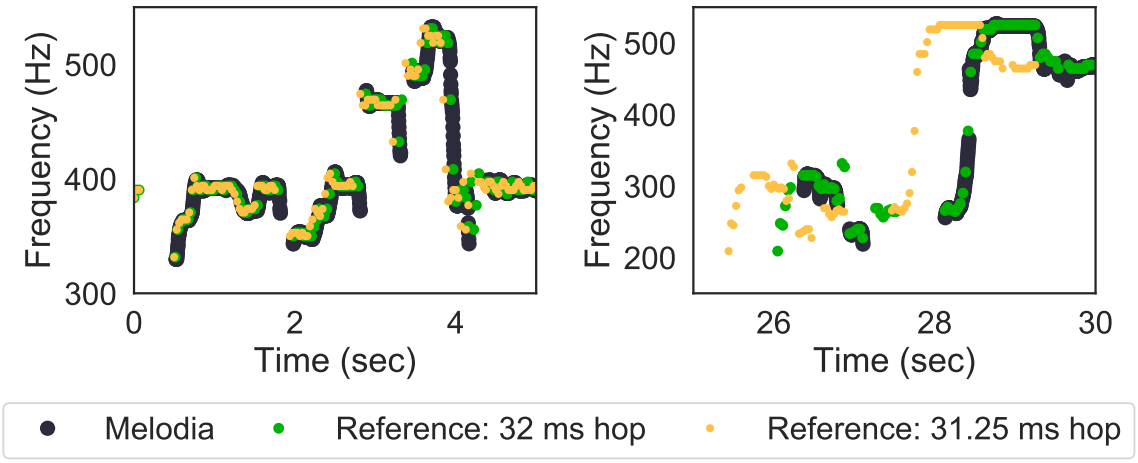


Figure 6.3 iKala reference annotations loaded using two different hop sizes (32 ms and 31.25 ms) versus the output of Melodia. (Left) the first 5 seconds of the track. (Right) the last 5 seconds of the track.

The iKala Dataset

We performed the same checksum experiment as for the Beatles dataset and compared checksums for four different researcher's copies of the iKala dataset. We found that all four versions (audio and annotations) were identical.

One challenge with the iKala dataset is that the vocal f_0 annotations are provided as newline-separated files with the pitch, but without timestamps, which must be inferred upon load. On the dataset's website, they state that the hop size is 0.032 seconds, but it does not state the alignment of the first time frame (left aligned or center aligned). The dataset's website also provides code for loading the annotation files, which uses a different hop size of 0.03125 seconds and center aligned frames (with the first time stamp at 0.5 * hop seconds).

To see how users infer the iKala time stamps, we performed a search of public code on Github.com for code which loads the iKala pitch annotations. We found 5 unique ways of loading the time stamps, consisting of 3 different hop sizes (0.032 s and 0.03125 as listed on the dataset website, and 0.032017, inferred from the duration of the audio files), and two different alignments (left and center). By far, the most common combination was using a hop size of 0.032 s and left aligned frames.

The differences in hop size have a major effect on the alignment of the audio and the annotation, especially over time. Figure 6.3 shows an example of the annotations loaded with two of the hop sizes, and the estimate of a melody extraction algorithm (Melodia [151]) for comparison. In the first 5 seconds, the differences are small, but in the last 5 seconds, we see a visible misalignment between the loaded annotations and the audio.

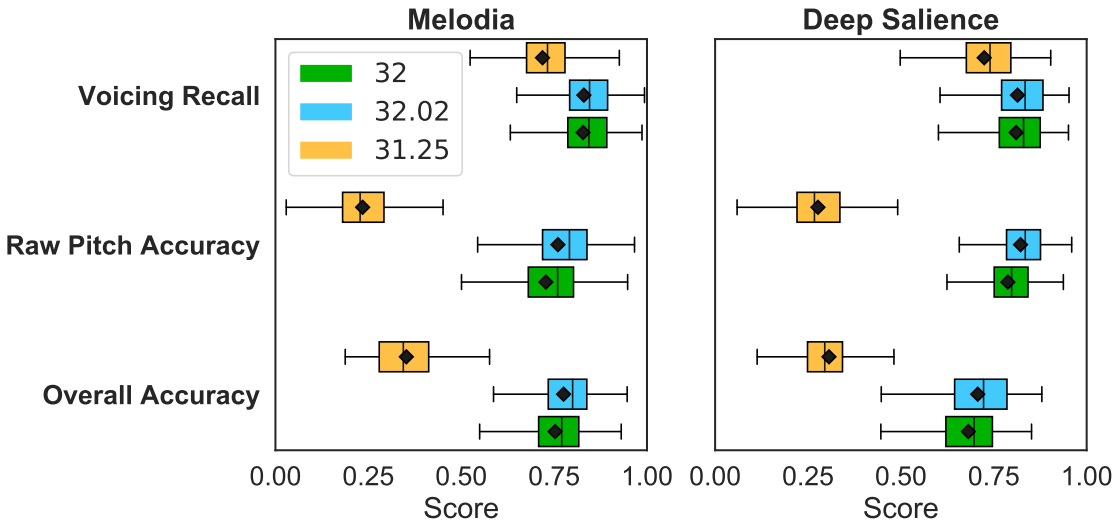


Figure 6.4 Melodia and Deep Salience melody metrics when evaluated against iKala's reference data loaded with 3 different hop sizes.

To investigate the severity of these differences, we ran two melody extraction algorithms, Melodia [151] and Deep Salience [9], on the iKala audio. We then compute melody evaluation metrics using `mir_eval` with reference times computed using the three different hop sizes, $h = 32$ ms, $h = 32.017$ ms and $h = 31.25$ ms, using *left* aligned frames. In Figure 6.4, we show boxplots of the results across tracks in the dataset for each of these reference hop sizes. The results for different hop sizes are quite different, and drastically so for the smallest hop size, 0.03125 s. Even the difference between $h = 32$ and $h = 32.017$ in Overall Accuracy is substantial for both datasets - a difference that is historically enough to claim state of the art over another algorithm. A paired T-test shows statistically significant differences for all pairs of hop sizes for each metric, with the exception of Voicing Recall for $h = 32$ and $h = 32.017$

Next, we compute the melody metrics for the same melody extraction algorithms using a hop size of $h = 32$ ms and compare left vs center aligned frames in the reference. Figure 6.5 shows the results per track of the two different reference alignments. The difference in metrics is smaller than for the hop size differences, but left alignment is statistically significantly worse than right alignment for Overall Accuracy and Raw Pitch Accuracy under a paired T-test.

This begs the question: which is the correct way to load the timestamps? Since it is quite unlikely that an incorrectly time aligned reference would produce higher scores than a correct alignment, it is likely that, despite the norm of using left-aligned frames, the annotations are intended to have center aligned timestamps. Indeed, if we look at a specific example of left vs. center aligned timestamps for a short excerpt compared with two different algorithm

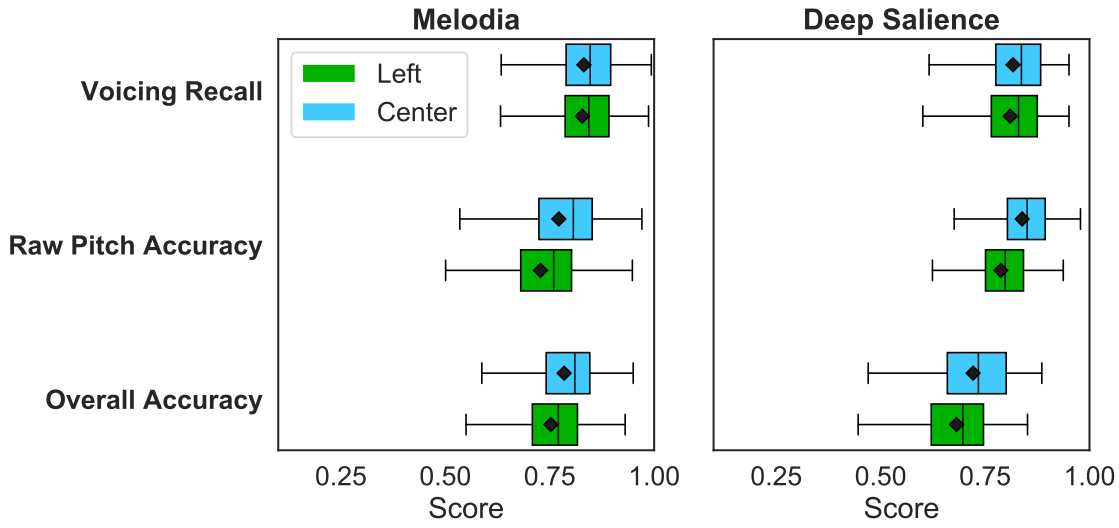


Figure 6.5 Melodia and Deep Salience melody metrics when evaluated against iKala's reference data loaded with left and center-aligned time stamps.

estimates, as in Figure 6.6, we see that the reference is better aligned with both estimates when using center aligned time frames. Note that in Figure 6.4, the reference hop of 32.02 ms resulted in higher metrics than the hop of 32 ms (both with left aligned frames), and a 32 ms hop with center aligned frames has higher metrics than all of the left-aligned hops. The data loaded with a hop size of 32.02 ms starts off misaligned but over time, approaches a center alignment, explaining the "better" score with this incorrect hop size.

The shocking result of this set of experiments is that every single example we found publicly – including the code found on the dataset's website – appears to be loading the f_0 data incorrectly, either with an incorrect hop size or an incorrect alignment - no example we found had both a hop size of 32 ms and center alignment.

Salami Dataset

The complete set of annotations for the Salami dataset was released in version 2.0 of the dataset, increasing the volume of the dataset with respect to previous versions. For instance, from version 1.9 to 2.0 additional annotations related to 539 tracks were added, 390 with multiple annotations and 189 with single annotations. Both versions are available in the dataset's main repository. However, as in other cases when a dataset is updated, there is no centralized version control that is transparent and ensures the awareness of the community to these changes.

Data-driven models are increasingly popular for addressing MIR tasks, including the task of boundary detection using the Salami dataset [175, 68]. One of the main reproducibility-

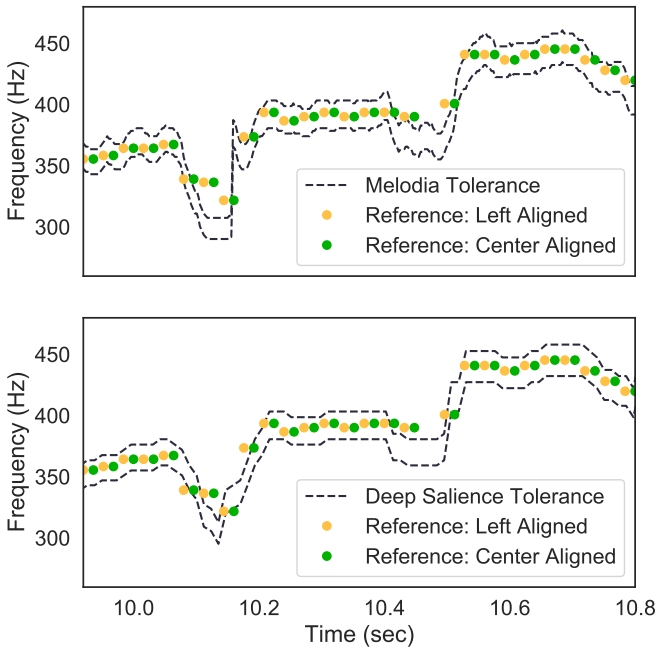


Figure 6.6 Left and center aligned time stamps for a track in iKala versus algorithm estimates from Melodia and Deep Salience. The dashed lines show the distance from the estimate where the algorithm would be considered correct in the standard melody extraction metrics.

related issues about data-driven models is their training, in particular, the amount of annotated data is crucial. When using Salami, it is important to avoid using an old version of the annotations, which could have a negative impact in a model's performance in comparison with the same model trained on the newest version of the dataset. In particular, if different data is used for training two different models with the aim of comparing their performance afterward, it is difficult to isolate possible causes of performance differences. An example of this situation is shown in [31], where the authors use a different subset of Salami than previous works, obtaining substantially different results when intending to re-implement other authors' model (e.g. 0.246 instead of the previously reported 0.523 F-measure with a ± 0.5 s tolerance window).

Another possible source of inconsistency with the use of this dataset relates to contributions from people other than the dataset creators. The authors in [117] manually edited 171 of the annotations in version 2.0, to correct formatting errors and enforce consistency with the annotation guide proposed by the dataset creators. However, this "corrected" version of the annotations was not included in the dataset's main repository. This third version of annotations is used in recent works [117, 174]; comparison of systems without awareness of the difference with the version 2.0 released annotations may lead to differences in performance that are beyond models' design.

MIR Dataset Loaders

In this section, we describe the `mirdata` python library, a potential solution to the current reproducibility issues with dataset versions and loaders. A driving philosophy of this library is to work with the imperfect situation we are faced with, with regards to the limited openness of MIR data. In an ideal scenario, all data would be freely sharable and version controlled; since this is not the case, we do our best to create tools to maximize reproducibility given the current constraints. Most importantly, we aimed to create a clean, transparent and easy to use interface to encourage reuse and contributions. The first release of the library will include loaders for Orchset [18], iKala [24], MedleyDB Melody and Pitch subsets [10], the Beatles dataset [71], Salami [163], the Million Song Dataset [7], Medley Solos-DB [104], RWC [62, 63, 28, 110], DALI [119], and GuitarSet [183].

Dataset Indexes and Checksums

Datasets, by their definition, are collections of data. In the case of MIR datasets, the data is often a collection of separate files, some of which correspond to e.g. a particular audio file. For example, the Beatles dataset contains four separate text files containing chord, beat, section, and key annotations for each audio file. Since each of these four annotation files are related to the same audio file, it is desirable to have a common way of linking them. In `mirdata`, we use a *dataset index* to link related files, in the form of a JSON file. This index contains a unique identifier for each group (for example, the name of the audio file), which is mapped to its corresponding file paths and their expected *checksums*, for example:

The use of an index for each dataset is advantageous for a number of reasons. First, it groups related data files in a transparent way, avoiding audio-annotation pairing mistakes, and removing the need for custom filename sub-string matching per dataset. Second, it gives a version-controlled record of all expected files in the dataset, preventing inconsistencies due to missing or extra files; we can check if all the expected files are present in a local copy of a dataset, and we load data to memory based on the index, ignoring files not included in it. Finally, it provides a way to verify if a local copy of a dataset is consistent with a canonical version on a file-by-file basis.

For each dataset, we also provide a `validate()` function, which checks for the existence of files locally and compares the expected checksums with checksums of the local copy. A checksum is a representation of a digital file similar to a fingerprint and usually computed by taking a hash of the bits in a file. The smaller (in size) representation created allows for efficient file comparisons at the bit level. If two files have the same checksum then a user can assume that the two files are exactly the same with high confidence. If the checksums between two files differ or the computed checksum is different from an expected checksum, then the user is at a minimum aware of discrepancies and can take appropriate action. In the `mirdata` use case, checksums allow users who have a local copy of a dataset to know

whether or not they are using the same data as others, simply by running the validation function.

Note that there is not always one "correct" version of a dataset, so it is difficult to decide which version of a dataset should be used to create the reference checksums. For this library, we compute checksums on the version that is as close to the "original" as possible, for example by obtaining a version from dataset creators.

Dataset Downloading

It can often be difficult or unclear how to get access to a particular dataset, and same data often exists in multiple places and may not be identical. In `mirdata`, we provide a `download()` function for each dataset. When data is openly available online, the function automatically downloads the data, and this version of the data is same the version used to create the checksums. When the data has been downloaded, we run validation to ensure it matches and warn the user if not (e.g. in the case of an incomplete download). When data is not openly available online, we provide instructions for how to obtain the data (e.g. by requesting access on a particular website). Once the data has been obtained, the user can then run validation to ensure the data is consistent.

Annotation Loaders

As highlighted previously, differences in implementations for loading annotation data to memory can have big effects on the resulting data. In `mirdata`, we remove the need for users to manually write loaders per dataset and annotation type by providing functions for loading all annotations for each dataset. These implementations are shared and transparent, allowing users to permanently correct mistakes in the way data is loaded.

As an example, for some of the beat annotations in the Beatles dataset, the beat position is missing for only a few observations. These missing positions can be inferred from the neighboring information (e.g. beats 1 and 3 have labels, and the one in between is absent), and in `mirdata`'s implementation we fill in this information on load.

Track Metadata

More often than not, in addition to data files containing e.g. time varying annotations, datasets provide track-level metadata. When loading annotations, we also link any available track level metadata with each track id group. This can be particularly useful when splitting data, such as for creating unbiased train-validation-test splits [116], or for analyzing evaluation metrics over different splits of a dataset.

6.4 Conclusions

Although data distribution challenges remain, we believe that the use of `mirdata` will result in reproducible usage of datasets and research moving forward. Future iterations of `mirdata` will include support for large (out of memory) datasets and an increased number of supported datasets. As datasets become more open and annotation formats standardize, the scientific need for this library will lessen, but it will remain a useful tool for ease of working with datasets.

Importantly, we designed `mirdata` to have a low barrier to entry for contributions. New datasets can be easily included independently with minimal interfacing with the rest of the library. With active community participation, we believe that `mirdata` can help ensure that MIR datasets are used in a consistent, reproducible manner.

Part III: Proposed approaches

Chapter 7

Analysis of Deep Learning Systems for Downbeat Tracking

Summary

In this chapter we offer a systematic investigation of the impact of largely adopted variants in deep learning-based downbeat tracking systems. We also investigate the potential of convolutional-recurrent networks for this task, which have not been explored for downbeat tracking before. Our findings are that temporal granularity has a significant impact on performance, post-processing helps in all cases, and the proposed convolutional-recurrent architecture performs competitively with the state-of-the-art, being able to improve the reference system's performance in some cases.

7.1 Motivation

The task of downbeat tracking has received considerable attention in recent years. In particular, the introduction of deep neural networks provided a major improvement in the accuracy of downbeat tracking systems [50, 13, 93], and the systems relying on deep learning have become the state-of-the-art. Among those methods, different design choices were taken at different stages of the processing pipeline, such as the temporal granularity of the input, low-level feature representations, network architecture, and the post-processing methods. Additionally, different proposals were evaluated using distinct training data and/or evaluation schemes (e.g., cross-validation vs leave-one-dataset-out) [52, 93, 15]. This variability makes it difficult to gain insights about the actual role of each design choice, and the delicate interactions between them.

Within this context, and in order to build upon state-of-the-art systems, we consider as a good strategy taking a close look at which design aspects of these systems have greater impact in their performance. Besides, we consider that re-implementing some of those sys-

tems and adopting some of their choices in an exploratory framework, would allow us to make fair comparisons between newly proposed systems and existing ones. Once a systematic comparison is addressed, we have good conditions to ask ourselves other questions: how do the current systems perform in terms of music consistency? Where are they failing and how could we address that?

In this chapter, we present a systematic investigation of the impact of different design choices in downbeat tracking models. In particular, we study the effect of the temporal granularity of the input representation (i.e., beat-level vs tatum-level), the output encoding (i.e., the label encoding used to train the networks), and their interactions with the post-processing stage and the internal network architecture. Besides, we propose a system that exploits convolutional-recurrent neural networks (CRNNs) for this task, within which we also explore these variations. This systematic analysis allows us to gain fresh understanding into some of the state-of-the-art approaches, showing that the proposed system behaves as the state of the art, and taking a step towards the systematic design of these systems.

7.2 Related work

As exposed in Chapter 2, downbeat tracking systems based on deep learning usually consist of three main stages, as illustrated in Figure 7.1. In the first stage, a low-level feature computation is commonly exploited, where several representations such as chroma [50] or spectral flux [93] have been adopted in previous works. This is usually followed by a stage of feature learning with neural networks, whose outcome is an activation function that indicates the most likely candidates for downbeats among the input audio observations. Then, a post-processing stage is often used, which consists of a dynamic model, typically a DBN, HMM or CRF [15, 54, 52].

Durand et al. [52] proposed a system for downbeat tracking that consists of an ensemble of four models each representing a different aspect of music: rhythm, harmony, bass content and melody. The authors developed a CNN for each musically inspired representation, and estimated the downbeat likelihood by averaging the likelihoods produced by each CNN in the ensemble. Then, the authors turn the soft state assignments of the CNN ensemble into hard assignments (*downbeat* vs *no-downbeat*) using an HMM. This approach showed the potential of CNNs for downbeat tracking and the complementarity of the different musically inspired features.

In parallel, Böck et al. [15], presented a system that jointly tracks beats and downbeats using Bi-LSTMs. The authors used three different magnitude spectrograms and their first order differences as input representations, in order to help the networks capture features with sufficient resolution in both time and frequency. The input representations were fed into a cascade of three fully connected Bi-LSTMs, obtaining activation functions for beat

and downbeat as output. Subsequently, a highly constrained DBN was used for inferring the metrical structure.

In turn, Krebs et al. [93] proposed a downbeat tracking system that uses two beat-synchronous features to represent the percussive and harmonic content of the audio signal. Those feature representations, based on spectral flux and chroma, are then fed into two independent Bi-GRUs [27]. Finally, the downbeat likelihood is obtained by merging the likelihoods produced by each Bi-GRU. The final inference for downbeat candidates relies on a constrained DBN. For further details on DBN, CNNs, LSTMs and GRUs please refer to Chapter 2.

More recently, combinations of CNNs and Recurrent Neural Networks (RNNs) such as GRUs or LSTMs have received increasing attention. For instance, Convolutional-Recurrent Neural Network architectures (CRNNs) have been proposed in other MIR tasks such as chord recognition [114] or drum transcription [176], and they are the state of the art in other audio processing domains such as sound event detection [23, 2].

7.3 Analysis of common variations

Given the large amount of possible variations in deep learning based systems, we focused our analysis on some of the design choices that were largely adopted and not formally compared in previous works, isolating other factors that might influence the methods' performance. We analyse their interaction in different datasets, intending to help identifying what is driving the success of such systems. The different design choices under study are indicated in red dashed boxes in Figure 7.1. In the following, we explain our methodology, the different variations we chose and the motivation behind those choices.

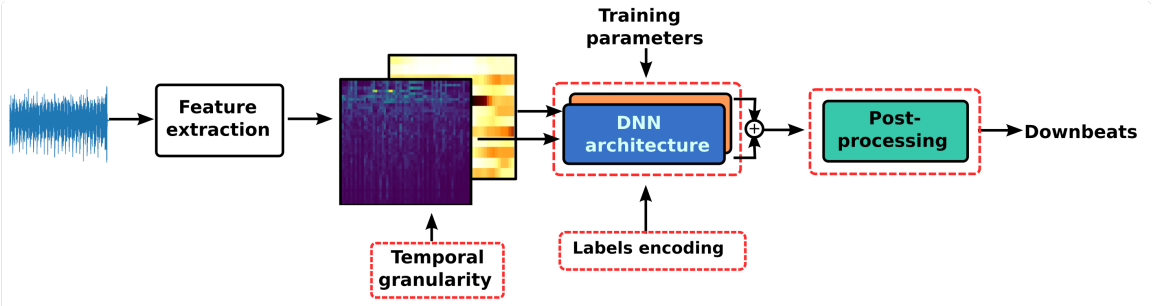


Figure 7.1 Different design choices under study in this chapter indicated by red dashed boxes: temporal granularity, DNN architecture, labels encoding and post-processing.

7.3.1 Input representations

Among the different input representations used in the literature, two out of three of the most recent state-of-the-art downbeat tracking systems based on deep learning use musically

inspired represention, showing their suitability for deep learning systems for this task. In particular, Krebs et al. [93] and Durand et al. [52] both exploit features that enhance the harmonic and the percussive content of the signal [93, 52]. We choose to work with the two representations that are used in the system presented by Krebs et al. [93], which are also similar to the representations used by the two best performing networks of the ensemble in [52]. We use the same input features for all the architectures under study, in all their variations.

The set of features describing percussive content, which we will refer to as PCF (*Percussive Content Feature*), is based on a multi-band spectral flux, computed using the short time Fourier transform with a Hann window, using a hop-size of 10ms and a window length of 2048 samples, with a sampling rate of 44100 Hz. The obtained spectrogram is filtered with a logarithmic filter bank with 6 bands per octave, covering the range from 30 to 17 000 Hz. The harmonic content's representation is the CLP (*Chroma-Log-Pitch*) [122] with a frame rate of 100 frames per second. The temporal resolution of the features is 4 subdivisions of the beat for the PCF, and 2 subdivisions for the CLP features. For computational efficiency, the authors in [93] assembled in matrices column-wise this resolution increment so the CLP feature set is of dimension 12×2 and the PCF is 45×4, which we maintained in this work. The representations are synchronized to the beat or tatum grid depending on the variation under study.

7.3.2 Temporal granularity: beats vs. tatums

The temporal granularity of the input observations (or temporal grid) relates to important aspects of the design of downbeat tracking systems. It determines the length of the context taken into account around musical events, which controls design decisions in the network architecture, such as filter sizes in a CNN, or the length of training sequences in an RNN.

Among the different downbeat systems, several granularities have been used. In particular, the latest state-of-the-art systems use either musically motivated temporal grids (such as tatums or beats) or fixed length frames. Systems that use beat- or tatum-synchronous input depend on reliable beat/tatum estimation upstream, so they are inherently more complex, and prone to error propagation [52, 93]. On the other hand, frame-based systems are not subject to these problems, but the input dimensionality is much higher due to the increased observation rate [15], which causes difficulties when training the models.

In this analysis, we focus on musically motivated temporal analysis grids, because they reduce the computational complexity of the systems considerably. We study the variations in performance using beat and tatum grids.

The beats for the beat-synchronous feature mapping are obtained using the beat tracker presented in [13], with the DBN introduced in [94].[1] We compute the tatums by interpolating

[1] In particular we used the DBNBeatTracker algorithm of the madmom package version 0.16 [12].

the beats, with a resolution of 4 tatums per beat interval.[2] We compare the interaction of the choice of temporal grid with those of the output encoding, the RNN or CRNN architectures and the post-processing stage. We adapt the sequence length used for training the networks in order to consider the same musical context in all cases.

7.3.3 DNN architecture: recurrent vs convolutional-recurrent

Recurrent neural network

To perform our analysis, we implemented with minor modifications the state-of-the-art downbeat tracking system presented by Krebs et al. [93], which we use as a baseline. This system consists of an ensemble of two neural networks, each one trained on a different input representation (CLP and PCF, see Section 7.3.1). In each network, the input features $X(t)$ at time t are mapped to a hidden state vector $h(t)$ by two consecutive Bi-GRU layers of 25 units each, and $h(t)$ is then mapped to a state prediction $p(t)$ by a dense layer, using a sigmoid activation. In our experiments, we observed that including BN layers consistently improves performance, so we included two BN layers, one after the input layer, and the other between the Bi-GRUs. Figure 7.2 summarizes the baseline architecture that processes the percussive input representation. A dropout of 0.1 and 0.3 is used during training to avoid over-fitting in the harmonic and percussive networks respectively, before the first Bi-GRU layer. The two networks are trained separately using the different input features and the obtained likelihoods are averaged.

The optimization of the model parameters is carried out by minimizing the binary-cross-entropy between the estimated and reference values, more details on this can be found in Section 7.4.

Convolutional-recurrent neural network

The architecture that we propose can be seen as an encoder-decoder system [26], where the encoder maps the input to an intermediate time series representation that is then mapped to the output space by the decoder. An interesting advantage of this kind of scheme is that several combinations of encoder-decoder can be explored, for instance by reusing the encoding but changing the decoder to specialize to a slightly different applications (e.g. beat and downbeat tracking).

The encoder architecture, which consists of a CRNN, is depicted in Figure 7.3. Each temporal unit (either beat or tatum) is fed into the CNN considering a fixed-length context window C of approximately one bar. The CNN processes each window $F \times C$ independently (F being the spectral/chroma dimension of the feature), and outputs a sequence of size

[2] This estimation is on the 16th note level, which we assumed as a good compromise to perform downbeat tracking.

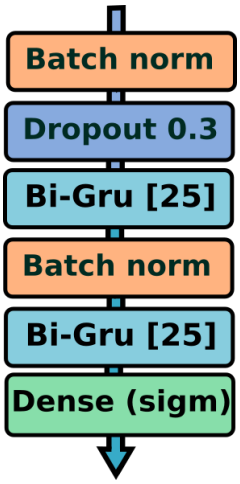


Figure 7.2 Arquitecture of baseline in [93], in the case of the percussive input feature and with added BN layers.

$T \times N$ (with T the length of the input sequence and N the output dimension of the CNN) that is fed to the Bi-GRU.

We base our CRNN architecture design on previous state-of-the-art choices. Particularly, our CNN design is based on the best CNN of the ensemble in [52], which we modify and combine with a single Bi-GRU layer, as explained below. The bi-directional version of GRUs integrates the information across both temporal directions, providing temporal smoothing to the local estimation of the CNN. CNNs are capable of extracting high level features that are invariant to both spectral and temporal dimensions, whereas RNNs model longer term dependencies accurately.

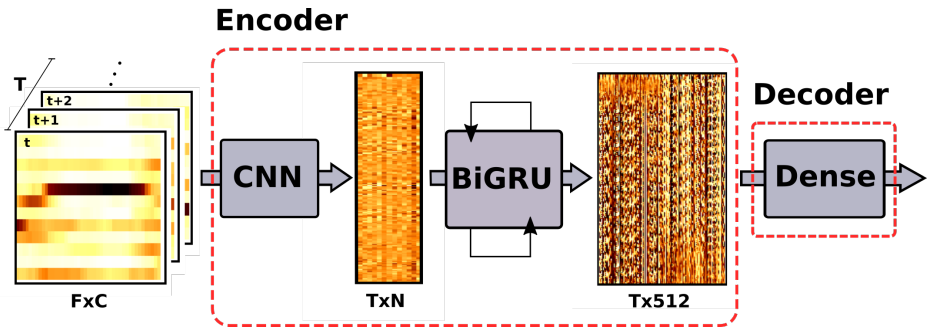


Figure 7.3 Encoder architecture: the input representation is either a beat/tatum-synchronous chromagram or multi-band spectral flux, of chroma/spectral dimension F. Each time unit is fed into the encoder with a context window C. The CNN outputs a sequence of dimension $T \times N$ which is fed to the Bi-GRU, T being the length of the input sequence and N the output dimension of the CNN). Finally, the encoder output dimension is $T \times 512$, that goes to the decoder and is mapped to a downbeat likelihood.

The CNN architecture consists of a cascade of convolutional and max-pooling layers, with dropout used during training to avoid over-fitting, to a total of eight layers. The motiva-

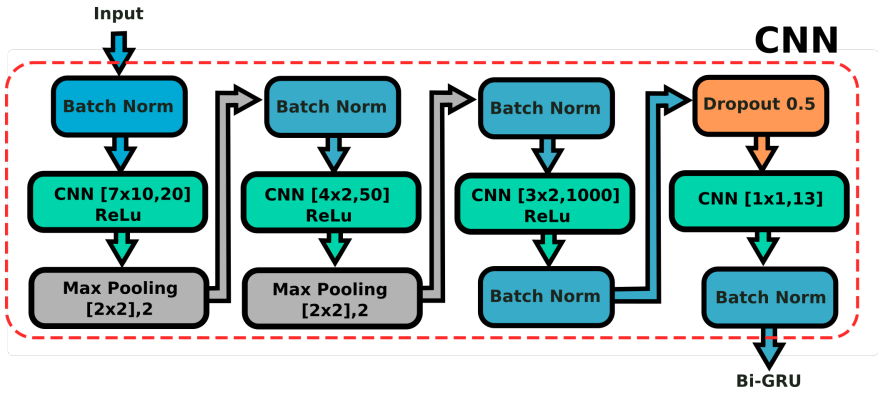


Figure 7.4 Summary of the CNN architecture.

tion behind the design of the convolutional layers' filters size is to capture local changes in harmony and spectral energy, so they are just a few tatums/beats length in time. Besides, the number of filters increases with layer depth (the second CNN layer has more filters than the first one and so on), enabling the network to learn complex combinations of the representations of previous layers then capturing complex structures. Max pooling layers help the network to be invariant to transposition, which is desirable in this context since transposition does not change downbeat perception. We add batch normalization layers to avoid too large or small values within the network that could slow the training of the encoder. Figure 7.4 represents the CNN's 2D filter sizes and the number of units, which is $[m \times n, u]$, with m and n operating on the spectral and temporal dimensions respectively, and u the number of units. The activation (if used) is indicated before the CNN description. Max pooling layers are notated as $[m' \times n'], s$ indicating frequency and time dimension and stride. The last layer of the CNN differs from the reference implementation in the number of units, which we set to 13 instead of 2 to fit features of bigger dimension to the Bi-GRU. We also remove the softmax activation of the last layer because the class discrimination is not carried out by the CNN.

The Bi-GRU consists of two independent GRUs, one running in each temporal direction, as explained in Section 2.2.3. Their hidden state vectors are concatenated to produce the bi-directional hidden state vector. We set the dimensionality of each GRU to 256, resulting in a total of 512.

Our decoder is a fully connected dense layer that maps each hidden state vector to the prediction state using a sigmoid activation, resulting in a downbeat likelihood at each time unit. The optimization of the model parameters is carried out by minimizing the binary-cross-entropy among the estimated and actual values.

7.3.4 Labels encoding: unstructured vs. structured

Among the downbeat tracking systems mentioned in Section 7.2, the common choice is to use an one-hot vector encoding to indicate the presence or absence of a downbeat at a particular position of the excerpt at training time. For instance, if using temporal analysis grid that is aligned on beats, a sequence of beats is usually encoded as $s = [1,0,0,0,1,0,0,0]$, indicating the presence of a downbeat at the first and 5th beat positions. We refer to this as *unstructured* encoding. Here, we also investigate whether incorporating a densely *structured encoding* with information about the metrical structure may help the neural networks perform a better downbeat tracking.

Structured encoding definition

We define the structured encoding as a set of classes that are active within the entire inter-beat interval. This is the set $\mathcal{C} = \{1,\dots,13\}$, where each class indicates the position of the beat inside a bar. We consider a maximum bar length of 12 beats, and an extra class *X* for labeling an observation in the absence of beats and downbeats, for a total of 13 classes (**$K = 13$**). For instance, to label a musical piece with time signature $4/4$, we use the subset of labels $\{1,2,3,4\}$, and we label consecutive time units corresponding to the same beat interval with the same class. Figure 7.5 illustrates the difference between the proposed and the unstructured encoding. In this structured class lexicon, the downbeats are represented by the label 1.

Figure 7.5 Audio excerpt in $4/4$ labeled with the *structured* encoding (top figure) and the *unstructured* encoding (bottom figure). The temporal granularity showed is tatum (quarter-beats). In the structured encoding each tatum receives a label corresponding to part of its metrical position.

We train the networks incorporating both the unstructured and the structured encoding. To incorporate the structure encoding we modified the decoder explained in Section 7.3.3 as shown in Figure 7.6. We use one dense layer to decode each class lexicon, and we evaluate the performance of the system using the unstructured output. The dense layers are connected so the information of the beginning of the bar is provided by the unstructured dense to the structured one as an extra input. We observed that this improves the downbeat

estimation, since it helps back-propagating information about the onset on the downbeat positions.

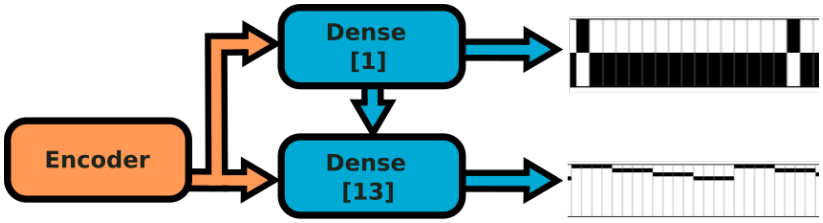


Figure 7.6 Decoder architecture with the structured encoding.

It is important to note that the unstructured coding has a clear interaction with the temporal grid in terms of the number of 1- vs 0- symbols in the training data, while the structured coding is consistent under any temporal granularity (i.e., the amount of class instances remains proportional). When changing to a finer granularity, i.e. from beats to tatum, the unbalance between the 0 and 1 classes becomes bigger, and the occurrences of the 1 class —which represents the downbeat— more sparse. In the structure encoding, the number of classes depends on the time signature and not on the granularity. In a 4/4 time signature, beats will have labels 1 to 4, and since the amount of tatums inside a beat are estimated so they are constant among beats, in the tatum grid there are more occurrences of all classes and the proportion remains the same.

7.3.5 Post-processing: threshold vs. DBN

The importance of the post-processing stage has been addressed in previous works [52, 93]. In this analysis, we assess the relative importance of this stage depending on the temporal granularity, the network architecture and the dataset. To that end, we compare the performance obtained using a simple threshold and the DBN presented in [93], which implements the BPM (see Section 2.3.3). This DBN models beats (or tatums) as states, where each state can assume a class representing the position inside a bar (i.e. 1, 2, 3, 4; for beats in a bar of 4/4). The state sequence is forced to always traverse the bar from left to right (i.e., transitions from beat 3 to beat 2 are not allowed), and imposing that time signature changes are unlikely. We consider bar lengths of 3 and 4 beats (12 and 16 tatums). We invite the interested reader to refer to [93] and Section 2.3.3 for further information.

7.4 Experimental setup

Model variations and implementation

The models were implemented with Keras 2.0.6 and TensorFlow 1.2.0 [30, 1]. We use the ADAM optimizer [89] with default parameters. We stop training after 10 epochs without

changes on the validation set, up to a maximum of 100 epochs. The low-level representations were extracted using the madmom library [12] and mapped by averaging the content in frames to the beat or tatum level. Table 7.1 summarizes the sixteen different variations under study and their codename. These variations convey:

- *Temporal granularity*: using low-level features synchronized to two temporal granularities (tatum and beat, which we will referred as *T* and *B*);
- *Architecture*: testing RNNs and CRNNs (*R* and *C*);
- *Output encoding*: with and without the addition of the structured encoding during training (referred as structured and unstructured, *S* and *U*);
- *Post-processing*: using either a threshold or a DBN (*t* and *d*).

All configurations are trained with the same input low-level representations, the same musical context—using patches of 15 beats or 60 tatums depending on the temporal grid—, and the same training parameters—using mini-batches of 64 patches per batch and a total of 100 batches per epoch—. This allows us to draw conclusions about the performance of the models in different conditions and to compare the architectures modularly.

Candidates for downbeats are obtained in two different manners. The first one is by thresholding the output activations with a threshold chosen to give the best F-measure result on the validation set. The second manner is to post-process the networks' outputs by adapting the DBN used in [93]. In this way we report the gain of using the DBN in each case. We use the DBN to model time signatures 3/4 and 4/4 following [93], and modifying it accordingly to the temporal grid (i.e., allowing bar lengths of {3,4} beats or {12, 16} tatums).

Table 7.1 Different variations under study and respective codenames (i.e. a CRNN model using unstructured encoding, tatum granularity and DBN would be CUTd).

	variations	codename
temporal granularity	tatum	T
	beat	B
DNN architecture	RNN	R
	CRNN	C
labels encoding	unstructured	U
	structured	S
post-processing	threshold	t
	DBN	d

Metrics and datasets

We investigate the performance of these configurations on 8 datasets of Western music, in particular: *Klapuri*, *R. Williams*, *Rock*, *RWC Pop*, *Beatles*, *Ballroom*, *Hainsworth* and *RWC Jazz*. These datasets cover various genres, including pop, rock and jazz, among others. We refer the reader to Section 4 for more details on the different datasets.

We perform leave-one-dataset-out evaluation and report the F-measure scores as in [93, 52], with a tolerance window of 70 ms around the annotated downbeat position. 25% of the training data is used for validation. The RWC Jazz dataset is only used to illustrate the performance of the systems in a challenging scenario where the beat estimation is less accurate and the music genre differs considerably from the training data, it is not used for training. Methods are evaluated independently on each dataset listed above for comparison to prior work. We also include an evaluation over the union of all datasets (denoted *ALL*).

To determine statistically significant differences, we conduct a Friedman test on the *ALL*-set results, followed by post-hoc Conover tests for pairwise differences using Bonferroni-Holm correction for multiple testing [58].

7.5 Results and discussion

We use as baseline the reported performance of two state-of-the-art downbeat tracking systems [93, 52], which achieved 78% and 78.6% mean F-measure across all datasets.[3] The performance of the models presented here across datasets is better than the baselines for all the cases when using the DBN as post-processing stage. The best results are obtained with RUBd (our reference implementation of [93], see Section 7.3.3) and CUTd, up to 82.4% and 82.8% respectively, as summarized in Table 7.2. A possible explanation for the difference in performance between the baseline in [93] and our implementation is the difference in the beat tracking performance, which is 3.3% better than the one reported in [93], which is likely to explain the 4.7% improvement in the RUBd model. To make a fair comparison, we use the RUBd model as a baseline, with the reasonable assumption that it behaves as the state-of-the-art. Figure 7.7 illustrates the performance of the different model variations across datasets.

The Friedman test on the *ALL* set rejected the null hypothesis (repeated measurements of the same individuals have the same distribution), with $p < 1\mathrm{e}{-10}$. Post-hoc analysis determined that all pair-wise comparisons were significantly different ($p < 1\mathrm{e}{-3}$), with the following exceptions: RUTt/RSTt, RUTd/RSTd, CUTt/CSBt, CUTd/CSBd, CUBd/CSTd, and RSBd/CUBd. A detailed analysis is presented in the following.

[3] For datasets that are not evaluated in [52], we report results in [93].

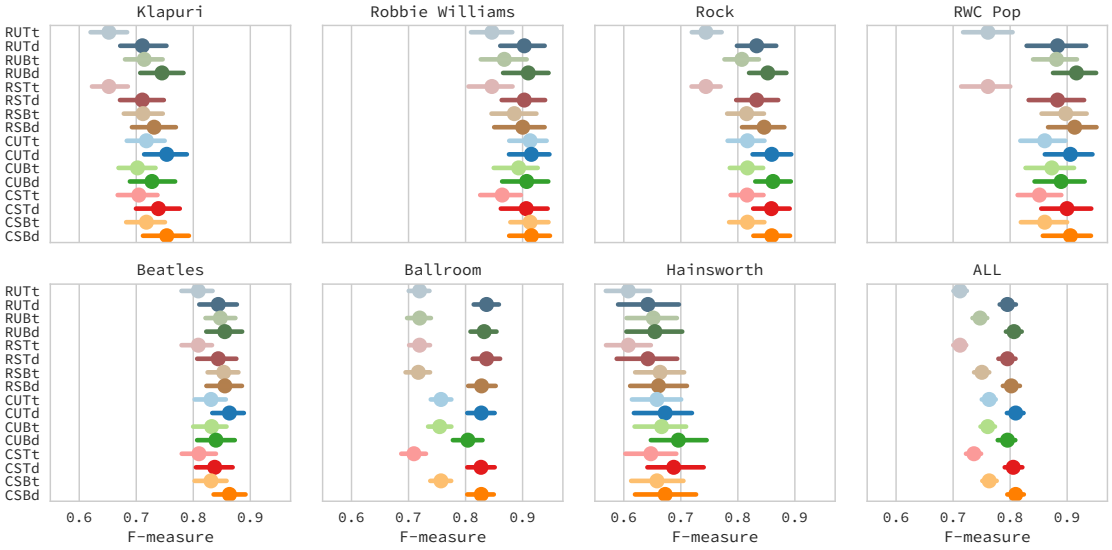


Figure 7.7 For each dataset, the estimated mean F-measure for each model under comparison. Error bars correspond to 95% confidence intervals under bootstrap sampling ($n = 1000$). *ALL* corresponds to the union of all test collections.

Table 7.2 Best performing systems among variations.

	F-measure
Krebs et al. [93]	78.0%
Durand et al. [52]	78.6 %
RUBd (baseline)	82.4%
CUTd (best variation)	82.8%

Effect of the temporal granularity

The temporal grid has an important effect on the performance of the RNN models, as illustrated in Figure 7.7 (T vs B variations). The RNN variations have more difficulty to model the temporal dependencies in the tatum grid, as shown by the thresholding results which are lower than the beat grid in most of the cases (e.g. RUTt vs. RUBt). The post-processing stage with the DBN becomes more important in the case of the tatum grid, helping the RNN models up to an extra 2.5% in mean F-measure over all datasets compared to the beat grid case.

By contrast, the CRNN models appear to be more robust to the temporal granularity change. In particular, for the case of the thresholding results, the difference in performance for the beat and tatum granularities (e.g. CUBt vs CUTt) is smaller than the case of the RNN variations. However, the increase in resolution seems to help the CRNN models in most

cases, showing a small increase from beat to tatum grid with the DBN (e.g., CUBd vs. CUTd). This indicates that the CRNN architecture is likely taking advantage of a finer temporal grid.

The results regarding the temporal granularity in the RNN and CRNN models are in line with the decisions of the authors in [93, 52], who in the first case decided to use tatums (with CNNs), and in the second case decided to use beats (with Bi-GRUs).

Effect of the DNN architecture

As shown in Figure 7.7, though statistically significant, the difference in performance between the two best versions of each architecture is small (being 82.4% and 82.8% F-measure for RUBd and CUTd respectively). This means that given the same data, input features and evaluation scheme, these two different DNN architectures can achieve equivalent performance, indicating that other aspects of the design such as training data and strategy (i.e. cross-validation vs. leave-one-dataset-out), or input representations might be more critical.

To see the performance of the systems in a difficult scenario, we performed an experiment on the RWC Jazz dataset, whose results are given in Figure 7.8. The DBN post-processing is used in all cases. Our results show that the CRNN models have better performance and less dispersion in the results than the RNN ones, showing robustness to unseen data whose genre is under-represented in the training data. Analogously to Figure 7.7, the RNN models show slightly better mean performance using the beat grid and the CRNN models with the tatum grid.

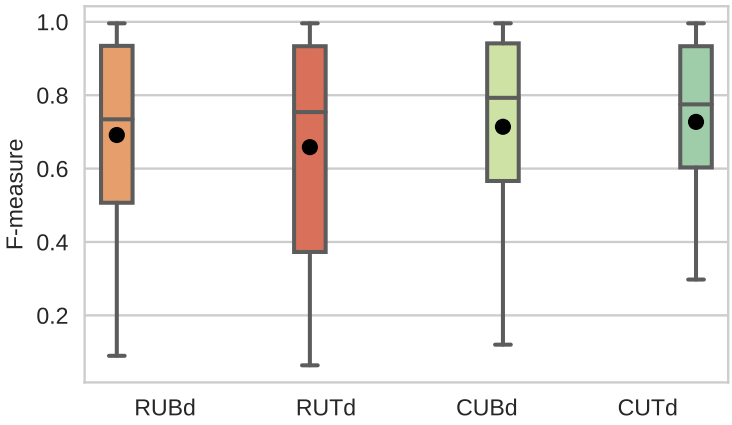


Figure 7.8 F-measure scores for the RWC Jazz dataset. Boxes show median value and quartiles, whiskers the rest of the distribution (0.99 quantiles). Black dots denote mean values. All results are obtained using the DBN post-processing.

Effect of the structural encoding

Regarding the structural encoding, the experiments show that it has no impact on the performance across databases (e.g. RU vs. RS and CU vs. CS in Figure 7.7). However, we observed that when using the structured encoding information, the downbeat activation's peaks are more evenly spaced and located consistently to an underlying time signature, reducing secondary peak occurrences. This presents a drawback when the estimation of the likelihood is inaccurate, because the errors tend to propagate over time, preventing the post-processing stage to correct them. Figure 7.9 illustrates this case.

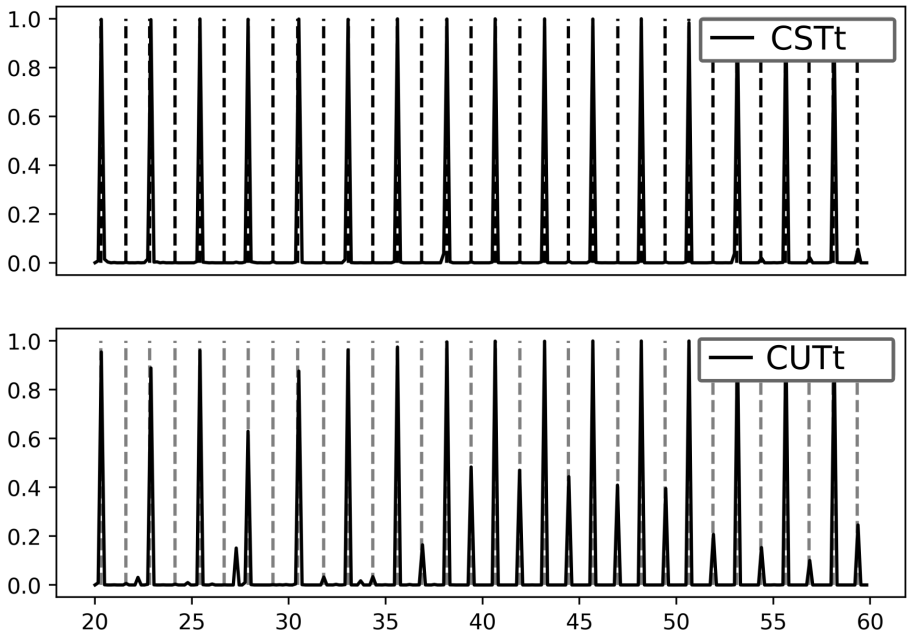


Figure 7.9 Example of downbeat activation's shape with and without structure encoding. The network's output using structured encoding presents less secondary peak occurrences.

We also observed that in some cases the use of structure encoding tends to decrease the accuracy of the peak positions in the estimated downbeat likelihood. We hypothesize that this could be a side effect of the encoding itself, since the entire beat interval is labeled with the same class, with no distinction of the onset position despite the case of the downbeat. This could change in a scenario with joint beat and downbeat tracking, where the unstructured encoding also contains the information of the location of beats. The addition of data augmentation could also contribute to help the system to learn the encoding properly.

The combined effects caused by the structured encoding results in the same overall performance as using the unstructured encoding only, indicating that including metrical information in this manner does not help in deep learning models for downbeat tracking.

Effect of post-processing

As shown in Figure 7.7, *d* vs. *t* model variants, the DBN post-processing helps in all cases, being particularly important with the tatum granularity and with the RNN models. The gap in performance between the models with and without post-processing is notable in the case of the Ballroom database, where in some cases, it is improving F-measure by 10%. The DBN increases the performance from RUTt to RUTd by 6.6% in the case of tatum grid across all databases, and 4.1% in the case of the beat grid (RUBt to RUBd). A similar trend is observed with the structured models (RS). The increase in the CRNN models performance is smaller, being 3.8% from CUTt to CUTd and 2.7% for CUBt to CUBd. The results obtained with the thresholding (t models) are more consistent over temporal granularities for the CRNN models, which suggests that the likelihood estimation of that model is more accurate and consistent over time.

7.6 Conclusions and future work

In this chapter we presented a systematic study of common decisions in the design of downbeat tracking systems based on deep neural networks. We explored the impact of the temporal granularity, output encoding, and post-processing stage in two different architectures. The first architecture is a state-of-the-art RNN, and the second is a CRNN introduced in this work. Experimental results show that the choice of the inputs' temporal granularity has a significant impact on performance, and that the best configuration depends on the architecture. The post-processing stage improves performance in all cases, with less impact in the case of the CRNN models whose likelihood estimations are more accurate. The proposed CRNN architecture performs as well as the state-of-the-art, improving robustness in a challenging scenario with an under-represented genre. We conclude that the addition of a densely structured output encoding does not help in the training of downbeat tracking systems. Nevertheless, the interaction of the structured encoding with multi-task training (beat and downbeat tracking) and data augmentation are interesting perspectives for future studies, for instance, as an extension of systems for the joint learning of beats and downbeats such us [15].

There remain open questions regarding the analysis of downbeat tracking systems, since this and other studies are not exhaustive in the whole universe of possibilities of these models [84]. For instance, the question of the impact of the training strategy (i.e. cross-validation vs. leave-one-dataset-out) was left out of the analysis presented here, and it has not yet been addressed formally. Among the different questions, probably the most important one is which are the remaining limitations of current state-of-the-art downbeat tracking models. As pointed out in [84], the performance of these systems in music with expressive timings

and in the presence of rare music variations (as in jazz or classical music) is relatively lower than in other Western music genres [52]. A question related to this, which has not been addressed before but we found particularly interesting, is whether the downbeat estimations obtained by these type of systems are musically consistent, i.e. whether the repeated structure of music is respected at different scales. This question is interesting because of two main reasons: 1) it relates to having more holistic retrieval systems; and 2) because it could potentially help to improve downbeat tracking estimation. To address this question, we explored several examples where the downbeat estimation failed in the Beatles dataset, which has annotations for both music structure (sections and boundaries) and downbeats, and presents pieces with variable meter. In the different examples studied, we observed that a considerable percentage of them had different downbeat estimations in repetitions of the same music segment. This means that two instances of a *verse* would have different downbeat estimations, whereas the most likely scenario is to have downbeats in the same position within the segment, i.e. to repeat the metrical structure. We hypothesize that the interaction of these systems with coarser musical structure might help closing the gap in the consistency of their estimations, and could also help downbeat tracking. In particular, the use of music structure information could bring valuable information, i.e. by repetitions of rare instances. The following chapter is dedicated to exploring these ideas.

Chapter 8

Structure-Informed Downbeat Tracking

Summary

In this chapter we study whether for a given system, the introduction of music structure information improves consistency and performance in downbeat tracking. We show in an oracle scenario how including long-term musical structure in the language model at the post-processing stage can enhance downbeat estimation. To that end, we introduce a skip-chain conditional random field language model for downbeat tracking designed to include music sections information in a unified framework. We combine this model with our state-of-the-art CRNN presented in Chapter 7, and we contrast its performance to the commonly used BPM. Our experiments show that this approach leads to more consistent downbeat estimations and it is able to handle rare music variations.

8.1 Motivation

In general, the pipeline of downbeat tracking systems based on deep learning consists of a first stage of low-level feature extraction, followed by a stage of feature learning, and a post-processing stage used to smooth the downbeat likelihood estimation obtained by the DNNs (see Chapter 2). Regardless of the architecture, most systems rely on HMMs or DBNs for the final downbeat inference. In particular, the bar pointer model (BPM) [181] has received considerable attention and it has been refined in different scenarios such as inferring tempo, time signature and downbeats [79] or long metrical cycles [166]. In these models the relation between latent-states are only modelled between time consecutive events or up to consecutive bars, ignoring longer term dependencies, which is a considerable simplification. Music has rich and interrelated dependencies within different time scales, thus accounting for music attributes over short and long temporal contexts is a more realistic approach.

In a diversity of music genres such as pop, rock or classical music the format of repeated sections is common practice. Typically, a song would feature an intro, verses and refrain,

notated using symbols such as '*AAB*'. It is then likely that music objects such as chord progressions or metric structure are similar among repetitions. Considering information from several instances of the same section is likely to provide complementary information and thus improve models' estimations [111]. An example of this situation is illustrated in Figure 8.1 for the song *Hello Goodbye* of the Beatles dataset. The estimation of the *CUBd* model (see Section 7.4) is denoted by the continuous curve, the dashed lines represent the annotated downbeat positions, and the top figure shows the annotated music sections. In this song, the estimations of the CRNN in different instances of the same segment (e.g. verse) are different, and sometimes wrong. In particular, the estimation of the instrumental verse could benefit from information about the other repetitions. Similarly to what happened with models such as the BPM, in which the inclusion of information from a coarser temporal scale (e.g. time signature) rules the inference of beats in a flexible way, we think that a model capable of handling structure information to help downbeat estimation is a novel idea that could bring interesting perspectives in terms of musical consistency within a piece.

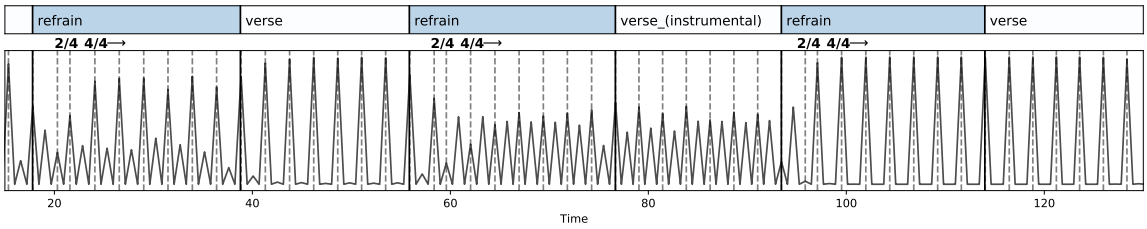


Figure 8.1 Excerpt of '*Hello Goodbye*'. Upper figure shows sections and bottom figure shows model's estimations. Dashed lines denote ground-truth downbeat positions, the continuous curve is the downbeat likelihood estimated by the *CUBd* model (see model variations Section 7.4).

In this chapter we study whether for a fixed system, the introduction of music structure information improves downbeat tracking performance. We propose a novel skip-chain CRF language model for downbeat tracking that incorporates structure information in a flexible way, and we assess its performance using a convolutional-recurrent network for the observations. We compare the performance of this language model with a popular DBN that implements the BPM [181, 93], showing its advantages. We also contrast the model to simpler approaches for including structural information. We validate our claim by assessing the different methods using annotated beats and sections on the Beatles songs dataset, to isolate noise due to beat/section estimation. Our experiments show that including music structure in language models for downbeat tracking helps in challenging cases, obtaining more consistent downbeat estimations.

8.2 Related work

The use of music structure to inform other MIR tasks has been previously explored. Dannenberg [34] proposed a system that incorporates structure information to perform beat tracking. He considers beat tracking as an optimization problem where the goal is to infer the best beat sequence given the constraint of a chroma similarity matrix. Mauch et al. [111] addressed the use of musical structure to enhance automatic chord transcription. Their method consists in averaging the chroma feature inputs in repeated sections and replace the occurrences by the average of the chroma. The authors showed the suitability of this approach for chord recognition, since it helps in obtaining consistent and more readable chord progressions. Although both methods showed promising results, they use very simple features and have limited flexibility to include other musically relevant information. Besides, they do not account for variations among different occurrences of the same segment. Papadopoulos et al. [133] proposed the use of Markov Logic Networks to include music structure information for chord transcription in a flexible way. The authors model the probability of a chord progression to occur in repeated occurrences of similar sections given the underlying chroma observations. This work showed that graphical models are capable of incorporating musical knowledge in different time scales in a flexible and unified manner. The main limitations of this work are the use of simple features and the very slow inference.

Among probabilistic graphical models, CRFs are discriminative classifiers for structured data prediction, which allow for modeling complex and interrelated properties both in the observations and output labels at different time scales, thus making them appealing for incorporating structure information in downbeat tracking systems. Linear-chain CRFs (LC-CRFs) have been successfully applied in MIR tasks such as beat tracking [57] and downbeat tracking [54], but these models are still limited to relating time neighbouring output labels. In turn, skip-chain CRFs (SCCRFs) extend LCCRFs by adding connections between nodes that do not represent consecutive events in time, and have been successfully applied in NLP. Sutton et al. [169] used SCCRFs in a simple speaker identification task on seminar announcements, showing that they outperform linear-chain CRFs while modelling more complex structure between words. In turn, Liu et al. [103] applied SCCRF to the problem of biomedical name entities recognition and showed it achieves significant improvements with respect to LCCRFs for the recognition of gene and protein mentions.

8.3 Proposed system

Our system consists of two main stages: first we compute the downbeat likelihood using the CRNN presented in Chapter 7, then we obtain the downbeat positions with a structure-informed SCCRF. The different components are explained in the following.

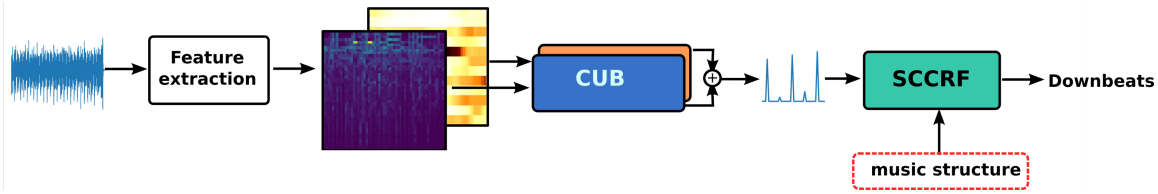


Figure 8.2 Overview of proposed model.

8.3.1 Convolutional-Recurrent Network

Briefly recalled, the CRNN model presented in Chapter 7 consists of an ensemble of two CRNNs, representing the harmonic and percussive content of the signal respectively. In particular, to simplify our analysis in terms of computational complexity, we use the CRNN of the *CUBd* variation (see Section 7.4), which has beat-synchronous features and unstructured encoding. We rename this model *CUB* to emphasize that we use the model's activations with different post-processing strategies different from the ones of the previous chapter. Note that the investigation presented here focuses on whether the use of structure information helps downbeat tracking for a fixed system, so the use of beat or tatum synchronous grid is indifferent for this study. We refer the interested reader to Section 7.3.3 for further information.

8.3.2 Skip-Chain Conditional Random Field model

The SCCRF model consists of a linear-chain CRF with additional long-distance connections between nodes, so called skip-connections [169]. Each node represents one time point in the sequence, in this case, beats. The evidence at one endpoint of the skip connection influences the label at the other distant endpoint, as illustrated in Figure 8.3. In its general formulation, the conditional probability of a label sequence $\mathbf{y} = (y_1, ..., y_T)$ of length T given an input sequence of observations $\mathbf{x} = (x_1, ..., x_T)$ is given by:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{T} \psi_t(y_t, y_{t-1}, \mathbf{x}) \phi(y_t, \mathbf{x}) \prod_{(u,v)\in\mathcal{I}} \psi_{uv}(y_u, y_v, \mathbf{x}), \tag{8.1}$$

where ψ_t is the local transition potential for the linear-chain (neighbouring nodes), ϕ is the observation potential and ψ_{uv} is the potential of the skip connections, which is defined for a pre-selected subset of nodes $\mathcal{I}$. In our model, the local transition potentials and skip potentials are independent of the observations, and the observation potential depends on the observation at time t, so Equation 8.1 becomes:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{T} \psi_t(y_t, y_{t-1}) \phi(y_t, x_t) \prod_{(u,v)\in\mathcal{I}} \psi_{uv}(y_u, y_v). \tag{8.2}$$

The skip potential also differs by modeling interactions between distant nodes, which is well justified given the flexibility of CRFs in terms of independence assumptions [178]. In Figure 8.3 the local transition potentials are indicated in blue, the skip potentials in green and the observations in orange.

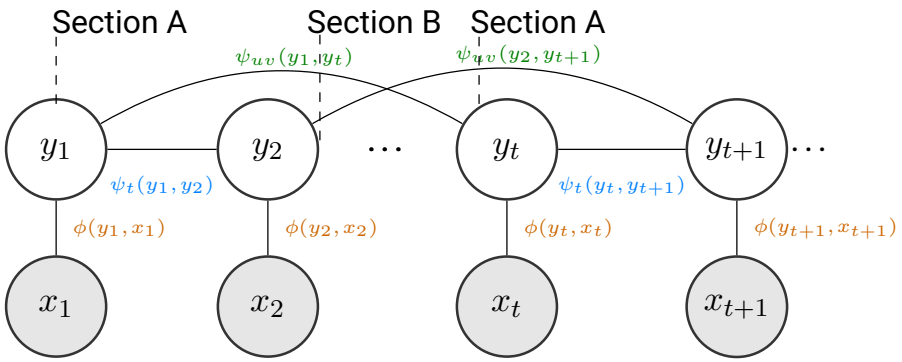


Figure 8.3 SCCRF graph. Observations and labels are indicated as gray and white nodes respectively. Beats of repeated section occurrences are connected to each other. Local transition, skip and observation potentials are indicated in blue, green and orange respectively.

We consider a set of labels $\mathcal{Y}$ which represents the beat position inside a bar, similarly to the BPM (see Section 2.3.3). Following [93, 15], and in order understand how this model behaves in a set up usually exploited in the literature, we consider bar lengths of 3 and 4 beats, corresponding to 3/4 and 4/4 meters. We model the beat position relating to different time signatures as separate labels. The first beat in a 3/4 time signature will have a different label compared to the first beat of a 4/4 time signature. The output labels y are a function of two variables: the beat position $b \in \mathcal{B} = \{1, ..., b_{max}(r)\}$ and the number of beats per bar $r \in \mathcal{R} = \{r_1, ..., r_n\}$, which relates to the time signature of the piece. This results in seven possible labels $\mathcal{Y} = \{1, .., 7\}$, one for each position b in $\mathcal{R} = \{3, 4\}$, i.e. the second beat of a 4/4 bar would be $b = 2$, $r = 4$ and $y = 5$.

Local transition potential ψ_t

The local transition potential applies only to nodes that are consecutive in time. Similar to [93, 54, 57], we impose that the beat position b inside a bar increase by one up to the maximum bar length considered, and switch to one at the end of the bar. Time signature changes are unlikely and only allowed at the end of the bar. Formally, this is expressed in

terms of $y_t = (b_t, r_t)$ as:

$$\psi_t(b_t, b_{t-1}, r_t, r_{t-1}) = \begin{cases} 1 & \text{if } b_t = b_{t-1} + 1 \\ 1-p & \text{if } r_t = r_{t-1},\, b_t = 1,\, b_{t-1} = r_{t-1} \\ p & \text{if } r_t \neq r_{t-1},\, b_t = 1,\, b_{t-1} = r_{t-1} \\ 0 & \text{otherwise,} \end{cases} \tag{8.3}$$

where $p = 10^{-6}$ is the probability of changing the time signature. We kept the value of p used in the previous chapter, which was set to the one in [93] for a fair comparison. This low value represents the probability of changing the time signature within a piece, which is rare for the datasets under study.

Observation Potential ϕ

The observation potential depends on the observation x_t at time t, given by the downbeat likelihood a_t computed by the CRNN. Formally:

$$\phi(b_t, a_t) = \begin{cases} a_t & \text{if } b_t = 1 \\ 1 - a_t & \text{otherwise.} \end{cases} \tag{8.4}$$

Skip potential ψ_{uv}

The skip potential depends on two labels y_u, y_v which are not neighbours in the time axis. It is given in terms of b and r by:

$$\psi(b_u, b_v, r_u, r_v) = \begin{cases} \alpha & \text{if } b_u = b_v,\, r_u = r_v \\ \frac{1-\alpha}{|\mathcal{Y}|-1} & \text{otherwise.} \end{cases} \tag{8.5}$$

When connected, two distant nodes y_u and y_v are constrained to have the same label by a factor α, and to have different labels by $\frac{1-\alpha}{|\mathcal{Y}|-1}$ (where the different labels are equally possible). Intuitively, the parameter α controls how strongly what is happening in distant nodes affects the local ones. We found the best value $\alpha = 0.3$ on a grid search between 0 and 1.

Graph structure

The subset $\mathcal{I}$, that determines which nodes are linked through the skip connections, is obtained using the musical section labels, given as input to the model. If a section s has multiple occurrences, we connect the first beats of each occurrence of s to each other, the second ones to each other, and so on, as shown in Figure 8.3. If the section repetitions are

of different lengths, we connect the beats until the shortest section length is reached. We connect the beats of all occurrences of s, i.e. the more repetitions, the more connections. All repetitions are connected to each other.

8.4 Experimental setup

Implementation details

The *CUB* model was implemented with Keras 2.0.6 and TensorFlow 1.2.0 [1, 9]. We use the ADAM optimizer [15] with default parameters. We stop training after 10 epochs without improvement on validation accuracy, up to a maximum of 100 epochs. The low-level representations were extracted using the madmom library in Python [12] and mapped to the beat grid by average. See Section 7.4 for detailed information. The SCCRF was implemented using the *factorgraph* package.[1]

Metrics and datasets

We use the Beatles dataset, since it has beat, downbeat and music structure annotations. We follow the leave-one-dataset-out evaluation scheme of [93, 52] and we train the *CUB* network with 6 Western music datasets leaving the Beatles dataset out (as in Chapter 7). Those datasets are: *Klapuri*, *R. Williams*, *Rock*, *RWC Pop*, *Ballroom* and *Hainsworth*, totalling 35h 03m of music. We report the F-measure score following previous works. To determine statistical significance, we perform a Friedman test followed by post-hoc Conover tests for pairwise differences using Bonferroni-Holm correction for multiple testing [58]. Message convergence in the LBP algorithm is given by $|\mu_{ij}^{m} - \mu_{ij}^{m-1}| < \tau \, \forall i, j$ where m is the current iteration and τ is a tolerance; or a maximum number of iterations is reached (see Section 2.3.4). We set $\tau = 10^{-8}$ and consider a maximum of 3000 iterations. Messages are normalized at each iteration to avoid values going to zero easily in practice. Inference takes a median time of 3.6s on 2m 30s of music using an Intel Xeon CPU E5-2643 v4 @ 3.40GHz.

8.5 Results and discussion

We compare the SCCRF performance to the DBN in [93], which is our state-of-the-art baseline. We employ the downbeat likelihood estimated by the CRNN as observations for both language models. To compare the SCCRF to simpler approaches aware of structure information, we enhance the CRNN estimation before performing inference with the DBN by:

[1] https://github.com/mbforbes/py-factorgraph — Last accessed 24/07/2019.

- averaging the input representations of section repetitions, replacing the occurrences by the average and feeding the network with the averaged features;
- applying the same idea but averaging the downbeat likelihood estimation of repeated sections instead.

We name these two approaches *DBN_AVF* and *DBN_AVA* respectively. We include a non structure-informed version of our model, without the skip potentials, in order to assess possible differences over the DBN due to the inference method. This model is a linear-chain CRF, and is denoted as *LCCRF*. Table 8.1 illustrates the different configurations we compare, and Figure 8.4 summarizes their performance.

Table 8.1 Summary of the different configurations studied, with and without the inclusion of music structure information.

model	music structure information	inference
SCCRF	skip connections	LBP
LCCRF	No	LBP
DBN	No	Viterbi
DBN_AVA	via likelihood average in repetitions	Viterbi
DBN_AVF	via features average in repetitions	Viterbi

The standard DBN approach benefits from adding structural information in terms of reducing variance in the performance. The SCCRF model brings the most benefit out of the compared models, due to improvement in difficult cases. The LCCRF performance is slightly worse than the DBN. This difference is presumably due to the inference algorithm. The LCCRF employs an inexact inference using the LBP algorithm, while the DBN uses Viterbi which gives the exact most likely solution. Mean and median performances of the SCCRF and DBN models are similar, with their difference not being statistically significant.

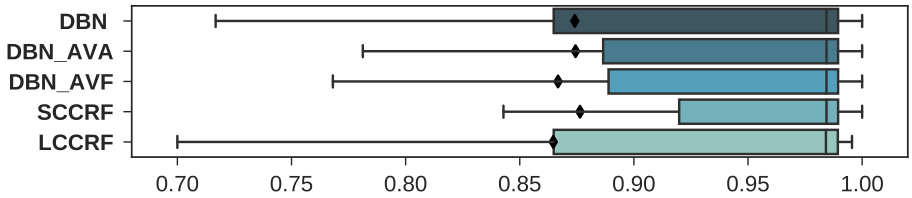


Figure 8.4 F-measure scores. Boxes show median value and quartiles, whiskers the rest of the distribution (0.99 quantiles). Black dots denote mean values.

Figure 8.5 illustrates an example where the inclusion of structure information and the flexibility of the SCCRF model help in the downbeat estimation. In this excerpt the downbeat likelihood estimation is inconsistent among different instances of the same section, and

in particular, it is correct in some instances and wrong in others. For instance, the downbeat likelihood has peaks in the right positions in one verse and the estimation is partially correct or incorrect in the two others. The SCCRF downbeat estimation is consistent over all section occurrences despite the discordant likelihood estimations, and more accurate in the overall performance. In turn, the DBN is not able to overcome the likelihood estimation errors, which is expected given the limited information it handles and the hard transition constraints. The time signature of this song is mostly 4/4, with the exception of one bar in 3/4 at the end of each verse. The SCCRF finds there is a 3/4 transition bar between the verse and the refrain, but it estimates that the 3/4 bar is in the refrain instead of the end of the verse. We hypothesize that this is due to the observation values which give more evidence of having a 3/4 bar in the position where the model finds it. The combination of the information in different time scales and the inference carried out globally make the model capable of identifying rare music variations and to fit the global time signature consistently.

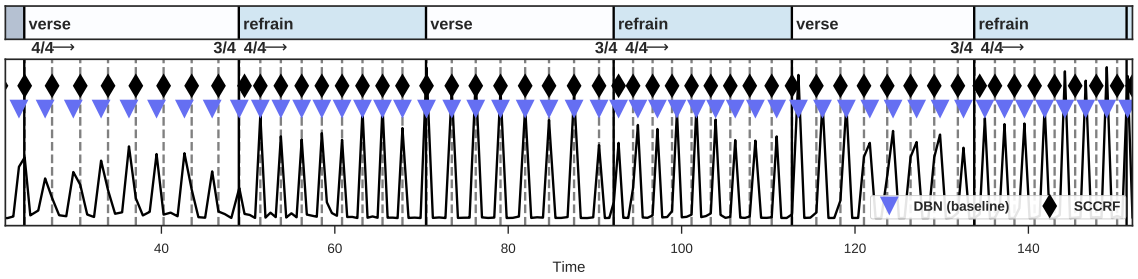


Figure 8.5 Excerpt of '*Blue Jay Way*'. Upper figure shows sections and bottom figure shows model's estimations. Dashed lines denote ground-truth downbeat positions, the continuous curve is the downbeat likelihood estimated by the CRNN (without any structure information). Time signature changes are denoted with numbers in between figures (4/4 and 3/4). The SCCRF improves downbeat tracking performance from 0.35 to 0.72 F-measure with respect to *CUBd*.

Another interesting example is shown in Figure 8.6. In this excerpt, *Hello Goodbye*, the downbeat likelihood estimation is accurate in some instances of the *verse*, which could be helpful for the DBN to infer the downbeat positions accurately. However, since the piece has time signature changes (4/4 $\rightarrow$ 2/4), it presents a challenging scenario for the DBN. As mentioned before, both methods—the SCCRF and the DBN— model time signatures of 3/4 and 4/4. This means that none of the methods are actually able to model the time signature 2/4, and as shown in Figure 8.6 they deal with this change in very different ways (also considering that the time signature changes are equally likely in both models). In the case of the DBN, the decoding is so that the piece's time signature is always 4/4, which might be supported by the observations (i.e. downbeat likelihood), since there are peaks at positions corresponding to the 4/4 meter. On the other hand, the imposition of consistency over segment repetitions in the SCCRF estimation implies that it finds a time signature change, which approximates as two bars with the 3/4 meter. This is the optimal way to substitute the actual time signa-

ture change with one that the model accounts for and that allows to be in synchrony within segment repetitions (2/4+4/4→ 3/4+3/4, both combinations with six beats).

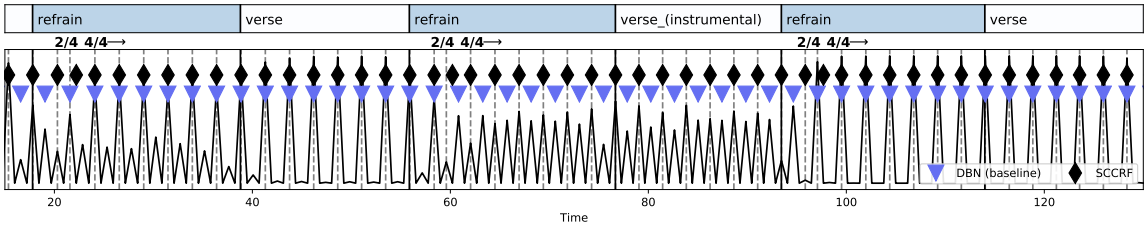


Figure 8.6 Excerpt of '*Hello Goodbye*'. Upper figure shows sections and bottom figure shows model's estimations. Dashed lines denote ground-truth downbeat positions, the continuous curve is the downbeat likelihood estimated by the CRNN (without any structure information). Time signature changes are denoted with numbers in between figures (4/4 and 2/4). The SCCRF improves downbeat tracking performance from 0.67 to 0.94 F-measure with respect to *CUBd*.

Figure 8.7 shows an example of the downbeat estimation with the DBN with structure-enhanced CRNN observations. The three likelihood estimations correspond to the three models DBN, DBN_AVF and DBN_AVA, and the downbeat positions found by the DBN using each set of observations are shown with dots of the respective color. We noticed that the inclusion of structure information through averaging features (DBN_AVF) has limited impact on the performance. Averaging the likelihood of different section occurrences presents the advantage that the likelihood has higher values where the CRNN finds strong evidence of downbeat occurrences and smaller values when it is unclear, so the average compensates with the correct estimation in many cases. Nevertheless, in examples like the one of Figure 8.7 which have shifted likelihood estimations and transition bars, the downbeat estimation of DBN_AVF and DBN_AVA do not achieve the consistency of the SCCRF on the different occurrences of the same section, indicating that it is necessary to have a flexible and robust language model to account for this information.

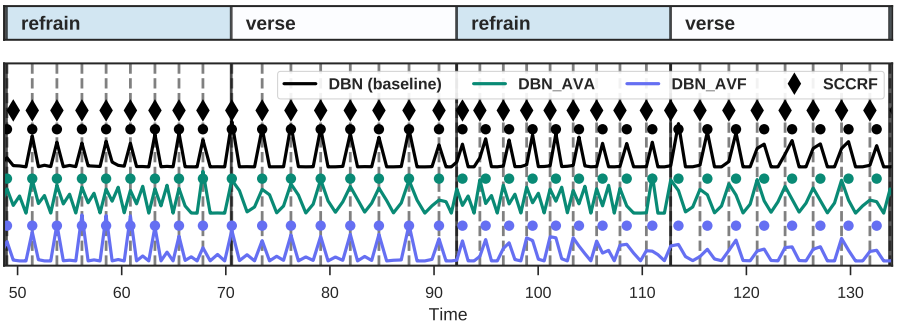


Figure 8.7 Excerpt of '*Blue Jay Way*'. Sections are shown on top and DBN estimations with enhanced CRNN observations in the bottom. Dots denote the downbeat positions estimated by the DBN in each case. Dash lines denote the ground-truth positions.

Finally, we noticed that the SCCRF does not outperform the DBN in all excerpts, in particular there are examples where the DBN performance is better than the SCCRF. Those are mainly examples where the annotations have contradictions such as two occurrences of the same section beginning in different parts of the bar, so the skip connections of the SCCRF model are misaligned and the information they provide is inaccurate. The SCCRF model is sensitive to this kind of misalignment, which impacts negatively in its performance. This makes it difficult to use estimated sections to infer the graph structure with current section estimation methods. In preliminary tests we perform with the *msaf* package [127], we found that the boundary estimations of the systems presented there are inconsistent with respect to the metric position where segment repetitions start—e.g. the same section starting before or after the downbeat in two different occurrences—, this happening even in the beat grid which is a coarser time scale in comparison to tatums or frames. This could be expected given that current methods for section labelling and boundaries detection do not consider meter structure explicitly, and also because the tolerances used for boundary detection are either 0.5 s or 3 s, which is considerably bigger than needed when looking for downbeats. This behaviour triggers the question whether if is not worth making interact these two musical levels the other way round: we saw that structure information can help downbeat tracking, is it the case that downbeat information can improve structure segmentation? We offer a discussion on this ideas in the next chapter.

8.6 Conclusions and future work

In this chapter we have addressed the question of whether for a fixed system, the inclusion of music structure information improves downbeat estimation, and in particular, if it leads to musically more consistent estimations. To that end, we presented a Skip-Chain Conditional Random Field language model which exploits music structure information in a unified and flexible manner. In our experiments, we have shown that using knowledge of repeating structure in the language model improves the downbeat estimation over state-of-the-art approaches by providing consistency among occurrences of the same section. We discussed different examples where approaches that integrate information in multiple scales will make the difference, showing promising perspectives especially in challenging cases with rare music variations such as fast time signature changes, where models like the BPM have limited performance due to the limited contextual information they handle. The proposed method can be directly applied to beat tracking, and easily extended to the joint tracking of beats and downbeats by incorporating suitable potentials. Considering information about rhythmic patterns as an intermediate temporal scale between bars and sections is also a promising idea to explore in the future.

The main limitation of the system comes down to inferring the structure of the skip-chain graph. In the present work we used the annotations, but ideally the graph structure could be obtained by estimating boundaries and labels of sections with an external algorithm, in a fully automatic fashion. The preliminary experiments we performed on this lead us to further questions in terms of music consistency among different scales. We noticed that state-of-the-art music structure estimation systems, at least the extensive list presented in [127], do not consider meter information in their estimation, leading to situations such as different sections' occurrences starting in different meter positions, and in particular, not many of the sections starting on a downbeat position, which is incorrect from a musical point of view in the dataset under investigation. From our point of view, the question that naturally arises from this is whether information flow is beneficial the other way round as well, and if the use of downbeat positions information can enhance musical consistency in a coarser scale, i.e. structure estimation. These are promising perspectives for future work that we discuss in Chapter 10. The following chapter is dedicated to explore the interplay with finer temporal scales that play a key role in the perception of expressiveness in music.

Chapter 9

Beat and Microtiming Tracking

Summary

In this chapter we explore the interplay between the beat level and the tatum level with the idea of integrating microtiming descriptors in automatic rhythm analysis systems. To that end, we formulate a common framework to describe microtiming in Brazilian *samba* and Uruguayan *candombe*, by focusing on the timekeeper instruments of those Afro-Latin American music traditions. We propose a language model based on CRFs that integrates deep learned beat and onset likelihoods as observations to track beats and microtiming profiles, and we assess our approach in controlled conditions suitable for these timekeeper instruments. We conduct preliminary studies on the microtiming profiles' dependency on genre and performer, illustrating promising aspects of this technique towards automatic tools for more comprehensive analysis of these music traditions.

9.1 Motivation

In previous chapters we investigated the interplay between downbeats, which play a key role in the organization of rhythm in many music genres, and music sections, which encode acoustic and semantic similarity and dictate the coarser structure of a music piece. Despite the fact that the automatic estimation of those two music objects is usually addressed separately in the MIR community, we have seen that modeling their interrelation is beneficial. In this chapter we address a different question, in a different time scale, but with the same vision. In some cases, the events in music present small-scale temporal deviations with respect to the underlying regular metrical grid, a phenomenon commonly referred to as microtiming.[1] Figure 9.1 illustrates an example of microtiming at the sixteenth note level. The interaction between microtiming deviations and other rhythmic dimensions contribute to what has been described as the sense of 'swing' or 'groove' [44, 107, 37]. The systematic use

[1] During this chapter we will refer to it as microtiming or microtiming deviations interchangeably.

of these deviations is of structural importance in the rhythmic and stylistic configuration of many musical genres. This is the case of jazz [83, 177, 45, 44], Cuban *rumba* [8], Brazilian *samba* [125, 182] and Uruguayan *candombe* [87], among others. Consequently, the analysis of these music genres without considering microtiming leads to a limited understanding of their rhythm. It is desirable that the computational rhythmic analysis of this music retrieve and describe such fundamental aspects.

The work presented in this chapter takes a first step towards fully-automatic and joint tracking of beats and microtiming deviations, applied to the analysis of two (usually underrepresented) Afro-Latin American music genres, namely Brazilian *samba* and Uruguayan *candombe*. The aim of this approach is to relate two time-scales that naturally interact in these music genres, beats and microtiming (which in these music genres is present at the tatum level, as discussed below), towards coherent and descriptive models of these music traditions.

More precisely, we introduce a CRF model that uses beat and onset activations derived from deep learning models as observations, and combines them to jointly track beats and microtiming profiles within rhythmic patterns at the sixteenth note level. This temporal granularity is in accordance with the type of microtiming deviations present in the music traditions under study. Following previous works [44], we derive microtiming labels from annotated onsets and use them to evaluate the proposed system, attaining promising results towards more holistic and descriptive models for rhythm analysis. We also study the usefulness of this approach in controlled conditions, as a first assessment of its capabilities. We explore our microtiming representation in some applications, namely the extraction of microtiming profiles of certain instruments, and the study of differences between musical genres based on their microtiming traits.

9.1.1 Microtiming in *Samba* and *Candombe* music

Samba and *candombe* are musical traditions from Brazil and Uruguay, respectively, that play a huge role in those countries' popular cultures. Both genres have deep African roots, partly evidenced by the fact that their rhythms result from the interaction of several rhythmic patterns played by large ensembles of characteristic percussive instruments. *Candombe* rhythm is structured in 4/4 meter, and is played on three types of drums of different sizes and pitches—*chico*, *repique* and *piano*—, each with a distinctive rhythmic pattern, the *chico* drum being the timekeeper.[2] *Samba* rhythm is structured in 2/4 meter, and comprises several types of instruments—*tamborim*, *pandeiro*, *chocalho*, *reco-reco*, *agogô*, and *surdo*, among others (see Chapter 5 for more details). Each instrument has a handful of distinct patterns [60], and more than one instrument may act as the timekeeper. Because of this com-

[2] In this musical context, the role of timekeeper is assigned to an instrument that plays an invariable rhythmic pattern (i.e., an *ostinato*) usually at a high rate, thus defining the subdivision of the beat.

bination of several timbres and pitches, the texture of a *samba* performance can become more complex than that of a *candombe* performance, where only three types of drums are present. Nevertheless, both rhythms have in common that they exhibit microtiming deviations at the sixteenth note level [102, 87, 125], with no deviations in the beat positions, as shown in Figure 9.1.[3] The main motivation of the work presented in this chapter is to take steps towards more insightful and descriptive automatic models for the analysis of rhythm. In this line, we consider that these music traditions are an excellent case of study, and we develop a method that is useful to illustrate some intrinsic characteristics of their rhythms, as well as commonalities and differences between them.

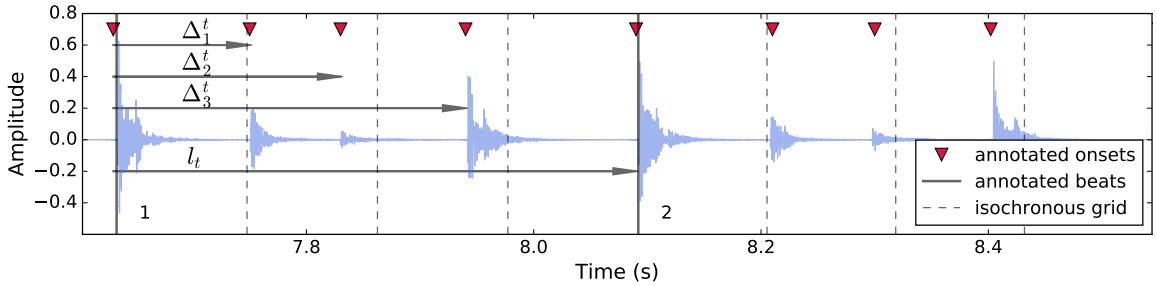


Figure 9.1 Example of microtiming deviations at the sixteenth note level for a beat-length rhythmic pattern from the *tamborim* in *samba de enredo*.

9.2 Related work

Most of the proposed methods for microtiming analysis are based on manually annotated data. Laroche et al. [101] proposed a method for the joint estimation of tempo, beat positions and swing in an ad-hoc fashion. The proposal exploits some simplifications: assuming constant tempo and swing ratio, and propagating beat positions based on the most likely position of the first beat. More recent works perform semi-automatic analysis still relying on informed tempo [45, 44], or using an external algorithm for its estimation [107]. Within the context of *candombe* and *samba*, microtiming characterization has also been addressed using either semi-automatic or heuristic methods [125, 64, 87].

In other rhythm-related MIR tasks such as beat and downbeat tracking, graphical models such as hidden Markov models or dynamic Bayesian networks are widely used [93, 79, 15, 166]. CRFs have been applied in MIR tasks such as beat tracking [57] or audio-to-score alignment [85], and have been successfully combined with deep neural networks (DNNs) [54, 92].

[3] In other musical forms, such as waltz, microtiming may be mostly on beats.

9.3 Proposed model

The proposed language model consists of a linear-chain CRF [100, 169], and is depicted in Figure 9.2 (see Chapter 2 for more information about CRFs).

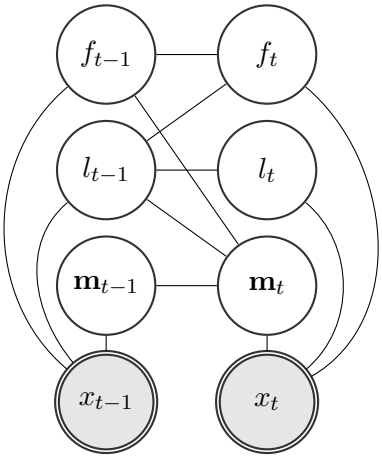


Figure 9.2 CRF graph. Observations and labels are indicated as gray and white nodes respectively. Double circles denote continuous variables, and simple circles discrete variables.

In our model, the output labels **y** are a function of three variables that describe the position inside the beat f_t, the length of the beat interval in frames l_t, and the microtiming profile within the beat-length pattern at the sixteenth note level $\mathbf{m}_t$. Formally,

$$y_t := (f_t, l_t, m_t), \tag{9.1}$$

where f_t is the frame counter with $f_t \in \mathcal{F} = \{1, ..., l_t\}$, $l_t \in \mathcal{L} = \{l_{\min}, ..., l_{\max}\}$ is the number of frames per beat, which relates to the tempo of the piece; and the microtiming $\mathbf{m}_t \in \mathcal{M} = \{\mathbf{m}_1, ..., \mathbf{m}_N\}$. The observations **x** are based on estimated beat and onset likelihoods, as detailed later. The problem of obtaining the beat positions and microtiming profiles is then formulated as finding the sequence of labels $\mathbf{y}^*$ such that $\mathbf{y}^* = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x})$.

Microtiming representation and tracking

Both in *samba* and *candombe*, timekeeper instruments usually play a beat-length rhythmic pattern that articulates several sixteenth notes [60], as shown in Figures 9.1 and 9.3 for the *tamborim*. In order to provide a common framework for comparing both music genres, we focus our study on the microtiming deviations of beat-length rhythmic patterns articulated by timekeeper instruments in groups of four sixteenth notes.[4] To this end, we consider the following hypothesis, which we explain further below:

[4] Note that minor adjustments to the proposed model allow for the tracking of microtiming deviations in other kinds of rhythmic patterns.

- The tempo is constant within a beat.
- The microtiming profile changes smoothly only at beat transitions.
- The tempo is between 120 and 135 BPM, to ensure an appropriate temporal resolution.

We define the microtiming descriptor $\mathbf{m}$ at frame t as:

$$\mathbf{m}_t := (m_t^1, m_t^2, m_t^3),$$

where $m_t^i := \frac{\Delta_t^i}{l_t} \in [\frac{i}{4} + \delta_L^i, \frac{i}{4} + \delta_U^i]$, and Δ_t^i is the distance in frames between an articulated sixteenth note and the beginning of the beat interval, as shown in Figure 9.1. Thus, each m_t^i models the position of the i-th sixteenth note with respect to the beginning of the beat, relative to the total beat length. For instance, the value of the microtiming descriptor for a rhythmic pattern of four isochronous sixteenth notes, i.e., located exactly on an equally-spaced metrical grid, is $\mathbf{m} = (0.25, 0.50, 0.75)$, indicating the articulation of events at $1/4, 1/2$ and $3/4$ of the beat interval respectively. To account for different microtiming profiles, the value of m_t^i is estimated within an interval determined by lower and upper deviations bounds, δ_L^i and δ_U^i, modeled as positive or negative percentages of the beat interval length. The proposed microtiming descriptor provides an intuitive idea of how the articulated sixteenth notes deviate within the rhythmic pattern from their isochronous expected positions. It is independent of tempo changes, since it is normalized by the estimated beat interval length, allowing for studies on microtiming–tempo dependencies. We will discuss the usefulness of this microtiming representation in Section 9.6.

The definition of the microtiming descriptor $\mathbf{m}_t$ can be related to the *swing-ratio*, s, proposed in previous work [44, 107], though the two differ in various aspects. The *swing-ratio* is defined in terms of the inter-onset intervals (IOIs) of a long–short rhythmic pattern, such that $s \geq 1$ is the ratio between the *onbeat* IOI (longer interval) and the offbeat IOI (shorter interval). In contrast, the $\mathbf{m}_t$ descriptor is composed by three *microtiming–ratios*, m_t^i, whose IOIs are defined with respect to the beginning of the beat instead of the previous onset as in [44, 107]. However, it is possible to convert the model proposed here into the *swing-ratio* by redefining $\mathbf{m}$ as $m_s := m_t^1$, and then, from the estimated m_s, computing $s = \frac{m_t^1}{1-m_t^1}$. With such modifications, the model could be applied to the studies presented in [44, 107].

Transition Potential ψ

The transition potential is given in terms of f_t, l_t, and m_t (see Equation 9.1) by:

$$\psi(y_t, y_{t-1}) := \psi_f(f_t, f_{t-1}, l_t, l_{t-1})\psi_m(\mathbf{m}_t, \mathbf{m}_{t-1}, f_{t-1}, l_{t-1}).$$

Similar to [93, 57], we force the frame counter f_t to increase by one, at each step, up to the maximum beat length considered, and to switch to one at the end of the beat. Beat duration

changes are unlikely (i.e., tempo changes are rare) and only allowed at the end of the beat. We constrain these changes to be smooth, giving inertia to tempo transitions. Those rules are formally expressed by:

$$\psi_f(f_t, f_{t-1}, l_t, l_{t-1}) := \begin{cases} 1 & \text{if } f_t = f_{t-1}\, l_{t-1}) + 1, \\ 1 - p_f & \text{if } l_t = l_{t-1}, \\ & \quad f_t = 1,\, f_{t-1} = l_{t-1} \\ \frac{p_f}{2} & \text{if } l_t = l_{t-1} \pm 1,\, f_t = 1 \\ & \quad f_{t-1} = l_{t-1} \\ 0 & \text{otherwise.} \end{cases}$$

The microtiming descriptor m_t changes smoothly and only at the end of the beat, that is:

$$\psi_m(\mathbf{m}_t, \mathbf{m}_{t-1}, f_{t-1}, l_{t-1}) := \begin{cases} 1 & \text{if } \mathbf{m}_t = \mathbf{m}_{t-1}, \\ & \quad f_{t-1} \neq l_{t-1} \\ 1 - p_m & \text{if } \mathbf{m}_t = \mathbf{m}_{t-1}, \\ & \quad f_{t-1} = l_{t-1} \\ \frac{p_m}{2} & \text{if } m_i^t = m_i^{t-1} \pm 0.02\ \forall i, \\ & \quad f_{t-1} = l_{t-1} \\ 0 & \text{otherwise.} \end{cases}$$

In the transition potential, p_f and p_m represent the probability of changing the beat interval length (i.e., tempo) and the probability of changing the microtiming profile at the end of the beat, respectively. The values of $1 - p_f$ and $p_f/2$ were chosen following previous works [15], whereas $1 - p_m$ and $p_m/2$ were similarly set in order to make the possible microtiming transitions equally likely.

Since m_t^i is given as a percentage with respect to the inter-beat-interval (IBI), the resolution with which microtiming can be estimated in the model is also percentual, and it is given by the relation between the sampling rate sr of the features and the bpm: $\mathrm{res} = \frac{\mathrm{bpm}}{60\mathrm{sr}}$. It has been shown in the literature that a resolution of 0.02 of the IBI is sufficient for representing microtiming deviations [125, 64]. To keep computational complexity low but at the same time guarantee a resolution $\mathrm{res} = 0.02$, we use observation features sampled at 110 Hz and we study pieces whose tempo is within a range of 120 to 135 bpm. Note that these assumptions are valid in the music under study, and they could be adapted to a different music genre, e.g. increasing sampling rate to increase the bpm interval.

Observation Potential ϕ

The observation potential depends on the beat and onset likelihoods, the frame counter f_t and the microtiming m_t:

$$\phi(f_t, \mathbf{m}_t, x_t) := \begin{cases} b_t & \text{if } f_t = 1 \\ o_t - b_t & \text{if } \frac{f_t}{l_t} \in \mathbf{m}_t \\ 1 - o_t & \text{otherwise,} \end{cases}$$

where b_t and o_t are beat and onset likelihoods, respectively. The onset likelihood was estimated using the ensemble of recurrent neural networks for onset activation estimation from *madmom* [12]—we refer the interested reader to [56, 11] for further information. We designed a simple DNN for the beat likelihood estimation and trained it on *candombe* and *samba*.[5] It consists of 6 layers, namely: batch normalization, dropout of 0.4, bidirectional gated recurrent unit (Bi-GRU) [27] with 128 units, batch normalization, another identical Bi-GRU layer, and a dense layer with two units and a softmax activation.

We use a mel-spectrogram as input feature for the DNN. The short–time Fourier transform is computed using a window length of 2048 samples and a hop of 401 samples, to ensure a sampling rate of 110 Hz with audio sampled at 44.1 kHz. We use 80 mel filters, comprising a frequency range from 30 Hz to 17 kHz.

9.4 Datasets

In our experiments we use a subset of the *candombe* dataset [150] and the BRID dataset (see Section 5.3) of Brazilian *samba*, explained in the following.

Candombe dataset

It comprises audio recordings of Uruguayan *candombe* drumming performances in which ensembles of three to five musicians play the three different *candombe* drums: *chico*, *piano* and *repique*. It has separated stems of the different drums, which facilitates the microtiming analysis. We focus our study on the *chico* drum, which is the timekeeper of the ensemble. We select a subset of the recordings in the dataset, in which the *chico* drum plays a beat-length pattern of four sixteenth notes, for a total of 1788 beats and 7152 onsets.

BRID dataset

It consists of both solo and ensemble performances of Brazilian *samba*, comprising ten different instrument classes: *agogô*, *caixa* (snare drum), *cuíca*, *pandeiro* (frame drum), *reco-reco*, *repique*, *shaker*, *surdo*, *tamborim* and *tantã*. We focus our study on the *tamborim*, which is one of the timekeepers of the ensemble. We select a subset of the solo tracks, in which the *tamborim* plays a beat–length rhythmic pattern of four sixteenth notes (shown in music notation in Figure 9.3), for a total of 396 beats and 1584 onsets.

[5] The training proved necessary because the timekeeper pattern of *candombe* rhythm has a distinctive accent displaced with respect to the beat that misleads beat–tracking models trained on "Western" music [147].

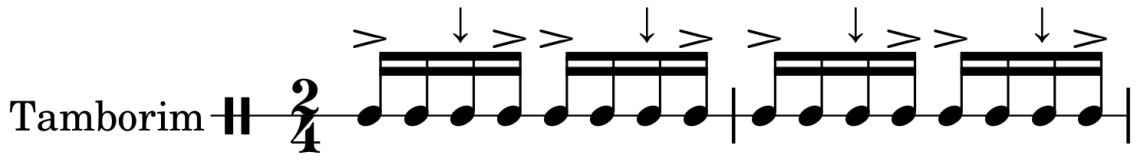


Figure 9.3 Example of the beat-length rhythmic pattern of the *tamborim* from the *samba* dataset in music notation. The symbol '$>$' refers to an accent, and '$\downarrow$' implies to turn the *tamborim* upside down and execute the strike backwards.

9.4.1 Ground-Truth Generation

The microtiming ground-truth is inferred following the approach of [44], in which the onsets are used to derive the swing-ratio annotations. Analogously, we compute the microtiming ground-truth using the annotated onsets, obtaining one value of $\mathbf{m} = (m_1, m_2, m_3)$ for each beat. In order to mitigate the effect of onset annotation errors and sensori-motor noise, we use a moving-median filter to smooth the microtiming ground-truth, with a centered rectangular window of length 21 beats, as shown in Figure 9.4.

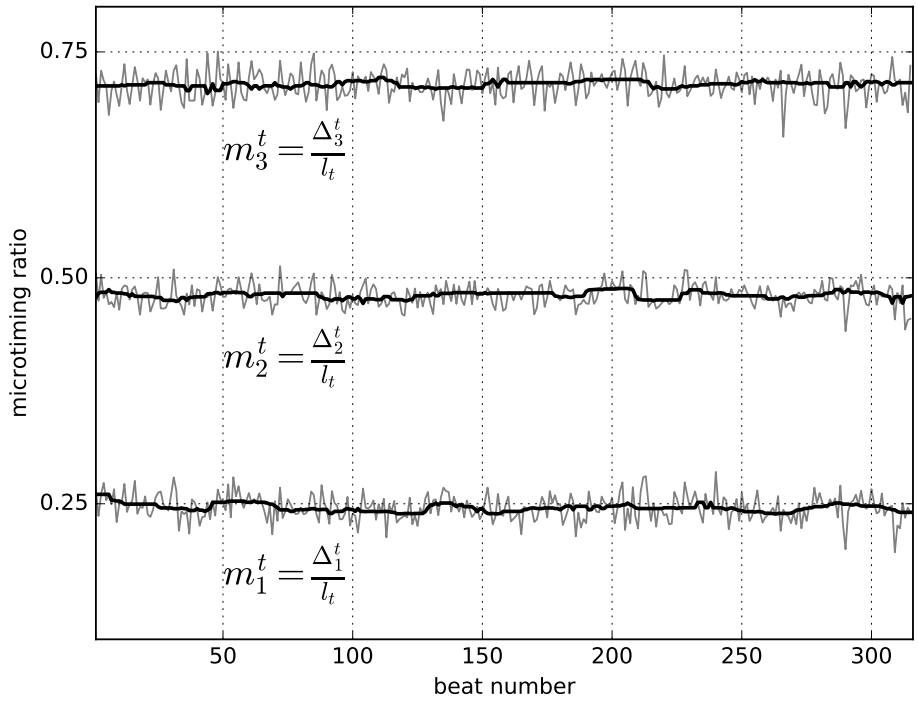


Figure 9.4 Example of the microtiming values for a *chico* drum recording in the *candombe* dataset. Dark and light lines represent the ground-truth with and without median filtering, respectively.

9.5 Evaluation setup

We investigate the performance of the model and whether the microtiming descriptor is useful for analysing the music at hand. To this end, we scale the *candombe* dataset to match the size of the *samba* dataset at test time by selecting excerpts in each track. We assess the model's performance using manually annotated onsets and beats, from which we derive our ground-truth as explained in Section 9.4.1. To evaluate if the microtiming estimation affects the beat tracking, we compare the model's performance with a simplified version of it that only tracks beats. This version has only variables f_t and l_t (see Figure 9.2); the same potential ψ_f is used, and the observation potential is simply b_t (the beat likelihood) at beat positions and $1 - b_t$ otherwise. We assess the microtiming estimation by: varying the p_m microtiming transition parameter—allowing smooth changes within the piece or no changes at all; and varying the tolerance used on the F-measure (F1) score. Finally, we discuss our main findings on the potential of jointly tracking beats and microtiming.

Implementation, Training and Evaluation Metrics

Similarly to previous chapters, the DNN beat likelihood model is implemented in Keras 2.2.4 and Tensorflow 1.13.1 [30, 1]. We use the ADAM optimizer [89] with default parameters. Training is stopped after 10 epochs without improvement in the validation loss, to a maximum of 100 epochs. We train the network with patches of 500 frames and a batch size of 64, leaving one track out and training with the rest, which we split in 30% and 70% for validation and training respectively, among the same genre. The onset activation was obtained with *madmom* version 0.16.1 [12], and the mel-spectra was computed using *librosa* 0.6.3 [118].

We evaluate the beat tracking model using the F1 score for beat tracking with a tolerance window of 70 ms, as implemented in *mir_eval* 0.5 [145]. To evaluate the microtiming estimation, we first select the correctly estimated beats, then compute F1 for each estimated m_t^i with $1 \leq i \leq 3$ and tolerance windows of different lengths, and the overall score as the mean F1 ($\mathrm{F1}_{\mathbf{m}_t} = \sum_i \mathrm{F1}_{m_t^i}/3$).

9.6 Results and discussion

The results on the microtiming tracking are depicted in Figure 9.5, which shows the F1 scores as a function of the tolerance. The different colors represent the different p_m values. We evaluate the model for the set of values $p_m = \{0, 0.001, 0.06\}$, that is no, very unlikely and more likely microtiming changes respectively. Those values were obtained from statistics on the data in preliminary experiments. We searched for microtiming ratios within the interval $[0.25, 0.29] \times [0.42, 0.5] \times [0.67, 0.75]$, for microtiming dimensions $i = 1, 2, 3$ respectively.

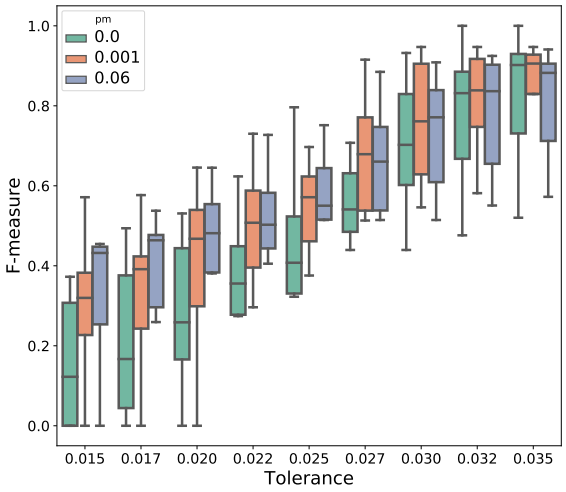


Figure 9.5 Mean microtiming F-measure score on the two datasets.

This corresponds to $\boldsymbol{\delta}_L = (0, -0.03, -0.08)$ and $\boldsymbol{\delta}_U = (0.04, 0, 0)$.[6] As illustrated in Figure 9.5, we found that the restriction of a constant microtiming profile ($p_m = 0$) along the piece relates to a worse performance, especially with small tolerances. We hypothesize that this occurs because in some *samba* excerpts the microtiming ratio changes are percentually bigger than those tolerances, leading to an inaccurate estimation. From considering the dependency of the F1 score with respect to the tolerance, we observe that is possible to achieve a reasonable F1 score from 0.025 on. The results with the best compromise in terms of variance and median are achieved with $p_m = 0.001$, which aligns with the hypothesis that microtiming profiles change very smoothly over time. We explored different tolerances since we are working with frames which are noisy, and the comparison with the smoothed ground truth still makes sense with large tolerances.

We found that the beat tracking performance of the model reaches a 95.7% F1 score, being equivalent to the beat tracking only version. The high F1 score in beat tracking is not surprising given that the DNN was trained using data of the same nature (acoustic conditions and genre) and the sets are homogeneous. As mentioned before, state-of-the-art beat tracking systems based on DNNs fail dramatically in this specific scenario [147], particularly tracking the beats in time-keeper instruments in *candombe*, because the data is too different from what was used in their training. We do not consider this as a challenging beat tracking case, but a training stage was needed to perform adequately.

During our experiments we observed that the microtiming descriptor $\mathbf{m}_t$ could be used to help beat tracking in some cases. Informing the microtiming profile a priori, by setting δ_L^i and δ_U^i, can disambiguate beat positions by helping the joint inference. This could allow

[6] Symmetric windows around the isochronus sixteenth note positions were used for the microtiming ratios in preliminary experiments with no gain in tracking performance and a higher computational burden.

to apply non pre-trained beat tracking models to *candombe* recordings, which usually fail in estimating the beat location by displacing it one sixteenth note (due to an accent in the rhythmic pattern). Aligned to that idea, the model could be useful in scenarios where onsets from other instruments are present. Besides, when the beat tracking is incorrect, the obtained microtiming profile can be descriptive of the type of mistake that occurs by contrasting the obtained profile with the expected one. The same case mentioned before—a lag of a sixteenth note in the beat estimation—shows in the microtiming estimation as unexpected forward positions in the second and third sixteenth notes, with a synchronous fourth one, which is the *candombe* microtiming profile lagged by a sixteenth note position.

Microtiming description and insights

Figure 9.4 illustrates an example of microtiming profile for an excerpt of the *candombe* dataset. This example shows the microtiming variations per beat interval along the complete recording. In the performance of the example, the rhythmic pattern is played with the same microtiming profile in the whole track. This microtiming template is characteristic of some patterns of *candombe* drumming [147], and it is present in several recordings of the dataset. We noticed that microtiming profiles do not present significant variations within tempo changes in the *candombe* dataset. However, the presented method can be used to characterize curves of microtiming vs. tempo that could be informative of musical phenomena for other music genres or datasets.

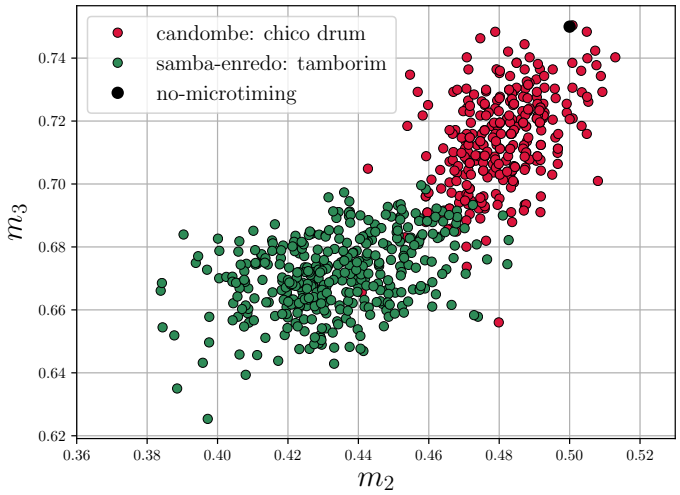


Figure 9.6 Microtiming distribution depending on style, view of the plane $m_t^2 m_t^3$, denoted as $m_2 m_3$ for simplicity. A dot at $(0.50, 0.75)$ indicates the expected position of a beat that shows no microtiming, that is, where all onsets are evenly spaced.

As shown in Figures 9.6 and 9.7, the microtiming descriptor $\mathbf{m}_t$ encodes musical features that are informative about the music genre, instrument type or performer. These two figures

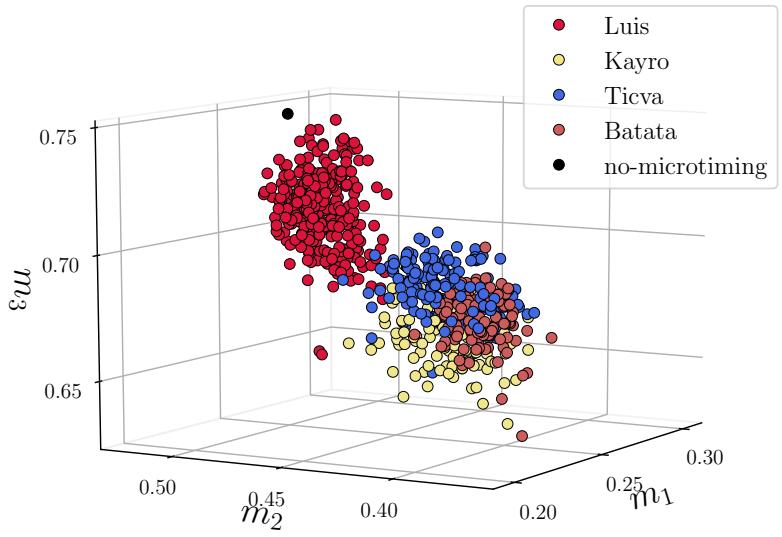


Figure 9.7 Microtiming distribution depending on performer (top musician plays *candombe* and the others play *samba*). A dot at $(0.25, 0.50, 0.75)$ indicates the point of no microtiming.

show the microtiming profile for all beats from *tamborim* and *chico* recordings, using the annotations for better visualization. Firstly, by observing the 'no-microtiming' reference in the figures that corresponds to $\mathbf{m}_t = (0.25, 0.50, 0.75)$, it becomes clear that both *samba* and *candombe* present considerable microtiming deviations in their time-keeper instruments. Even though the rhythmic patterns from both instruments present deviations that tend to compress the IOI in a similar manner, the microtiming profile differs for each music style, being more drastic in the case of the *tamborim*. This analysis should be extended to other *samba* instruments in order to determine if differences are due to the rhythmic pattern of a particular instrument; or if different patterns within the same genre tend to follow the same microtiming profile (characteristic of the genre). Figure 9.7 shows the microtiming profiles of each performer. It is quite clear that performers tend to be consistent with their microtiming, opening the perspective of studying microtiming profiles for performer characterization, as was done for jazz [44].

9.7 Conclusions and future work

In this chapter, we introduced a common framework for the representation and automatic tracking of two music objects—beats and microtiming deviations—with the objective of taking further steps towards more holistic models for the analysis of rhythm. In particular, we addressed two Afro-Latin American music traditions—Uruguayan *candombe* and Brazilian *samba*—that represent an interesting case of study given the particularities of their rhythm

which exhibit characteristic microtiming deviations. We introduced for the first time a CRF language model that exploits beat and onset likelihoods as observations, and we framed the retrieval of beats and microtiming deviations as a sequence labelling problem. We focused our study on beat-length rhythmic patterns of timekeeper instruments, with four articulated sixteenth notes. The results obtained with our method using a ground-truth derived from annotated onsets illustrates the potential of more informative methods for the automatic study of these rhythms.

The information that this framework conveys allows for richer analysis about the musical content of audio signals, and opens other perspectives for the computational analysis of rhythm in these musics: can we retrieve useful microtiming information to characterize *samba* and *candombe* performances? Is this information useful for music generation? Or for the computational assistance of music students? Besides, other interesting research questions are also open: how could our model be adapted to describe the microtiming profile depending on the nature of the rhythmic pattern being played? i.e., whether they articulate 2, 3, 4 or more notes; Or, how useful is our model in challenging scenarios in comparison to heuristic methods? As a general remark, considering multiple musical dimensions in the automatic study of rich, complex rhythms allows for more interesting, complex questions. In this matter, we hope this work contributes in diversifying the research problems usually addressed in the MIR community.

Chapter 10

Conclusions and Future work

Summary

This concluding chapter is dedicated to summarize the work carried out in this dissertation, along with its contributions, the main results obtained and conclusions. The chapter concludes with a discussion on future perspectives and research directions, and the results of the ideas explored in this thesis.

In this dissertation we have addressed the problem of building computational models for the analysis of rhythm in music audio signals. The originality of the methods presented here is that they exploit multi-scale interactions among semantic and temporal dimensions, providing a framework for studying the interactions between various musical attributes in a flexible manner. Our objective was to show that incorporating information from multiple temporal and semantic levels may lead to more coherent estimations and more informative models, and even improve the performance of the systems in some cases. To that end, we developed models for downbeat tracking that learn representations at multiple temporal scales and incorporate segment repetition information, and a model that estimates beats and microtiming profiles jointly. We have shown that integrating knowledge from interrelated musical dependencies leads to more coherent, informative and robust systems.

Part of the work performed in this PhD project was devoted to the creation of tools and annotated data for the MIR community, resulting in two datasets of Brazilian Afro-rooted Latin American music for computational rhythm analysis and a set of software tools for datasets usability and reproducibility.

The interactions between different music attributes are countless, and building models that handle such interrelated structure as humans do is a very challenging task, in which a lot remains to be done. Our hope is that the ideas and models presented in this dissertation are a step towards this research direction. We present in the following the main contributions and results, as well as our perspectives for future work.

10.1 Summary of contributions

Identification of multi-scale schemes for rhythmic analysis

Along this dissertation we offered a discussion on automatic musical rhythm analysis from a multi-scale perspective, mentioning challenges and limitations of current methods for the automatic analysis of rhythm. We argued that by extending current 'well established' MIR tasks to account for other intrinsically related music phenomena, we could think of new questions and challenges that enrich the research problem definitions in the context of computational music research. We formulated and addressed two problems following these ideas: the interaction between the coarser and pivot scales, music segments and downbeats; and the interaction between one of the fundamental scales and one of the finest, microtiming deviations and beats.

Analysis of common variations

We first presented a systematic study of common variations in the design of downbeat tracking systems based on deep neural networks in a rigorous manner. We conducted several experiments accounting for different design choices regarding the temporal granularity of the input representations, the output encoding used to train the neural networks and the impact of the post-processing stage. We explored these distinct schemes in 8 datasets covering various music genres, namely rock, pop, ballroom dances and jazz. This systematic investigation was carefully carried out considering the same training data, input features and evaluation setup.

In our experiments, we found that the choice of the inputs' temporal granularity has a significant impact on performance; the post-processing effect depends on the dataset and the addition of a densely structured output encoding does not help in the training of downbeat tracking systems in comparison to using a classical sparse encoding. The main contribution of the presented analysis is that it allowed us to draw conclusions about the different decisions in the systems' pipeline that would be difficult to obtain in other conditions, e.g. where training data or the evaluation scheme are different.

Downbeat tracking by learning at multiple temporal scales

We proposed a novel CRNN architecture, introduced for the first time for downbeat tracking. We compared the proposed CRNN architecture modular-wise to a state-of-the-art RNN architecture, and concluded that it performs as the state-of-the-art. Moreover, we showed that a system based on CRNNs is more robust in a challenging scenario where the test data—in this case a dataset of jazz recordings— is a strongly under-represented genre in the training set. An originality of the proposed architecture is that it exploits information from multiple

temporal scales within the same deep learning model, which is useful to capture complementary information that improves the robustness of downbeat estimations.

Structure-informed downbeat tracking system

We further extended our proposed system based on CRNNs by including music structure information in the post-processing stage. To that end, we proposed an SCCRF language model which exploits music structure information in a unified and flexible manner. This model incorporates information about music segment repetitions with suitable potentials, that were designed to account for local and distant temporal interrelationships. In our experiments, we have shown that taking into account information from music segment repetitions in the post-processing stage improves the downbeat estimation over the popular BPM in controlled conditions, by providing consistency among occurrences of the same section and accounting for rare music variations.

Moreover, we discussed different examples showing promising perspectives especially in challenging cases with rare music variations such as fast time signature changes, where models like the BPM have limited performance due to the limited contextual information they can handle. An additional value of the proposed method is its flexibility, which allows it to be adapted to other tasks: it can be directly applied to beat tracking, and easily extended to the joint tracking of beats and downbeats.

Common framework for beat and microtiming estimation

We proposed a common framework for the representation and automatic tracking of beats and microtiming deviations. We first focused on the definition of commonalities for the analysis of microtiming deviations in two Afro-Latin American music genres—in particular Brazilian samba and Uruguayan candombe—. In these two Afro-rooted music genres the small-scale deviations represent a key component of their rich rhythmic structure. We draw this analogy by focusing on time-keeper instruments of both genres—chico and tamborim—, which have common beat-length rhythmic pattern structures. We then formulated the problem of automatically analysing microtiming deviations and beats as a sequence labelling problem, and we introduced for the first time a LCCRF model that exploits beat and onset likelihoods as observations for the retrieval of beats and microtiming profiles.

The findings obtained with our method, while preliminary, illustrate the potential of more informative methods for the automatic study of these rhythms. Our analysis revealed interesting applications of these holistic approaches that consider both music attributes, in particular for the identification of styles within commonly rooted genres and the distinction of performers' styles within the same genre.

Contributions to the creation of datasets and common tools

Part of the work carried out in this dissertation was devoted to conceptualizing and developing datasets and tools for the use of the MIR community. The goal we pursued in this matter was two-fold: 1) to increase the diversity within the available datasets for computational rhythm analysis; 2) to develop software tools to improve reproducibility and facilitate the usability of common datasets. For this, we contributed in the generation of comprehensive collections of Brazilian popular genres—*samba*, *samba de enredo*, *partido alto*, *capoeira*, among others—, some of them available for computational rhythm analysis for the first time. Two datasets of Brazilian music for computational rhythm analysis were built: BRID—consisting of solos and mixtures of percussion instruments and various Brazilian music styles—, and Sambaset—consisting of over 40 hours of historical *samba de enredo* recordings—. Second, we developed *mirdata*, a software package for common loaders for MIR datasets.This package facilitates a set of tools for the easy usability of annotated data, while checking consistency of annotations and audio data from canonical sources, to guarantee a common framework in the usability of datasets which is indispensable in the context of MIR.

10.2 Future work

The work introduced in this dissertation took a few steps towards more holistic multi-scale computational models for the analysis of musical rhythm. Nevertheless, there are many issues and many potential areas of improvement in the methods presented here. Besides, from the analysis of some of the ideas presented in this work, it is clear that we have only scratched the surface of many interesting problems related to the computational study of rhythm under this perspective. We present in the following some points we wish to address in future work as well as more general directions for future research.

Concerning the analysis of common variations in downbeat tracking systems of Chapter 7, the proposed structured encoding is an interesting idea that could be further explored. In particular, the perspective of including such an encoding—or a similar one—within a multi-task DNN for beat and downbeat tracking seems promising, especially with a frame-wise grid, where beat and downbeat events are very sparse. Besides, we currently use a context of 15 beats for the learning of our downbeat tracking system based on CRNNs. This context showed to be a good compromise in preliminary experiments, but the effect of a longer context should be assessed.

With regards to the SCCRF model introduced in Chapter 8, the results obtained are encouraging but further improvements are needed. The experiments carried out relied on controlled conditions: annotated beats and structure information. The first assumption could be relaxed by using available beat tracking methods to estimate the beats. At the present time, the developed system infers the structure of the skip-chain graph from the annotations,

which could be addressed differently and in a data-driven fashion. A possible alternative for this is to compute self-similarity matrices from the audio signal and design the potentials to weight the connections accordingly in a fully-connected graph. The disadvantage of such an approach is that the inference on a fully-connected graph might be computationally expensive, however, since the model is designed to deal with the beat level, it should not be prohibitively expensive and it will allow for extending this method to data without structure annotations. Besides, the self-similarity matrix could include information about rhythmic patterns as intermediate temporal scales between bars and sections which might also increase the robustness of downbeat estimations.

Concerning the model for the estimation of beats and microtiming of Chapter 9, there are several open questions that still have to be answered. The method is currently evaluated on a few examples that we annotated manually, but it should be evaluated on more data and compared with a baseline based on heuristics, to properly assess its potential and weaknesses, which we only saw at glance from our experiments. As it is right now, the model's complexity increases considerably when dealing with a big set of possible tempos and microtiming profiles. This should be addressed to ensure the scalability and usability of the method. A possible fix is to follow previous approaches and efficiently sample the state space [98] plus to use inexact inference [165]. Once these complexity limitations are addressed, the model should be extended to account for different rhythmic patterns, becoming more general and applicable to other music genres. We plan to address these limitations in future work along with a more detailed study of the presented framework's usability to characterize *samba* and *candombe* performances.

As a general remark, considering multiple temporal and hierarchical levels in the automatic study of rhythm raises interesting research questions. Along this line, we consider that there is still a huge amount of work to be done in diversifying the research problems we usually address in the MIR community, and the work of this dissertation only took a few steps towards new problem formulations. Fortunately, recent new resources and methods are contributing to the possibility of expanding the research goals within computational rhythm analysis, and further exploring the ideas presented here. A few examples are the recently proposed variant of CNNs called *dilated convolutional networks* (DCNs) [185], or the *Harmonix Set* [128]. DCNs are natural candidates to explore the interplay between temporal scales in music, given their capacity to sample from different temporal resolutions. Furthermore, the recently created *Harmonix Set* consists of an extensive collection containing annotations of beats, downbeats and music sections in the same set, which is a great opportunity for the community to propose new multi-scale approaches. On our side, we wish to further contribute to this fascinating research direction in the future.

Bibliography

[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

[2] Adavanne, S., Pertilä, P., and Virtanen, T. (2017). Sound event detection using spatial features and convolutional recurrent neural network. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[3] Araujo Junior, S. M. (1992). *Acoustic labor in the timing of everyday life: a critical contribution to the history of samba in Rio de Janeiro*. PhD thesis, University of Illinois.

[4] Arom, S. (1984). Structuration du temps dans les musiques d'afrique centrale: périodicité, mètre, rythmique et polyrythmie. *Revue de Musicologie*, 70(1):5–36.

[5] Bello, J. P., Daudet, L., Abdallah, S., Duxbury, C., Davies, M., and Sandler, M. B. (2005). A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13(5):1035–1047.

[6] Bello, J. P., Rowe, R., Guedes, C., and Toussaint, G. (2015). Five perspectives on musical rhythm. *Journal of New Music Research*, 44(1):1–2.

[7] Bertin-Mahieux, T., Ellis, D. P., Whitman, B., and Lamere, P. (2011). The million song dataset. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 591–596.

[8] Bilmes, J. A. (1993). *Timing is of the essence: Perceptual and computational techniques for representing, learning, and reproducing expressive timing in percussive rhythm*. Master's thesis, Massachusetts Institute of Technology, Cambridge, USA.

[9] Bittner, R. M., McFee, B., Salamon, J., Li, P., and Bello, J. P. (2017). Deep salience representations for f0 estimation in polyphonic music. In *International Society for Music Information Retrieval Conference*, ISMIR.

[10] Bittner, R. M., Salamon, J., Tierney, M., Mauch, M., Cannam, C., and Bello, J. P. (2014). MedleyDB: A multitrack dataset for annotation-intensive MIR research. In *International Society for Music Information Retrieval Conference*, ISMIR.

[11] Böck, S., Arzt, A., Krebs, F., and Schedl, M. (2012). Online real-time onset detection with recurrent neural networks. In *15th Int. Conf. on Digital Audio Effects (DAFx)*, York, UK.

[12] Böck, S., Korzeniowski, F., Schlüter, J., Krebs, F., and Widmer, G. (2016). madmom: a new python audio and music signal processing library. In *Proceedings of the 24th ACM International Conference on Multimedia*, ACMMM.

[13] Böck, S., Krebs, F., and Widmer, G. (2014). A Multi-model Approach to Beat Tracking Considering Heterogeneous Music Styles. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 603–608, Taipei, Taiwan.

[14] Böck, S., Krebs, F., and Widmer, G. (2015). Accurate tempo estimation based on recurrent neural networks and resonating comb filters. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 625–631, Málaga, Spain.

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

[16] Böck, S. and Schedl, M. (2011). Enhanced beat tracking with context-aware neural networks. In *Proc. Int. Conf. Digital Audio Effects*, pages 135–139.

[17] Bogdanov, D., Wack, N., Gómez Gutiérrez, E., Gulati, S., Herrera Boyer, P., Mayor, O., Roma Trepat, G., Salamon, J., Zapata González, J. R., and Serra, X. (2013). Essentia: An audio analysis library for music information retrieval. In *International Society for Music Information Retrieval Conference*, ISMIR.

[18] Bosch, J. J., Marxer, R., and Gómez, E. (2016). Evaluation and combination of pitch estimation methods for melody extraction in symphonic classical music. *Journal of New Music Research*, 45(2):101–117.

[19] Bottou, L. (1991). Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, 91(8):12.

[20] Brossier, P. M. (2006). *Automatic Annotation of Musical Audio for Interactive Applications*. PhD thesis, Department of Electronic Engineering, Queen Mary University of London.

[21] Cannam, C., Jewell, M. O., Rhodes, C., Sandler, M., , and d'Inverno, M. (2010a). Linked data and you: Bringing music research software into the semantic web. *Journal of New Music Research*, 39(4):313–325.

[22] Cannam, C., Landone, C., and Sandler, M. (2010b). Sonic Visualiser: An open source application for viewing, analysing, and annotating music audio files. In *ACM Multimedia 2010 Int. Conf.*, pages 1467–1468, Florence, Italy.

[23] Çakır, E., Parascandolo, G., Heittola, T., Huttunen, H., and Virtanen, T. (2017). Convolutional recurrent neural networks for polyphonic sound event detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(6):1291–1303.

[24] Chan, T.-S., Yeh, T.-C., Fan, Z.-C., Chen, H.-W., Su, L., Yang, Y.-H., and Jang, R. (2015). Vocal activity informed singing voice separation with the ikala dataset. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 718–722. IEEE.

[25] Cheng, T., Fukayama, S., and Goto, M. (2018). Convolving Gaussian Kernels for RNN-Based Beat Tracking. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pages 1905–1909. IEEE.

[26] Cho, K., Courville, A., and Bengio, Y. (2015). Describing multimedia content using attention-based encoder-decoder networks. *IEEE Transactions on Multimedia*, 17(11):1875–1886.

[27] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

[28] Cho, T. and Bello, J. P. (2011). A feature smoothing method for chord recognition using recurrence plots. In *International Society for Music Information Retrieval Conference*, ISMIR.

[29] Choi, K., Fazekas, G., Cho, K., and Sandler, M. (2017). A tutorial on deep learning for music information retrieval. *arXiv preprint arXiv:1709.04396*.

[30] Chollet, F. et al. (2015). Keras. https://keras.io.

[31] Cohen-Hadria, A. and Peeters, G. (2017). Music structure boundaries estimation using multiple self-similarity matrices as input depth of convolutional neural networks. In *Audio Engineering Society Conference: 2017 AES International Conference on Semantic Audio*. Audio Engineering Society.

[32] Coughlan, J. (2009). A tutorial introduction to belief propagation. *The Smith-Kettlewell Eye Research Institute*.

[33] Cunha, F. L. (2018). Samba locations: An analysis on the carioca samba, identities, and intangible heritage (Rio de Janeiro, Brazil). In Cunha, F. L., Santos, M., and Rabassa, J., editors, *Latin American Heritage*, The Latin America Studies Book Series, chapter 1, pages 3–20. Springer International Publishing.

[34] Dannenberg, R. B. (2005). Toward automated holistic beat tracking, music analysis and understanding. In *International Society for Music Information Retrieval Conference*, ISMIR.

[35] Davies, M. E., Degara, N., and Plumbley, M. D. (2009). Evaluation methods for musical audio beat tracking algorithms. *Queen Mary University of London, Centre for Digital Music, Tech. Rep. C4DM-TR-09-06*.

[36] Davies, M. E. P. (2007). *Towards automatic rhythmic accompaniment*. PhD thesis, University of London.

[37] Davies, M. E. P., Madison, G., Silva, P., and Gouyon, F. (2013). The effect of microtiming deviations on the perception of groove in short rhythms. *Music Perception*, 30(5):497–510.

[38] Davies, M. E. P. and Plumbley, M. D. (2007). Context-dependent beat tracking of musical audio. *IEEE Transactions on Audio, Speech and Language Processing*, 15(3):1000–1009.

[39] De Clercq, T. and Temperley, D. (2011). A corpus analysis of rock harmony. *Popular Music*, 30(1):47–70.

[40] de Tomaz Júnior, P. D. (2016). Separação automática de instrumentos de percussão brasileira a partir de mistura pré-gravada. Master's thesis, Federal University of Amazonas, Manaus, Brazil.

[41] Degara, N., Rúa, E. A., Pena, A., Torres-Guijarro, S., Davies, M. E., and Plumbley, M. D. (2012). Reliability-informed beat tracking of musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):290–301.

[42] Deutsch, D. and Feroe, J. (1981). The internal representation of pitch sequences in tonal music. *Psychological review*, 88(6):503.

[43] Di Giorgi, B., Zanoni, M., Sarti, A., and Tubaro, S. (2013). Automatic chord recognition based on the probabilistic modeling of diatonic modal harmony. In *nDS'13; Proceedings of the 8th International Workshop on Multidimensional Systems*, pages 1–6. VDE.

[44] Dittmar, C., Pfleiderer, M., Balke, S., and Müller, M. (2018). A swingogram representation for tracking micro-rhythmic variation in jazz performances. *Journal of New Music Research*, 47(2):97–113.

[45] Dittmar, C., Pfleiderer, M., and Müller, M. (2015). Automated estimation of ride cymbal swing ratios in jazz recordings. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 271–277, Málaga, Spain.

[46] Dixon, S. (2006). Onset detection revisited. In *9th Int. Conf. on Digital Audio Effects (DAFx-06)*, pages 133–137, Montreal, Canada.

[47] Dixon, S. (2007). Evaluation of the audio beat tracking system BeatRoot. *Journal of New Music Research*, 36(1):39–50.

[48] Dixon, S., Gouyon, F., Widmer, G., et al. (2004). Towards characterisation of music via rhythmic patterns. In *International Society for Music Information Retrieval Conference*, ISMIR.

[49] Downie, J. S. (2003). Music information retrieval. *Annual review of information science and technology*, 37(1):295–340.

[50] Durand, S., Bello, J. P., David, B., and Richard, G. (2015). Downbeat tracking with multiple features and deep neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[51] Durand, S., Bello, J. P., David, B., and Richard, G. (2016). Feature adapted convolutional neural networks for downbeat tracking. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*.

[52] Durand, S., Bello, J. P., David, B., and Richard, G. (2017). Robust Downbeat Tracking Using an Ensemble of Convolutional Networks. *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, 25(1):76–89.

[53] Durand, S., David, B., and Richard, G. (2014). Enhancing downbeat detection when facing different music styles. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3132–3136. IEEE.

[54] Durand, S. and Essid, S. (2016). Downbeat Detection With Conditional Random Fields And Deep Learned Features. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 386–392, New York, USA.

[55] Ellis, D. P. W. (2007). Beat tracking by dynamic programming. *Journal of New Music Research*, 36(1):51–60.

[56] Eyben, F., Böck, S., Schuller, B., and Graves, A. (2010). Universal onset detection with bidirectional long-short term memory neural networks. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 589–594, Utrecht, The Netherlands.

[57] Fillon, T., Joder, C., Durand, S., and Essid, S. (2015). A conditional random field system for beat tracking. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 424–428, South Brisbane, Australia.

[58] Garcia, S. and Herrera, F. (2008). An extension on"statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research*, 9:2677–2694.

[59] Gemmeke, J. F., Ellis, D. P., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., and Ritter, M. (2017). Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780. IEEE.

[60] Gonçalves, G. and Costa, O. (2000). *The Carioca Groove: The Rio de Janeiro's Samba Schools Drum Sections*. Groove, Rio de Janeiro, Brazil.

[61] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

[62] Goto, M., Hashiguchi, H., Nishimura, T., and Oka, R. (2002). RWC Music Database: Popular, Classical and Jazz Music Databases. In *International Society for Music Information Retrieval Conference*, volume 2 of *ISMIR*, pages 287–288.

[63] Goto, M., Hashiguchi, H., Nishimura, T., and Oka, R. (2003). RWC music database: Music genre database and musical instrument sound database. In *International Society for Music Information Retrieval Conference*, ISMIR.

[64] Gouyon, F. (2007). Microtiming in 'Samba de Roda' — preliminary experiments with polyphonic audio. In *11th Brazilian Symposium on Computer Music (SBCM)*, pages 197–203, São Paulo, Brazil.

[65] Gouyon, F., Klapuri, A., Dixon, S., Alonso, M., Tzanetakis, G., Uhle, C., and Cano, P. (2006). An experimental comparison of audio tempo induction algorithms. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1832–1844.

[66] Greaves-Tunnell, A. and Harchaoui, Z. (2019). A statistical investigation of long memory in language and music. *arXiv preprint arXiv:1904.03834*.

[67] Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. (2016). LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232.

[68] Grill, T. and Schlüter, J. (2015). Music boundary detection using neural networks on combined features and two-level annotations. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 531–537.

[69] Hainsworth, S. W. and Macleod, M. D. (2004). Particle filtering applied to musical tempo tracking. *EURASIP Journal on Advances in Signal Processing*, 2004(15):927847.

[70] Hamanaka, M., Hirata, K., and Tojo, S. (2014). Musical Structural Analysis Database Based on GTTM. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 325–330, Taipei, Taiwan.

[71] Harte, C. (2010). *Towards automatic extraction of harmony information from music signals*. PhD thesis, QMUL.

[72] He, X., Zemel, R. S., and Carreira-Perpiñán, M. Á. (2004). Multiscale conditional random fields for image labeling. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–II. IEEE.

[73] Hennig, H., Fleischmann, R., Fredebohm, A., Hagmayer, Y., Nagler, J., Witt, A., Theis, F. J., and Geisel, T. (2011). The nature and perception of fluctuations in human musical rhythms. *PloS one*, 6(10):e26457.

[74] Hertzman, M. A. (2009). A Brazilian counterweight: Music, intellectual property and the African diaspora in Rio de Janeiro (1910s–1930s). *Journal of Latin American Studies*, 41(4):695–722.

[75] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

[76] Hockman, J., Davies, M. E., and Fujinaga, I. (2012). One in the jungle: Downbeat detection in hardcore, jungle, and drum and bass. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 169–174.

[77] Holzapfel, A., Davies, M. E. P., Zapata, J. R., Oliveira, J. L., and Gouyon, F. (2012). Selective sampling for beat tracking evaluation. *IEEE Transactions on Audio, Speech and Language Processing*, 20(9):2539–2548.

[78] Holzapfel, A. and Grill, T. (2016). Bayesian meter tracking on learned signal representations. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 262–268.

[79] Holzapfel, A., Krebs, F., and Srinivasamurthy, A. (2014). Tracking the 'odd': Meter inference in a culturally diverse music corpus. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 425–430, Taipei, Taiwan.

[80] Humphrey, E. J., Salamon, J., Nieto, O., Forsyth, J., Bittner, R. M., and Bello, J. P. (2014). JAMS: A JSON Annotated Music Specification for Reproducible MIR Research. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 591–596.

[81] Ioffe, S. and Szegedy, C. (2015). Batch normalization: accelerating deep network training by reducing internal covariate shift. In *32nd International Conference on Machine Learning (ICML)*.

[82] IPHAN (2014). *Matrizes do Samba no Rio de Janeiro: partido-alto, samba de terreiro, samba-enredo*. National Institute of Historic and Artistic Heritage (IPHAN), Brasília, Brazil. IPHAN dossier 10.

[83] Iyer, V. (2002). Embodied mind, situated cognition, and expressive microtiming in African-American music. *Music Perception*, 19(3):387–414.

[84] Jia, B., Lv, J., and Liu, D. (2019). Deep learning-based automatic downbeat tracking: a brief review. *Multimedia Systems*, pages 1–22.

[85] Joder, C., Essid, S., and Richard, G. (2011). A conditional random field framework for robust and scalable audio-to-score matching. *IEEE Transactions on Audio, Speech, and Language Processing (TASLP)*, 19(8):2385–2397.

[86] Jozefowicz, R., Zaremba, W., and Sutskever, I. (2015). An empirical exploration of recurrent network architectures. In *International Conference on Machine Learning*, pages 2342–2350.

[87] Jure, L. and Rocamora, M. (2016). Microtiming in the rhythmic structure of Candombe drumming patterns. In *4th Int. Conf. on Analytical Approaches to World Music (AAWM)*, New York, USA.

[88] Khadkevich, M., Fillon, T., Richard, G., and Omologo, M. (2012). A probabilistic approach to simultaneous extraction of beats and downbeats. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 445–448. IEEE.

[89] Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[90] Klapuri, A. P., Eronen, A. J., and Astola, J. T. (2005). Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):342–355.

[91] Kolinski, M. (1973). A cross-cultural approach to metro-rhythmic patterns. *Ethnomusicology*, 17(3):494–506.

[92] Korzeniowski, F., Böck, S., and Widmer, G. (2014). Probabilistic extraction of beat positions from a beat activation function. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 513–518.

[93] Krebs, F., Böck, S., Dorfer, M., and Widmer, G. (2016). Downbeat tracking using beat synchronous features with recurrent neural networks. In *International Society for Music Information Retrieval Conference*, ISMIR.

[94] Krebs, F., Böck, S., and Widmer, G. (2011). An efficient state space model for joint tempo and meter tracking. In *16th International Society for Music Information Retrieval Conference (ISMIR)*.

[95] Krebs, F., Böck, S., and Widmer, G. (2013a). Rhythmic pattern modeling for beat and downbeat tracking in musical audio. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 227–232.

[96] Krebs, F., Böck, S., and Widmer, G. (2013b). Rhythmic pattern modelling for beat and downbeat tracking from musical audio. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 227–232, Curitiba, Brazil.

[97] Krebs, F., Böck, S., and Widmer, G. (2015a). An efficient state-space model for joint tempo and meter tracking. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 72–78, Málaga, Spain.

[98] Krebs, F., Holzapfel, A., Cemgil, A. T., and Widmer, G. (2015b). Inferring metrical structure in music using particle filters. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(5):817–827.

[99] Kumar, S. and Hebert, M. (2004). Discriminative fields for modeling spatial dependencies in natural images. In *Advances in neural information processing systems*, pages 1531–1538.

[100] Lafferty, J., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. pages 282–289.

[101] Laroche, J. (2001). Estimating tempo, swing and beat locations in audio recordings. In *IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 135–138, New Paltz, USA.

[102] Lindsay, K. A. and Nordquist, P. R. (2007). More than a feeling: Some technical details of swing rhythm in music. *Acoustics Today*, 3(3):31–42.

[103] Liu, J., Huang, M., and Zhu, X. (2010). Recognizing biomedical named entities using skip-chain conditional random fields. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*, pages 10–18. Association for Computational Linguistics.

[104] Lostanlen, V., Cella, C.-E., Bittner, R., and Essid, S. (2018). Medley-solos-DB: a cross-collection dataset for musical instrument recognition.

[105] Maddage, N. C. (2006). Automatic Structure Detection for Popular Music. *IEEE MultiMedia*, 13(1):65–77.

[106] Manilow, E., Seetharaman, P., and Pardo, B. (2018). The Northwestern University Source Separation Library. In *International Society for Music Information Retrieval Conference*, ISMIR.

[107] Marchand, U. and Peeters, G. (2015). Swing ratio estimation. In *18th Int. Conf. on Digital Audio Effects (DAFx)*, pages 423–428, Trondheim, Norway.

[108] Mauch, M., Cannam, C., Davies, M., Dixon, S., Harte, C., Kolozali, S., Tidhar, D., and Sandler, M. (2009a). OMRAS2 metadata project 2009. In *International Society for Music Information Retrieval Conference*, ISMIR.

[109] Mauch, M. and Dixon, S. (2010). Simultaneous estimation of chords and musical context from audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1280–1289.

[110] Mauch, M., Fujihara, H., Yoshii, K., and Goto, M. (2011). Timbre and melody features for the recognition of vocal activity and instrumental solos in polyphonic music. In *International Society for Music Information Retrieval Conference*, ISMIR.

[111] Mauch, M., Noland, K. C., and Dixon, S. (2009b). Using musical structure to enhance automatic chord transcription. In *International Society for Music Information Retrieval Conference*, ISMIR.

[112] McAdams, S. (1989). Psychological constraints on form-bearing dimensions in music. *Contemporary Music Review*, 4(1):181–198.

[113] McFee, B. (2018). *Statistical Methods for Scene and Event Classification*, pages 103–146. Springer International Publishing, Cham.

[114] McFee, B. and Bello, J. P. (2017). Structured training for large-vocabulary chord recognition. In *International Society for Music Information Retrieval Conference*, ISMIR.

[115] McFee, B., Humphrey, E. J., Nieto, O., Salamon, J., Bittner, R., Forsyth, J., and Bello, J. P. (2015a). Pump up the JAMS: V0. 2 and beyond. *Music and Audio Research Laboratory, New York University, Tech. Rep*.

[116] McFee, B., Kim, J. W., Cartwright, M., Salamon, J., Bittner, R. M., and Bello, J. P. (2019). Open-source practices for music signal processing research: Recommendations for transparent, sustainable, and reproducible audio research. *IEEE Signal Processing Magazine*, 36(1):128–137.

[117] McFee, B., Nieto, O., Farbood, M. M., and Bello, J. P. (2017). Evaluating hierarchical structure in music annotations. *Frontiers in psychology*, 8:1337.

[118] McFee, B., Raffel, C., Liang, D., Ellis, D. P. W., McVicar, M., Battenbergk, E., and Nieto, O. (2015b). librosa: Audio and music signal analysis in Python. In *14th Python in Science Conf. (SciPy)*, pages 18–24, Austin, USA.

[119] Meseguer-Brocal, G., Cohen-Hadria, A., and Peeters, G. (2018). Dali: A large dataset of synchronized audio, lyrics and notes, automatically created using teacher-student machine learning paradigm. In *19th International Society for Music Information Retrieval Conference*.

[120] Moehn, F. J. (2005). 'The disc is not the avenue': Schismogenetic mimesis in samba recording. In Green, P. D. and Porcello, T., editors, *Wired for Sound: Engineering and Technologies in Sonic Cultures*, chapter 3, pages 47–83. Wesleyan University Press.

[121] Müller, M. (2015). *Fundamentals of Music Processing*. Springer Verlag, Berlin, Germany.

[122] Müller, M. and Ewert, S. (2011). Chroma toolbox: Matlab implementations for extracting variants of chroma-based audio features. In *International Society for Music Information Retrieval Conference*, ISMIR.

[123] Murphy, K. P. and Russell, S. (2002). *Dynamic Bayesian networks: representation, inference and learning*. PhD thesis.

[124] Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814.

[125] Naveda, L., Gouyon, F., Guedes, C., and Leman, M. (2011). Microtiming patterns and interactions with musical properties in samba music. *Journal of New Music Research*, 40(3):225–238.

[126] Naveda, L., Leman, M., Gouyon, F., and Guedes, C. (2009). Multidimensional microtiming in samba music.

[127] Nieto, O. and Bello, J. P. (2016). Systematic exploration of computational music structure research. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 547–553.

[128] Nieto, O., McCallum, M., Davies, M. E., Robertson, A., Stark, A., and Egozy, E. (2019). The HARMONIX Set: beats, downbeats, and functional segment annotations of Western popular music.

[129] Nunes, L., Rocamora, M., Jure, L., and Biscainho, L. W. P. (2015). Beat and downbeat tracking based on rhythmic patterns applied to the uruguayan candombe drumming. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 264–270, Málaga, Spain.

[130] Oliveira, J. L., Gouyon, F., Martins, L. G., and Reis, L. P. (2010). IBT: A real-time tempo and beat tracking system. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 291–296, Utrecht, The Netherlands.

[131] Orio, N. et al. (2006). Music retrieval: A tutorial and review. *Foundations and Trends® in Information Retrieval*, 1(1):1–90.

[132] Papadopoulos, H. and Peeters, G. (2010). Joint estimation of chords and downbeats from an audio signal. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(1):138–152.

[133] Papadopoulos, H. and Tzanetakis, G. (2013). Exploiting structural relationships in audio music signals using markov logic networks. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*.

[134] Papadopoulos, H. and Tzanetakis, G. (2016). Models for music analysis from a markov logic networks perspective. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(1):19–34.

[135] Parascandolo, G., Huttunen, H., and Virtanen, T. (2016). Recurrent neural networks for polyphonic sound event detection in real life recordings. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6440–6444. IEEE.

[136] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch.

[137] Paulus, J. and Klapuri, A. (2002). Measuring the similarity of rhythmic patterns. In *International Society for Music Information Retrieval Conference*, ISMIR.

[138] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

[139] Peeters, G. and Fort, K. (2012). Towards a (Better) Definition of the Description of Annotated MIR Corpora. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 25–30, Porto, Portugal.

[140] Peeters, G. and Papadopoulos, H. (2010). Simultaneous beat and downbeat-tracking using a probabilistic framework: Theory and large-scale evaluation. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(6):1754–1769.

[141] Polak, R., London, J., and Jacoby, N. (2016). Both isochronous and non-isochronous metrical subdivision afford precise and stable ensemble entrainment: a corpus study of Malian jembe drumming. *Frontiers in neuroscience*, 10:285.

[142] Prado, Y. (2015). Padrões musicais do samba-enredo na era do Sambódromo. *Música em Perspectiva*, 8(1):155–195.

[143] Purwins, H., Li, B., Virtanen, T., Schlüter, J., Chang, S.-Y., and Sainath, T. (2019). Deep learning for audio signal processing. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):206–219.

[144] Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

[145] Raffel, C., McFee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., and Ellis, D. P. W. (2014). mir_eval: A transparent implementation of common MIR metrics. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 367–372, Taipei, Taiwan.

[146] Randel, D. M. (2003). *The Harvard Dictionary of Music*. Belknap Press of Harvard University Press, Cambridge, USA, 4 edition.

[147] Rocamora, M. (2018). *Computational methods for percussion music analysis : The Afro-Uruguayan Candombe drumming as a case study"*. PhD thesis, Universidad de la República (Uruguay). Facultad de Ingeniería. IIE.

[148] Rocamora, M., Jure, L., and Biscainho, L. W. P. (2014). Tools for detection and classification of piano drum patterns from candombe recordings. In *9th Conf. on Interdisciplinary Musicology (CIM14)*, pages 382–387, Berlin, Germany.

[149] Rocamora, M., Jure, L., Marenco, B., Fuentes, M., Lanzaro, F., and Gómez, A. (2015a). An audio-visual database of candombe performances for computational musicological studies. In *IN Proceedings of the International Congress on Science and Music Technology 2015*, CICTeM.

[150] Rocamora, M., Jure, L., Marenco, B., Fuentes, M., Lanzaro, F., and Gómez, A. (2015b). An audio-visual database of Candombe performances for computational musicological studies. In *II Congreso Int. de Ciencia y Tecnología Musical (CICTeM)*, pages 17–24, Buenos Aires, Argentina.

[151] Salamon, J. and Gómez, E. (2012). Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio. Speech. Lang. Processing*, 20(6):1759–1770.

[152] Sandroni, C. (2008). *Feitiço Decente: Transformações do samba no Rio de Janeiro (1917-1933)*. Jorge Zahar Ed./Ed. UFRJ, Rio de Janeiro, Brazil, 2 edition.

[153] Schedl, M., Gómez, E., Urbano, J., et al. (2014). Music information retrieval: Recent developments and applications. *Foundations and Trends® in Information Retrieval*, 8(2-3):127–261.

[154] Schlüter, J. and Böck, S. (2014). Improved musical onset detection with convolutional neural networks. In *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 6979–6983. IEEE.

[155] Schuster, M. and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

[156] Serra, X. (2011). A multicultural approach in music information research. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 151–156, Miami, USA.

[157] Serra, X. (2014). Creating research corpora for the computational study of music: the case of the compmusic project. In *Audio engineering society conference: 53rd international conference: Semantic audio*. Audio Engineering Society.

[158] Settles, B. (2005). Abner: an open source tool for automatically tagging genes, proteins and other entity names in text. *Bioinformatics*, 21(14):3191–3192.

[159] Sha, F. and Pereira, F. (2003). Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 134–141. Association for Computational Linguistics.

[160] Sigtia, S., Benetos, E., and Dixon, S. (2016). An end-to-end neural network for polyphonic piano music transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(5):927–939.

[161] Silla Jr., C. N., Koerich, A. L., and Kaestner, C. A. A. (2008). The Latin music database. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 451–456, Philadelphia, USA.

[162] Sjöberg, J. and Ljung, L. (1995). Overtraining, regularization and searching for a minimum, with application to neural networks. *International Journal of Control*, 62(6):1391–1407.

[163] Smith, J. B. L., Burgoyne, J. A., Fujinaga, I., De Roure, D., and Downie, J. S. (2011). Design and creation of a large-scale database of structural annotations. In *International Society for Music Information Retrieval Conference*, volume 11 of *ISMIR*, pages 555–560. Miami, FL.

[164] Sousa, J. M., Pereira, E. T., and Veloso, L. R. (2016). A robust music genre classification approach for global and regional music datasets evaluation. In *2016 IEEE Int. Conf. on Digital Signal Processing*, pages 109–113, Beijing, China.

[165] Srinivasamurthy, A., Holzapfel, A., Cemgil, A. T., and Serra, X. (2015). Particle Filters for Efficient Meter Tracking with Dynamic Bayesian Networks. In *International Society for Music Information Retrieval Conference*, ISMIR.

[166] Srinivasamurthy, A., Holzapfel, A., Cemgil, A. T., and Serra, X. (2016). A generalized bayesian model for tracking long metrical cycles in acoustic music signals. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 76–80, Shanghai, China.

[167] Srinivasamurthy, A., Holzapfel, A., and Serra, X. (2017). Informed automatic meter analysis of music recordings. In *International Society for Music Information Retrieval Conference*, ISMIR.

[168] Srinivasamurthy, A. and Serra, X. (2014). A supervised approach to hierarchical metrical cycle tracking from audio music recordings. In *ICASSP*, pages 5217–5221. IEEE.

[169] Sutton, C. and McCallum, A. (2006). An introduction to conditional random fields for relational learning. In Getoor, L. and Taskar, B., editors, *Introduction to Statistical Relational Learning*, chapter 4, pages 93–128. MIT Press, Cambridge, USA.

[170] Sutton, C. and McCallum, A. (2012). An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373.

[171] Theano Development Team (2016). Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688.

[172] Tingle, D., Kim, Y. E., and Turnbull, D. (2010). Exploring automatic music annotation with acoustically-objective tags. In *Proceedings of the international conference on Multimedia information retrieval*, pages 55–62. ACM.

[173] Torralba, A. and Efros, A. A. (2011). Unbiased look at dataset bias. In *CVPR*, volume 1, page 7. Citeseer.

[174] Tralie, C. J. and McFee, B. (2019). Enhanced hierarchical music structure annotations via feature level similarity fusion. *arXiv preprint arXiv:1902.01023*.

[175] Ullrich, K., Schlüter, J., and Grill, T. (2014). Boundary detection in music structure analysis using convolutional neural networks. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 417–422.

[176] Vogl, R., Dorfer, M., Widmer, G., and Knees, P. (2017). Drum transcription via joint beat and drum modeling using convolutional recurrent neural networks. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 150–157.

[177] Waadeland, C. H. (2001). "It don't mean a thing if it ain't got that swing"—Simulating expressive timing by modulated movements. *Journal of New Music Research*, 30(1):23–37.

[178] Wallach, H. M. (2004). Conditional random fields: An introduction. *Technical Reports (CIS)*, page 22.

[179] Werbos, P. J. et al. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

[180] White, L., Togneri, R., Liu, W., and Bennamoun, M. (2018). Datadeps.jl: Repeatable data setup for replicable data science. *CoRR*, abs/1808.01091.

[181] Whiteley, N., Cemgil, A. T., and Godsill, S. J. (2006). Bayesian modelling of temporal structure in musical audio. In *International Society for Music Information Retrieval Conference*, ISMIR. Citeseer.

[182] Wright, M. and Berdahl, E. (2006). Towards machine learning of expressive microtiming in Brazilian drumming. In *ICMC*. Citeseer.

[183] Xi, Q., Bittner, R. M., Pauwels, J., Ye, X., and Bello, J. P. (2018). Guitarset: A dataset for guitar transcription. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 453–460.

[184] Yedidia, J. S., Freeman, W. T., and Weiss, Y. (2003). Understanding belief propagation and its generalizations. *Exploring Artificial Intelligence in the New Millennium*, 8:236–239.

[185] Yu, F. and Koltun, V. (2015). Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*.

[186] Zahray, L., Nakamura, E., and Yoshii, K. (2019). Beat and downbeat detection with chord recognition based on multi-task learning of recurrent neural networks.

[187] Zapata, J. R., Davies, M. E., and Gómez, E. (2014). Multi-feature beat tracking. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(4):816–825.

[188] Zapata, J. R., Holzapfel, A., Davies, M. E. P., Oliveira, J. L., and Gouyon, F. (2012). Assigning a confidence threshold on automatic beat annotation in large datasets. In *International Society for Music Information Retrieval Conference*, ISMIR, pages 157–162, Porto, Portugal.

[189] Zhu, C., Zhao, Y., Huang, S., Tu, K., and Ma, Y. (2017). Structured attentions for visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1291–1300.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Analyse Numérique Multi-Échelle du Rythme Musical : un cadre unifié pour les sections, premiers temps, temps et microtiming

Mots clés: Traitement du signal, Extraction d'informations musicales, Apprentissage profond

Résumé: La modélisation computationnelle du rythme a pour objet l'extraction et le traitement d'informations rythmiques à partir d'un signal audio de musique. Cela s'avère être une tâche extrêmement complexe car, pour traiter un enregistrement audio réel, il faut pouvoir gérer sa complexité acoustique et sémantique à plusieurs niveaux de représentation. Les méthodes d'analyse rythmique existantes se concentrent généralement sur l'un de ces aspects à la fois et n'exploitent pas la richesse de la structure musicale, ce qui compromet la cohérence musicale des estimations automatiques.

Dans ce travail, nous proposons de nouvelles approches tirant parti des informations multi-échelles pour l'analyse automatique du rythme. Nos modèles prennent en compte des interdépendances intrinsèques aux signaux audio de musique, en permettant ainsi l'interaction entre différentes échelles de temps et en assurant la cohérence musicale entre elles. En particulier, nous effectuons une analyse systématique des systèmes de l'état de l'art pour la détection des premiers temps, ce qui nous conduit à nous tourner vers des architectures convolutionnelles et récurrentes qui exploitent la modélisation acoustique à court et long terme; nous introduisons un modèle de champ aléatoire conditionnel à saut de chaîne pour la détection des premiers temps. Ce système est conçu pour tirer parti des informations de structure musicale (c'est-à-dire des répétitions de sections musicales) dans un cadre unifié. Nous proposons également un modèle linguistique pour la détection conjointe des temps et du micro-timing dans la musique afro-latino-américaine.

Nos méthodes sont systématiquement évaluées sur diverses bases de données, allant de la musique occidentale à des genres plus spécifiques culturellement, et comparés à des systèmes de l'état de l'art, ainsi qu'à des variantes plus simples. Les résultats globaux montrent que nos modèles d'estimation des premiers temps sont aussi performants que ceux de l'état de l'art, tout en étant plus cohérents sur le plan musical. De plus, notre modèle d'estimation conjointe des temps et du microtiming représente une avancée vers des systèmes plus interprétables. Les méthodes présentées ici offrent des alternatives nouvelles et plus holistiques pour l'analyse numérique du rythme, ouvrant des perspectives vers une analyse automatique plus complète de la musique.

Title: Multi-Scale Computational Rhythm Analysis : a framework for sections, downbeats, beats, and microtiming

Keywords: Signal processing, Music Information Retrieval, Deep Learning

Abstract: Computational rhythm analysis deals with extracting and processing meaningful rhythmical information from musical audio. It proves to be a highly complex task, since dealing with real audio recordings requires the ability to handle its acoustic and semantic complexity at multiple levels of representation. Existing methods for rhythmic analysis typically focus on one of those levels, failing to exploit music's rich structure and compromising the musical consistency of automatic estimations.

In this work, we propose novel approaches for leveraging multi-scale information for computational rhythm analysis. Our models account for interrelated dependencies that musical audio naturally conveys, allowing the interplay between different time scales and accounting for music coherence across them. In particular, we conduct a systematic analysis of downbeat tracking systems, leading to convolutional-recurrent architectures that exploit short and long term acoustic modeling; we introduce a skip-chain conditional random field model for downbeat tracking designed to take advantage of music structure information (i.e. music sections repetitions) in a unified framework; and we propose a language model for joint tracking of beats and micro-timing in Afro-Latin American music.

Our methods are systematically evaluated on a diverse group of datasets, ranging from Western music to more culturally specific genres, and compared to state-of-the-art systems and simpler variations. The overall results show that our models for downbeat tracking perform on par with the state of the art, while being more musically consistent. Moreover, our model for the joint estimation of beats and microtiming takes further steps towards more interpretable systems. The methods presented here offer novel and more holistic alternatives for computational rhythm analysis, towards a more comprehensive automatic analysis of music.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France