Compositional hierarchical model for music information retrieval

A dissertation presented

by

Matevž Pesek

to

The Faculty of Computer and Information Science

in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

in the subject of

Computer and Information Science

Ljubljana, 2018

APPROVAL

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text.

— Matevž Pesek —
September 2018

The submission has been approved by

dr. Denis Trček
Professor of Computer and Information Science
examiner

dr. Andrej Košir
Professor of Electrical Engineering
examiner

dr. Juan Pablo Bello
Associate Professor of Music Technology
external examiner
New York University

dr. Matija Marolt and dr. Aleš Leonardis
Professors of Computer and Information Science
advisors

PREVIOUS PUBLICATION

I hereby declare that the research reported herein was previously published/submitted for publication in peer reviewed journals or publicly presented at the following occasions:

[1] M. Pesek, A. Leonardis, and M. Marolt. A compositional hierarchical model for music information retrieval. Y-H. Yang, J. H. Lee, editors, *Proc. of ISMIR 2014*, pages 131–136, Taipei (TW), 2014.

[2] M. Pesek, A. Leonardis, and M. Marolt. Robust real-time music transcription with a compositional hierarchical model. *PloS one*, 12(1):1–21, 2017. 10.1371/journal.pone.0169411

[3] M. Pesek, A. Leonardis, and M. Marolt. SymCHM—An Unsupervised Approach for Pattern Discovery in Symbolic Music with a Compositional Hierarchical Model. *Applied Sciences*, 7(11):1–20, 2017. 10.3390/app7111135

[4] M. Pesek, and M. Marolt. Compositional hierarchical model for music understanding. In Vempala N., Russo F., editors, *Proc. of CogMIR 2013*, Toronto (CA), 2013.

[5] M. Pesek, F. Mihelič. Hidden Markov model for chord estimation using compositional hierarchical model features. In Zajc B., Trost A., editors, *Proc. of ERK 2013*, pages 145–148, Portorož (SI), 2013. IEEE.

[6] M. Pesek, Guna J, A. Leonardis, and M. Marolt. Visualization of a deep architecture using the compositional hierarchical model. In Stojmenova Duh E., editor, *Proc. of ICWUD 2013*, pages 145–148, Ljubljana (SI), 2013.

[7] M. Pesek, and M. Marolt. Chord estimation using compositional hierarchical model. In Ramirez R., Conklin D., Manuel Ińesta J., editors, *Proc. of MML 2013*, Prague (CZ), 2013.

[8] M. Pesek, A. Leonardis, and M. Marolt. Boosting audio chord estimation using multiple classifiers. In Muštra M., editor, *Proc. of IWSSIP 2014*, pages 107–110, Zagreb (HR), 2014. IEEE.

[9] M. Pesek, A. Leonardis, and M. Marolt. A preliminary evaluation of robustness to noise using the compositional hierarchical model for music information retrieval. In Zajc B., Trost A., editors, *Proc. of ERK 2014*, pages 104–107, Portorož (SI), 2014. IEEE.

[10] M. Žerovnik, M. Pesek, and M. Marolt. Ocenjevanje osnovnih frekvenc z uporabo kompozicionalnega hierarhičnega modela. In Zajc B., Trost A., editors, *Proc. of ERK 2014*, pages 265–268, Portorož (SI), 2014. IEEE.

[11] M. Pesek, A. Leonardis, and M. Marolt. Compositional hierarchical model for pattern discovery in music. Berge P., editor, *Proc. of EuroMAC 2014*, pages 288, Leuven (BE), 2014.

[12] M. Pesek, A. Leonardis, and M. Marolt. Towards pattern discovery in symbolic music representations using a compositional hierarchical model. In Zajc B., Trost A., editors, *Proc. of ERK 2014*, pages 57–60, Portorož (SI), 2015. IEEE.

[13] M. Pesek, L. Zakrajšek, and M. Marolt. WEBCHM: an online tool for music analysis, transcription and annotation. In Mueller M., Wiering F., editors, *Proc. of ISMIR 2015*, Malaga (ES), 2016.

[14] M. Pesek, A. Leonardis, and M. Marolt. Pattern discovery and music similarity with compositional hierarchical model. In Vempala N., Russo F., editors, *Proc. of CogMIR 2016*, New York (NY), 2016.

[15] M. Pesek, A. Leonardis, and M. Marolt. SymCHMMerge—hypothesis refinement for pattern discovery with a compositional hierarchical model. In Ramirez R., Conklin D., Manuel Iñesta J., editors, *Proc. of MML 2016*, Riva del Garda (IT), 2016.

[16] M. Pesek, M. Žerovnik, A. Leonardis, and M. Marolt. Modeling song similarity with unsupervised learning. In Ramirez R., Conklin D., Manuel Iñesta J., editors, *Folk music analysis : 8th International Workshop*, Thessaloniki (GR), 2018.

I certify that I have obtained a written permission from the copyright owner(s) to include the above published material(s) in my thesis. I certify that the above material describes work completed during my registration as graduate student at the University of Ljubljana.

This dissertation is a result of doctoral research, in part financed by the European Union, European Social Fund and the Republic of Slovenia, Ministry for Education, Science and Sport in the framework of the Operational programme for human resources development for the period 2007 – 2013.

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Matevž Pesek
Kompozicionalni hierarhični model za pridobivanje informacij iz glasbe

POVZETEK

S porastom globokih arhitektur, ki temeljijo na nevronskih mrežah, so se v zadnjem času bistveno izboljšali rezultati pri reševanju problemov na več področjih. Zaradi popularnosti in uspešnosti teh globokih pristopov, temelječih na nevronskih mrežah, so bili drugi, predvsem kompozicionalni pristopi, odmaknjeni od središča pozornosti raziskav.

V pričujoči disertaciji se posvečamo vprašanju, ali je mogoče razviti globoko arhitekturo, ki bo presegla obstoječe probleme globokih arhitektur. S tem namenom se vračamo h kompozicionalnim modelom in predstavimo kompozicionalni hierarhični model kot alternativno globoko arhitekturo, ki bo imela naslednje značilnosti: transparentnost, ki omogoča enostavno razlago naučenih konceptov, nenadzorovano učenje in zmožnost učenja na majhnih podatkovnih bazah, uporabnost modela kot izluščevalca značilk, kot tudi zmožnost uporabe transparentnosti modela za odkrivanje vzorcev.

Naše delo temelji na kompozicionalnih modelih, ki so v glasbi intuitivni. Predlagani kompozicionalni hierarhični model je zmožen nenadzorovanega učenja večnivojske predstavitve glasbenega vhoda. Model omogoča pregled naučenih konceptov skozi transparentne strukture. Lahko ga uporabimo kot generator značilk – izhod modela lahko uporabimo za klasifikacijo z drugimi pristopi strojnega učenja. Hkrati pa lahko transparentnost predlaganega modela uporabimo za analizo (raziskovanje naučene hierarhije) pri odkrivanju vzorcev, kar je težko izvedljivo z ostalimi pristopi, ki temeljijo na nevronskih mrežah.

Relativno kodiranje konceptov v samem modelu pripomore k precej manjšim modelom in posledično zmanjšuje potrebo po velikih podatkovnih zbirkah, potrebnih za učenje modela. Z vpeljavo biološko navdahnjenih mehanizmov želimo model še bolj približati človeškemu načinu zaznave. Za nekatere mehanizme, na primer inhibicijo, vemo, da so v človeški percepciji prisotni na nižjih nivojih v ušesu in bistveno vplivajo na način zaznave. V modelu uvedemo prve korake k takšnemu načinu procesiranja proti končnemu cilju izdelave modela, ki popolnoma odraža človeško percepcijo.

V prvem poglavju disertacije predstavimo motivacijo za razvoj novega modela. V drugem poglavju se posvetimo dosedanjim objavljenim dosežkom na tem področju. V nadaljnjih poglavjih se osredotočimo na sam model. Sprva opišemo teoretično zasnovo modela in način učenja ter delovanje biološko-navdahnjenih mehanizmov. V naslednjem koraku model apliciramo na več različnih glasbenih domen, ki so razdeljene glede na tip vhodnih podatkov. Pri tem sledimo časovnici razvoja in implementacijam modela tekom doktorskega študija. Najprej predstavimo aplikacijo modela za časovno-frekvenčne signale, na katerem model preizkusimo za dve opravili: avtomatsko ocenjevanje harmonij in avtomatsko transkripcijo osnovnih frekvenc. V petem poglavju predstavimo drug način aplikacije modela, tokrat na simbolne vhodne podatke, ki predstavljajo glasbeni zapis. Pri tem pristopu se osredotočamo na odkrivanje vzorcev, s čimer poudarimo zmožnost modela za reševanje tovrstnih problemov, ki je ostalim pristopom še nedosegljivo. Model prav tako evalviramo v vlogi generatorja značilk. Pri tem ga evalviramo na problemu melodične podobnosti pesmi in razvrščanja v variantne tipe. Nazadnje, v šestem poglavju, pokažemo zadnji dosežek razvoja modela, ki ga apliciramo na problem razumevanja ritma v glasbi. Prilagojeni model analiziramo in pokažemo njegovo zmožnost učenja različnih ritmičnih oblik in visoko stopnjo robustnosti pri izluščevanju visokonivojskih struktur v ritmu.

V zaključkih disertacije povzamemo vloženo delo in rezultate ter nakažemo nadaljnje korake za razvoj modela v prihodnosti.

Ključne besede pridobivanje informacij iz glasbe, globoke arhitekture, avtomatsko ocenjevanje harmonij, ocenjevanje osnovnih frekvenc, odkrivanje vzorcev, modeliranje ritma

University *of Ljubljana*
Faculty *of Computer and Information Science*

Matevž Pesek

Compositional hierarchical model for music information retrieval

ABSTRACT

In recent years, deep architectures, most commonly based on neural networks, have advanced the state of the art in many research areas. Due to the popularity and the success of deep neural-networks, other deep architectures, including compositional models, have been put aside from mainstream research.

This dissertation presents the compositional hierarchical model as a novel deep architecture for music processing. Our main motivation was to develop and explore an alternative non-neural deep architecture for music processing which would be transparent, meaning that the encoded knowledge would be interpretable, trained in an unsupervised manner and on small datasets, and useful as a feature extractor for classification tasks, as well as a transparent model for unsupervised pattern discovery.

We base our work on compositional models, as compositionality is inherent in music. The proposed compositional hierarchical model learns a multi-layer hierarchical representation of the analyzed music signals in an unsupervised manner. It provides transparent insights into the learned concepts and their structure. It can be used as a feature extractor—its output can be used for classification tasks using existing machine learning techniques. Moreover, the model's transparency enables an interpretation of the learned concepts, so the model can be used for analysis (exploration of the learned hierarchy) or discovery-oriented (inferring the hierarchy) tasks, which is difficult with most neural network based architectures.

The proposed model uses relative coding of the learned concepts, which eliminates the need for large annotated training datasets that are essential in deep architectures with a large number of parameters. Relative coding contributes to slim models, which are fast to execute and have low memory requirements. The model also incorporates several biologically-inspired mechanisms that are modeled according to the mechanisms that exists at the lower levels of human perception (e.g. lateral inhibition in the human ear) and that significantly affect perception.

The proposed model is evaluated on several music information retrieval tasks and its results are compared to the current state of the art.

The dissertation is structured as follows. In the first chapter we present the motivation for the development of the new model. In the second chapter we elaborate on the related work in music information retrieval and review other compositional and transparent models. Chapter three introduces a thorough description of the proposed model. The model structure, its learning and inference methods are explained, as well as the incorporated biologically-inspired mechanisms. The model is then applied to several different music domains, which are divided according to the type of input data. In this we follow the timeline of the development and the implementation of the model. In chapter four, we present the model's application to audio recordings, specifically for two tasks: automatic chord estimation and multiple fundamental frequency estimation. In chapter five, we present the model's application to symbolic music representations. We concentrate on pattern discovery, emphasizing the model's ability to tackle such problems. We also evaluate the model as a feature generator for tune family classification. Finally, in chapter six, we show the latest progress in developing the model for representing rhythm and show that it exhibits a high degree of robustness in extracting high-level rhythmic structures from music signals.

We conclude the dissertation by summarizing our work and the results, elaborating on forthcoming work in the development of the model and its future applications.

Key words music information retrieval, deep learning architectures, automated chord estimation, multiple fundamental frequency estimation, pattern discovery, rhythm modeling

ACKNOWLEDGEMENTS

I am grateful to my family, especially my wife and my mother, for supporting me during the endless weekends at the Laboratory. I am thankful to the members of my Laboratory, Alenka, Ciril, Manca and Saša, for the debates and brainstorming sessions, and both my supervisors for their leadership and help with the model's development and evaluation. I thank my colleagues Manca, Luka, Urša and Špela who have been involved and have helped developing this model. Their faith in the compositional model has been inspiring. I look forward to the coming years in which we will further collaborate, expand and improve the compositional hierarchical model.

I dedicate this dissertation to my son Gabriel.

The world is compositional.

— Matevž Pesek, Ljubljana, September 2018.

CONTENTS

1

Introduction

1.1 Motivation

The field of computer science involves a vast mixture of interdisciplinary approaches that aim to automate the existing or invent new processes that assist our needs in everyday life. Computers are already ubiquitous in most of our activities, integrated into end-user products, such as smart TVs, kitchen appliances or self-driving cars, and used for storing, managing, organizing and classifying the ever-growing amounts of data we produce. It is therefore natural that computer science has also touched seemingly opposite fields such as art and music—in music, the junction has been formed on several layers: in music creation, music organization, and music analysis.

As an attempt to analyse, retrieve and organize music, the field of music information retrieval (MIR) has emerged in the last two decades [1]. It has grown since its early beginnings, encompassing a number of topics, including music perception, cognition and information retrieval, bordering on several well-established fields, such as psychology (e.g. [2–4]), neuroscience (e.g. [5–7]), musicology (e.g. [8, 9]), and computer science (e.g. [10–12]).

One part of MIR researches deals with the extraction of semantic descriptions from music in its various forms. As in many related areas, a significant increase in algorithm accuracy and efficiency has been achieved in recent years for tasks, such as melody estimation (e.g. [13, 14]), chord estimation (e.g. [15–18]), beat tracking (e.g. [19, 20]), mood estimation (e.g. [21, 22]), music recommendation (e.g. [23]), genre classification (e.g. [24–26]), and pattern analysis (e.g. [27–29]). In many cases, the increased accuracy can be attributed to the introduction of deep learning to the field [30]. Several deep learning approaches have been proposed for a number of tasks, including melody transcription (e.g. [31]), genre classification (e.g. [32]), onset detection (e.g. [33]), drum pattern analysis (e.g. [34]), and chord estimation (e.g. [35]). In its broad definition, a deep learning algorithm constructs multiple levels of data abstraction (a hierarchy of features) in order to model high-level representations present in the observed data [36].

Most deep learning approaches are based on neural networks. When such networks are trained, the high-level representations in the training data are encoded in a multilayer hierarchy, however, the encoded knowledge is implicit and difficult to explain in a transparent (non black-box) way. Many approaches for visualization of the learned concepts in neural networks have been developed [37], but they are still far from explaining the encoded knowledge in a fully transparent way. For example, input occlusion on images [38] attempts to identify the regions within the image which trigger the observed response. Deconvolution generates salience maps by inverting the process. The salience maps can be used for object localization [39]. A neural network may be trained to invert feature representations, so that given a feature vector, the network predicts the expected average image that could have produced the given feature vector, thus explaining the feature [40].

Deep neural networks typically have a large number of trainable parameters to cover the entire target domain, which necessitates large datasets for training. Such large datasets may be difficult to acquire due to the scarcity of the appropriate data, potential copyright issues or pure storage requirements. Another pertinent drawback is that in most cases, training datasets need to be annotated for supervised learning. The annotations are a) most commonly subjective (e.g. genre classification in music, object tracking in computer vision) and therefore require multiple annotators to approximate the human perception of the problem, b) often require an expert (e.g. music transcription) and c) require a significant amount of time and manpower.

While deep neural architectures are highly successful and commonly used for discrimination—to classify whether the observed input belongs to one class or another, we are not aware of their uses for discovery tasks—given unknown input, the model produces its own observations of high-level abstractions in the input. A desirable feature of a deep model should also be to provide a set of responses to unfamiliar input given its previously gathered knowledge. If there is more than one explanation of a given input, the model's output should provide several alternative explanations, as well as their likelihoods.

With these properties of current deep neural architectures in mind, our main motivation is to develop and explore an alternative non-neural deep architecture for music processing. We base our work on compositional models, as compositionality is inherent in music, and introduce the compositional hierarchical model as a deep architecture with the following properties:

- a transparent compositional structure enabling explicit interpretation of the learned concepts;
- the ability to unsupervisedly train on small datasets and generalize well to larger datasets;
- the ability to tackle discriminative and discovery tasks with a single model;

- the ability to produce alternative hypotheses about the input, based on the knowledge acquired through training.

1.2 Scientific Contributions

The proposed dissertation consists of the following expected scientific contributions:

- *A short-time compositional hierarchical model featuring biologically-inspired mechanisms for music information retrieval.* A compositional hierarchical model for processing of music signals has been developed. The proposed model is a white-box model that provides insights into the learned concepts on all processing layers. To add to its robustness, the model includes mechanisms inspired by the human auditory system. The developed model has been evaluated on several music information retrieval tasks using standard annotated datasets.

- *Extension of the model to time-dependent music processing.* The short-time automatic gain control mechanism has been developed as a short-term time-dependent mechanism. The model has also been applied to modeling (time-evolving) melodic and rhythmic patterns.

- *Extension of the model for discriminative tasks.* The compositional hierarchical model was used for several different discriminative tasks, including chord estimation, multiple fundamental frequency estimation and tune family classification.

1.3 Dissertation overview

This dissertation begins with an overview of related work. We continue with a description of the proposed model—the compositional hierarchical model—which aims to include the aforementioned desired features in a single model. We also describe modifications of the model for the analysis of spectral and symbolic music representations. To show that the model meets the desired goals, we apply it to several MIR tasks: automated chord estimation, multiple fundamental frequency estimation, pattern discovery in symbolic music, and rhythmic processing. For the first task, we demonstrate that the model can be trained on small datasets and used for chord estimation as a frame-based feature extractor. For multiple fundamental frequency estimation, we train the model on small monophonic datasets and apply it to polyphonic music, where the model acts as an explicit classifier, thus eliminating the need for additional classifiers. We also show the model's robustness in unfamiliar situations by applying it to several different datasets. The task of pattern discovery in symbolic music showcases the model's transparent structure and its ability to tackle discriminative as well as discovery tasks. Finally, we present the application of the model to rhythmic processing of music, where we show its ability of identifying rhythmic representations robustly across corpora and in live audio. We conclude this dissertation with an overview of the presented model, a discussion about the model's performance, and a fraction of the endless list of future plans.

1.4 Abbreviations used in this dissertation

The list of abbreviations used in this dissertation is presented in Table 1.1.

Table 1.1
List of abbreviations

Abbreviation	Description
MIR	Music information retrieval
MIREX	Music Information Retrieval Evaluation eXchange
ISMIR	International Society for Music Information Retrieval Conference
CHM	Compositional hierarchical model developed in this dissertation
AGC	Automatic gain control mechanism in CHM
MAPS	MIDI Aligned Piano Sounds database—a piano database for multipitch estimation and automatic transcription of music
DNMF	Discriminant Non-Negative Matrix Factorization
SymCHM	An adjustment of the compositional hierarchical model for symbolic music representation
SymCHMMerge	An improvement of the SymCHM, which includes a refined pattern output procedure
JKU-PDD	The development dataset for the discovery of repeated themes and sections task
MTC-ANN	The Dutch folk song dataset
ACE	Automated chord estimation
MFFE	Multiple fundamental frequency estimation

2

Related Work

The goal of this dissertation is to develop a deep learning architecture, based on transparency, relativity and shareability of encoded structures, and unsupervised learning. This chapter first presents an overview of hierarchical modeling and continues with a deliberation on the hierarchical approaches in MIR. Additionally, an overview of several deep learning architectures, which are currently the prevalent approach to solving a variety of MIR tasks is provided. The chapter concludes with an overview of related work for five MIR tasks our model was evaluated on: automated chord estimation, multiple fundamental frequency estimation, discovery of repeated themes and sections, music similarity and rhythm analysis.

2.1 Hierarchical models

The main principle of hierarchical models lies in the hierarchical nature of our perception of the world. Just like the human visual system can discern complex forms by combining basic elements, such as edges, lines, contrasts, and colors into increasingly more complex percepts, so can the human auditory system group frequency components into auditory events, multiple tonal events into harmonies and their time evolution into melodies and harmonic progressions.

The compositional hierarchical structures are therefore intuitively similar to our conscious perception. We begin this section with an overview of such models in the field of computer vision, from which the idea for the model proposed in this dissertation originates, followed by an overview of hierarchical models in MIR.

2.1.1 Hierarchical models in computer vision

In the field of computer vision, several non-neural deep approaches have recently been introduced. In this section, we present several hierarchical approaches, including the *learned Hierarchy of Parts*, which was an inspiration for the development of our compositional hierarchical model.

Cascaded hierarchical model Seyedhosseini et al. [41] presented a cascaded hierarchical model, which learns contextual information in a hierarchical framework for image segmentation. At each level of the hierarchy, a classifier is trained based on down-sampled input images and outputs from previous levels. This model then incorporates the resulting multi-resolution contextual information into a classifier to segment the input image at the original resolution. The procedure is repeated by cascading the hierarchical framework to improve the segmentation accuracy. Their approach uses multiple classifiers. The best results are given by the logistic disjunctive normal networks (LDNN), which have also been developed by the authors. An LDNN consists of three layers: an adaptive layer of feature detectors implemented by logistic sigmoid functions, a layer of logical units that compute conjunctions, and a layer for disjunctions.

While the approach employs a hierarchical structure, the incorporated classifiers (LDNNs) are not transparent.

Hierarchical compositional network George et al. [42] recently presented the Hierarchical compositional network (HCN). The approach is a directed generative model able to ”discover and disentangle, without supervision, the building blocks of a set of binary images”. The model is composed of binary features which are defined hierarchically as a composition. The compositions are formed from the features on the layer below and form compositions on the consecutive layers. To achieve transparency, the authors proposed new inference and learning processes. They introduced max-product message passing (MPMP), a significantly extended approach of the well-known sum-product message passing. According to the authors, the MPMP ”can learn features that are composable, interpretable and causally meaningful” [p. 2][42]. The features can be employed for image reconstruction and therefore enable an insight into the model.

Learned Hierarchy of Parts In 2007, Leonardis and Fidler presented a statistical approach to learning a hierarchy of parts [43] in the field of computer vision. They proposed a novel approach to constructing a hierarchical representation of visual input which aims to enable recognition and detection of a large number of object categories. Their approach is statistically driven and inspired by several principles: efficient indexing, robust matching and compositionality. Their idea is driven by the need for robust and flexible representations which they denote as ”parts”, as shown in Fig. 2.1. The principle of compositionality is employed throughout the model's structure; by merging statistically significant trivial features, new layers of compositions (parts composed of parts) are built. Although the model is used for object categorization, the lower layers are learned in a category-independent way to ”obtain complex, yet shareable visual building blocks, which is a crucial step towards a scalable representation” [43].

The model has been further developed and included in a *Histogram of Compositions*

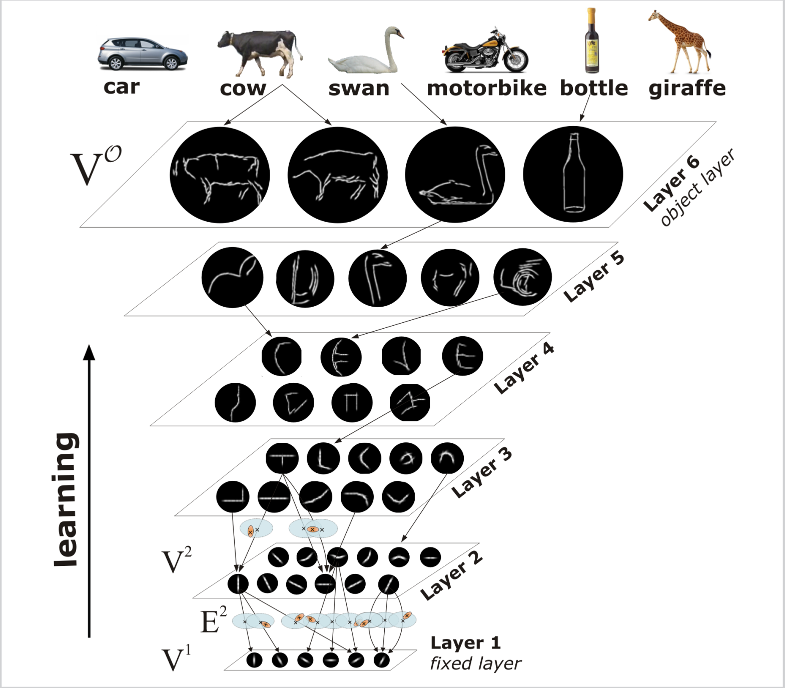


Figure 2.1
Representation of the lHoP model. Bottom layers are learned in an unsupervised manner. The top layer is learned supervisedly to categorize objects.

(HoC) descriptor [44]. The HoC descriptor uses the generative model of hierarchical compositions for feature extraction and performs a hypothesis verification of detections produced by the hierarchical compositional model. Tabernik et al. [44] evaluated the proposed descriptor and demonstrated its superiority in robustness on significantly occluded objects in comparison to the state of the art convolutional neural network.

The structure of our model is inspired by the learned Hierarchy of Parts (lHoP) model.

Other hierarchical compositional models Several other *hierarchical compositional models* for computer vision have previously been proposed. Zju and Yuille [45] proposed a hierarchical compositional system for object detection. The proposed system is defined as a set of nodes, which are compositions of sub-nodes. The authors empirically evaluated the system's performance in terms of robustness and speed, surpassing several other systems of that time. Similar to our proposed model, the system's structure is a transparent hierarchy of compositions.

Kortylewski et al. [46] focused on unsupervised learning of this model and proposed a greedy EM-variant of the learning algorithm. The aim was to provide a simple probabilistic and generative learning procedure. The reported results indicate comparable performance of the model on the domain adaptation task.

Töpfer et al. [47] proposed a compositional hierarchical model for road scene analysis. Their model is encoded as a tree-structured graph, where vertices represent compositional parts and edges their connections which encode spatial constraints of the compositions. Evaluation showed that the approach significantly reduces time complexity of detection while retaining the accuracy.

2.1.2 Hierarchical models in music

Hierarchical representations are intuitive in music when one considers its spectral and temporal structures. Many approaches for hierarchical music modeling stem from the field of music theory, which offers well-established (hierarchical) rule-based systems for music analysis. First, an overview of several algorithms for automatic generation of these rule-based models is presented, most of which have been introduced in the last decade. Additionally, two hierarchical models, which address similar problems to the ones discussed in this dissertation, are presented in more detail: the Multiple Viewpoint System and the Information Dynamics of Thinking architecture.

Rule-based models The generative theory of tonal music (GTTM) by Lerdahl and Jackendoff [8] offers an approach of explicit hierarchical music modeling in musicology that is very well-known in contemporary music theory. The GTTM attempts to formalize a system, which reflects the ways of the listener's understanding of music. The GTTM proposes four hierarchical aspects: grouping and metrical structures, time-span reduction, and prolongational reduction structures. It operates under the constraints provided by the sets of rules for each structure. Another hierarchical approach was proposed by Heinrich Schenker [48]. Named after the author, the Schenkerian analysis attempts to unveil the underlying fundamental structure (*Ursatz*) in music.

Though the GTTM and the Schenkerian analysis mostly rely on expert rules, the concept of hierarchical structuring is perceived as natural, since it is based on the patterns of human perception and thinking processes. As the rules are not very strictly defined, it is difficult to automate such analysis, although several systems for GTTM or Schenkerian analysis were introduced in the past (e.g. [49–51]). For example, Marsden [51] proposed a system for automatically deriving a Schenkerian reduction of an extract of tonal music. He provided a proof of concept that such analysis can be done automatically; however, he also discovered several issues with his implementation. The proposed procedure demands significant computational space and power. Additionally, the system yields a large amount of possible analyses, which differ in their quality. The author concluded that although the concept had been proven successful, additional research was required in order for the system to be usable for real-world analysis.

Moreover, the human perception has been explored and often described as one or multiple hierarchical systems. For example, Farbood [52] explored the interconnection between the limitations of working memory and the hierarchical structures in music. They reported that the differences in optimal timing for tonal harmonies versus rhythm and pitch contour imply different processing for each of these modalities. Attempts, such as [53, 54], have also been made to empirically determine the presence of such hierarchical representations produced by human cognitive processes. Finally, we must note that hierarchical models are abound in the analysis of music perception from the point of view of computational biology and neuroscience [9, 55–57].

Multiple viewpoint system Conklin and Anagnostopoulou [58] proposed a multiple viewpoint pattern discovery algorithm based on the suffix-tree. For a selected viewpoint (a transformation of a musical event into an abstract feature) the algorithm builds a suffix tree of viewpoint sequences (transformed music pieces). After selecting the patterns that meet the specified frequency and significance thresholds, the leafs of the suffix tree are reported as the longest significant patterns in the corpus. Conklin and Bergeron [59] present two algorithms based on viewpoints for statistical modeling of the melody [60]. A viewpoint is a function which computes values for the events in a sequence; a pattern is a sequence of such feature sets, where the latter represent a logical conjunction of multiple viewpoints. The authors present a complete algorithm which can find all 'maximal frequent patterns' and an optimization algorithm using a faster heuristic approach, where the found patterns may not always be the maximal frequent patterns. The maximal frequent pattern represents a pattern whose component feature set cannot be further specialized without the pattern becoming infrequent.

Information Dynamics of Thinking architecture Wiggins and Forth described a cognitive architecture that is close to our proposed model and named it *Information Dynamics of Thinking* (IDyOT) [61]. The architecture is a step towards a model which includes ”aspects of human creativity and other forms of cognitive processing in terms of a preconscious predictive loop” [61, p. 127]. It is a hierarchical architecture that includes a number of *generators* on the first layer, employed to sample the input. Each generator produces an output distribution based on the input sequence. The architecture attempts to model a cognitive cycle, based on the statistical observations of input sequences. The latter are *atomic percepts*, such as pitch, timbre, amplitude and time. The generators' predictions are formed into chunks based on the selection.

Predictions, which match the perceptual input, are grouped into sequences. If a sequence matches the information profile, a *chunk* is detected. The generator stores the chunk which is then included in the statistical model. This dynamic aspect results in an incremental learning process. The proposed architecture offers an alternative deep approach. While it seems to address several limitations discussed in this dissertation, there is, to our knowledge, no available implementation or published results of this system being applied to MIR tasks.

2.2 Deep neural network architectures

As deep neural network architectures have become the preferred approach for classification and segmentation, as well as other tasks that involve the processing of images, videos and sound, they are given a more focused overview in this section. To fully familiarize the reader with neural-network-based deep architectures, we first briefly elaborate on the history and evolution of neural networks, followed by a short description of some of the prevalent deep neural network architectures. We conclude the section with an overview of their applications in MIR.

2.2.1 Neural networks

Artificial neural networks were first introduced in the early sixties by Rosenblatt [62, 63], who defined the *perceptron* as a three layer structure with one input layer, a second non-adaptive layer with hand-coded features, and an output layer. Although perceptrons were an innovative and promising algorithm, they were limited in their learning capacity (only linear problems) and were also not learned but hand-coded.

Decades later, when the backpropagation algorithm for weight adjustment was introduced, first generation perceptrons were extended by discarding the need for hand-coding of weights, as well as by introducing non-linear activation functions [64]. The latter is also called the backward propagation of errors and a generalization of the *delta rule* [65]. Based on an annotated training set, the outputs of the neural network for the given input are compared to the annotations. The error is calculated as the difference between the expected and the produced outputs and is used to adjust the weights of the network's hidden layer. The algorithm is repeated for each layer backwards—from the output to the input layer. The algorithm can be iterated several times until a the error is satisfactorily small. The whole process can be time-consuming, depending on the number of training samples and network layers. Although backpropagation-based artificial neural networks have successfully been used in a variety of problem domains, they possess several shortcomings. Large networks that would, for example, model complex perceptual tasks are difficult to train, as the size of the appropriate annotated datasets increases and learning becomes unstable. The training algorithm may often converge to a local minimum and thus a good solution may not be found. Deep neural networks are essentially neural networks with a high number of layers. In recent years they have become the preferred algorithm for solving a large number of tasks involving multimedia materials. Why deep architectures are more successful than the shallow ones is still unclear. The reasons may lie in the hierarchical nature of tasks we are trying to solve, the number of neurons needed for the same accuracy (shallow networks could be larger than deep for the same task), and the fact that shallow networks are more difficult to train. Many different deep neural network architectures have been introduced over the years; here we summarize several of the more prominent ones.

Deep belief networks Deep belief network approach [66] emerged as a new approach in 2006, when kernelized support vector machines were outperformed on the MNIST database of handwritten digits, addressing some of the issues of shallow networks by introducing gradual layer-by-layer learning and the ability to train on non-annotated data.

A deep belief network (DBN) is a generative model, comprised of several layers of latent variables. The units at the lowest layer represent the input vector of the data, while the subsequent layers represent latent variables. The connections between these layers are directed in a top-down manner. In contrast, the top two layers are linked with un-

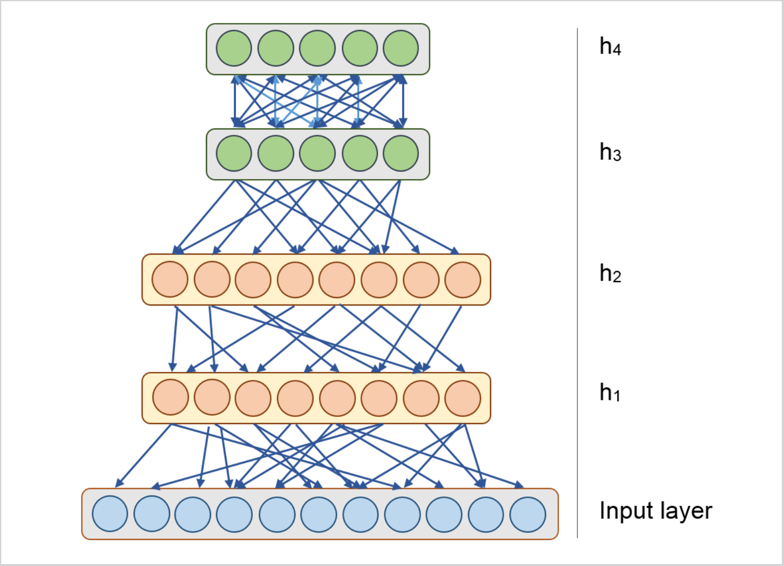


Figure 2.2
An abstract representation of the DBN structure. This example shows a model with an input data layer at the bottom, followed by two latent layers (h_1 and h_2), and two output layers, representing high-level concepts extracted from the data. The highest two layers are connected undirectedly, whereas the latent and the input layers form top-down directed connections.

directed connections in order to form associative memory. The units of the latent layers can be observed as feature detectors.

Deep belief networks reflect a hierarchy by processing the signal through several stages, extracting simple features at lower layers and modeling complex structures at higher layers. Such deep learning embodies the idea of learning the less-complex abstract representations of the data on one layer and later composing these representations into more complex high-level structures present in the data.

The model can be applied to a specific task in two stages: the first stage consists of layer-by-layer learning or pre-training of the model on a training set. At the second stage, the model is applied to the dataset of interest. Training a DBN may seem a difficult problem; however, by symmetrically connecting the hidden and output layers, the model can be observed as a restricted Boltzmann machine [67]. Each layer of a DBN is learned independently, thus facilitating the learning process compared to the previous attempts with multi-layer artificial neural networks. The layer-wise unsupervised learning process may also be implemented by a greedy approach for weight optimization [68]. The most discernible features from different classes are stimulated. While inferring the DBN over a given dataset, the information is extracted and passed from the input layer to the highest layer over a number of latent layers. The output of the highest computed DBN layer may be used as an input for standard machine-learning classification techniques. The highest output layer may also be hand-coded, depending on the problem task. For example, the output layer may contain only a single node summing all the outputs of the previous layer and applying a threshold function for a binary classification.

Convolutional neural networks Convolutional neural networks (CNNs) also consist of an input and an output layer connected by a number of hidden layers. As the name implies, their main difference from the DBNs are the convolutional layers, which apply correlations with the (learned) filters to their input and provide the resulting *feature maps* as outputs. Since a filter is only applied to a small portion of the input—its receptive field—it only has a small number of parameters, which is beneficial when compared to a fully-connected standard network layer. Additionally, to reduce the size of the feature maps produced by the network filters, pooling layers, which reduce the size of the maps by grouping and summarizing blocks of activations on a previous layer into single outputs, can be included. The entire network commonly consists of tens or even hundreds of convolutional layers, optionally followed by one or more fully connected layers used for classification. Specialized CNN architectures, such as inception [69] and residual networks [70], have been introduced for specific domains.

Recurrent neural networks Neural networks provide an abstraction of a single or a small amount of neighboring input entities. When observing time-domain signals, their long-term evolution is also important. To model this aspect, recurrent neural networks (RNNs) were proposed. In RNNs, feed-forward connections from lower to higher layers are complemented by feedback connections from higher to lower layers. These connections can model delays in the signal and thus represent memory-like sequence modeling units. RNNs can therefore model temporal sequences. Several recurrent network models have been introduced, such as the long-short term memory (LSTM) by Hochreiter and Schmidhuber [71].

Generative adversarial networks In 2014, Goodfellow et al. [72] proposed the generative adversarial network (GAN), a combination of two neural networks. The proposed model is an attempt to overcome two difficulties of existing deep generative networks, as

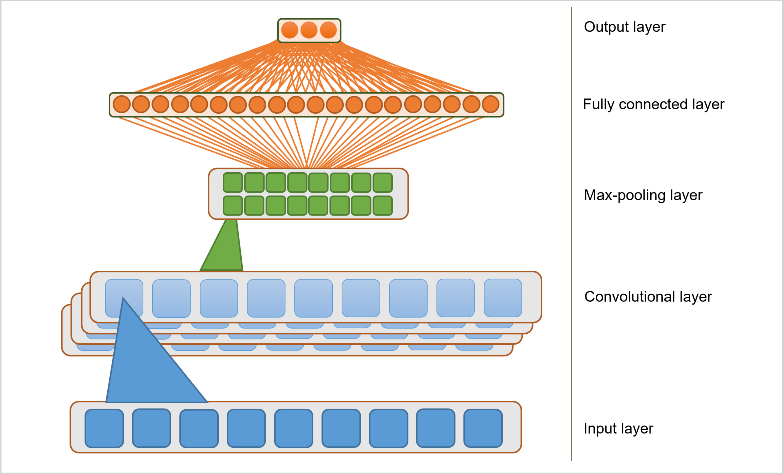


Figure 2.3
An abstract representation of the CNN structure. This example shows a model with an input data layer at the bottom, followed by a convolutional layer and a max-pooling layer. The highest two layers are fully connected.

expressed by the authors: the difficulty of approximating many intractable probabilistic computations which arise in maximum likelihood estimation and related strategies; and the difficulty of leveraging the benefits of piece-wise linear units in the generative context. The approach consists of two models: a generative Model G and a discriminative Model D. While the generative Model models the data distribution and generates samples based on a latent space, the discriminative Model determines whether a sample originates from the Model's distribution or the data distribution. The Model D is trained to maximize the probability of assigning the correct label to training samples and samples generated by G. The Model G is trained to minimize the difference between the G's and the training data distributions, thus trying to *fool* D. The GANs have been mainly applied to computer vision problems, such as video generation (e.g. [73]) and object categorization (e.g. [74]).

2.3 *Approaches in MIR*

Music information retrieval involves a wide variety of tasks that encompass creative, analytic and retrieval aspects of working with music in all its different digital forms. In this section, we present the state of the art in tasks, which we chose to demonstrate the possibilities of our compositional hierarchical model: automated chord estimation, multiple fundamental frequency estimation, pattern discovery and melodic similarity in symbolic music, and rhythm processing.

Many of these tasks have been formalized within the Music Information Retrieval Evaluation eXchange (MIREX)[1] in an attempt to establish a framework for evaluation and comparison of different MIR approaches [75]. The MIREX evaluation campaign is now well established in the MIR community, it is run annually and its results are presented at the ISMIR conference.

2.3.1 Automated Chord Estimation

Chord progressions and melody are two of the most recognizable building blocks of Western music—they are usually adequate for recalling a music piece. Automatic chord estimation can therefore be used for transcription [76–79], music classification [80] and other tasks. Chord estimation can also be used for information aggregation, or metadata extraction, providing information for high-level chord progression [16, 81] and pattern analysis [3, 82].

Chord estimation algorithms most commonly consist of two models: an acoustic model, transforming the audio signal into features, and a language model, modeling the time relations between chords.

In traditional approaches, chroma vectors [76, 83] or pitch class profile (PCP) vectors [84] are the most commonly used features. They provide an intermediate-level representation of an audio signal and usually contain 12 dimensions, each representing the strength of a pitch class in the signal. Each chroma vector component is calculated from the corresponding octave-wrapped frequencies in the signal spectrum. As chroma vectors retain pitch-class information, they can be used for chord estimation with standard machine-learning algorithms, such as support vector machines. However, such classification ignores time-dependent information as vectors are treated independently. Hidden Markov models (HMM) are therefore commonly applied as language models for time-dependent processing [16, 85, 86] with chords as hidden states and features as observations.

Recently, deep learning has often been used for chord estimation. Boulanger-Lewandowski et al. [87] proposed a RNN model for this task. By training the model on the

[1] http://www.music-ir.org/mirex/wiki/MIREX_HOME

whole dataset, they report 93.5% average accuracy per chord. However, they also elaborate on the results, stating that ”this scenario is strongly prone to overfitting: from a machine learning perspective, it is trivial to design a non-parametric model performing at 100% accuracy” [87, p. 5]. Sigtia et al. [17] proposed a hybrid recurrent neural network for audio chord recognition. They first applied a 5-layer deep neural network to the input, obtained with a Constant-Q transform. The DNN was used as an acoustic model, eliminating the need for chroma vectors or similar features. Second, they introduced a hybrid recurrent neural network (RNN) as a language model, which models the relationships between the outputs. This model effectively replaces the HMM used in the traditional approaches. The approach was tested on the MIREX 2014 dataset with four-fold cross validation, where the training set was further split into 80% for training and 20% for validation. The results show increased performance over acoustic-only models (about 3% in frame-level accuracy). Deng and Kwok [35] proposed a hybrid Gaussian-HMM-deep-learning approach. First, the Gaussian-HMM model is used for segmentation of chromagrams and forwarded to a chord classifier implemented as a deep learning model. The authors propose two deep learning models, a deep belief network and a long-short-term-memory RNN. They evaluate several variants of the proposed model combinations and show that their model achieves favourable results over an existing system Chordino on datasets where large chord vocabularies are used; however, it is outperformed by Chordino on small chord vocabularies. Korzeniowski and Widmer [88] proposed a deep chroma chord recognition system, based on a deep neural network. They evaluated their approach on five available datasets—the Beatles, Queen and Zweieck datasets, the RWC pop dataset and the Robbie Williams dataset. The aforementioned authors also proposed a second approach, employing a fully convolutional deep auditory model for chord recognition [89]. The system uses the convolutional network for feature extraction and a conditional random field, which models inter-frame dependencies, to for incorporating dependencies between predictions. Both approaches were evaluated at MIREX 2016 and achieve above average results.

2.3.2 *Multiple Fundamental Frequency Estimation*

The goal of music transcription is to estimate a music score (notes played) from an audio signal. Its essential part is the multiple fundamental frequency estimation (MFFE), where the goal is to estimate all the fundamental frequencies (corresponding to pitches) in individual time-frames of a music signal. As an important MIR goal, transcription has been researched since the early 1970s and a variety of approaches have been developed (e.g. [90–93]). Some approaches use the note hypothesis evaluation based on the signal spectrum (e.g. [94, 95]), while others (e.g. [96–98]) model the audio signal as a composition of sources. Several approaches are tuned to the transcription of specific instruments (e.g. [99–102]) or focus on transcribing instrument-specific symbolic data (e.g. [103]).

Several neural-network-based deep approaches were also presented for multiple fundamental frequency estimation [31, 104, 105]. Bock and Schedl [104] used a recurrent neural network model for a piano transcription, while Nam et al. [105] combined deep belief networks with support vector machines and a Hidden Markov model for the same task. Rigaud and Radenen [31] proposed a combination of two deep neural networks for a transcription of singing voice.

Due to the lack of annotated datasets, many deep network approaches for MFFE [31, 104–106] use a large proportion of the dataset for training. The MAPS dataset [107] is one of the most commonly used datasets for training and evaluating MFFE algorithms. It consists of 30 songs, played using Disklavier and synthesized using 7 piano samples (roughly one million note events). Bock and Schedl [104] evaluated a recurrent neural network model on four piano music datasets, including MAPS MIDI and MAPS D. They reported a high F_1 score (up to 93.5%) for note onset detection; however, they also used a significant amount of the datasets for training and validation (approximately 75% for training and 9.4% for validation). Nam et al. [105] reported results for 30 second excerpts from the MAPS dataset (74.4% frame-level F_1 score) by using roughly 60% of the dataset for training and 25% for validation. As these approaches use a significant part of a limited-sized dataset for training and validation, the results may be overly optimistic. Bittner et al. [108] proposed a model for multiple fundamental frequency estimation, based on a fully connected convolutional network. The model employed salience representations and achieved state-of-the-art results on two out of three MFFE evaluation datasets, and surpassed the state-of-the-art approaches in melody extraction. Among the latest, Hawthorne et al. [109] presented a combination of convolutional neural networks and Long Short Term Memory networks. On a frame level, they achieve 78.30% F_1 score, while on the note level, they surpass the results of other approaches by roughly 30 percent, achieving 82.29%. The authors provide the results only as a proof-of-concept for their work, and stress the issues of training and evaluating on such a small dataset.

It is therefore difficult to apply the achieved scores to ”real-world” scenarios, including recordings, which may not have been recorded in ideal studio environments or with professional performers. The approaches are rarely evaluated in such conditions mainly due to the lack of diverse annotated datasets—most datasets consist mainly of synthesized recordings, which are easily obtainable, and contain only a small number of annotated real recordings. Consequently, the robustness of the algorithms may suffer, as they may overfit the small datasets and the instrument timbres, which leads to poor performance on diverse materials and in the presence of noise.

2.3.3 *Pattern Discovery*

The discovery of repeated patterns is a known problem in various domains, including computer vision (e.g. [110]), bioinformatics (e.g. [111]) and music information retrieval (MIR). Although a common problem, its definition, as well as pattern discovery algorithms, differ significantly across these fields. In music, the importance of repetition has been addressed and discussed by a number of music theorists and, more recently, also by researchers, who have developed algorithms for semi-automatic music analysis, such as Marsden [51]. The MIREX community established several tasks dealing with patterns and structures in music, including structural segmentation, symbolic melodic similarity and pattern matching, and pattern discovery.

The aim of the *discovery of repeated themes and sections* task is to find repetitions, which represent one of the more significant aspects of a music piece [28]. The MIREX task definition states that ”the algorithms take a piece of music as input, and output a list of patterns repeated within that piece” [112]. The task may also seem similar to the well-known pattern matching task [113], but while a pattern matching algorithm aims to find the place of a searched pattern within a dataset and usually has a clear quantitative relation between a query and a match, a discovery of repeated patterns finds locations of multiple similar sequences of data in the dataset, without any information about the searched pattern. As noted by Wang et al. [114], the pattern discovery task differs from the structural segmentation task, where segments cover the whole musical piece and represent disjoint sets of events. In the pattern discovery task, patterns may partially overlap or be subsets of another pattern.

A variety of approaches have been proposed for pattern discovery in music. Hsu et al. [115] attempt to discover nontrivial patterns. They define this as a set of repeated instances without any variation. Patterns which are included in other patterns are ignored. They present two approaches for extracting all nontrivial repeating patterns in the feature string of a music object. The first approach uses a correlative matrix to generate all nontrivial repeating patterns. The second approach uses a string-joining approach, where the longest repeating pattern can be found by repeatedly joining shorter repeating patterns. The approaches are tested and compared on monophonic music. Knopke [116] analyzed 101 masses written by Giovanni Pierluigi da Palestrina. The presented algorithm works well with the chosen corpus since it integrates the specifics of the era and the artist. He uses a *suffix array structure* for pattern discovery, similar to the commonly-used suffix tree (e.g. [117]). The algorithm evaluation is however performed analytically as a comparison of the given results and expert knowledge. Chiu [118] proposed two algorithms for pattern discovery in polyphonic music, both based on string operations. The identified patterns do not contain pattern variations. They also evaluate the approaches in terms of computational complexity. Meek [119] presents an approach for 'keywords' or themes discovery in music. Thus the approach focuses on a sub-task of pattern discovery and attempts to find the musical theme in all identified repeated patterns. The algorithm was evaluated on a dataset of 60 music pieces from the Baroque, Classical, Romantic and contemporary periods. It is able to identify themes in 98% of the cases when compared to the Barlow's expert annotations [120]. Although the algorithm accepts polyphonic music as input, it only finds themes in the top voice.

Rolland [121] presented the FlExPat (Flexible Extraction of Patterns) algorithm for extracting sequential patterns from sequences of data. The algorithm first identifies the equipollent passage pairs and produces the similarity graph, representing the relations between each two passages; second, patterns are extracted from the similarity graph. The author evaluated the approach on a set of ten Charlie Parker solos from the subset of Owens' corpus [122]. Cambouropoulos et al. [117] introduced an approach for extraction of patterns from abstract strings of symbols, allowing for partial overlap of various abstract symbolic classes. They also focused on the time complexity of their solution and addressed the problem of approximate pattern matching. Based on their previous work [123], they presented the PAT algorithm for segmentation based on maximal repeated patterns. Meredith [124] described multiple point-set compression algorithms, including COSIATEC, COSIATECCompress and Forth's algorithm. The author evaluated these approaches on three music-analytical tasks: the classification of folk-song melodies into tune families, the discovery of entries of subjects and countersubjects in fugues, and the discovery of repeated themes and sections in polyphonic works.

Meredith [125] also evaluated his SIATECCompressSegment algorithm for pattern discovery. SIATECCompressSegment is a greedy compression algorithm based on the previously introduced SIATEC approach [28]. The algorithm evaluates patterns based on the assumption that the perceptually interesting patterns correspond to *maximal translatable patterns* (MTP). The approach produces a compact encoding of a musical piece, in the form of a set of *translational equivalence classes* (TECs) of MTPs. The MTP with a defined particular vector is a set of points, which can be translated by that vector to yield a set of new points in the point-set representation. The authors observed that the MTPs often correspond to perceptually significant repeated patterns in music. The TEC defines a set of all the patterns which are translationally equivalent to a pattern defining the specific TEC. The SIATECCompressSegment approach generates an ordered list of TECs, which may overlap (in contrast to his other related algorithms such as COSIATEC). Recently, Velarde and Meredith [126] extended the approach to melodic segmentation [127] for melodic classification and segmentation, where the symbolic input is first segmented, then compared and hierarchically clustered. Finally, the clusters are ranked, taking into account the cumulative length of all occurrences within each cluster. Based on their results, it can be assumed that the output is additionally filtered by a threshold defining the number of output patterns.

Lartillot [128] introduced the PatMinr algorithm [129] which uses an incremental one-pass approach to identify pattern occurrences. To avoid redundancy, the author addresses two issues: closed pattern mining, which filters out the patterns which have more occurrences than their more specific patterns, thus providing more robust patterns, and pattern cyclicity, which removes redundant matches for successive occurrences of a single underlying pattern. The most recent approach submitted to the MIREX task by Ren [130] also employs a closed pattern approach commonly used in data mining. Nieto [131] proposed the MotivesExtractor which obtains a harmonic representation of the audio or symbolic input and extracts patterns based on a produced self-similarity matrix. Using a score-based greedy algorithm [132] the approach extracts repeated segments, allowing the patterns to overlap. Finally, the segments are grouped into clusters and provided in the algorithm's output as patterns.

To our knowledge, neural-networks have not yet been applied to this task. Such models perform well for classification tasks; however, for pattern extraction, an obvious obstacle lays in their black-box representation of knowledge, which makes it difficult to extract the learned concepts.

2.3.4 Melodic similarity

The concept of similarity in music has been explored in the context of different research areas: cognitive science, musicology, music cognition and music information retrieval. Music similarity is closely related to pattern discovery and identification, and the question of whether two music pieces are similar is relevant in many scenarios, such as song identification, classification, systematization, categorization.

In this dissertation, we apply our model to the task of categorization of folk songs into tune families, where a tune family represents ”a set of folk songs which have a common origin in history” [133].

Some of the best performing algorithms for this task are based on alignment algorithms; one of the first approaches was presented by Mongeau and Sankoff [134]. The alignment algorithms and profile modeling for classification and retrieval tasks were also applied to pop and rock songs datasets [135]. Walshaw [136] investigated enhancements of well-established local alignment algorithms to classify Dutch songs [137] into tune families. Bountouridis et al. [138] explored biologically-inspired techniques, which originate in the field of bioinformatics, for MIR tasks. They identified several shared concepts between music and bioinformatics, such as melody (DNA), oral transmission (evolution), variations (homologues), tune families (homology) etc., and showed that bioinformatics algorithms are applicable to tasks dealing with music similarity. Savage et al. [139] also used an adapted alignment algorithm from bioinformatics to classify songs into four diverse tune families (two English, two Japanese).

Many approaches for classifications into tune families were evaluated on the Dutch folk song dataset compiled by van Kranenburg et al. [140]. Among the most recent, the alignment approach [137] produces the best classification accuracy. This approach models various features of music as substitution scoring functions, which are incorporated in the Needleman-Wunsch-Gotoh [141] algorithm. The model employs several 'viewpoints', such as pitch, duration, score time, time in bar, onset, current bar number, current phrase number, upbeat, current meter, free meter, accented, inter-onset-interval ratio, normalized metrical weight and time position within phrase. Van Kranenburg had analyzed the combinations of these attributes and discovered that the best results were obtained with pitch and position within phrase attributes. Despite the high accuracy, the metadata used as features in this dataset are rarely available in music collections. To eliminate the need for expert knowledge, Velarde et al. [142] classified Dutch songs using Haar-wavelet filters. The results are not on a par with Van Kranenburg's [140], however the approach does not require any encoded expert knowledge.

2.3.5 *Rhythm*

Rhythm, melody and harmony represent the main music modalities. Rhythm is directly related to tempo; moreover, rhythm may affect and change the perception of tempo without changing the latter. Rhythmic patterns significantly affect both the melodic and harmonic aspects of a music piece. By changing the rhythmic patterns underneath, two versions of a song may be classified into different genres and imply different dancing styles.

The perceptual aspect of rhythm is complex. The rhythmic structures represent the base for one's perception of the song's structure through segmentation and repetition. As with harmonic and melodic perception, the listener's music knowledge aids their perception and understanding of music. Schaal et al. [143] explored the differences in memory capacity between musicians and non-musicians in a rhythm memory task. They showed that the musicians perform significantly better than non-musicians. De Fleurian et al. [144] addressed rhythm perception by proposing five measures from information theory and algorithmic complexity to measure rhythmic complexity. The human judgment of the latter was evaluated by comparing formal complexity measurements to judgments of human listeners on a novel rhythm perception task. Results showed the influence of musical expertise on complexity judgments.

Several MIREX tasks related to rhythm have been proposed, such as genre estimation, tempo estimation, beat tracking and downbeat estimation. The audio genre classification task is closely related to rhythm, since rhythmic patterns represent one of the key features for differentiation between music genres. For example, already in 2004 Dixon et al. [145] tackled the problem of dance music genre classification by identifying different patterns, which define each music genre. They evaluated their approach on the Ballroom music dataset, which is distributed into 8 music genres: Jive, Cha cha, Quickstep, Rumba, Samba, Tango, Viennese Waltz and (English) Waltz. They showed that the rhythmic patterns are a useful feature for genre classification. The tempo estimation task, as one of the first MIREX tasks, is also closely related with the rhythmic aspect of music. In recent years, deep learning has been used for tempo estimation. For example, Böck et al. [146] proposed an approach based on recurrent neural networks in combination with comb filters. As a generalization of tempo estimation, the goal of beat tracking is to identify the positions of beats in the audio. Although the task seems relatively trivial, current F-measures of the best approaches on different datasets still only reach around 0.6 (MIREX 2017), so a lot of room for improvement still remains [147]. Derived from the latter, the goal of downbeat estimation is to identify the first beats within each measure. To reduce the dominance of 3/4 and 4/4 meters prevalent in Western music, several non-Western music datasets are used for evaluating algorithms for this task, including the Turkish, Cretan and Carnatic datasets. Due to the strong interrelationship of meter, beat and tempo, several approaches attempt to model more than one aspect of rhythm. For example, Krebs et al. [148] proposed a Hidden Markov model-based system, which they applied to beat tracking and downbeat estimation. They also evaluated the results on the Ballroom dataset. Esparza et al. [149] proposed a neural network for rhythm genre classification and evaluated it on the Latin Music Dataset where they achieved state of the art results. They further explored the underlying rhythmic structures and pointed out several limitations in the dataset. Their research also showed, that a single genre is not necessarily defined by a single specific rhythm in dance music due to inter-genre influence.

Since rhythm in non-Western music contains a larger variety of different meters, several works have been dedicated specifically to the exploration and rhythm modeling in non-Western music. For example, Holzapfel [150] observed the rhythmic patterns (usul) in Turkish Makam music. He investigated how these rhythmic events are related to note events and what can be inferred from these results regarding meter as a latent mental construct. By investigating the rhythmic patterns in a large corpus of Turkish music he proposed a methodology capable of identifying differences between Western and Turkish music, and applied a maximal likelihood criterion for rhythm classification. In a similar manner, London et al. [151] explored African rhythm patterns, focusing on three different music pieces. They also compared their data to Turkish Makam music [150]. Panteli and Dixon [152] investigated the invariance of audio features for description of rhythmic content of diverse music styles, including Afro-American, North-Indian, African, Latin-Brazilian and classical music styles.

3

Model Definition

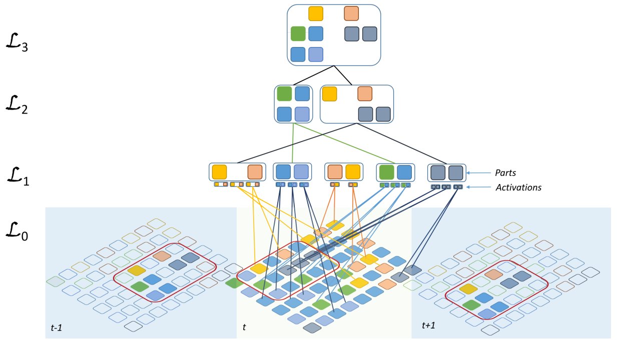
In this chapter we describe the proposed compositional hierarchical model for music processing. The model can learn a hierarchical representation of music in an unsupervised manner, starting from the simple components on the lowest layer, up to the high-level concepts on the highest layers.

The structure of our model was inspired by research in the field of computer vision, specifically the *learned Hierarchy of Parts* (lHoP) model presented by Leonardis and Fidler [43, 153]. Their model represents objects in images in a hierarchical manner, structured in layers, from simple to complex image parts. The model is learned from the statistics of natural images and can be employed as a robust statistical engine for object categorization and other computer vision tasks. While our model shares the inspiration for its hierarchical composition of structures and statistical learning with lHoP, it was developed from the ground up as a new model that incorporates features specific for music processing.

The proposed model is built on the assumption that a complex signal can be decomposed into a hierarchy of building blocks - *parts*. Parts exist at various levels of granularity and represent sets of entities describing the signal. With regard to their complexity, parts can be structured across layers from the less to the more complex. Parts on higher layers are expressed as compositions of parts on lower layers, analogous to the fact that a chord is composed of several pitches, and each pitch of several harmonic partials. A part can therefore describe individual frequencies in a signal, their combinations, as well as pitches, chords and temporal patterns, such as melodic or chord progressions. The entire structure is *transparent*, so that the role of each part can be observed and interpreted.

3.1 Model structure

The compositional hierarchical model consists of an input layer $\mathscr{L}_0$ and several compositional layers $\{\mathscr{L}_1, \dots, \mathscr{L}_\mathrm{N}\}$. Each compositional layer $\mathscr{L}_n$ contains a set of parts $\{\mathrm{P}_1^n, \dots, \mathrm{P}_\mathrm{M}^n\}$, where a part is a composition of parts from $\mathscr{L}_{n-1}$ and may itself be part of any number of compositions on $\mathscr{L}_{n+1}$. Thus, the compositional model forms a hierarchy of parts, as may be observed in Fig. 3.1, where connections between parts represent the structure of compositions.



Part activations are depicted underneath each part on $\mathscr{L}_1$ and represent occurrences of concepts represented by parts in the input. Due to the relative encoding of parts, a part may activate at different locations within the input music signal. The entire structure of the model is transparent and allows for detailed analysis of the learned abstractions.

Figure 3.1
An abstract representation of the compositional hierarchical model. An abstract representation of the compositional hierarchical model. Colors are used for clarity and represent different types of events in the input signal. Each layer includes a set of parts which are compositions of parts from the previous layer.

3.1.1 Compositional layers

Layers $\{\mathscr{L}_1, \dots, \mathscr{L}_N\}$ contain parts which are compositions of parts from lower layers. Formally, we define composition i on layer n - P_i^n - as:

$$P_i^n = \{P_{k_0}^{n-1}, \{P_{k_j}^{n-1}, (\mu_j, \sigma_j)\}_{j=1}^{K-1}\}. \tag{3.1}$$

P_i^n is a composition of K parts from layer $\mathscr{L}_{n-1}$ - *subparts*. The composition is governed by parameters $\mu_{1,\dots,K-1}$ and $\sigma_{1,\dots,K-1}$, which model relations between subparts. These relations are *relative*, meaning that compositions are always defined by the relative distances (*offsets*) between the subpart $P_{k_0}^{n-1}$ and subparts $P_{k_1}^{n-1}, \dots, P_{k_{K-1}}^{n-1}$. The offsets are encoded by parameters $\mu_{1,\dots,K-1}$ and $\sigma_{1,\dots,K-1}$ and are always defined relative to $P_{k_0}^{n-1}$, which we denote as the composition's *central* part.

The model is automatically constructed by unsupervised learning on a set of examples (see section 3.2) and the learned parts and their parameters encode concepts learned from the examples (e.g. pitches, chords, melodic patterns, rhythmic patterns). When new input is presented to the model, part *activations* are calculated. A part *activation* indicates that the concept it represents has been found in the input signal.

An activation has three components: *location*, which maps the (relatively encoded) part onto the absolute axis (e.g. pitch or scale), thus making it absolute; *time* which represents the absolute time of activation in the input; and *magnitude*, representing the activation's strength. A part can activate only if all of its subparts are activated with magnitude greater than zero (this constraint can be relaxed by the hallucination mechanism defined later in this chapter). Due to the relative encoding of concepts in the model, a part can have many simultaneous activations (at different locations and/or times), indicating that the concept it represents has been found at several locations in the input signal.

More formally, the activation A is defined as a triplet $\langle A_T, A_L, A_M \rangle$ of time, location and magnitude. The activation location A_L and time A_T of part P_i^n are defined as:

$$\begin{aligned} A_L(P_i^n) &= A_L(P_{k_0}^{n-1}), \\ A_T(P_i^n) &= A_T(P_{k_0}^{n-1}). \end{aligned} \tag{3.2}$$

Compositions thus propagate their locations and onset times upwards through the hierarchy. Such propagation can be usefully employed as an indexing mechanism and allows for top-down analysis of activations.

The activation magnitude A_M represents the strength of the composition's match with the input and is defined as a weighted sum of subpart magnitudes:

$$A_M(P_i^n) = \tanh\left(\tfrac{1}{K}\textstyle\sum_{j=0}^{K-1} w_j A_M(P_{k_j}^{n-1})\right), \tag{3.3}$$

where weights w_j are defined by the match between the learned and the observed relative subpart activation location and are bounded by the difference in their activation times:

$$\begin{aligned} w_j &= \begin{cases} 1: & j = 0 \\ \mathcal{N}(\delta_{Lj}, \mu_j, \sigma_j): & j > 0 \wedge \delta_{Tj} < \tau_W \\ 0: & \delta_{Tj} \geq \tau_W \end{cases} . \\ \delta_{Lj} &= A_L(P_{k_j}^{n-1}) - A_L(P_{k_0}^{n-1}) \\ \delta_{Tj} &= A_T(P_{k_j}^{n-1}) - A_T(P_{k_0}^{n-1}) \end{aligned} \tag{3.4}$$

The rationale behind Eqs.3.3 and 3.4 is that the more the input fits the learned subpart differences encoded by μ_j and σ_j, the higher the activation should be. The more the input diverges, the lower the activation. The role of the *tanh* function, which stems from neural-network architectures, is that it guarantees saturated output with the maximum limited to 1. Other activation functions could also be used. We decided to use *tanh*, because it is a monotonically increasing function with smooth gradient and its value approaches 1 when the input goes towards infinity. Since activation magnitudes are directly used to calculate activation magnitudes on higher layers, they need to be normalized.

Parameter τ_W represents the maximal difference between the activation times of two subparts which still produces an activation. Such a limit must be imposed in order to avoid combinatorial explosion in calculation of part activations. Thus, only subpart activations which are *close enough* with regard to τ_W are considered when calculating part activation. Intuitively, it makes sense to limit the distance between related activations, as we also perceive music as a stream of events by combining nearby events and not events spaced far apart. Thus, if subpart activations fall within τ_W, the part activation magnitude is calculated according to the match between the observed (δ_{L_j}) and learned (μ_j, σ_j) relative subpart distances. A part will activate with maximal magnitude when its subparts activate at distances according to the learned representation encoded by μ_j and σ_j.

Note that onset times of events do not directly influence the activation magnitude - it is thus not dependent on the temporal distance between subpart activations (within τ_W) and is the same whether they are adjacent or separated by other events. This becomes a very useful mechanism when modeling temporal signals such as music, as it permits for gaps between detected events and can robustly locate the learned concepts in the presence of other signals.

3.1.2 Input

Any music representation can be used as input to the model, providing that it describes music events consisting of three components: time, a specific location (e.g. frequency, pitch) and magnitude (greater than 0). We therefore define the input representation $\mathcal{I}$ as a set of triplets $\mathbf{X}$:

$$\mathcal{I} : \{\mathbf{X} : \mathbf{X} = [X_t, X_l, X_m]\}. \tag{3.5}$$

In the following chapters, the model will be applied to spectral (time-frequency-magnitude) and symbolic (time-pitch-magnitude) representations.

3.1.3 Input layer

The $\mathscr{L}_0$ layer consists of a single part P_0^0, which represents all atomic events in the model input. P_0^0 activates on all input events, encoding the events' time, location and magnitude. Depending on the input representation and task, a part activation may represent an individual frequency component (transcription), a note event (pattern discovery) or a pair of rhythmic events (rhythm). Formally, the activation of the part P_0^0 is defined as:

$$A(P_1^0) = \langle A_T, A_L, A_M \rangle \leftarrow [X_t, X_l, X_m]. \tag{3.6}$$

3.2 Learning

The model is constructed layer-by-layer with unsupervised learning on a set of training examples, starting with $\mathscr{L}_1$. We view learning as an optimization problem, where we aim to find a minimal set of compositions for the learned layer, which will explain the maximal amount of information present in the input data. The learning process is driven by statistics of part activations which capture regularities in the input data.

To formalize the problem, we first define *coverage* C of a part activation as a set of input events (consequently $\mathscr{L}_0$ activations) which caused the activation. This set can be obtained efficiently by observing the tree formed by the activated subparts through indexing encoded in the locations of their central parts down to layer $\mathscr{L}_0$ as:

$$C(A(P_i^n)) = \bigcup_{j=0}^{K-1} C(A(P_{k_j}^{n-1})). \tag{3.7}$$

Coverage at $\mathscr{L}_0$ is defined by the presence of an event at the given activation as:

$$C(A(P_1^0)) = A(P_1^0) = X : X_t = A_T \wedge X_l = A_L. \tag{3.8}$$

The coverage of an entire layer $\mathscr{L}_n$ is the set of events in the input data, which covered by all the parts in the layer:

$$C(\mathscr{L}_n) = \bigcup_{P_i^n \in \mathscr{L}_n} C(A(P_i^n)) \tag{3.9}$$

The goal of learning of a new layer $\mathscr{L}_n$ is to minimize the amount of uncovered events in the input data and, on the other hand, to limit the number of parts added to the layer, which can be expressed as:

$$min(\sum_i |\bigcup_{X \notin C(\mathscr{L}_n)} X| + \lambda|\mathscr{L}_n|), \tag{3.10}$$

where λ is a regularization factor which balances between the number of parts and the adequacy of the coverage.

The problem of finding an optimal coverage is a special case of the well-known *set cover* problem, which is NP-complete. We therefore approximate the solution by using a greedy algorithm, which incrementally adds compositions to the new layer. With each iteration the algorithm chooses a composition that covers the largest amount of uncovered data. The entire learning algorithm is composed of two steps: finding new candidate compositions and adding compositions to the new layer.

3.2.1 Finding candidate compositions

When a new layer $\mathscr{L}_n$ is learned, we first need to form a set of parts (compositions), which will be considered for inclusion in the new layer. We first perform inference on the training set up to the layer $\mathscr{L}_{n-1}$. Then, we observe co-occurrences of $\mathscr{L}_{n-1}$ part activations over the entire training set. The co-occurrences indicate parts, which frequently activate simultaneously and are thus good candidates for forming compositions, as they are believed to form common concepts.

We calculate histograms of co-occurring activations according to the distances between activation locations for all parts. New compositions are formed from parts where the number of co-occurrences exceeds a learning threshold τ_L. Composition parameters μ and σ are estimated from the corresponding histogram (Fig 3.3) and each new composition is added to the set of candidate compositions $\mathscr{C}$. The pseudo-code of the procedure is shown in Fig. 3.2.

3.2.2 Selecting compositions

Due to the NP-completeness of the set cover problem, we use a greedy approach to select a subset of compositions from the set of candidates $\mathscr{C}$, which leaves a minimal amount of information in the training set uncovered (according to Eq.3.10).

Based on coverage, the greedy part selection algorithm is as follows:

```
procedure CANDIDATECOMPOSITIONS($\mathscr{L}_n$)
$\mathscr{C} = \{\}$
for $(\mathrm{P}_1, \mathrm{P}_2) \in \mathscr{L}_{n-1} \times \mathscr{L}_{n-1}$ do
    $hist \leftarrow array(maxSize)$
    for $Act_1 \in \mathrm{P}_1$ do
        for $Act_2 \in \mathrm{P}_2$ do
            if $withinWindow(Act_1, Act_2)$ then
                $loc \leftarrow Act_2[\mathrm{A_L}] - Act_1[\mathrm{A_L}]$
                $hist[loc] \leftarrow hist[loc] + Act_1[\mathrm{A_M}] + Act_2[\mathrm{A_M}]$
            end if
        end for
    end for
    $peak\ getspeakPick(hist, \tau_\mathrm{L})$
    while $peak \neq \emptyset$ do
        $[\mu, \sigma] \leftarrow estimateGaussian(hist, peak);$
        $\mathscr{C} \leftarrow \mathscr{C} \cup newPart(\mathrm{P}_1, \mathrm{P}_2, \mu, \sigma)$
        $hist \leftarrow removeFromHist(hist, peak, \mu, \sigma)$
        $peak \leftarrow peakPick(hist, \tau_\mathrm{L})$
    end while
end for
return $\mathscr{C}$
```

Figure 3.2
The algorithm for generating histograms and candidate compositions

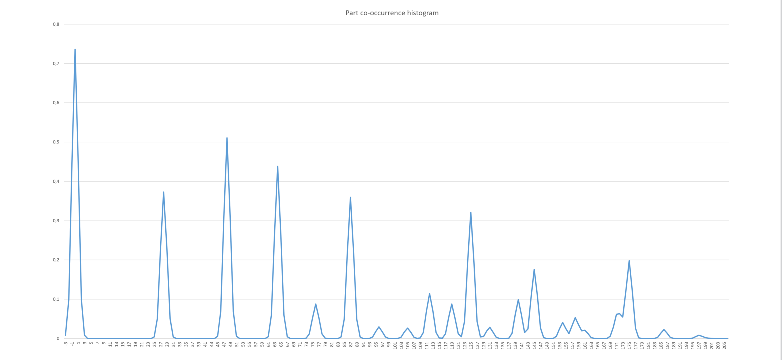


Figure 3.3
Co-occurrence histogram for a $\mathscr{L}_2$ part in the spectral CHM. The normalized co-occurrence histogram represents the distribution of the distances (offsets) of $\mathscr{L}_2$ subparts that activate simultaneously. The distances are shown relative to a chosen central $\mathscr{L}_2$ part.

- the coverage of each part from $\mathscr{C}$ is calculated according to Eq.3.7,
- parts are iteratively added to the new layer $\mathscr{L}_n$ by choosing the part that *adds* most to the coverage of the entire training set in each iteration (according to Eq.3.9). This ensures that only compositions that provide enough coverage of *new* data with regard to the currently selected set of parts will be added,
- the algorithm stops when the added coverage falls below the learning threshold τ_C or the overall coverage reaches the threshold τ_P.

The learning procedure is repeated for each layer until a desired number of layers is reached. The desired number of layers is dependent on the underlying problem. The reader should also note that the number of layers governs the maximal length of encoded concepts, as discussed in evaluation.

A formal definition of the learning algorithm is provided in Fig. 3.4.

3.3 *Inference*

A trained model captures the repetitive concepts in the training data, which are relatively encoded and may be observed through the inspection of the model's parts on its various layers. When a trained model is presented with new input data, the learned concepts may be located in the input through the process of *inference*. The inference calculates part activations on the input data according to equations 3.2 and 3.3. They are calculated bottom-up layer-by-layer, whereby the input data activates layer $\mathscr{L}_0$. As already mentioned, activation of a part represents a specific occurrence of the concept it represents in the input. An activation has three components: *location* and *onset time*, which map the relative concept onto a specific set of values within the input sequence of events (thus making it absolute), and *magnitude*, representing its strength. A part can concurrently activate at several different locations, which indicates multiple occurrences of the represented concept in the input representation.

Inference may be exact or approximate, whereby in the latter case two additional mechanisms, *hallucination* and *inhibition*, enable the model to find the learned concepts also when its input is noisy or contains changed, added or deleted events. In this way, the model's predictive power and robustness are increased.

```
procedure SELECTCOMPOSITIONS($\mathscr{C}$)
$prev\mathrm{Cov} \leftarrow 0$
$cov \leftarrow \emptyset$
$\mathscr{L}_n \leftarrow \emptyset$
$sum\mathrm{Input} \leftarrow |\mathscr{I}|$
repeat
    for $\mathrm{P} \in \mathscr{C}$ do
        $c \leftarrow 0$
        $\mathscr{F} \leftarrow coverage(\mathscr{L}_n \cup \mathrm{P})$
        $c \leftarrow c + |\mathscr{F}|$
        $cov[\mathrm{P}] \leftarrow c / sum\mathrm{Input}$
    end for
    $Chosen \leftarrow \underset{\mathrm{P}}{\mathrm{argmax}}(cov)$
    $\mathscr{L}_n \leftarrow \mathscr{L}_n \cup Chosen$
    $\mathscr{C} \leftarrow \mathscr{C} \square Chosen$
    if $cov[Chosen] - prev\mathrm{Cov} < \tau_{\mathrm{C}}$ then
        break
    end if
    $prev\mathrm{Cov} \leftarrow cov[Chosen]$
until $prev\mathrm{Cov} > \tau_{\mathrm{P}} \vee \mathscr{C} = \emptyset$
return $\mathscr{L}_n$
```

Figure 3.4
The greedy algorithm for the selection of compositions from the candidate set $\mathscr{C}$. Compositions that add the most to the coverage of information in the learning set are prioritized.

3.3.1 Hallucination

When calculating activations, the default model behavior is very conservative—a part is activated only if all of its subparts are activated. *Hallucination* relaxes this condition and enables the model to produce activations even in the case of incomplete (missing, masked or damaged) input. The model generates activations of parts, which most fittingly cover the information present in the input signal, where fragments, which are not present, are ”hallucinated”. The missing information is thus extrapolated from the knowledge acquired during learning, encoded into the model structure.

Hallucination changes the conditions under which a part may be activated. With hallucination, the part P_i^n is activated when the percentage of events in the input signal it

```
procedure INFERENCE($\mathscr{L}_n$)
$\mathscr{A} \leftarrow \emptyset$
for $\mathrm{P} \in \mathscr{L}_n$ do
    $(\mathrm{P}_0^{n-1}, \mathrm{P}_1^{n-1}, \mu, \sigma) \leftarrow \mathrm{P}$
    for $\mathrm{A}ct_0 \in \mathrm{P}_0^{n-1}$ do
        for $\mathrm{A}ct_1 \in \mathrm{P}_1^{n-1}$ do
            if $is\mathrm{W}ithin(\mathrm{A}ct_1, \mathrm{A}ct_2, \mu, \sigma)$ then
                $\mathrm{A} \leftarrow new\mathrm{A}ctivation()$
                $\mathrm{A}[\mathrm{A_L}] \leftarrow \mathrm{A}ct_0[\mathrm{A_L}]$
                $\mathrm{A}[\mathrm{A_T}] \leftarrow \mathrm{A}ct_0[\mathrm{A_T}]$
                $\mathrm{A}[\mathrm{A_M}] \leftarrow calc\mathrm{M}agnitude(\mathrm{A}ct_0, \mathrm{A}ct_1, \mu, \sigma)$
                if $exceeds\mathrm{H}allucination(\mathrm{P}, \mathrm{A}, \tau_\mathrm{H})$ then
                    $\mathscr{A} \leftarrow \mathscr{A} \cup \mathrm{A}$
                end if
            end if
        end for
    end for
end for
$\mathscr{A} \leftarrow inhibit(\mathscr{A}, \tau_\mathrm{I})$
return $\mathscr{A}$
```

Figure 3.5
The inference algorithm returning the set of activations of compositions on layer $\mathscr{L}_n$.

covers exceeds a hallucination threshold τ_H, which can be defined for each layer separately:

$$\frac{|\{\mathrm{X} : \mathrm{X} \in \mathrm{C}(\mathrm{A}(\mathrm{P}_i^n)) \wedge \mathrm{X}_m > 0\}|}{|\mathrm{A}(\mathrm{P}_i^n)|} \geq \tau_\mathrm{H}. \tag{3.11}$$

If we set τ_H to 1, we obtain the default behavior (all of the events that a part represents must be present in the input for the part to activate). By lowering the parameter value, we increase the number of activations in each layer, as parts are also activated when their input is incomplete.

By allowing activations in the presence of incomplete input, hallucination not only enables the model to fill-in the missing information, but also to yield alternative explanations of the input. Namely, different parts of the model can explain the same fragments of information in the input. Hallucination boosts these alternative representations and enables the model to produce multiple explanations of the same input.

3.3.2 *Inhibition*

Inhibition performs hypothesis refinement by reducing the number of part activations on individual layers. It provides a balancing factor in the model by reducing redundant activations, similar to lateral inhibition in the human auditory system [154]. Although the learning algorithm penalizes parts redundantly covering the signal, some redundant parts are always present. During inference, each layer may therefore produce multiple redundant activations covering the same information in the input signal (hallucination also adds to the number of such activations).

An activation of the part P^n_i is *inhibited* (removed), when one or multiple parts $P^n_{j_1}, \dots, P^n_{j_K}$ already cover most events that the part P^n_i covers, but with stronger magnitude. More formally, the activation of part P^n_i is inhibited when the following conditions are met:

$$\exists\,\{P^n_{j_1} \dots P^n_{j_k} \dots P^n_{j_K}\} : \frac{|C(A(P^n_i)) \setminus \bigcup_{k=1}^{K} C(A(P^n_{j_k}))|}{|C(A(P^n_i))|} < \tau_I \tag{3.12}$$

and

$$\forall P^n_{j_k} \in \{P^n_{j_1} \dots P^n_{j_k} \dots P^n_{j_K}\} : A_M(P^n_{j_k}) > A_M(P^n_i). \tag{3.13}$$

$C(A)$ represents activation coverage (Eq.3.7), A_M the activation magnitude (Eq.3.3) and τ_I a parameter that controls the strength of inhibition. If the τ_I is set to zero, no inhibition occurs; the larger its value, the more activations are inhibited and fewer are propagated to higher layers. Notably, only activations with magnitudes larger than that of the inhibited part P^n_i are considered in the inhibition process.

Next to reducing the number of activations, the inhibition mechanism can also be used for producing alternative explanations of the input. If activations of the strongest part, which inhibits other competing hypotheses, are removed from the model, the next best hypothesis is selected during the inference, thus providing an alternative explanation of the input through activations of different parts.

Alongside the hypothesis refinement, the removal of redundant activations also reduces noise in the input, which is usually manifested in a number of low-magnitude activations of parts on various layers. In combination with hallucination, the inhibition process provides an efficient way to control the explanatory power and robustness of the proposed model.

3.4 *Relativity and shareability*

The proposed model has two important features that set it apart from similar architectures.

The *relativity* of parts enables a single part to represent an abstract high-level concept regardless of its location in the input signal. Relative perception naturally occurs in the human learning process. It is an important part of the abstraction of the object of interest, and enables the formation of a complete percept, regardless of its environment. It minimizes the amount of memory needed to store the learned concepts and enables their robust identification in previously unobserved sensory inputs, such as within noisy audio signals and in the presence of non-musical events.

Relativity is inherent in our model and can be observed in the definitions of part composition and activation (Eqs 3.1 and 3.3). Although the parts are relative and only represent abstract concepts with no direct absolute representation (e.g. the model cannot encode the pitch G5 explicitly, but only the concept of pitch), the part's activation indicates where and when its encoded concept appears in the signal. Since this can occur at several locations, a part can have multiple activations at different locations.

This is also shown in Fig. 3.1, where $\mathscr{L}_1$ parts have several activations, meaning that the concepts they represent are present at several locations in the input.

The relative nature of the parts also enables efficient *shareability* of the parts. A part on layer $\mathscr{L}_{n-1}$ may be a subpart of several compositions on layer $\mathscr{L}_n$. Consequently, any two or more $\mathscr{L}_{n-1}$ parts may form a number of different $\mathscr{L}_n$ compositions at different offsets μ. Thus, they may be combined into several more complex abstractions, themselves relative.

The consequence of relativity and shareability is that the model can very efficiently encode complex concepts. As an example: a part representing the concept of a pitch may be shared by several compositions on a higher level that encode different intervals. This encoding is general, compact and efficient if we consider the alternative of encoding all the intervals in an absolute manner. This is also evident in the evaluation of the proposed model, where a learned hierarchy with a small number of compositions is shown to be robust and to generalize well in modeling musical events in audio signals, which differ from the ones used for training in quality, the amount of noise and the number and the type of sources present in the signal.

4

The Compositional Hierarchical Model for Time-Frequency Representations

4.1 Model Description

In this chapter, we present the implementation of the compositional hierarchical model for processing audio recordings, where its input consists of a time-frequency representation of the input audio signal. The input $\mathcal{I}$ thus consists of frequency components from a time-frequency representation of an audio signal. The atomic part $P_1^0 \in \mathcal{L}_0$, activates on all time-frequency components, thus:

$$A(P_1^0) = \langle A_T, A_L, A_M \rangle \leftarrow [X_t, X_l, X_m], \tag{4.1}$$

where X_t represents the time-frame, X_l represents the frequency bin, and X_m the magnitude of the frequency bin X_l at time X_t.

In this way, the compositions on higher layers combine the individual frequency components into more complex units, learning to compose the components into harmonic templates, tones, intervals and chords. All compositions and their parameters are learned in an unsupervised manner according to the algorithm presented in the previous chapter.

A simple depiction of a learned model is shown in Fig.4.1. Four layers of the model are shown, including $\mathcal{L}_0$, which represents individual frequency components. The parts on higher layers combine these components according to their learned parameters μ and σ which govern the relative distances between components (cents are used instead of Hz to encode frequencies, as their differences are octave independent).

For example, P_2^2 is defined as:

$$P_2^2 = \{P_1^1, \{P_3^1, (1200, 25)\}\}, \tag{4.2}$$

where μ and σ are given in cents. It thus represents a composition of P_1^1 with P_3^1 spaced approximately 1200 cents (one octave) apart, where σ governs the allowed deviation from this value.

Relativity of the encoded structures has a large effect on the size of the learned hierarchy. Since all relationships in the model are relatively encoded, rather than having encoded the specific instances of a music concept (e.g. the tone A5), the proposed model learns generalized concepts (e.g. a tone is a set of frequency components at some relative positions). This leads to small models which can learn on small datasets.

The mapping from relatively defined to absolutely positioned concepts (e.g. a generalized tone concept to tone A5) is performed during inference, by calculating part activ-

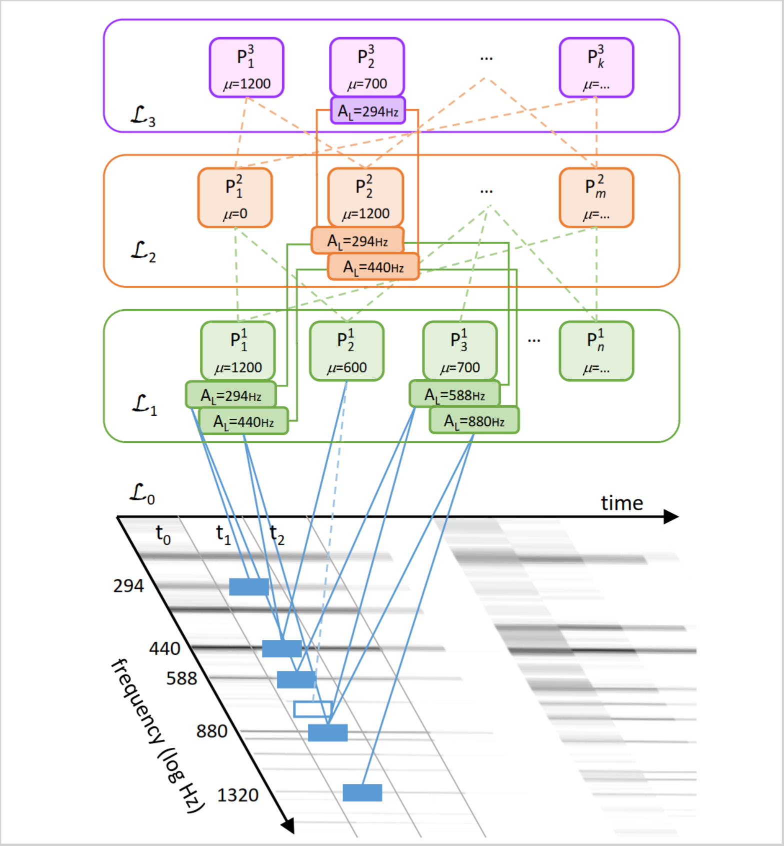


Figure 4.1
The compositional hierarchical model for time-frequency representations. The input layer corresponds to the signal components in the time-frequency representation. The parts on higher layers are compositions of the lower-layer parts (depicted as connections between parts, parameter μ is given in cents). A part may be contained in several compositions, e.g. P_1^1 is contained in compositions P_1^2, P_2^2 and P_m^2. Active parts have activation locations displayed underneath; a part can have several activations on different locations. The entire structure is transparent, thus we can discern that the activation of P_2^2 at 440Hz represents the harmonic series of 440Hz, 880Hz and 1320Hz by observing the subtree leading from the activation down to $\mathcal{L}_0$.

ations upwards through all the layers. In Fig. 4.1, two activations of P_2^2 are shown: one for subparts P_1^1 and P_3^1 at 294 Hz and 588 Hz (corresponding to tone C4); and one for the subparts at 440 Hz and 880 Hz (corresponding tone A4). The same part is thus responsible for representing the series of frequency components over the entire range of pitches.

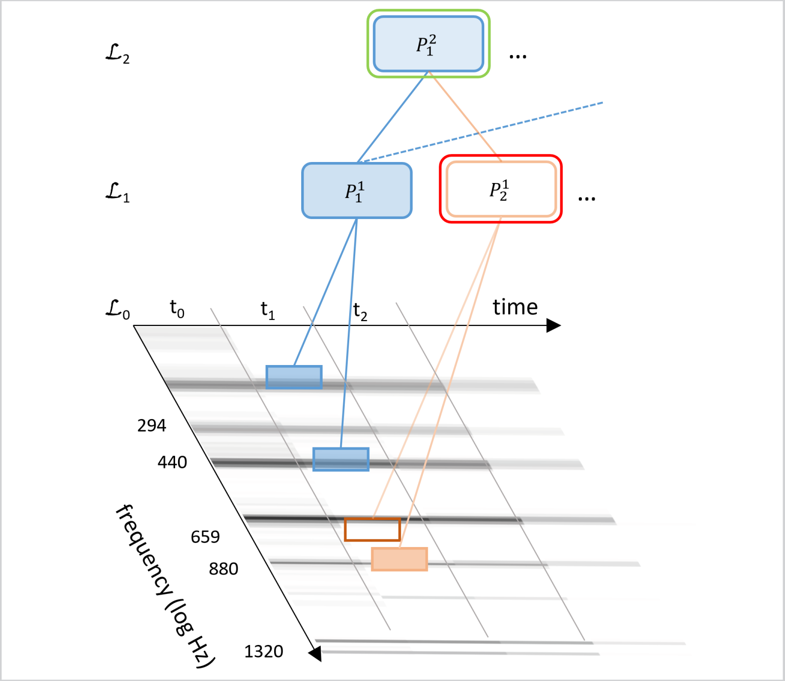


Figure 4.2
Depiction of the hallucination mechanism. The description of the figure is given in the text.

4.1.1 Mechanisms

Three inference mechanisms are used in this version of the model: hallucination, inhibition and automatic gain control.

The hallucination and inhibition mechanisms are applied within each time-frame independently and are implemented as described in the previous chapter. The effect of hallucination is shown in Fig.4.2, where parts P_1^1 and P_2^1 compose part P_1^2. At time-frame t_1 only part P_1^1 is activated, whereas part P_2^1 fails to generate an activation due to the missing frequency component at 659Hz. Nevertheless, part P_1^2 (marked in green) still produces an activation if hallucination is used and the threshold τ_H is set to a value lower than 0.75, as in this case enough of the covered signal will be present in the input (3 out of 4 frequency components).

The inhibition mechanism is employed mainly for two reasons: it removes the com-

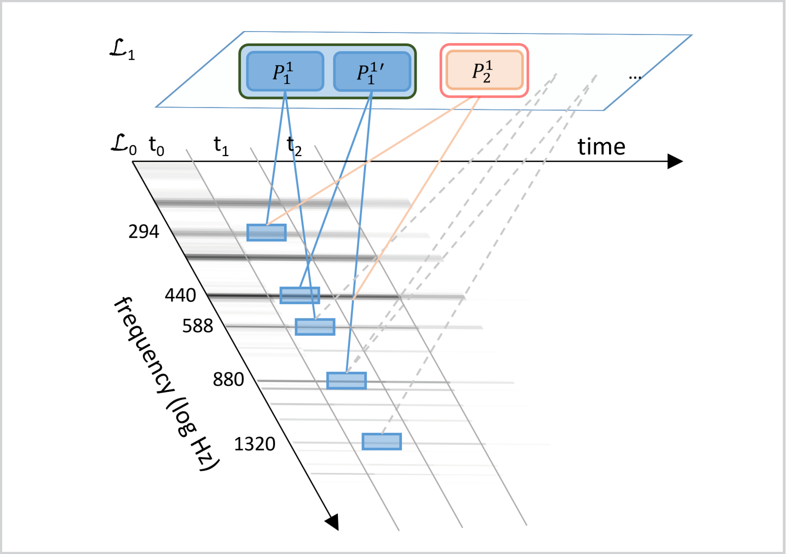


Figure 4.3
Depiction of the inhibition mechanism. Description of the figure is given in the text.

peting hypotheses and thus prevents noisy low-magnitude activations to propagate to higher layers, while also providing a counter-balance to the hallucination mechanism. An example is shown in Fig.4.3, which shows that part P_1^1 was activated at two locations: 294Hz and 440Hz. Concurrently, part P_2^1 activates at 294 Hz, covering the frequency components at 294 Hz and 440 Hz. As its input is already covered by the stronger P_1^1 part, the activation of P_2^1 is inhibited.

Automatic gain control The presented implementation of the model is time-independent, it processes each time-frame independently from the others. To model short-time dependencies between frames, we introduce an additional mechanism named automatic gain control (AGC). It operates on principles similar to automatic gain control contrast mechanism in human [155] and animal [156] perceptual systems. The mechanism in our model allows for linking of part activations through time by introducing time dependencies between activations.

The AGC influences the activation of a part in the following manner: when the part is activated at a new location and its activation persists, the activation magnitude is initially

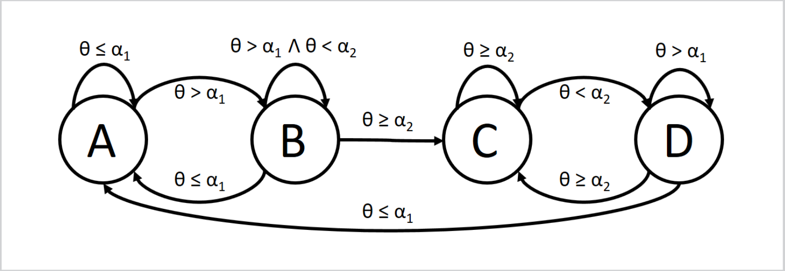


Figure 4.4
A finite state machine implementing the AGC mechanism. State A represents the normal behavior of a part, state B the boosting (onset), state C the sustain and state D the decay of activation magnitude.

boosted to accentuate the onset and later suppressed towards a stable value (see Fig 4.5). The AGC is defined with a four-state finite state machine, as shown in Fig 4.4. The four AGC states represent: (A) normal part behavior, (B) onset, which boosts activations, (C) sustain, which keeps activations at a steady state and (D) decay, which leads to the normal state.

The transitions between states are conditioned on the density of part activations θ within the time window W. The density for part P_i^n at time t is defined as:

$$\theta = \frac{1}{W} \left\| [A_M^{(t-W+1)}(P_i^n), \dots, A_M^{(t)}(P_i^n)] \right\|_0 . \qquad (4.3)$$

α_1 and α_2 are thresholds that control the transitions between states.

Given the four state model, the magnitude of a part activation for individual states is calculated as:

$$A_M^{(t)}(P_i^n) = \begin{cases} A_M^{(t)}(P_i^n) : & A, D \\ \sum_{f=t-W+1}^{t} A_M^{(f)}(P_i^n) : & B \\ \tau_S : & C \end{cases} , \qquad (4.4)$$

where τ_S represents the constant activation magnitude in the sustain state. Thus, for states A and D, the magnitude is not influenced by the AGC mechanism. When the density of activations exceeds α_1, magnitudes are boosted by accumulating them within the time window W. When the density exceeds α_2, the sustain state is reached and constant magnitudes are produced. Finally, when magnitudes fall again, the decay state is reached, where the AGC model may return back to the sustain state or stop and return to the normal state.

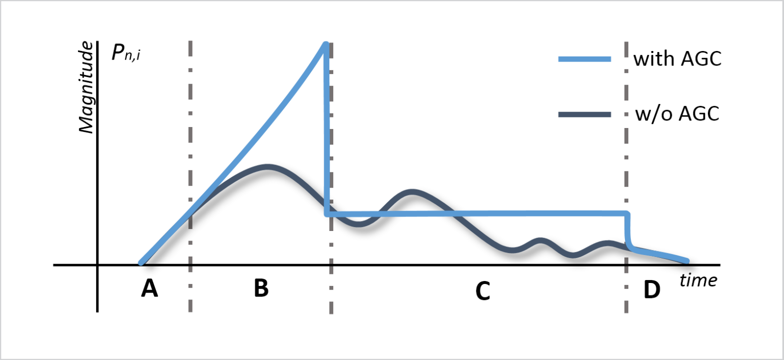


Figure 4.5 An abstract representation of the AGC influence on part activations. Without the AGC, the activation magnitudes may notably fluctuate, especially towards the end of an event. The AGC boosts the onset of an event and later keeps the activation magnitude on a fixed level until the offset.

The mechanism operates on all layers; it has a short-term effect on lower layers and longer-term effect on higher layers (the window size W increases for each consecutive layer) in line with the complexity of concepts represented on different layers. The mechanism's effect on the activation magnitude is shown in Fig 4.5. The AGC stabilizes activations, boosts event onsets and produces an overall smoother model output with less fluctuations.

4.2 Evaluation

The proposed model is applicable to various MIR tasks in the audio domain. In this section, we demonstrate its usefulness for two MIR tasks: automated chord estimation, as a proof of the concept (presented at ISMIR 2014 [157]) and multiple fundamental frequency estimation (presented in the PLOS ONE Journal paper [158]).

4.2.1 Automated chord estimation

According to MIREX, the goal of this task is to transcribe a sequence of chords from a music recording. Chords are a useful representation for a variety of other tasks, from music segmentation and music similarity to genre classification. Moreover, chord progressions are mid-level features which can, when combined with a melody, be used as a basic symbolic representation of a piece. Different approaches use different chord representations, from basic major/minor chords, to the full characterization of chords—root, quality, and bass note. In our experiments, we used the basic major/minor representation.

To obtain the input representation for our model, the audio signal was transformed with the constant-Q transform of 345 frequency bins spaced 25 cents apart between 55 and 8000 Hz, with a step size of 50 ms and the maximal window size of 100 ms. The transformed signal was used as the input to the model.

The rationale for the chosen parameters is as follows. The constant-Q transform was chosen to provide a logarithmically-spaced time-frequency representation, which corresponds more to the way the human ear perceives pitches. 25 cent spacing was chosen as it yields an appropriate frequency resolution also for non-ideally tuned instruments (such as the human voice), where frequency components are not perfectly aligned with any standard tuning scale and fluctuate a lot. The frequency range was chosen to cover the range used by most instruments (e.g. the piano ranges from 27.5 Hz to 4200 Hz), while the window sizes chosen are somehow a standard compromise between time and frequency resolution (e.g. many other music processing approaches use similar values). The maximal window size of 100 Hz limits the resolution between neighboring frequency bins of the FFT to 10 Hz, which is not enough to resolve neighboring tones in the low frequencies, however we decided not to extend the windows on account of improved time resolution. Due to its relativity, the model learns generic relative structures, which are valid across the entire frequency range, and thus learning is not much affected by this choice. Inference is, and especially with chord estimation, where time resolution is not so important, accuracy might be improved by tuning the window sizes.

Several layers of compositions were then learned in an unsupervised manner, as described in the previous chapter. The values of model parameters are listed in Table 4.1.

To train the model, we performed several training runs, each with a different dataset, including a subset of the Beatles dataset[1], the Queen dataset[2], a Slovenian folk song dataset, several piano pieces and single piano notes from the MAPS dataset. Our first finding was that the models do not differ very much, regardless of the dataset used. Each learned model contained a few tens of parts per layer.

The structure of parts on the first two layers corresponded to different relatively-encoded harmonic series forming a general concept of pitch. For higher layers, we expec-

[1] http://www.isophonics.net/content/reference-annotations-beatles

[2] http://www.isophonics.net/content/reference-annotations-queen

Table 4.1
Parameter values for the ACE experiment

Parameter	Description	Value
τ_H	Hallucination parameter that allows for incomplete input	0.5
τ_I	Inhibition parameter reducing the number of competing activations	0.5
τ_C	Learning threshold for added coverage which needs to be exceeded in order for a candidate composition to be retained while learning the model	0.005
τ_P	Learning threshold for cumulative coverage which, when exceeded, stops the candidate selection procedure	0.92
τ_W	Window size in the input layer	$100ms$

ted to find compositions that would correspond to more complex entities, such as intervals or chords. However, contrary to what we expected, the structures on layers $\mathscr{L}_3$ and $\mathscr{L}_4$ still corresponded more or less to pure harmonic series with a varying amount of higher partials—only very few compositions corresponded to intervals. This behaviour emerged due to the statistical nature of learning—namely, pure harmonic series are much more common than interval combinations, so it is logical that the model tries to build more and more complex (and common) harmonic series instead of the (rarer) interval or chord structures.

We could add additional rules to avoid such behavior when learning higher layers; however, we decided not to follow this path and to retain a more general form of our model and test it in a different setting. For our further experiments, we trained two layers of the model by using a small set of 88 piano keys as our training set. Such a model did not differ much when compared to (the first two layers of) the models trained on more complex music. Due to the relativity and shareability of parts, the first two layers contained only 23 and 12 parts respectively.

We used this model as a feature generator for chord classification. As our representation of pitch is relative, we added an additional *octave-invariant* third layer, which mapped $\mathscr{L}_2$ activations onto a chroma-like octave invariant feature vector according to their (octave-wrapped) locations. For the ACE task, the $\mathscr{L}_3$ layer was used as the output of the model. The model thus produced a single 12-dimensional vector per time-frame, where the magnitudes of multiple activations of a single $\mathscr{L}_3$ part were summed up.

To assess the performance of our model for chord estimation, we used the layer $\mathscr{L}_3$ features as inputs to a Hidden Markov model (HMM) for decoding chord labels. The process used in this experiment is outlined in Fig.4.6. We decided to use the basic 24 chord labels for the 12 major and 12 minor keys. The architecture is similar to most chord estimation approaches (until the arrival of deep architectures) which were based on HMMs, commonly decoding chord labels from chroma-like features.

To evaluate the model, we used the standard *Beatles* dataset, publicly provided by C. Harte. We used activations of the octave-invariant $\mathscr{L}_3$ layer as features and made the classification by using a Hidden Markov model with 24 states, each representing a chord, as described by [85]. We used cross-validation for evaluation; one album was used for HMM training and the rest of the dataset for estimation. The initial values of the HMM transition matrix were based on the doubly-nested circle of fifths, as proposed in previous works [16, 85]. Initial state probabilities were the same for all chords.

Our per-frame classification accuracy on the given dataset was 69.45% with 14.94% standard deviation. Compared to other per-frame approaches, we find our results comparable to Papadopoulos et al. [16] and Bello et al. [85]. The latter evaluated their approach on the first two albums of the dataset and reported an average accuracy of 66.87 percent, while our approach achieves 69.78 percent on these two albums. We must also note that the window sizes used for this experiment significantly differ among the three approaches. The approach by Bello et al. used the window size of 743 ms, while Papadopoulos reported the window size of 480 ms. On the other hand, our maximal window size was 100 ms, which leads to improved temporal resolution, but is also more sensitive to the variations of the input. The segments in chord progressions are usually longer, up to several seconds, so the results could potentially improve if a larger window was used. Larger windows could lower the accuracy around chord changes, while increasing the accuracy within each segment.

The results achieved and reported here were gathered while implementing the proof of concept of the proposed model in the early beginning of this research. It is important to stress the progress of other deep learning approaches in recent years. Korzeniowski and Widmer [88] achieved most of the highest results at MIREX evaluation in 2016

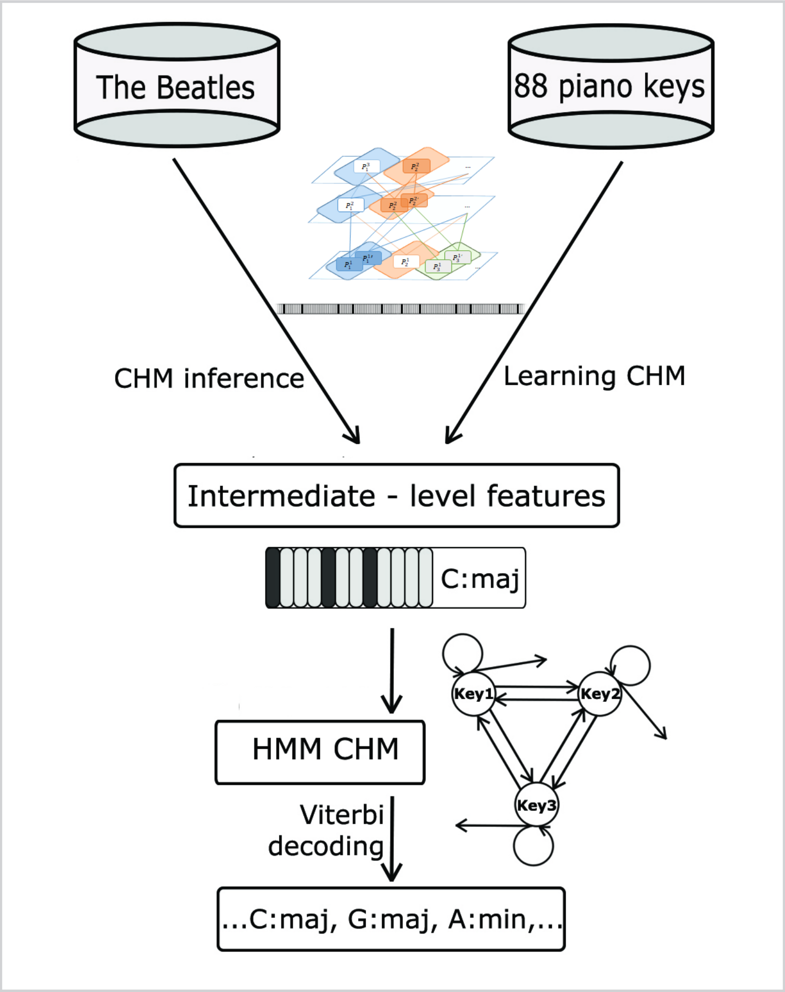


Figure 4.6 An outline of the chord estimation experiment. The CHM is trained on a small database of 88 piano keys and used as feature extractor for chord estimation on *The Beatles* dataset. The intermediate-level features are provided as input to the HMM. Chord labels are estimated with Viterbi decoding.

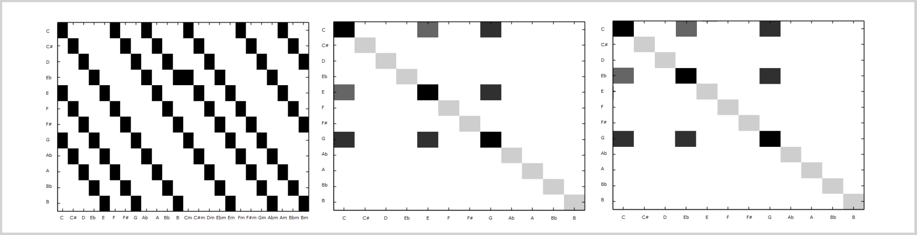


Figure 4.7

Initial values of the μ matrix (left) and σ matrix for C major and minor chord (center and right). Each row of the μ matrix represents a chord template (tonic, mediant and dominant) for each key starting with C major to B major and C minor to B minor.

and 2017, using a deep neural network, on different datasets, including the Isophonics, Billboard, RWC Pop, US Pop and Robbie Williams collections. McFee and Bello [159] proposed a deep convolutional-recurrent model and applied it to a dataset of 1217 pieces, aggregated from the Isophonics, Billboard, RWC Pop, and MARL datasets. The annotations contain 170 different classes, including augmented, diminished, seventh and other chord types. Their approach surpassed the baseline methods.

Although we did not reach our initial goal of forming interval and chord-level hierarchies with the chord estimation experiment, the model turned out to be a robust pitch estimator, which can learn features superior to manually derived features, such as chroma vectors for chord estimation. In our further experiment, we decided to make use of this property for fundamental frequency estimation.

4.2.2 Robustness to noise

To demonstrate the model's ability to robustly extract information from recordings, we applied the model to recordings mixed with different amounts of noise. The experiment was performed using the existing three-layer compositional structure with two compositional layers and an octave-invariant layer. We evaluated the model on the first two albums of The Beatles dataset. We added different amounts of pink and white noise to the original audio recordings in order to observe the degradation of chord classification accuracy.

Robustness to noise can be measured in different ways. Boulanger-Lewandowski et al. [160] provided an evaluation to a set of noise-types with variable signal-to-noise ratio. Lardeur et al. [161] evaluated the possible use of prior knowledge for robustness by adding equalization, reverberation and compression effects. Mauch and Ewert developed [162] the Audio Degradation toolbox (ADT) designed especially for such evaluation. The ADT offers a variety of audio distortions such as noise, reverberation and other effects that resemble real situations. For our evaluation we followed a similar line as Boulanger-Lewandowski et al. [160] and generated noisy audio input by adding pink and white noise to our recordings with SNRs between [20, 0] dB with a step of 5 dB.

We performed the experiment as follows. The hierarchical model was trained on 88 piano keys. For the experiment, we detected chords in songs from the first two albums of the Beatles dataset. The audio files were distorted using both pink and white noise at the given signal-to-noise ratios (yielding a total of 280 noisy audio files and 28 originals). Using the trained hierarchical model, we produced octave-invariant features for all audio files, including the originals.

We then performed the classification with a Hidden Markov model trained on clean (noise-free) features and tested on the noisy versions. For example, we trained the HMM on noise-free *Album1* and classified the noisy versions of *Album2*. We repeated the process by switching the *Album2* as the training set and noisy versions of *Album1* for the test set. Figure 4.8 shows that the classification performance slowly degrades up to 0 dB SNR, where especially for pink noise performance drops significantly. We may conclude, that the CHM features seem to be robust to this type of distortions.

4.2.3 Multiple fundamental frequency estimation

The goal of multiple fundamental frequency estimation (MFFE) is to estimate the pitches (fundamental frequencies) of all the tones present in each time-frame of a time-freqquency representation of a music signal. MFFE is typically one of the steps in music transcription, which aims to extract the played notes from an audio signal, and is thus an important MIR task. In our work, we evaluated how our hierarchical model can be used to directly infer the pitches in music recordings.

Experiment The input to the model was the same as in the chord estimation experiment: the audio signal was transformed with the constant-Q transform of 345 frequency bins spaced 25 cents apart between 55 and 8000 Hz, with a step size of 50 ms and the maximal window size of 100 ms.

We trained three layers of compositions on top of $\mathscr{L}_0$ in an unsupervised manner as described in the previous chapter. Such four-layer structure was sufficient for the model

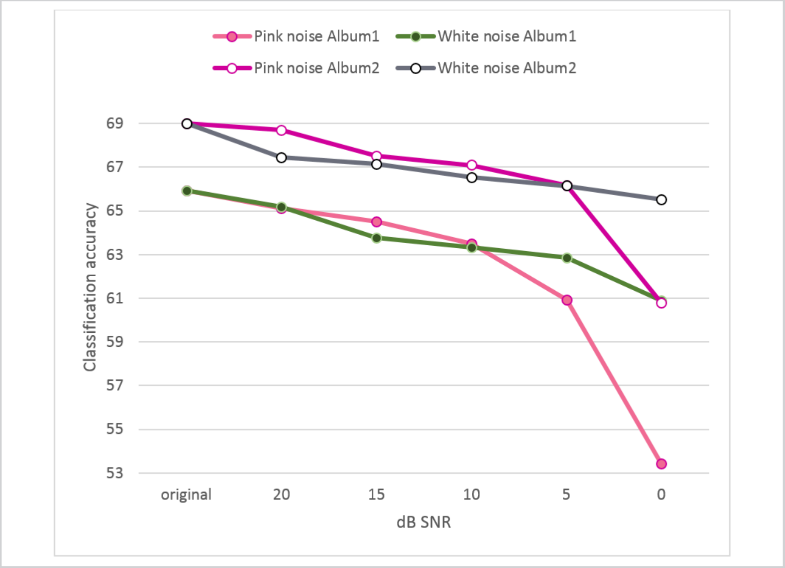


Figure 4.8 The graph represents the degradation of classification accuracy of the automated chord estimation task for the first two albums of the Beatles dataset. Values marked with *Album1* represent the results of classifying the noisy versions of the first album using a HMM learned on the second album, *Album2* represents the results by learning on the first and testing on the second album.

to learn a robust representation of pitch, as shown in our results.

To assess how different training datasets influence the structure of the model, we trained the model on several large and small datasets: three small datasets consisting of individual isolated instrument sounds (piano, flute, and guitar), two medium-sized datasets of popular music (the Beatles and Queen albums) and a large dataset of polyphonic piano music.

A comparison of the learned structures showed that the size of the learned models did not vary significantly. All the models contained a small number of compositions on all layers (in total between 50–60), with very similar structures. To compare the models, we calculated the Jaccard similarity coefficient for each layer of the hierarchies. The Jaccard similarity coefficient of two sets A and B is calculated as the size of the intersection of the two sets, divided by the size of the union of the two sets:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}. \qquad (4.5)$$

We compared the models layer per layer, where the compositions with matching subpart parameters μ were considered identical. The average Jaccard similarity coefficient per layer was 0.586 and 0.381 for $\mathscr{L}_1$ and $\mathscr{L}_2$, respectively. For both, the models trained only on individual instrument samples or only on polyphonic music, the Jaccard coefficient was higher (0.764 and 0.56 for individual instruments, and 0.778 and 0.522 for polyphonic music respectively). Only the identical compositions were counted when calculating the index, although others could also be very similar (e.g. the compositions that have three out of four subparts and parameters μ in common). The small model size and the similarity of the learned compositions is the consequence of relativity and shareability of parts, thus all models learn a generalized representation of pitch, and even a small dataset may be sufficient for learning.

We therefore decided to perform all our MFFE experiments on a model trained on the individual Bösendorfer model 225[3] piano notes (these were not included in the test datasets), which makes the training fast, but still yields good results. The learned model contained only 23, 12 and 16 parts on layers 1, 2 and 3, respectively.

To use the model for MFFE, we exploited its transparency, which enabled us to interpret the activations of the parts on layer $\mathscr{L}_3$ and directly map them onto the frequency axis. This allowed us to directly extract a set of fundamental frequencies at each time-frame and no additional supervised machine learning models were thus needed.

Results To assess the robustness of the learned pitch concepts, we tested the model for MFFE on four distinct datasets: MAPS M [107], containing piano-synthesized MIDI files, MAPS D, containing the recordings of the Disklavier [107], Su & Yang dataset [163], containing mixtures of piano and string instruments, and a dataset of folk songs sung by choirs of 2–4 singers (available at http://musiclab.si/folkmusic.zip).

For all datasets, we compared our results to three other methods: DNMF decomposition of the time-frequency representation [101], where DNMF was trained on 70% of the individual dataset and tested on the remaining 30%, Klapuri's multiple F_0 estimation method [92], and two approaches presented by Benetos and Weyde [90, 164]. For Klapuri's method, we used 30% of the annotated dataset to fine-tune the salience threshold parameter.

The results are given in Table 4.2. We report the average frame-level F_1 score, proposed in the MIREX multiple fundamental frequency estimation task. The F_1 score is the

[3] From the EastWest Ultimate Piano Collection

Table 4.2

Comparison of CHM, DNMF, Klapuri and Benetos approaches. F_1 scores in %, running times and memory usage for 1 minute of audio for different datasets and different transcription methods are shown. F_1 scores are frame-based scores calculated in accordance to MIREX MFFE evaluations [165].

Dataset	CHM	DNMF [101]	Klapuri [92]	Benetos [90]	Benetos [164]
MAPS MIDI	52.6	**61.6**	56.0	56.7	56.7
MAPS D	51.8	57.1	52.5	50.1	**62.6**
Su & Yang	**48.9**	32.6	48.0	40.3	55.6
Folk song	**49.3**	35.0	31.8	27.5	16.2
Average F_1	**50.7**	46.6	47.1	43.7	47.8
Running time (s)	6.2	5.7	19.4	188.1	87
RAM Usage (MB)	63.8	120.0	43.2	1914.2	716.5

harmonic average of precision and recall, where the precision is defined as ”the portion of correct retrieved pitches for all pitches retrieved for each frame” and the recall as ”the ratio of correct pitches to all ground truth pitches for each frame”.

The results show that the proposed model learns a robust representation of pitch and has a good ability to generalize, as it yields consistent results on the different datasets. While other approaches, such as DNMF or Benetos’, achieve better scores on some datasets, we should consider that they were trained on each dataset separately and likely overfit the specific timbres (e.g. DNMF was trained on the majority of the MAPS dataset). Also, their performance suffers on datasets where timbre is not so well defined, such as the folk song dataset which contains singing.

Although trained only on piano notes, the proposed model unsupervisedly learned the concept of pitch in a robust manner, without (over)fitting to specific templates of a single instrument. It is the most accurate of all the compared approaches on the folk song dataset, where it demonstrates its robustness. Singing transcription is difficult for most algorithms based on harmonic templates (which include all the compared algorithms), as the vocal timbre changes not only between songs (different performers), but also within a song (different vowels, stress etc.). It is therefore difficult to capture the timbre with a template, which results in poor transcription performance, especially in terms of precision. In addition, these songs originate from field recordings of folk music which are performed by amateur singers and recorded in everyday environments with portable audio equipment. Thus, they significantly differ from the studio-level or synthesized recordings. The CHM, with its multilayer representation, hallucination, inhibition, and AGC mechanisms, achieves performance comparable to other datasets, while the compared methods perform significantly worse (Kruskal-Wallis test $\chi^2 = 56.8$, $p < 10^{-11}$).

Choice of parameters Our model has several parameters, which influence the learning and the inference steps. We first evaluated the sensitivity of the proposed model to different values of its two most significant parameters: τ_H and τ_I. For performance reasons, the comparison was made only on one MAPS folder, and not on the entire dataset. Results in Fig 4.9 show that the model's performance for MFFE is mostly stable, apart from the extreme values. If τ_H that controls the hallucination is set to a low value, the amount of activations increases drastically, as the parts are allowed to hallucinate almost freely and vice versa. High values produce few activations, so in both cases the performance suffers. Similarly, a low value of τ_I (inhibition) results in a large number of part activations and subsequently a worse performance.

Other parameters also have well defined roles and effects. During learning, the model is invariant to changes of the learning parameter τ_P above approximately 0.75 due to limitations imposed by τ_C. High values of the latter result in small part candidate sets and insufficient coverage of the signal. The AGC parameters α_1 and α_2 influence the stability of activations over time and only affect the performance if set to extreme values.

The training and the inference parameters used for all MFFE experiments were set to the values described in Table 4.3.

Error analysis We analyzed the model's output with respect to the manually annotated ground truth to assess the most typical errors made by the proposed model. Four types of errors are frequent: offset localization, semitone errors, harmonic (octave) errors and pitch fluctuation.

Offset localization errors frequently appear in recordings with strong reverberation, where an event is prolonged and is detected after the instrument has stopped playing. The AGC mechanism may additionally prolong the detected offsets, so the combination of both factors reflects in longer durations of identified events, as shown in Fig 4.10-A.

A singer's vibrato can cause the detected pitch to shift up or down in individual frames,

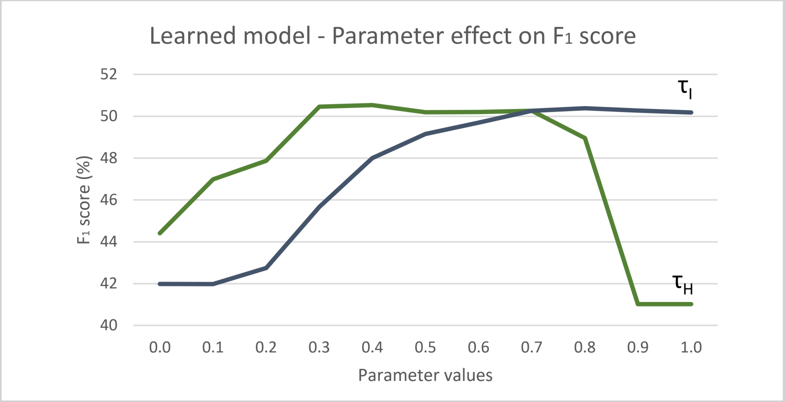


Figure 4.9
MFFE performance on the *AkPnBcht* folder from the MAPS dataset with different τ_H and τ_I values. The *x* axis represents the parameter values, the *y* axis the F_1 score.

Table 4.3
Model's parameter settings for the MFFE experiment

Parameter	Description	Value
τ_H	Hallucination parameter that allows for incomplete input	0.7
τ_I	Inhibition parameter reducing the number of competing activations	0.5
τ_C	Learning threshold for the added coverage which needs to be exceeded in order for a candidate composition to be retained while learning	0.005
τ_P	Learning threshold for cumulative coverage which, when exceeded, stops the candidate selection procedure	0.9
α_1	Density threshold which governs the transition of the AGC mechanism from normal behaviour to the onset state	0.1
α_2	Density threshold which governs the transition of the AGC mechanism from the onset state to the sustain state	0.5

which may cause semitone errors, as the ground truth usually reflects the desired and not

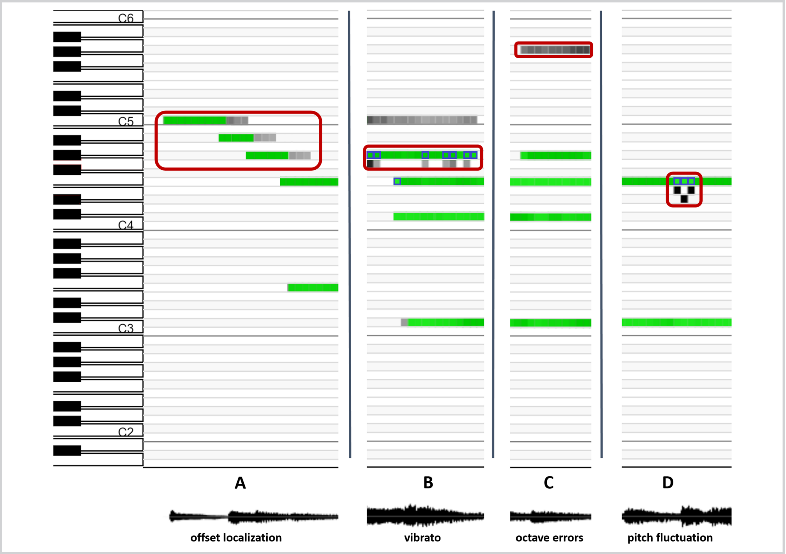


Figure 4.10 The most frequent errors of the model. Ground truth annotations are displayed in green, the CHM activations are shown in grey. Activations which are not aligned with the ground truth represent false positive errors. Additionally, false negatives are outlined in blue.

the actual pitch within a time-frame (Fig 4.10-B).

Octave and other harmonically related errors are a common source of errors for most algorithms due to the sharing of the harmonics between harmonically related tones. CHM is no exception, especially in recordings where instruments contain many strong harmonic components (4.10-C).

Voice fluctuations are commonly present in singing, especially when singers sing a cappella (without the support of instruments). Pitch may fluctuate at the onsets of syllables, resulting in the spread of energy over several semitones, similar to the vibrato, which leads to pitch estimates that differ from the ground truth, as may be observed in Fig 4.10-D.

4.2.4 Discussion

Our experiment shows that the proposed model can learn a compact hierarchical representation of the basic units of a music signal and detect these units through activations of the learned concepts over several layers. We demonstrated its effectiveness by using a model learned in a completely unsupervised manner for multiple fundamental frequency estimation. This was possible due to the model's transparency, where part activations can be interpreted meaningfully and projected to the input layer. When compared to specialized approaches, the proposed algorithm may not perform as well as the current state of the art, which is expected, as it is not tuned for a specific task. For comparison, one of the best MAPS M transcription scores is 77.1%, reported by Weninger et al. [100]. His approach differs significantly from ours—it is based on a support vector machine classifier, which was trained on a large portion of the MAPS dataset (approx. 80% of the dataset), so it likely overfits the timbre.

The deep network approaches for MFFE [31, 104–106] also typically use a large proportion of the dataset for training. Böck and Schedl [104] evaluated a recurrent neural network model on four piano music datasets, including MAPS MIDI and MAPS D. They reported a high F_1 score (up to 93.5%) for transcription accuracy around the onsets; however, they also used a significant amount of the datasets for training and validation (approximately 75% and 9.4% on average per dataset for training and validation, respectively). Nam et al. [105] reported the results for 30-second excerpts from the MAPS dataset (74.4% frame-level F_1 score) by using roughly 60% of the dataset for training and 25% for validation. Most recently, Hawthorne et al. [109] reported one of the highest reported results on the MAPS dataset, with their combination of convolutional neural networks and Long Short Term Memory networks. They evaluated their approach on the frame level, note level, and note-with-offset level. For the latter two, they reported around 30% accuracy increase, compared to their implementation of Kelz et al. [106] and Sigtia et al. [166]. However, the authors stress the drawbacks of the evaluation on the MAPS dataset and express their concern that results are not representative in terms of performance in real-world transcription scenarios.

One of the main issues of using deep approaches on small transcription datasets is that there is no guarantee that the training datasets contain an approximately equal distribution of all notes—very low and high notes, or notes from scales which are not very common may be underrepresented. If this is the case, the deep approaches will estimate some pitches better, depending on their prevalence in the training set, and will fail to recognize others, as they learn absolute representations of the pitches. The CHM avoids this problem with its relatively encoded structure, where all pitches present in the input, regardless of their location, may contribute to the representations learned by the model.

The reason that a large proportion of the dataset is usually used for training in most SoA systems is that all MFFE datasets are relatively small. This is due to the fact that annotations require expert knowledge and a significant amount of time. The annotations can thus not be crowd-sourced, as for example in image labelling, where deep networks are very successful. It becomes necessary to include a significant amount of the available data into the training set, retaining only a small portion (down to 10 % in several cases) for testing. The results are assumed to generalize over the whole dataset, but there is no information on how these models would perform on more diverse datasets, and for instruments with different timbres.

In comparison, our model was trained on only a small set of 88 piano key samples not included in any of the datasets. Although the CHM does not reach the accuracy of such tuned approaches, it is able to generalize and perform well in a variety of cases where the source is not so well defined, as shown in our evaluation on the Su & Yang and folk song datasets. We may therefore conclude that the CHM extracts timbre-invariant features from the audio signal, which, combined with a robust inference mechanism, leads to stable performance in various scenarios.

Real-time performance An added feature of the proposed approach lies in the small sizes of the learned models, which are the consequence of part relativity and shareability. The computational complexity of inference with such small models is low, so CHM can be used for transcription in real-time scenarios. Table 4.2 lists the running times and memory consumption of all the compared algorithms for one minute of audio measured on a system with a 16 GB RAM and an Intel Xeon E5520 2.26 GHz processor using a single thread. The CHM and DNMF are the fastest, with approximately ten-to-one ratio of audio length over processing time, followed by Klapuri (approx. three-to-one ratio). Both approaches by Benetos and Weyde with one-and-a-half-to-three and one-to-three ratio are not usable in real-time scenarios, as, in addition to high running times, they also require the entire audio file for processing. The memory consumption of the proposed approach is also low—it uses approximately half the memory in comparison to DNMF, and around 50% more than Klapuri's approach.

In addition, the approach is parallelizable, as parts on a layer can be inferred independently and thus in parallel. The speed, small memory consumption and robustness of our approach make it suitable for real-world use, and applicable within embedded systems and mobile devices with multiple cores and low processing power per core.

A different deep architecture The compositional hierarchical model shares some similarities with deep learning architectures. It is similar in terms of learning a variety of signal abstractions on several layers of granularity. The learning procedure is similar to DBNs: the structure is built layer-by-layer. However, unlike most deep architectures, CHM learns in an entirely unsupervised manner, so no annotated datasets are needed for training and validation. In addition, several aspects of the model set it apart from other architectures.

Transparency is manifested in the compositional nature of the model. Parts are compositions of subparts and their activations are directly observable and interpretable (each activation can be projected to the input layer and its effect observed). In contrast, most other neural-network-based deep architectures offer no clear explanation of the underlying feature extraction process and the meaning of the extracted features, with the exception of convolutional neural networks, which partially and indirectly offer explanations of their nodes [38]. Transparency enables the model to be used directly as a classifier by observing and interpreting part activations, as we show in our evaluation for the MFFE task.

Relativity and shareability of parts enable efficient encoding of the learned concepts and lead to a small number of parts needed to represent complex concepts. A part in the proposed model is defined by the relative distance between its subparts and can be activated on different locations along the frequency axis. Therefore, the large amount of layer units, which, for example, convolutional networks need for a full frequency range coverage, is not necessary. Moreover, the presence of an event at a specific location is not necessary for the CHM to learn the concept of the event, if similar events occur at other locations. The ability of relative encoding eliminates the need for a large dataset containing, for instance, pitch occurrences at all frequencies.

Relativity is accompanied by part shareability: parts on a layer may be shared by many compositions on the higher layers. Although this feature is similar to other deep representations, relativity takes shareability a step further: a set of subparts may form several new relative compositions on a higher layer, representing different entities, and may thus be efficiently reused. The learned models therefore contain a small number of parts, which also enables the use of small datasets for training and a very fast inference. This is evident in the presented evaluation, where a small set of samples was used to train a three-layer model, which performed well on several different datasets.

Compositional Hierarchical Model for Symbolic Music Representations

5.1 Model description

In this chapter, we present how the compositional hierarchical model can be applied to knowledge extraction from symbolic music representations. We present the model's implementation for symbolic data, which we denote as the symbolic compositional hierarchical model (SymCHM), and evaluate it for melodic pattern extraction and the identification of tune families.

The input of the SymCHM consists of a list of note pitches and their onset times (we currently ignore note durations). Any symbolic encoding that encodes these values can be used, such as MusicXML, MIDI or text-based representations. We define the model's input representation as a set of note onset (e.g. in seconds) and note pitch (e.g. MIDI pitch) tuples. Additionally, MIDI note velocities could be mapped to input event magnitudes; however, we currently ignore this parameter and set all magnitudes to the value of 1:

$$\mathscr{I} : \{\mathbf{X} : \mathbf{X} = [\mathrm{N}_o, \mathrm{N}_p, 1]\}. \tag{5.1}$$

The input layer of the SymCHM $\mathscr{L}_0$ consists of a single atomic part P_1^0, which activates for all note events as:

$$\mathrm{A}(\mathrm{P}_1^0) = \langle \mathrm{A}_\mathrm{T}, \mathrm{A}_\mathrm{L}, \mathrm{A}_\mathrm{M} \rangle \leftarrow \langle \mathrm{N}_o, \mathrm{N}_p, 1 \rangle \tag{5.2}$$

Activation locations A_L are equal to note pitches, onset times A_T to note onsets, while magnitudes A_M are assumed to be 1 for all events (they could also represent note dynamics, if greater importance was to be put on accented notes).

The parts on higher layers are defined as compositions of their subparts according to parameters μ and σ (see chapter 3). These encode the relative distances (offsets) between each subpart and the composition's central part. In the SymCHM, offsets are modeled in the pitch domain, thus a composition encodes the pitch distance between various subparts (e.g. in semitones if a MIDI pitch is used to represent the event location). Standard deviation σ is set to a small fixed value, which does not allow for deviations from the offset encoded by μ. This condition may be relaxed in the future work to potentially achieve a similar robustness as in chromatic to morphetic pitch translation [167].

An example of the model is shown in Fig.5.1. The SymCHM provides a hierarchical representation of a symbolic music piece, from individual notes on the lowest layer, up

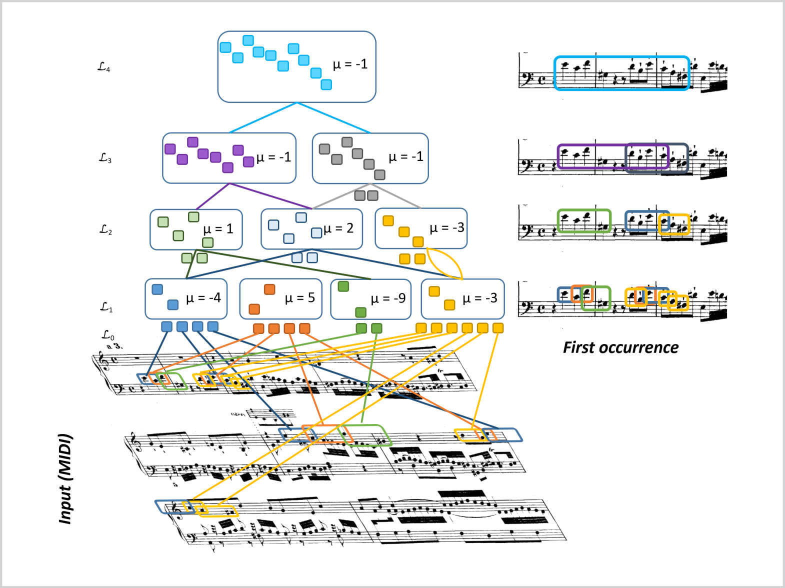


Figure 5.1 An abstraction of the SymCHM. The score is modelled from individual note events on the first layer up to more complex sequences on higher layers. Relative distances between the notes/patterns are encoded as pitch (semitone, in this case) distances by μ. Each part can have multiple activations representing the occurrences of the pattern it encodes in the score. First activations of each part are also shown in the score on the right side of the figure. Individual part structure representations are scaled in height.

to complex musical patterns on the higher layers. Part P_1^0 is activated for each input note event. The parts on the first layer represent intervals, e.g. the first $\mathscr{L}_1$ part represents a descending major third (-4 semitones) and is activated for all such intervals in the input regardless of gaps, with notes spaced maximally τ_W apart. The first part of layer 2 represents a composition of two subparts with offset 1 ($\mu = 1$), meaning its pattern is a concatenation of two sub-patterns spaced 1 semitone apart. Such relative encoding enables the model to learn position and time-independent patterns. Note that, as onset times are not modelled in the representation, the encoded patterns may span different time scales, as well as contain gaps (compared to the input representation).

5.1.1 Inference

A trained model captures the repetitive patterns in the training data, which are relatively encoded and may be observed through the inspection of the model's parts on its various layers. When a trained model is presented with new input data, the learned patterns may be located in the input through the process of *inference*. Inference calculates part activations on the input data (and thus absolute pattern positions) according to Equations 3.2 and 3.3. They are calculated bottom-up layer-by-layer, whereby the input data activates the layer $\mathscr{L}_0$. An activation of a part represents a specific occurrence of the pattern it represents in the input. Its *location* and *onset time* map the relative pattern onto a specific set of pitches within the input sequence of events (thus making it absolute), while its *magnitude* represents its strength. A part can concurrently activate at different locations, which represent multiple occurrences of the pattern in the input representation.

5.1.2 Mechanisms

Inference may be exact or approximate, where in the latter case the hallucination and inhibition mechanisms enable the model to find patterns with deletions, changes or insertions, thus increasing its predictive power and robustness.

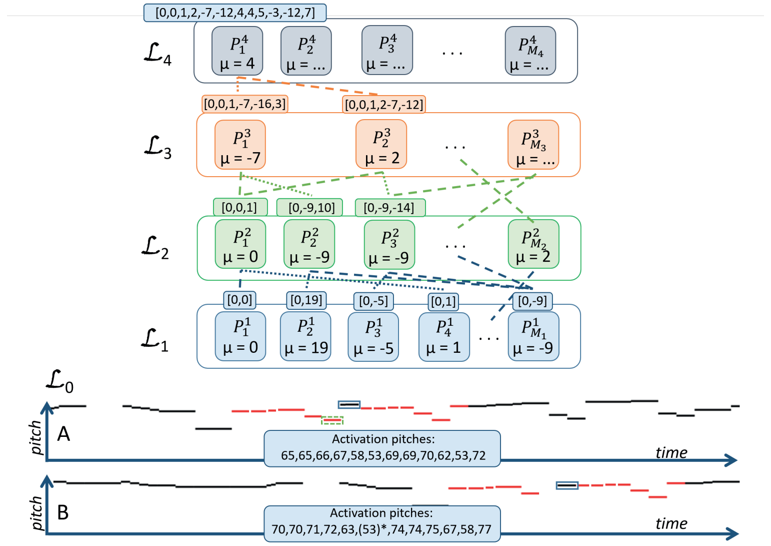
Hallucination provides means to activate a part even when the input is incomplete or changed. For the SymCHM such changes often occur in melodic variations and ornamentation. Hallucination enables the model to robustly identify patterns with variations.

Inhibition is also essential in the SymCHM for the removal of redundant hypotheses. As the model does not rely on any musicological rules, the parts may produce a large number of competing patterns. Inhibition may be used to reduce the number of activations and find the patterns that best correspond to the learned hierarchy.

5.2 Evaluation: discovery of repeated patterns

The goal of the pattern discovery task is to identify repeating patterns in a corpus. As a part of MIREX, the *discovery of repeated patterns and sections* task is defined as an *intra-opus* pattern discovery task, where an algorithm takes a single music piece as input and outputs a list of patterns repeated within that piece. Although the definition itself is rather vague (there is no exact definition of what a 'pattern' is), the task description attributes great importance to the task, labeling it crucial for "understanding and interpreting a musical work" [168]. The goal of the task is therefore to find patterns, described as a "set of onset time-pitch pairs that occurs at least twice". The patterns are evaluated against the reference annotations, provided by three experts. Patterns may significantly vary in size, from short four-event-long motifs to tens-of-events-long sections.

The SymCHM can be used for this task due to its transparency, relative encoding



The structure of a part is displayed above each part in the Figure, represented by a sequence of pitch values relative to the first subpart (e.g. [0,0,1] for part P^2_1). A part may be contained in several compositions, e.g. P^1_n is a part of compositions P^2_2 and P^2_3. The entire structure is transparent, thus we can observe the entire sub-tree of part P^4_1. A part activates when (a part of) the pattern it represents is found in the input. As an example, P^4_1 activates twice (input A and B); however, there are differences in the found patterns. Pattern A is positioned 5 semitones higher than B. Pattern B is missing one event (dotted green rectangle), and the pitch of one event (blue rectangle) differs between the two patterns.

Figure 5.2
The symbolic compositional hierarchical model. The input layer corresponds to a symbolic music representation (a sequence of pitches). Parts on higher layers are compositions of lower-layer parts (depicted as the connections between parts, parameter μ is given in semitones).

of the learned structure and the ability to unsupervisedly learn a hierarchy on a small dataset (e.g. a single song). During the learning process, the frequently co-occurring parts are joined into new compositions. Since the compositions are relatively encoded, multiple occurrences of an encoded structure result in multiple activations of a single composition. The learned compositions can therefore be observed as patterns, while their activations represent pattern occurrences.

5.2.1 Pattern selection

The SymCHM can be trained on a single or multiple symbolic music representations. It learns a hierarchical representation of the patterns occurring in the input, where the patterns encoded by the parts on the higher layers are compositions of the patterns on the lower layers. The inference produces part activations, which expose the learned patterns (and their variations) in the input data. The shorter and more trivial patterns naturally occur more frequently, the longer patterns less frequently. On the other hand, the longer patterns may entirely subsume the shorter patterns.

The occurrences of the melodic patterns in a given piece are discovered by observing the activations of the learned model's parts, where each activation of a part is interpreted as an *occurrence* of the pattern encoded by the part. As activations may be abundant and may overlap, while the patterns on the lower layers get subsumed by the patterns on the higher ones, it is beneficial to select a subset of the found patterns as the model's output. In this section, we present two approaches for pattern selection.

Basic selection In basic pattern selection, we output all the patterns of sufficient complexity, represented by parts from the layer L up to the highest layer N. First, we select all parts from the layers $\mathscr{L}_{\mathrm{L}} \ldots \mathscr{L}_{\mathrm{N}}$. Since the parts on the higher layers are compositions of the parts on the lower layers, we exclude from this set all the parts which are subparts of a composition on a higher layer, to avoid redundancy. The final selection of parts can be formulated as:

$$\bigcup_{l=\mathrm{L}}^{\mathrm{N}} \{\mathrm{P}_i^l \in \mathscr{L}_l : (\neg\exists\, \mathrm{P}_j^{l+1})[\mathrm{P}_j^{l+1} \in \mathscr{L}_{l+1} \wedge \mathrm{P}_i^l \in \mathrm{P}_j^{l+1}]\} \tag{5.3}$$

Pattern selection is performed simply by inference on a given piece (or set of pieces). The observed activations of the selected parts, their locations and times represent the found patterns. The hallucination and the inhibition mechanisms are applied during the inference to provide balance between the producing hypotheses, which partially match the input representation (hallucination) and the amount of competitive hypotheses (inhibition).

SymCHMMerge: merging overlapping patterns An analysis of the basic pattern selection approach showed a lack of pattern diversity, as the found patterns were often very similar and overlapping, as shown in the top part of Fig.5.3. We improved the algorithm by merging redundant patterns and adjusting the learning and inference parameters. We named the resulting pattern extraction algorithm SymCHMMerge.

As our model is trained in an unsupervised manner, several parts may represent similar and overlapping patterns (e.g. patterns shifted by a few notes). Inhibition reduces the redundant activations of such parts; however, it is usually not enforced strongly, as it

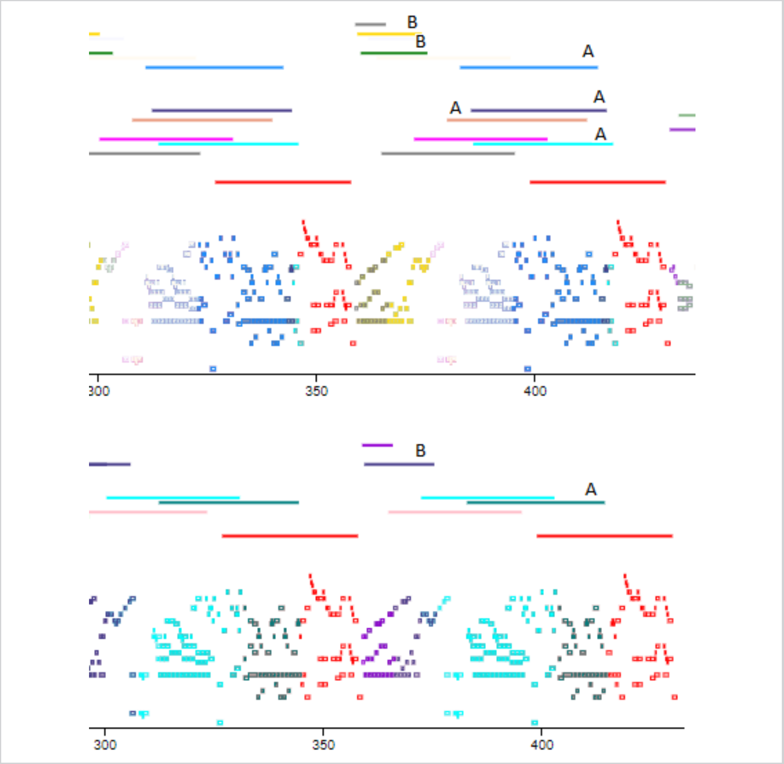


Figure 5.3
Patterns extracted by the basic selection algorithm (top). The abscissa represents time in seconds, the ordinate axis represents MIDI pitch. The extent of each pattern occurrence is marked by a horizontal line above each graph. Two sets of similar overlapping pattern occurrences (A and B) are clearly visible. These occurrences can be merged into single occurrences if overlaps are consistent over most of the piece (bottom) with the SymCHMMerge approach. Source: Žerovnik [169].

could overly reduce the number of activations and found patterns. Therefore, to reduce the number of such overlapping patterns for the pattern finding task, we merge them into longer patterns.

Let $C(A(P_i^n))$ represent a pattern occurrence defined by the coverage of the part's activation, as defined by Eq. 3.7. Ψ_i^n represents the set of all such pattern occurrences given activations of the part:

$$\Psi_i^n = \bigcup_k \{C(A_k(P_i^n))\}. \tag{5.4}$$

We express the overlap of two pattern occurrences a_i and a_j, produced by parts P_i^n

and P_j^m, by calculating the Jaccard similarity coefficient between the two occurrences:

$$\begin{aligned} a_i = C(A(P_i^n)), a_j &= C(A(P_j^m)) \\ J(a_i, a_j) &= \frac{|a_i \cap a_j|}{|a_i \cup a_j|} \end{aligned} \tag{5.5}$$

We aim to merge the patterns of two parts, if they often significantly overlap. We therefore calculate the proportion of overlapping patterns as:

$$\frac{1}{|\Upsilon_i^n| + |\Upsilon_j^m|} \sum_{a_i \in \Upsilon_i^n} \sum_{a_j \in \Upsilon_j^m} |J(a_i, a_j) > \tau_R|. \tag{5.6}$$

τ_R governs the amount of the allowed overlaps. If the proportion of significantly overlapping patterns (overlap exceeds τ_R) exceeds a merging threshold τ_M, all redundant pattern occurrences of the two parts are merged.

For evaluation, the thresholds τ_R and τ_M were both set to 0.5, meaning that the pattern occurrences produced by two parts had to share at least 50% of the events in the input layer and have such overlap in at least 50% of cases, to be merged.

To address the problem of pattern diversity, we needed to increase the number of patterns found by the model. This was achieved with three simple adjustments. First, we lowered the candidate selection thresholds in the greedy phase of the learning process to add more parts to each layer (evaluation showed that on average 16% more parts were added). Second, more layers were considered when searching for pattern occurrences. And third, the hallucination threshold was increased during inference. All these modifications could also be made with the basic pattern selection approach; however, they would result in an even higher number of redundant patterns. With SymCHMMerge, redundant occurrences are merged and thus the diversity of the found patterns increases.

5.2.2 *Evaluation metrics*

The evaluation metrics from the MIREX discovery of repeated themes and sections task were used for evaluation. This subsection provides a short description and a formalization of the definitions found in the MIREX task definition [112].

The *establishment* measure (precision P_{est}, recall R_{est} and F score F_{1est}) evaluates the algorithm's ability to find at least one occurrence of each pattern shifted in time and pitch. Two *occurrence* measures F_{1occ} evaluate the extent of the model's ability to find all the pattern occurrences, where the factor $c = \{0.5, 0.75\}$ represents the inexactness tolerance threshold. Meredith [124] proposed an additional three-layer metric (P_3, R_3, TLF_1) that provides balance between the establishment and the occurrence measures. The exact precision, recall and F-score measures (P, R, F_1) show the algorithm's performance in matching the found patterns with the reference annotations in an exact manner. This exact score is expected to be significantly lower than the aforementioned establishment and occurrence scores, because it discards all the inexact matches, which are not completely identical to the reference annotations, while the former scores tolerate inexact beginnings and endings of patterns. The metrics are formally defined using the following set of symbols:

- $n_{\mathscr{P}}$ - the number of patterns in the ground truth
- $\Pi = \{\mathscr{P}_1, \mathscr{P}_2, \dots, \mathscr{P}_{n_{\mathscr{P}}}\}$ - a set of ground truth patterns
- $\mathscr{P} = \{\mathrm{P}_1, \mathrm{P}_2, \dots, \mathrm{P}_{m_{\mathrm{P}}}\}$ - occurrences of a pattern $\mathscr{P}$
- $n_{\mathscr{Q}}$ - the number of patterns in the algorithm's output
- $\Xi = \{\mathscr{Q}_1, \mathscr{Q}_2, \dots, \mathscr{Q}_{n_{\mathscr{Q}}}\}$ - a set of patterns returned by the algorithm
- $\mathscr{Q} = \{\mathrm{Q}_1, \mathrm{Q}_2, \dots, \mathrm{Q}_{m_{\mathrm{Q}}}\}$ - occurrences of a pattern $\mathscr{Q}$.
- k - the number of ground truth patterns identified by the algorithm

Standard precision is defined as $\mathrm{P} = k/n_{\mathscr{Q}}$, recall as $\mathrm{R} = k/n_{\mathscr{P}}$, and the F_1 score as $\mathrm{F1} = 2 * \mathrm{PR}/(\mathrm{P} + \mathrm{R})$. As it is very difficult to discover exact patterns, more robust versions of the standard metrics are provided: occurrence and establishment scores. First, the cardinality score is used to determine similarity between the annotated and the discovered patterns:

$$s_c(\mathrm{P}_i, \mathrm{Q}_j) : |\mathrm{P}_i \cap \mathrm{Q}_j| / \max\{|\mathrm{P}_i|, |\mathrm{Q}_j|\} \tag{5.7}$$

A score matrix is calculated based on this similarity as follows:

$$s(\mathscr{P}, \mathscr{Q}) = \begin{bmatrix} s(\mathrm{P}_1, \mathrm{Q}_1) & s(\mathrm{P}_1, \mathrm{Q}_2) & \dots & s(\mathrm{P}_1, \mathrm{Q}_{m_{\mathrm{Q}}}) \\ s(\mathrm{P}_2, \mathrm{Q}_1) & s(\mathrm{P}_2, \mathrm{Q}_2) & \dots & s(\mathrm{P}_2, \mathrm{Q}_{m_{\mathrm{Q}}}) \\ \vdots & \vdots & \ddots & \vdots \\ s(\mathrm{P}_{m_{\mathrm{P}}}, \mathrm{Q}_1) & s(\mathrm{P}_{m_{\mathrm{P}}}, \mathrm{Q}_2) & \dots & s(\mathrm{P}_{m_{\mathrm{P}}}, \mathrm{Q}_{m_{\mathrm{Q}}}) \end{bmatrix} \tag{5.8}$$

Based on the score matrix, the establishment matrix is calculated from the set of annotated patterns Π and the set of algorithm's output patterns Ξ:

$$\mathrm{S}(\Pi,\Xi) = \begin{bmatrix} \mathrm{S}(\mathscr{P}_1,\mathscr{Q}_1) & \mathrm{S}(\mathscr{P}_1,\mathscr{Q}_2) & \dots & \mathrm{S}(\mathscr{P}_1,\mathscr{Q}_{n_\mathscr{Q}}) \\ \mathrm{S}(\mathscr{P}_2,\mathscr{Q}_1) & \mathrm{S}(\mathscr{P}_2,\mathscr{Q}_2) & \dots & \mathrm{S}(\mathscr{P}_2,\mathscr{Q}_{n_\mathscr{Q}}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{S}(\mathscr{P}_{n_\mathscr{P}},\mathscr{Q}_1) & \mathrm{S}(\mathscr{P}_{n_\mathscr{P}},\mathscr{Q}_2) & \dots & \mathrm{S}(\mathscr{P}_{n_\mathscr{P}},\mathscr{Q}_{n_\mathscr{Q}}) \end{bmatrix} \tag{5.9}$$

The establishment precision is defined as:

$$\mathrm{P}_{est} = \frac{1}{n_\mathscr{Q}} \sum_{j=1}^{n_\mathscr{Q}} \max\{\mathrm{S}(\mathscr{P}_i,\mathscr{Q}_j)\,|\,i = 1 \dots n_\mathscr{P}\} \tag{5.10}$$

The establishment recall is defined as:

$$\mathrm{R}_{est} = \frac{1}{n_\mathscr{P}} \sum_{j=1}^{n_\mathscr{P}} \max\{\mathrm{S}(\mathscr{P}_i,\mathscr{Q}_j)\,|\,i = 1 \dots n_\mathscr{Q}\} \tag{5.11}$$

And the establishment F_1 score is calculated as:

$$\mathrm{F1}_{est} = 2 * \mathrm{P}_{est}\mathrm{R}_{est}/(\mathrm{P}_{est} + \mathrm{R}_{est}) \tag{5.12}$$

The establishment metric rewards a single match between the annotated and the algorithm's patterns, so it measures the success of an algorithm in finding different patterns. The occurrence metric, on the other hand, rewards the algorithm's ability to find *all* the occurrences of a single pattern. Its calculation also considers inexact matches, so that the found pattern occurrences need not exactly match the ground truth. The factor *c* (the values used are 0.5 and 0.75) defines the threshold in the establishment matrix that determines whether a match is considered similar enough to the reference pattern and is thus considered as discovered.

If we define $\mathscr{I}$ as the set of indices in the establishment matrix with values greater or equal to *c*, the occurrence matrix $\mathrm{O}(\Pi,\Xi)$ is calculated as follows. Starting with an empty $n_\mathscr{P} \times n_\mathscr{Q}$ matrix and the establishment indices $\mathscr{I}$:

$$\forall (i,j) \in \mathscr{I} : \mathrm{O}(\Pi,\Xi)[i,j] = s(\mathscr{P}_i,\mathscr{Q}_j). \tag{5.13}$$

The occurrence precision score defined as:

$$\mathrm{P}_{occ} = \frac{1}{n_{col}} \sum_{j=1}^{n_\mathscr{Q}} \mathrm{O}(i,j)\,|\,i = 1 \dots n_\mathscr{P}, \tag{5.14}$$

where n_{col} represents the number of non-zero columns in the occurrence matrix O. The occurrence recall score is analogously calculated as:

$$\mathrm{R}_{occ} = \frac{1}{n_{row}} \sum_{j=1}^{n_{\mathscr{P}}} \mathrm{O}(i,j) | i = 1 \dots n_{\mathscr{Q}}, \tag{5.15}$$

where n_{row} represents the number of non-zero rows in the occurrence matrix O.

5.2.3 *Experiment*

We evaluated the proposed model for the discovery of repeated themes and sections in symbolic monophonic music pieces. As the goal of the task is to search for patterns within a given piece (and not across an entire corpus), we trained a model independently for each piece and inferred the patterns on the same piece. All model parameters were kept constant during all the evaluations and were not tuned to each specific case. Their values are shown in Table 5.1. The τ_W parameter limiting the time span of activations was set to $\tau_W = 2^{n+2}$ events, thus growing with each layer n. The values of the hallucination and inhibition parameters τ_H and τ_I were based on the stable performance achieved in the mid-range around 0.5 (see the *Sensitivity to parameter values* subsection). Merging parameters τ_R and τ_M were set to 50%, as explained previously, while the learning thresholds τ_P and τ_C were retained from the spectral CHM, where they were evaluated empirically.

5.2.4 *Results*

MIREX The SymCHM with the basic pattern selection algorithm was submitted to MIREX 2015 [112] for evaluation within the Discovery of repeated themes and sections in monophonic symbolic music task. The results are shown in Table 5.2. The submitted model learned a six layer hierarchy on each piece, where the activations of parts on the layers 4–6 were output as the found pattern occurrences.

Overall, the SymCHM exhibited good occurrence metrics and lower establishment metrics, thus not finding enough patterns but many occurrences of the ones that were found. Two state-of-the art approaches by Velarde and Meredith (VM2) [126] and Lartillot (OL1) [128] achieved better overall results, while other approaches that had been proposed in previous MIREX evaluations—such as NF1'14 [131] and DM1'13 [170]—were comparable to SymCHM.

Table 5.1
Parameter values used in the experiment

Parameter	Description	Value
τ_H	Hallucination parameter that allows for incomplete input	0.5
τ_I	Inhibition parameter reducing the number of competing activations	0.4
τ_R	Redundancy parameter determining the the amount of overlap of two patterns to be considered for merging	0.5
τ_M	Merging parameter determining the percentage of redundant pattern occurrences for two patterns to be merged into one	0.5
τ_C	Learning threshold for added coverage which needs to be exceeded in order for a candidate composition to be retained while learning	0.005
τ_P	Learning threshold for cumulative coverage which, when exceeded, stops the candidate selection procedure	0.9
τ_W	Window size limiting the time span considered when calculating activations, defined per layer $\mathscr{L}_n$	2^{n+2}

The table also shows that the results vary a lot among different approaches and measures. The reason behind the variations of different F scores lays in their definitions. All the F scores, except for the F_1 score shown in the third row, are purposely defined not to fully penalize an algorithm. For example, the establishment F score does not change when an algorithm does not identify all the occurrences of the established patterns. On the other hand, the occurrence F score only penalizes the unidentified occurrences of those patterns which had previously been established. Moreover, the occurrence F score also considers inexact matches of the identified pattern occurrences (governed by threshold c).

The second reason for the variations is the vague definition of *pattern discovery*. There is no information provided about the amount and size of the patterns to be discovered.

Table 5.2
MIREX results for the discovery of repeated themes and sections (monophonic) task

Algorithm	P_{est}	R_{est}	$P_{occ(c=.75)}$	$R_{occ(.75)}$
SymCHM MIREX 2015	53.36	41.40	81.34	59.84
NF1 MIREX 2014 [131]	50.06	54.42	59.72	32.88
DM1 MIREX 2013 [170]	52.28	60.86	56.70	75.14
OL1 MIREX 2015 [128]	61.66	56.10	87.90	75.98
VM2 MIREX 2015 [126]	65.14	63.14	60.06	58.44

	$P_{occ(c=.5)}$	$R_{occ(c=.5)}$	P	R
SymCHM MIREX 2015	73.34	62.48	10.64	6.50
NF1 MIREX 2014 [131]	54.98	33.40	1.54	5.00
DM1 MIREX 2013 [170]	47.20	74.46	2.66	4.50
OL1 MIREX 2015 [128]	78.78	71.08	16.0	23.74
VM2 MIREX 2015 [126]	46.14	60.98	6.20	6.50

	F_{1est}	$F_{1occ(c=.75)}$	$F_{1occ(c=.5)}$	F_1	TLF_1
SymCHM MIREX 2015	42.32	67.92	67.24	5.12	37.78
NF1 MIREX 2014 [131]	50.22	40.86	40.80	2.36	33.28
DM1 MIREX 2013 [170]	54.80	62.42	56.94	3.24	43.28
OL1 MIREX 2015 [128]	49.76	80.66	74.50	12.36	42.72
VM2 MIREX 2015 [126]	62.74	57.00	51.52	6.2	42.20

Individual approaches therefore employ different techniques to tackle this task. While some approaches perform well in terms of pattern establishment (e.g. VM2), others may identify fewer different patterns but perform better in finding all the occurrences of the identified patterns (e.g. OL1).

SymCHMMerge Based on the performance of basic pattern selection, we developed the improved SymCHMMerge algorithm, where we aimed to increase diversity and decrease the redundancy of the found patterns. To compare its performance with the basic model, we evaluated both on the publicly available JKU PDD dataset[1] [171], which consists of five pieces:

- Bach's Prelude and Fugue in A minor (BWV889)—731 note events, 3 patterns, 21 pattern occurrences,
- Beethoven's Piano Sonata in F minor (Op. 2, No. 1), third movement—638 note events, 7 patterns, 22 pattern occurrences,
- Chopin's Mazurka in B flat minor (Op. 24, No. 4)—747 note events, 4 patterns, 94 pattern occurrences,
- Gibbons's The Silver Swan—347 note events, 8 patterns, 33 pattern occurrences,
- Mozart's Piano Sonata in E flat major, K. 282-2nd movement—923 note events, 9 patterns, 38 pattern occurrences.

In the SymCHMMerge, the activations of parts on layers 2–6 were considered for finding pattern occurrences, where each layer included 16% more parts on average, due to the more relaxed learning conditions.

The comparison of both algorithms on the JKU PDD dataset is given in Table 5.3. The SymCHMMerge achieved significantly better results (Friedman's test: $\chi^2 = 7.2, p < .01$). It mostly improved in the establishment measures, which indicates the improvement of the algorithm's ability to discover at least one occurrence of a pattern, tolerating for time shift and transposition [112]. On the other hand, the occurrence measures $F_{1occ(c=.75)}$ and $F_{1occ(c=.5)}$, which evaluate the algorithm's ability to find all the occurrences of the established patterns, dropped by 5%. We attribute this drop to the higher number of established patterns, for which the occurrence measure is calculated. Finally, absolute precision, recall and F scores significantly increased due to the SymCHMMerge's pattern merging procedure and the increased pattern diversity.

Detailed results for each music piece in the JKU PDD dataset are displayed in Table 5.4. Exact matches were found only on music pieces by Bach, Gibbons and Mozart. The exactly matched patterns are relatively short (Bach has 7 to 20-event long patterns), whereas the longest patterns (over 300 events and over one minute long segments) were

[1] The dataset is publicly available on this link: https://dl.dropbox.com/u/11997856/JKU/JKUPDD-Aug2013.zip.

Table 5.3

Evaluation of the SymCHM and SymCHMMerge for the discovery of repeated themes and sections (monophonic) task

Algorithm	P_{est}	R_{est}	$P_{occ(c=.75)}$	$R_{occ(.75)}$	$P_{occ(c=.5)}$	$R_{occ(c=.5)}$	P	R	F_{1est}	$F_{1occ(c=.75)}$	$F_{1occ(c=.5)}$	F_1	TLF_1
SymCHM JKU PDD	67.92	45.36	93.90	82.72	78.53	72.99	25.00	13.89	51.01	86.85	75.41	17.18	51.75
SymCHMMerge JKU PDD	67.96	50.67	88.61	75.66	83.23	68.86	35.83	20.56	56.97	80.02	73.88	25.63	52.89

not identified due to limitations on pattern length imposed by the number of layers. The number of discovered patterns is very low for the piece by Gibbons, where several patterns in the reference annotation describe very long sections, which were not identified by our model due to the aforementioned limitations.

5.2.5 Sensitivity to parameter values

To assess the sensitivity of the SymCHMMerge to the changes of the model's parameters, we analyzed its performance by varying the inhibition and hallucination parameters τ_I and τ_H, which affect inference. We observed the behavior of the occurrence and establishment measures in order to estimate the balance between the two. Due to the large number of possible parameter combinations, we evaluated the effect of changes in one parameter (set for all layers) on the model's performance, when all other parameters were fixed.

Inhibition The top part of Figure 5.4 shows how changes in the inhibition parameter τ_I affect the results. An increase of τ_I increases the inhibition and removes activations, which are only partially covered by others, while a decrease will allow for more overlapping activations to propagate to higher layers. The plots show that a reduced inhibition has a positive effect on occurrence recall, which is expected as more activations

Table 5.4

A detailed list of the JKU PDD results for the SymCHMMerge algorithm. n_{P} and n_{Q} columns represent the number of annotated patterns and the number of discovered patterns, respectively. Song names are shortened, using a four letter abbreviation of the composer's name.

Piece	n_{P}	n_{Q}	P_{est}	R_{est}	$\mathrm{P}_{occ(c=.75)}$	$\mathrm{R}_{occ(c=.75)}$
bach	3	2	100.00	66.67	100.00	45.65
beet	7	7	65.81	60.02	80.71	80.71
chop	4	5	47.95	49.81	62.36	51.96
gbns	8	3	78.16	35.49	100.00	100.00
mzrt	9	8	47.88	41.39	100.00	100.00
Average	6.2	5	67.96	50.67	88.61	75.66

Piece	P_3	R_3	$\mathrm{P}_{occ(c=.5)}$	$\mathrm{R}_{occ(c=.5)}$	P	R
bach	62.96	41.97	100.00	45.65	100.00	66.67
beet	77.38	64.95	79.24	72.44	0.00	0.00
chop	46.96	39.92	57.00	46.29	0.00	0.00
gbns	81.82	34.33	100.00	100.00	66.67	25.00
mzrt	57.21	47.54	79.92	79.92	12.50	11.11
Average	65.27	45.74	83.23	68.86	35.83	20.56

Piece	F_{1est}	TLF_1	$\mathrm{F}_{1occ(c=.75)}$	$\mathrm{F}_{1occ(c=.5)}$	F_1
bach	80.00	50.37	62.68	62.68	80.00
beet	62.78	70.62	80.71	75.69	0.00
chop	48.86	43.15	56.69	51.09	0.00
gbns	48.81	48.37	100.00	100.00	36.36
mzrt	44.40	51.93	100.00	79.92	11.77
Average	56.97	52.89	80.02	73.88	25.63

are produced. It is even more interesting that it also positively affects occurrence precision, which might be explained by the fact that overlapping activations are successfully merged by the SymCHMMerge merging algorithm. For the establishment metrics, the

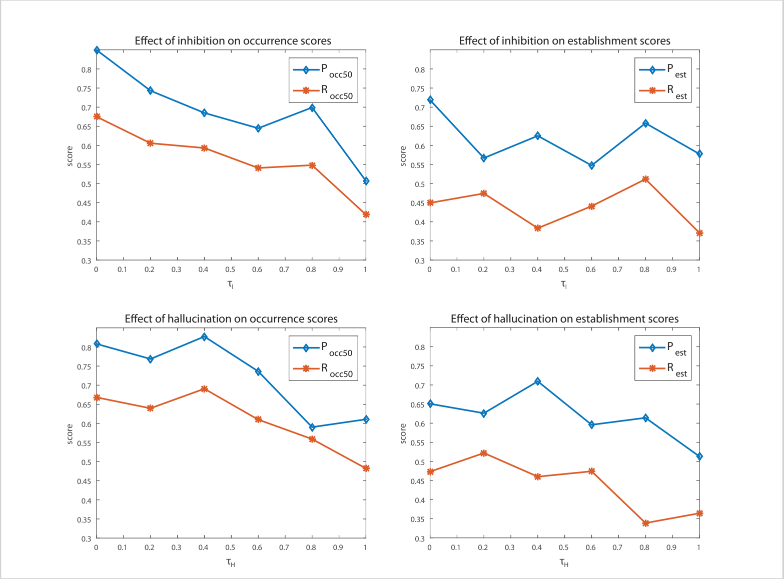


Figure 5.4
Sensitivity of the model to the changes of the inhibition parameter τ_I (top) and the hallucination parameter τ_H (bottom). When one parameter was changed, all others remained fixed.

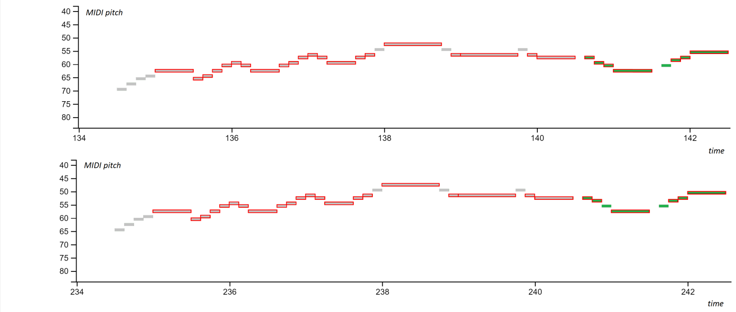
effect of change in inhibition is not so obvious, and apart from the extreme values, the performance is stable.

Hallucination The bottom part of Figure 5.4 shows how changes in the hallucination parameter τ_H affect performance. As described previously, larger τ_H values decrease hallucination and thus the number of activations. Decreased hallucination affects both the occurrence and the establishment of patterns, as there is little tolerance for pattern variations. With more hallucination, both measures gradually decrease.

5.2.6 *Error Analysis*

To increase our understanding of the model's performance, we performed an analysis of its most common types of errors.

Incomplete matches We observed that the occurrence metrics increase when we allow for partially incomplete patterns to be discovered (hallucination). However, the exact



The reference annotation is colored in gray and the identified pattern occurrences outlined in red. Even though similar, the events on the right side (shown in green) are not part of the reference annotations; however, they are included in the model's output due to their co-occurrence with other events.

Figure 5.5
Incomplete match of two pattern occurrences in Bach BWV 889 Fugue in A minor (from the JKU PDD dataset). Two pattern occurrences are presented in the Figure (top and bottom).

F_1 scores do not always increase. After observing the pattern occurrences, which do not contribute to the rise in the F_1 score, we discovered that these patterns do not completely match the reference annotations, as shown in Figure 5.5.

The difference between a reference annotation and the model's output usually occurs at the edges of a pattern, where the model assumes that one or more preceding or succeeding events belong to the pattern. These events frequently occur at the same locations (relative to the pattern), with similar time and pitch offsets. This is the reason why the model adds these events to the pattern occurrence, causing mismatch with the reference annotation. Such errors could be resolved by incorporating theoretical rules governing the beginnings and endings of patterns, e.g. the gap rule ([172], p. 68).

Unidentified patterns The patterns which were not identified by the model usually belong to one of the two types—section patterns and short patterns.

The section patterns, such as in Mozart's Piano Sonata in E flat major, K. 282-2nd movement, remain unidentified. These section patterns represent large segments of music (50–137 events). The six layers in our model have the potential of encoding patterns of up to 64 events. While some of the reference patterns could be identified, the model did not contain a sufficient amount of layers to cover the largest patterns. We consequently focused on observing the absence of the shorter section patterns (between 50 and 64 events). While incomplete (often overlapping) matches of these patterns were found on the $\mathscr{L}_5$ and $\mathscr{L}_6$ layers (sub-patterns), there were no complete matches between the found patterns and the reference annotations. Also, the overlap of such parts on $\mathscr{L}_5$ was not high enough in order for these sub-patterns to merge during pattern merging.

The second subgroup—the short patterns—also frequently occurs in evaluation datasets. These patterns are 4–5 events long. They are identified by the model on the layers $\mathscr{L}_2$ and $\mathscr{L}_3$, but the respective parts also frequently form compositions on higher layers. If this is the case, the pattern selection procedure excludes the short patterns from the model's output.

The discovery of long patterns could be improved by building additional compositional layers, while training the model, and by adjusting the merging rules for long patterns. To find more short patterns, we could add supplementary criteria that would counterbalance the promotion of longer patterns during pattern selection. For example, event duration could be used when considering the importance of short events.

5.2.7 Drawbacks of the evaluation

As thoroughly discussed by Meredith [124], the pattern finding MIREX task possesses many drawbacks and thus might not be an optimal tool for evaluation of melodic pattern finding algorithms.

First and foremost, the definition of a pattern is vague. Some of the patterns in the ground truth represent themes, while others represent entire sections. Without any prior knowledge about the goal (pattern length or ratio between the length and the variation within the pattern), the metrics are logically leaning towards awarding the approach which finds the most occurrences of the discovered pattern. Designing an algorithm capable of finding a ”pattern” seems impossible when the definition of a pattern varies among the annotators. The three-layer F-score proposed by Meredith is a step towards a metric which provides the balance between the establishment and the occurrence metrics.

The size of the dataset is also small: the combined JKU PDD and MIREX datasets represent ten (classical) musical pieces in total. It is thus difficult to claim that the datasets provide a representative sample of any kind of music or genre. However, we acknowledge the effort put in the creation of the datasets and the tasks; we believe the size of the datasets is affected by the effort needed.

Nevertheless, it is rather difficult to create an experiment which would provide a clearer evaluation of discovery algorithms and we believe that the MIREX task is currently still the best approach for their comparison.

5.2.8 Applying the SymCHMMerge to polyphonic pattern discovery

The proposed model is general, so it was also tested for the task of finding patterns in polyphonic music, where multiple events can occur at the same time on different locations.The results on the MIREX polyphonic pattern discovery task on the JKU PDD dataset and the model's comparison to the reported results of other approaches are shown in Table 5.5.

The results of the proposed model are, according to the three-layer F1 metric, comparable to other algorithms—four algorithms perform better, while four perform worse than the proposed model. The occurrence metrics significantly change with the stricter tolerance *c*. We can conclude that the algorithm has problems when identifying the exact pattern boundaries—when the level of exactness is relaxed, the score is significantly raised.

We find the results good, especially as we implemented no specific mechanisms for determining the importance of patterns, such as various compression metrics used in some of Meredith's approaches. Such a mechanism could help improve the selection of relevant patterns—namely, based on the MIREX task definition, any repeated sequence is a pattern. However, an expert annotator does not classify a repeated sequence as a pattern only due its repetitiveness.

Nevertheless, it is evident from the visualization of the discovered patterns in Figure 5.6 that the found patterns are relevant. In the example, our model finds 13 patterns, while 11 are present in the ground truth. The discovered patterns represent partial or whole repeated structures, some are over-segmented (we find several shorter occurrences instead one longer), some subsumed (such as the shorter ground truth patterns).

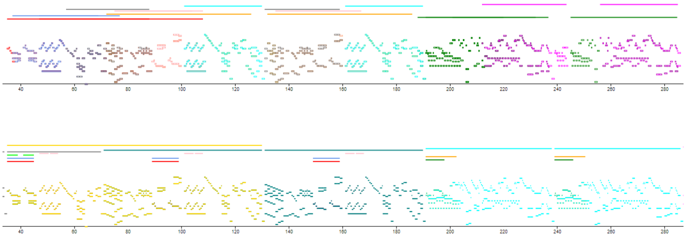
Parameter sensitivity We have re-evaluated the model's sensitivity to parameters for the polyphonic version of the task on the JKU PDD dataset and the obtained results shown in Fig.5.7. The sensitivity to parameter values shows similar behaviour to the monophonic results with less fluctuations. The model's performance is stable in the middle range of the parameters. For some of the parameters (e.g. inhibition parameter and occurrence scores) the scores seem to show some variability, which we attribute to the small size of the dataset.

Table 5.5
Results of polyphonic pattern finding on the JKU PDD dataset.

Algoritm	$P_{occ(c=.75)}$	$R_{occ(c=.75)}$	$P_{occ(c=.5)}$	$R_{occ(c=.5)}$
MotivesExtractor	0.58	0.45	0.54	0.52
Motives_poly	0.66	0.51	0.60	0.50
SIATECCompressSegment	0.50	0.78	0.44	0.78
SIATECCompressRaw	0.18	0.09	0.30	0.28
SIATECCompressBB	0.33	0.49	0.41	0.64
COSIATECSegment	0.65	0.75	0.57	0.72
COSIATECRaw	0.15	0.15	0.31	0.30
COSSIATECBB	0.26	0.31	0.54	0.64
SymCHMMerge	0.17	0.17	0.49	0.53

Algoritm	P_{est}	R_{est}	P_3	R_3
MotivesExtractor	0.59	0.59	0.46	0.48
Motives_poly	0.55	0.53	0.44	0.48
SIATECCompressSegment	0.51	0.68	0.44	0.56
SIATECCompressRaw	0.19	0.37	0.14	0.32
SIATECCompressBB	0.39	0.62	0.33	0.52
COSIATECSegment	0.44	0.64	0.40	0.54
COSIATECRaw	0.18	0.36	0.16	0.35
COSSIATECBB	0.30	0.54	0.29	0.52
SymCHMMerge	0.45	0.37	0.46	0.37

Algoritm	$F_{1occ(c=.75)}$	$F_{1occ(c=.5)}$	$F1_{est}$	TLF_1
MotivesExtractor	0.49	0.53	0.54	0.41
Motives_poly	0.57	0.53	0.48	0.39
SIATECCompressSegment	0.59	0.56	0.57	0.48
SIATECCompressRaw	0.12	0.29	0.25	0.19
SIATECCompressBB	0.39	0.50	0.48	0.40
COSIATECSegment	0.69	0.63	0.50	0.44
COSIATECRaw	0.15	0.30	0.24	0.21
COSSIATECBB	0.28	0.58	0.38	0.36
SymCHMMerge	0.17	0.51	0.39	0.40



Horizontal lines above each piano roll indicate the length of individual patterns.

Figure 5.6
The Figure represents a part of Mozart's Piano Sonata in E flat major, K. 282-2nd movement. The graph above depicts the results returned by our algorithm while the graph below represents the expert annotations of the repetitive patterns.

5.2.9 Discussion

In our experiments, we showed that the model can be used to find patterns in symbolic music and that it can learn to extract patterns in an unsupervised manner without hard-coding music theory rules. In the audio-related tasks of the previous chapter, we used the model for classification, while in this chapter, we demonstrated that the model can also be applied to unsupervised pattern extraction, where its transparency enables interpretation of the learned knowledge.

As pointed out, this evaluation contains many potential drawbacks, but it is currently the best choice for pattern discovery evaluation. The definition of the 'pattern' itself is elusive and may contain many different explanations, varying from strictly music-theoretical to mathematical formalizations. The human perception of patterns in music itself is too difficult to explain and incorporate in a single formalized task. However, with the proposed model we have demonstrated that a deep transparent architecture can be used to tackle pattern discovery. Unsupervised extraction of knowledge may better approximate listeners' recognition of patterns than rule based systems. Due to its transparency, hallucination and inhibition, the model is also very suitable for inclusion into a semi-automatic exploration and pattern discovery tool. The model can produce multiple hypotheses on several layers, which can be used as reference points for deeper semi-automatic music analysis by an expert.

By analyzing the results, we have identified the most common problems of the model, which provide ideas for further improvements, specifically establishment of very long and very short patterns, better pattern selection based on importance, inclusion of note durations and better establishment of pattern boundaries.

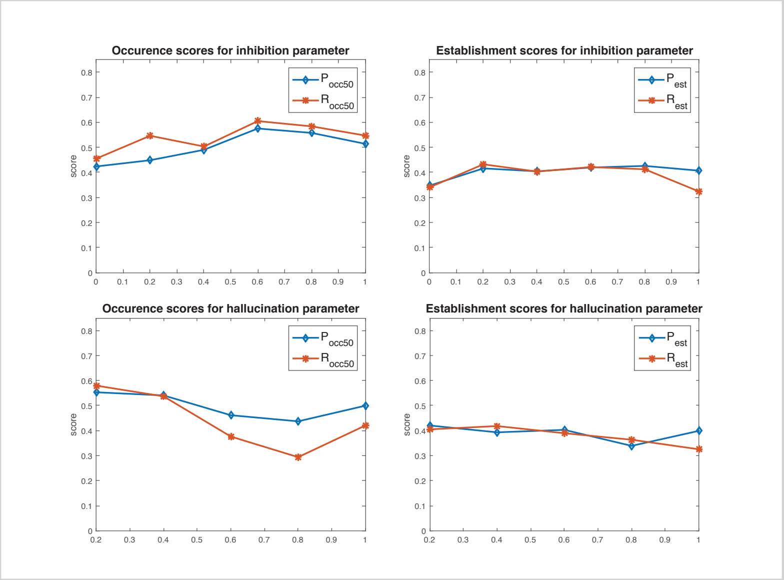


Figure 5.7 Sensitivity of the model to the changes of the inhibition parameter τ_I (top) and the hallucination parameter τ_H (bottom) for polyphonic music. When one parameter was changed, all others remained fixed.

5.3 *Evaluation: classification of melodies into tune families*

In the previous section, we applied the SymCHM to pattern extraction from symbolic music. Due to the drawbacks of the task and especially its evaluation, as discussed in Section 5.2.7, we also decided to test the model as a feature extractor for classification of melodies into tune families.

The goal of tune family classification is to group a set of tunes according to their common 'ancestor', thereby forming families of related tunes. The field of ethnomusicology has been leading this research for decades, and most of the work has been done manually by aligning and comparing groups of related melodies [173]. In recent years, researchers from computational musicology have proposed several automated methods for this problem.

As the amount of shared melodic patterns between two songs represents an important factor in determining whether two songs belong to the same tune family, we decided to explore whether our compositional model can be used for the task. We performed the evaluation on two datasets: the Slovenian folk song dataset and the Dutch folk song dataset.

The SymCHMMerge was trained unsupervisedly, using the same procedure as in the previously described pattern discovery task. Instead of training the model on a single music piece, it was trained to find patterns on a set of training examples and then inferred on a test dataset.

Instead of outputting the learned patterns for each song, the model was used as a feature generator—for each song, a feature vector was calculated. Each model part was mapped to a vector component, whose value corresponded to the sum of the magnitudes of all the part activations on the song. The reasoning is that the songs which contain similar patterns will activate similar parts and will thus have similar feature vectors.

5.3.1 Experiment 1: Slovenian folk songs

The experiment was performed on the manuscript collection of Slovenian folk songs 'OSNP' (Slovenian—'zbirka Odbora za nabiranje slovenskih narodnih pesmi z napevi'), which was created between 1906 and 1913, and contains about 12,000 folk songs. The songs are the part of the digital archive 'Ethnomuse', maintained by the Institute of Ethnomusicology, Scientific Research Centre of the Slovenian Academy of Sciences and Arts. The songs in the collection are both polyphonic and monophonic, and contain a variety of annotations, including classification into tune families. The tune families in the dataset can be very small (just a single song), or may contain hundreds of songs. Therefore, we selected a subset of the dataset for the experiment to obtain a uniform distribution of songs in the included tune families. We randomly chose ten tune families with ten songs per family.

We used stratified sampling to split the dataset into a training set, containing 80 percent of the songs, and a test set containing the rest. The model was trained on the training set.

For classification into tune families, we used a random forest classifier, which classified the model's feature vectors into the respective tune families. The classifier was trained on the training set. On the test set, the classifier correctly classified 34 % of tunes into their tune families, which is better than random (10 %); however, the percentage was much lower than initially expected. Analysis of the model's performance identified several problems with the used dataset.

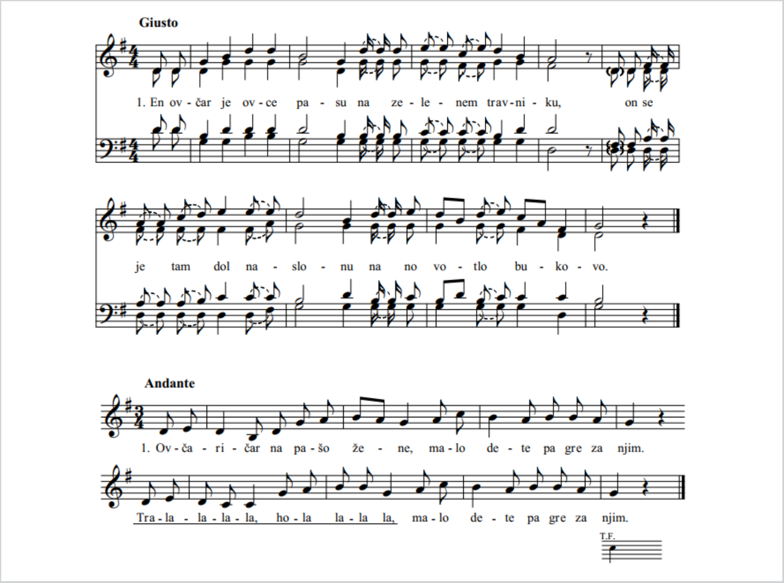

Giusto
1. En ov - čar je ov - ce pa - su na ze - le - nem trav - ni - ku, on se
je tam dol na - slo - nu na no vo - tlo bu - ko - vo.
Andante
1. Ov - ča - ri - čar na pa - šo že - ne, ma - lo de - te pa gre za njim.
Tra - la - la - la - la, ho - la la - la la, ma - lo de - te pa gre za njim.
T.F.

Figure 5.8
Both songs are part of the same tune family. The melodies are very different but lyrics describe a similar topic. Source: Žerovnik [169].

LYRICS VERSUS MELODY The tune family attributes in the OSNP dataset were based mostly on the song contents (lyrics), while the melody could significantly differ among different variants. An example is given in Fig.5.8.

SIMPLICITY OF FOLK SONGS Folk songs consist of relatively simple melodies. In some cases, the model identified patterns shared among the songs which do not belong to the same tune family. As the model does not incorporate additional know-how of pattern structure, such as bar placement, note duration or other modalities of sheet music, it treats the songs sharing common patterns as similar, although an annotator might decide otherwise. An example is shown in Fig.5.9.

Comparison to human performance To put the obtained model performance in perspective, we performed an additional experiment, for which we asked two annotators to blindly classify the song melodies (without lyrics) from the dataset into 10 classes according to their similarity. The annotators possessed basic music knowledge obtained

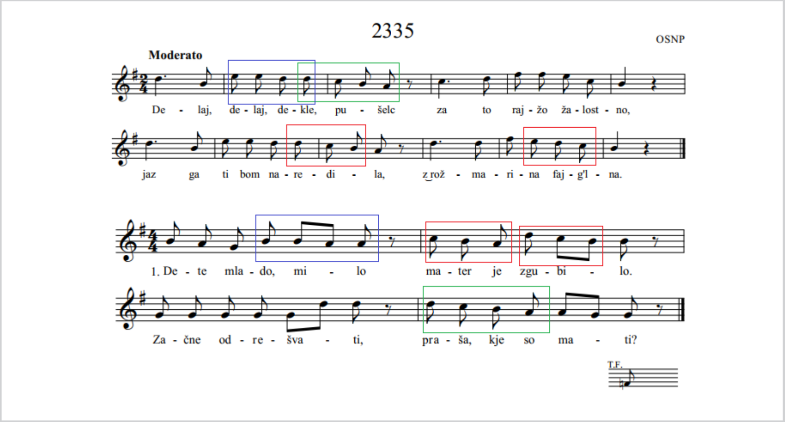

2335
OSNP
Moderato
De - laj, de - laj, de - kle, pu - šelc za to raj - žo ža - lost - no,
jaz ga ti bom na - re - di - la, z rož - ma - ri - na faj - g'l - na.
1. De - te mla - do, mi - lo ma - ter je zgu - bi - lo.
Za - čne od - re - šva - ti, pra - ša, kje so ma - ti?
T.F.

Figure 5.9
Two different folk songs with similar patterns. The similarity is however coincidental, and if note duration and bar placement were considered, such common pattern occurrences could be ignored. The model misclassified the songs as members of the same tune family. Source: Žerovnik [169].

Table 5.6
classification accuracy of the model and two human annotators.

	SymCHM	Annotator 1	Annotator 2
Classification accuracy	0.34	0.36	0.35

within a standardized 6-year lower music school programme and had several years of experience in performing music. Additionally, both annotators had previous experience with music analysis, transcription and annotation.

To eliminate the potential bias, no additional information about the songs was provided. The contents of the classes were not predefined, only their number (there was no option of enlarging or reducing the number of classes). The annotators were asked to classify the songs according to their melodic similarity.

Using this procedure, we obtained human annotations, which we compared to the annotated tune families. Melodies from each class were assigned to the tune family prevalent in the class. The results of this classification are given in Table 5.6. It is evident that the human annotators can also not classify songs into distinct classes/tune families very accurately, as they reach approximately the same score as our model.

5.3.2 Experiment 2: Dutch folk songs

To compare the model to other approaches for tune family classification, we performed an experiment with the Dutch folk song dataset provided by van Kranenberg et al. [174]. The collection consists of several annotated subsets [175], including the MTC-ANN dataset containing 360 Dutch folk songs classified into 26 tune families. All songs are monophonic.

On this dataset, Van Kranenburg et al. reported excellent results for a retrieval task based on nearest neighbours. Each tune is first characterized by a feature vector of 88 global features gathered from several sources. 51 features were collected by Steinbeck and Jesser [176] by exploring the Essen Folk Song Collection, which contains 10,000 folk songs from Germanic regions and China. 37 features were collected by McKay [177] and were collected for general genre and song analysis. The features incorporate hand-crafted measures, which are based on statistical observations of note occurrences, combinations with fine-tuned threshold and specific combination of spectral features. The nearest neighbour classification is used. Each song is classified into the tune family of a song which is the closest in the vector feature space. The comparison of vectors is performed by using the cosine distance.

We used the same retrieval setup in this experiment as in the previous Slovene songs dataset experiment; however, the feature vectors were obtained as follows. We trained one SymCHM model on all 360 songs of the MTC-ANN dataset and obtained a model with 3750 parts across layers 3–7. The model was then inferred on each song and its output encoded into a feature vector where each part was mapped onto one vector element whose value represented the sum of the part's activations. The vector values were then adjusted as described in Van Kranenburg et al. [178]. For each element, the values were standardized across the dataset to have zero mean and a standard deviation of 1. As with Van Kranenberg et al. [178], the cosine distance was used for comparison of vectors.

Our model reached 74.4 % classification accuracy on the dataset. The confusion matrix is depicted in Figure 5.10. The results are about 20 percent lower when compared to [178]. However, we believe the results are interesting, considering the fact that a pattern discovery model, relying only on onset-pitch notation, was used for this task. The model was not specifically trained or parameter-tuned for this task and was applied to the dataset without any dataset-specific adjustment. The model provided compositions of relatively-encoded melodic patterns learned in an unsupervised manner. Other ap-

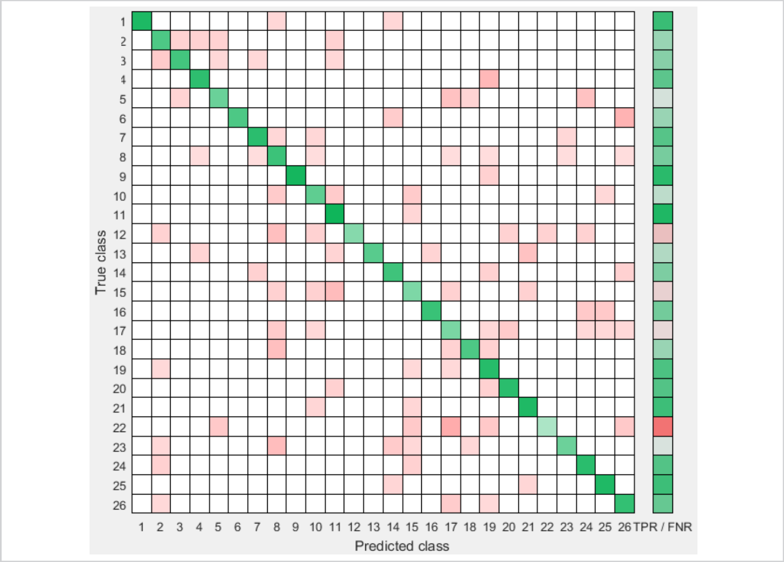


Figure 5.10
The confusion matrix of tune family classification with SymCHM. The reference annotations are represented in rows (left) and the predicted classes in columns (bottom).

proaches applied to the MTC-ANN dataset used additional spectral features, e.g. Van Kranenburg [178], and symbolic features, e.g. Walshaw [136] who used bar indicators. In contrast, no know-how about the dataset or folk and western music in general was used in the procedure. Of course, inclusion of such knowledge could also be beneficial and will be explored in our future work.

6

Compositional Hierarchical Model for Rhythm Modeling

In this chapter, we present how the compositional hierarchical model can be used for modeling rhythm. We thus focus on the temporal aspects of music and ignore the harmonic and melodic aspects that were discussed in the previous two chapters. Our motivation stems from the fact that some of the model's features are intuitively applicable to rhythm. For example, the relative encoding of time in rhythmic structures is commonly used in rhythmic representations. In live music, such relative encoding comes natural—a rhythmic pattern may vary in duration due to tempo changes, yet it retains its inner structure. When studying rhythm in music corpora, rhythmic patterns occur in different tempi across pieces, so their relative encoding is necessary if they are to be studied. In addition, the model's biologically-inspired mechanisms aid in handling the variability of rhythmic patterns, which commonly occur in the transitions between segments (e.g. drum transitions) and in segment repetitions (e.g. half-feel and double-feel).

This chapter summarizes our latest work. The results represent the work in progress, leaving several aspects of the model's development for future work. Nevertheless, we show how the model can be used for modeling rhythm and demonstrate its abilities through several examples, which are connected with the rhythm-related tasks, such as tempo estimation, rhythmic classification and beat tracking.

6.1 Model Description

Input of the rhythmic model consists of the onset times and the magnitudes of music events. These may be extracted either from audio recordings (with an onset detector) or from symbolic representations. The input thus contains onset times and their magnitudes. In contrast to the SymCHM, pitch information is ignored:

$$\mathcal{I} : \{\mathbf{X} : \mathbf{X} = [\mathrm{N}_o, 0, \mathrm{N}_m]\}. \tag{6.1}$$

As in the previous implementations, the first layer $\mathcal{L}_0$ consists of a single atomic part P_1^0. Since any *rhythm* is composed of at least two events (i.e. a single event cannot by itself represent rhythm), P_1^0 activates for all the pairs of input events $i_1 = [\mathrm{N}_o^1, 0, \mathrm{N}_m^1]$ and $i_2 = [\mathrm{N}_o^2, 0, \mathrm{N}_m^2]$, where i_1 occurs before i_2, as:

$$\mathrm{A} = \langle \mathrm{A_T}, \mathrm{A_L}, \mathrm{A_M} \rangle \leftarrow \langle \mathrm{N}_o^1, \mathrm{N}_o^2 - \mathrm{N}_o^1, (\mathrm{N}_m^2 + \mathrm{N}_m^1)/2 \rangle \tag{6.2}$$

The onset time A_T is defined by the onset time of the first event, the magnitude A_M is the average magnitude of both events. The role of activation location A_L is different in this model, as it represents the *scale* of activation on the time axis. On the first layer, A_L is defined as the difference of onset times of both events (the difference in their length). Namely, as each rhythmic pattern in our model is relatively encoded, the activation scale represents the timing (speed) with which it has been located in the model's input. Scale will distinguish between two pattern occurrences found at the same onset, one faster (small scale), and one slower (large scale).

6.1.1 Rhythmic compositions

The definition of parts (compositions) on the higher layers was extended in the rhythmic model. In addition to the Gaussian that regulates the relationship between subpart locations (difference in frequency or pitch in the previous models), we introduce an additional Gaussian, so that the part definition changes to:

$$P_i^n = \{P_{k_0}^{n-1}, \{P_{k_j}^{n-1}, (\mu_{1,j}, \sigma_{1,j}), (\mu_{2,j}, \sigma_{2,j})\}_{j=1}^{K-1}\}. \tag{6.3}$$

The role of μ_1 and μ_2 is as follows. μ_1 defines the size of the subpart *relative* to the size of the central part. Values of μ_1 larger than one indicate that the subpart is longer than the central part; the values smaller than one indicate the reverse. When the model is trained on a corpus, the values of μ_1 usually converge to the integer ratios commonly present in different time signatures, e.g. $1/5$, $1/4$, $1/3$, $1/2$, 1, 2 etc. The given activations of the subpart $P_{k_1}^{n-1}$ and the central part $P_{k_0}^{n-1}$, μ_1 are calculated as:

$$\mu_1 = \frac{A_L(P_{k_1}^{n-1}) * (1 + \mu_1(P_{k_1}^{n-1}))}{A_L(P_{k_0}^{n-1}) * (1 + \mu_1(P_{k_0}^{n-1}))} \tag{6.4}$$

The second parameter μ_2 defines the placement (onset) of the subpart *relative* to the size of the central part. Thus, values larger than one indicate that the subpart's onset comes after the end of the central part (there is a gap in between), the value of one means that the subpart starts exactly at the end of the central part, while smaller values indicate an earlier onset—an overlap between both subparts. The parameter is calculated as:

$$\mu_2 = \frac{A_T(P_{k_1}^{n-1}) - A_T(P_{k_0}^{n-1})}{A_L(P_{k_0}^{n-1}) * (1 + \mu_1(P_{k_0}^{n-1}))} \tag{6.5}$$

Figure 6.1 shows four simple compositions of $\mathscr{L}_0$ parts with different parameters.

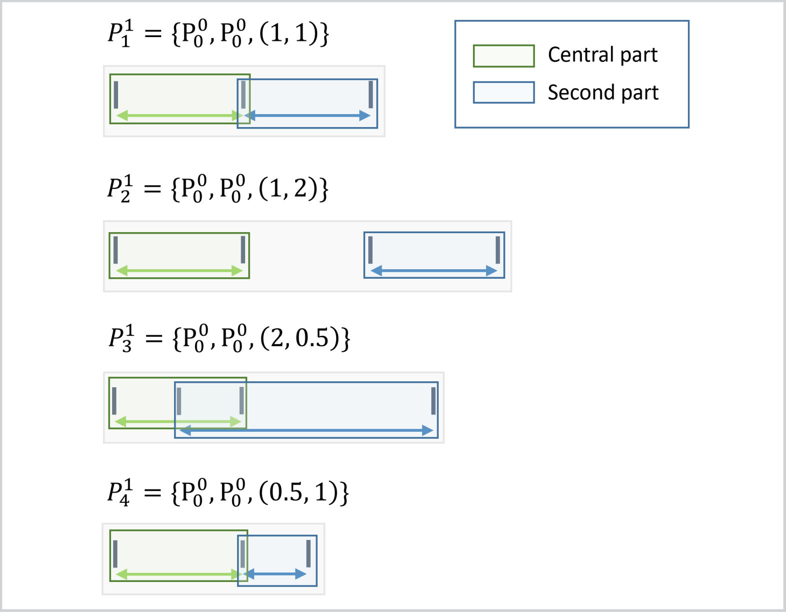

$P_1^1 = \{\mathrm{P}_0^0, \mathrm{P}_0^0, (1,1)\}$
Central part
Second part
$P_2^1 = \{\mathrm{P}_0^0, \mathrm{P}_0^0, (1,2)\}$
$P_3^1 = \{\mathrm{P}_0^0, \mathrm{P}_0^0, (2,0.5)\}$
$P_4^1 = \{\mathrm{P}_0^0, \mathrm{P}_0^0, (0.5,1)\}$

Figure 6.1
An example of four different RhythmCHM parts. In P_1^1, the second part is the same length as the central part and begins at an onset positioned exactly one length of the latter after the central part's position. In P_2^1, the second part is the same length as the central part and occurs at twice its length after the central part. In P_3^1, the second part is twice the length of the central part and starts at half its length after the central part. In P_3^1, the second part is half the length of the central part and starts at exactly its length after the central part.

6.1.2 Activations on higher layers

The activations of compositions on the higher layers are calculated in the same manner as presented in Chapter 3. The key difference is semantic—while the $\mathrm{A_L}$ component represented the frequency and the pitch-related information in the previous models, it represents the scale of the pattern's occurrence here. Activation components $\mathrm{A_T}$ and $\mathrm{A_M}$ retain their meaning and represent the pattern's onset time and magnitude.

The role of $\mathrm{A_L}$ is illustrated in Figure 6.2, which shows a simple $\mathscr{L}_1$ part with several activations on different scales. The fact that the part's encoding of the events in a rhythmic pattern is relative and its scale is only established during the inference and encoded in activations means that the model's parts encode rhythmic information independently of tempo, and that the model may easily follow the patterns in pieces with changing tempo or in corpora that contain pieces of varying tempi.

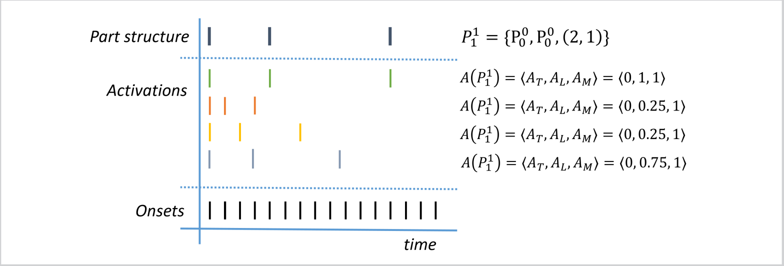

Part structure
$P_1^1 = \{\mathrm{P}_0^0, \mathrm{P}_0^0, (2,1)\}$
Activations
$A(P_1^1) = \langle A_T, A_L, A_M \rangle = \langle 0, 1, 1 \rangle$
$A(P_1^1) = \langle A_T, A_L, A_M \rangle = \langle 0, 0.25, 1 \rangle$
$A(P_1^1) = \langle A_T, A_L, A_M \rangle = \langle 0, 0.25, 1 \rangle$
$A(P_1^1) = \langle A_T, A_L, A_M \rangle = \langle 0, 0.75, 1 \rangle$
Onsets
time

Figure 6.2
An $\mathscr{L}_1$ part with three events is activated on a simple input signal. Four activations are shown, all occurring at the same time, but with different scales encoded in A_L (between 0.25 and 1).

6.2 *Learning and inference*

The learning and the inference in the rhythm model do not significantly differ from the previous models. A single music piece or a corpus of many pieces can be used for training the model. Learning is performed layer-by-layer, each new layer consists of compositions of parts from the previous layer. However, several small changes needed to be introduced due to the adjustments of part structure.

6.2.1 *The learning algorithm*

The overall learning algorithm is very similar to the one described in Chapter 3. The main difference is in the algorithm for the generation of new candidate compositions, which has been adjusted according to the changed part structure. Introduction of the second Gaussian parameter changes the dimensionality of the generated histograms to three dimensions. The μ_1 and μ_2 parameters are relatively defined, so the first and the second histogram dimensions become ratios, making the histograms a ratio-ratio-frequency representations, as shown in Fig. 6.3.

As in previous models, new compositions are formed from parts where the number of co-occurrences exceeds the learning threshold τ_L. The composition parameters μ_1, μ_2 and σ_1, σ_2 are estimated from the corresponding histograms and each new composition is added to the set of candidate compositions $\mathscr{C}$. From this point on, the candidate picking procedure remains unchanged, as described in Chapter 3.

6.2.2 *Inference*

The inference algorithm is identical as for the symbolic SymCHM model. The trained model contains rhythmic structures encoded in compositions, starting from simple 3-

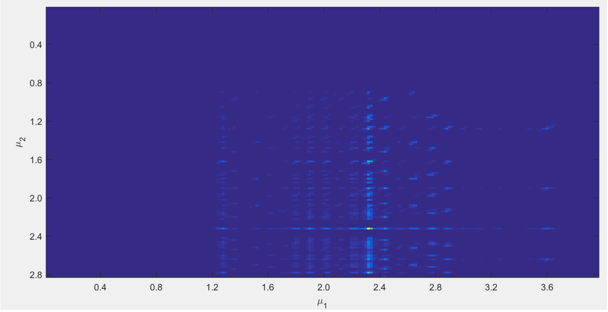


Figure 6.3
An example of the ratio-ratio-frequency histogram.

and 4-event structures on layer $\mathscr{L}_1$. Part activations are calculated when the model is inferred. The scaling factor, encoded in A_L, determines the length of the structure, A_T determines the onset time of the structure and A_M its magnitude.

The hallucination and inhibition mechanisms can both be used to robustly uncover rhythmic regularities in music. Both mechanisms behave identically as in the SymCHM version of the model.

Approximate inference aids to the model's ability to find rhythmic patterns with deletions, changes or insertions, thus increasing its robustness. The inhibition mechanism has a more pronounced role in the rhythmic model, because of the relativity of the encoded patterns and, typically, a high regularity of the input signal. Especially on the lower layers, a high number of activations representing simple straight rhythmic patterns emerges, typically out of a few simple parts activating at different scales and onsets. An example is given in Fig. 6.4, which shows a series of activations of a simple $\mathscr{L}_1$ part on a regular input signal. For clarity, only two different scales are shown in this example.

6.3 *Analyses*

We demonstrate the usability of the model for the extraction of rhythmic patterns in two simple experiments. In the first experiment, we assess how the model can extract patterns from different dance music genres and how the extracted patterns characterize different genres. In the second experiment, we show how the model can extract patterns from individual music pieces, in which the rhythm changes due to changing time signatures, as well as tempo variations.

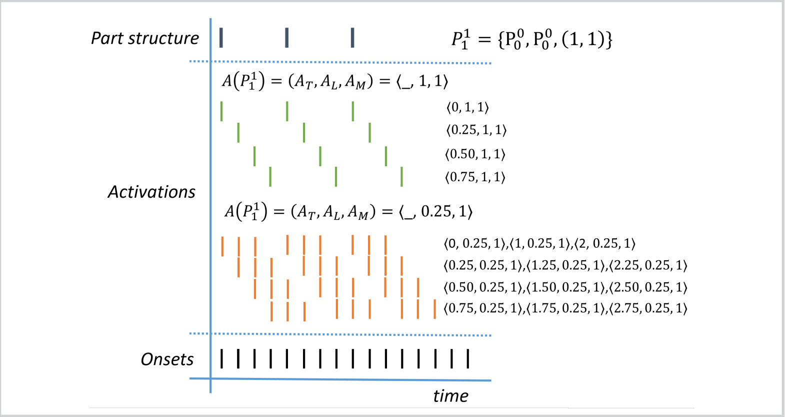

Part structure
$P_1^1 = \{P_0^0, P_0^0, (1,1)\}$
$A(P_1^1) = (A_T, A_L, A_M) = \langle \_, 1, 1\rangle$
$\langle 0, 1, 1\rangle$
$\langle 0.25, 1, 1\rangle$
$\langle 0.50, 1, 1\rangle$
$\langle 0.75, 1, 1\rangle$
Activations
$A(P_1^1) = (A_T, A_L, A_M) = \langle \_, 0.25, 1\rangle$
$\langle 0, 0.25, 1\rangle, \langle 1, 0.25, 1\rangle, \langle 2, 0.25, 1\rangle$
$\langle 0.25, 0.25, 1\rangle, \langle 1.25, 0.25, 1\rangle, \langle 2.25, 0.25, 1\rangle$
$\langle 0.50, 0.25, 1\rangle, \langle 1.50, 0.25, 1\rangle, \langle 2.50, 0.25, 1\rangle$
$\langle 0.75, 0.25, 1\rangle, \langle 1.75, 0.25, 1\rangle, \langle 2.75, 0.25, 1\rangle$
Onsets
time

Figure 6.4
The example shows overlapping activations of a single P_1^1 part. Inhibition can be employed to efficiently reduce such overlapping activations.

In both experiments we process audio recordings. To obtain the required onset-magnitude input representation, we first process the recordings with the CNN onset detector to extract the onset times of music events. As the onset detector does not output onset magnitudes by default, we set all the magnitudes to the same value of one.

6.3.1 Ballroom dancing

We first evaluated the model's ability to extract rhythmic patterns of different dance music genres. For the purpose, we used the Ballroom dataset that is also used in many of the MIREX tasks (tempo and genre estimation, beat tracking). The Ballroom dataset is publicly available online[1].

There are eight different genres in the Ballroom dataset—jive, rumba, cha cha, quickstep, samba, tango, Viennese waltz and (English) waltz. We generated a model for a subset of genres for which we expected to find distinctive rhythmic structures. For example, we expected to find swing patterns in jive, and similar patterns in rumba and cha cha but in different tempi. In the following subsections, we report on the learned structures.

Jive Jive music is based on a distinct swing rhythm, which contains unevenly spaced stressed events. In its everyday occurrence on radios, the swing rhythm is commonly associated with jazz. Jive is a medium to fast swing (commonly denoted as ”uptempo”)

[1] http://mtg.upf.edu/ismir2004/contest/tempoContest/node5.html

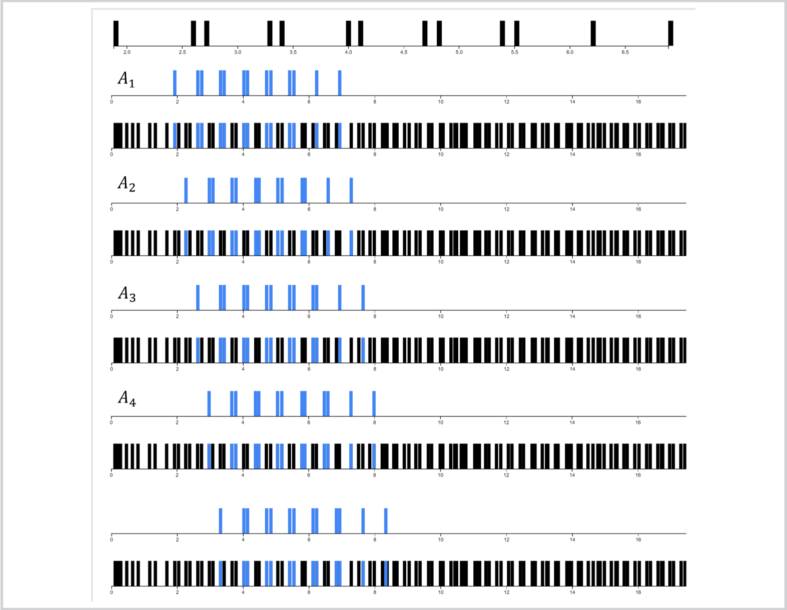


Figure 6.5 The $\mathscr{L}_5$ part representing the basic swing rhythm commonly found in jive songs. Below the part's structure, the first four activations are depicted individually with their projections onto the input layer. The decomposition of the part's structure is shown in Fig. 6.6.

and usually contains a happy or goofy tune. Compared to other genres in the Ballroom dataset, jive is the only genre based on the swing rhythm.

In the model trained on all jive pieces in the dataset, we expected to observe distinct swing-like rhythmic structures. The learned model contained 38 parts on five layers (number of parts per layer: $\{2, 8, 10, 9, 9\}$). The analysis of the model revealed several compositions on different layers which contained the distinct swing rhythmic structure.

An example of such a structure on the fifth layer is given in Fig. 6.5. More interestingly, as shown in Fig. 6.6, the $\mathscr{L}_5$ part was generated from a single $\mathscr{L}_4$ part, the latter from a single $\mathscr{L}_3$ part and so on to the first layer. The first layer contains only two parts, where the first part P_1^1 represents three evenly separated events (straight rhythm), with $\mu_1 = 1, \mu_2 = 1$ and the second part P_2^1 an uneven rhythm with $\mu_1 = 0.86, \mu_2 = 1$. This part composes all the parts with swing rhythm structures on the higher layers $\mathscr{L}_2 \dots \mathscr{L}_5$.

Along with these structures, the model also learned the parts representing the basic straight rhythm. The analysis of their activations showed that these parts activate on the downbeats of the rhythm, interpreting the input as a straight meter and ignoring the

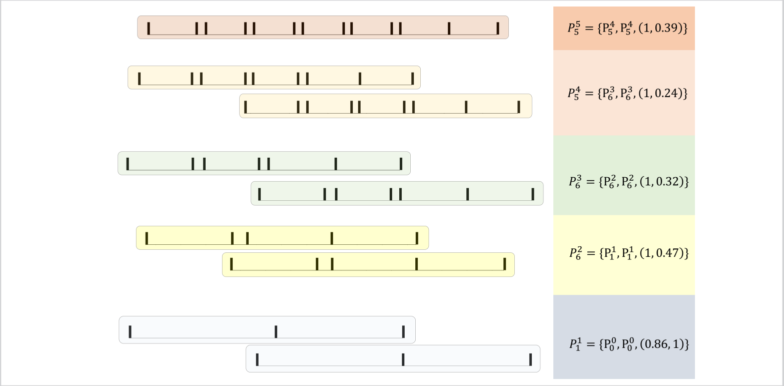
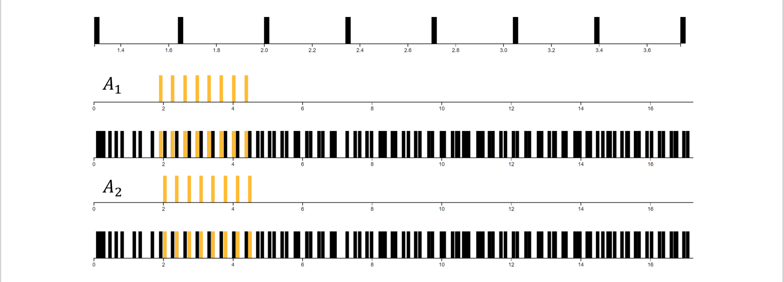

$P_5^5 = \{\mathrm{P}_5^4, \mathrm{P}_5^4, (1, 0.39)\}$
$P_5^4 = \{\mathrm{P}_6^3, \mathrm{P}_6^3, (1, 0.24)\}$
$P_6^3 = \{\mathrm{P}_6^2, \mathrm{P}_6^2, (1, 0.32)\}$
$P_6^2 = \{\mathrm{P}_1^1, \mathrm{P}_1^1, (1, 0.47)\}$
$P_1^1 = \{\mathrm{P}_0^0, \mathrm{P}_0^0, (0.86, 1)\}$

Figure 6.6
The structure of part P_5^5. All subpart compositions were formed from two instances of a single part on layers $\{\mathscr{L}_1 \dots \mathscr{L}_4\}$. The compositions' parameters are shown on the right side. Each part is shown twice with the offset used in the consecutive layer's composition.



The first activation A_1 covers eight consecutive downbeats, whereas the second activation A_2 covers the syncopation of the swing rhythm in the input, imitating the straight rhythm with a shifted phase.

Figure 6.7
The $\mathscr{L}_5$ part representing a straight rhythm. Below the part's structure, two typical activations are depicted individually with their projections onto the input layer.

syncopated second beat. In addition, such parts also activated on the syncopated beats. This second group of activations therefore acted as the straight meter with a shifted phase. An example is given in Fig. 6.7.

Samba Samba has a distinctive rhythm which partially resembles jive, with a distinctive difference in timing of the second beat, which is played in the straight rhythm. Due to its syncopation of the second beat, we can expect distinctive structures in the the learned model, which cover the specific offset of the second beat in the rhythmic pattern. As shown in Fig. 6.8, the samba's basic beat structure is sufficiently extracted from the input. The structure itself may seem similar to the previously shown jive structure.

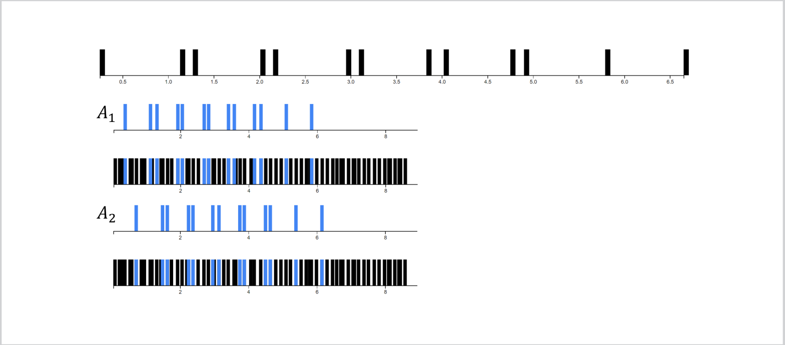


Figure 6.8
The $\mathscr{L}_5$ part representing the a basic rhythmic structure of a samba song. Below the part's structure, the first two activations are depicted individually with their projections onto the input layer. The part's structure is similar to jive, but contains different ratios between the first and the second beat.

However, it is not, since the ratio between the first and second beat is different. Compared to the jive structure, the difference in the ratios is about 15 percent.

Rumba and Cha Cha Both music genres belong to the group of Latin-American dances. While the rumba dance and music are associated with sensual topics, the cha cha contains more bright, powerful and uptempo beats. Both music genres contain a distinctive ”four-and-one” syncopation, which also defines the basic steps in the dance. Additionally, both genres are played in straight meter with strong accents on all four beats.

We built a separate model for each genre and analyzed the learned structures. In contrast to the previously analyzed genres, these two models shared a greater deal of straight rhythmic parts. The rumba-trained model had 36 parts on five layers, while the cha-cha-trained model contained 33 parts.

Straight patterns dominated in both models. The distinctive patterns, containing the ”four-and-one” beats, were not as dominant as we initially expected. This was mainly the effect of the variety of percussive and brass instruments playing granulated rhythmic riffs on and between all beats, which the onset detector does not distinguish from the others. Therefore, straight patterns dominate. The most closely associated typical pattern that was found is depicted in Fig. 6.9.

Tango Although tango possesses South American roots, the widely known international (European) version of the dance contains several aspects influenced by the Euro-

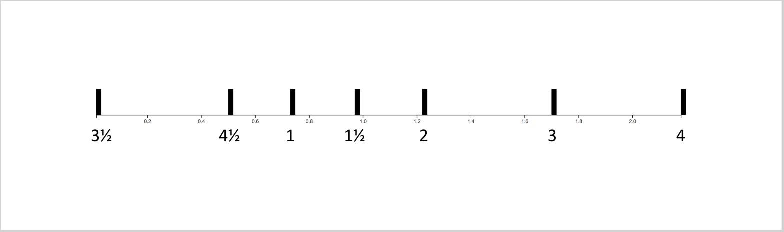


Figure 6.9
The most prototypical part found in the cha cha trained model on layer $\mathscr{L}_3$. The "four-and-one" pattern is partially explained. However, two other eight-note events are also included.

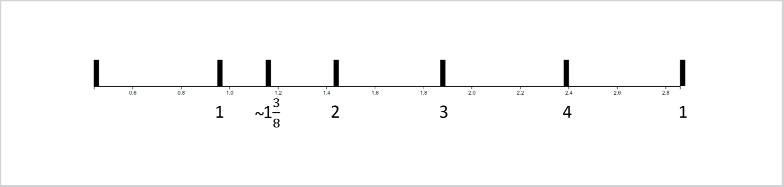


Figure 6.10
A tango part representing the regular "1-2-3-4" beats that match the meter and a syncopated off-beat (third onset from the left).

pean culture, such as the closed position of the dancers, more common in standard dances. The music associated with this dance is usually orchestral, played with severe stress on the syncopation (usually played by the violin and other string instruments) on the offbeat before the downbeat.

In the trained model we observed several parts forming a structure similar to the one displayed in Fig. 6.10. The off-beat is usually located between two beats and creates a syncopated rhythm, for example the first and the second beat in the meter, and can also appear on other locations. Onsets of these beats have a fast attack and decay and usually a higher velocity. The beat after is usually omitted to create a deeper sensation of the stressed beats. Even though these properties are not reflected in the input, which only contains the onset times, the learned structures do reflect the specifics of the rhythm.

6.3.2 Extracting patterns from live music

To show the model's ability to extract different rhythmic patterns from a single music piece, we looked at live music recordings on Youtube. In this context, an interesting live recording was produced by the Croatian singer Severina in a song "Djevojka sa sela"[2]. The live version performed at her concert in 2009 was accompanied by trumpeters who played a 7/8 meter in verse, whereas the rock band played a 4/4 meter in the

[2] Song available on Youtube https://youtu.be/heJQAckM-eI

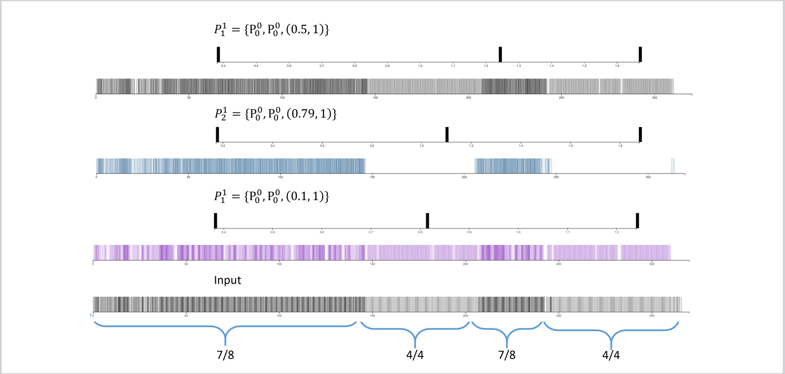


Figure 6.11
The three parts forming the $\mathscr{L}_1$ layer for the Severina song. It may be clearly observed that the P_2^1 part covers the song parts played in 7/8 meter, while the P_1^1 and P_3^1 parts activate across both 7/8 and 4/4 meters.

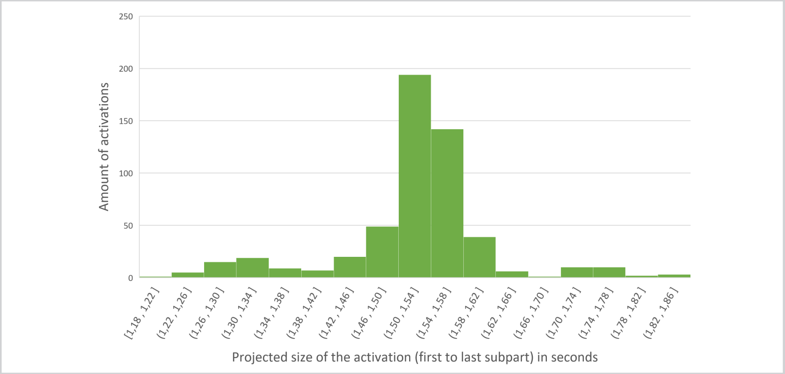


Figure 6.12
Varying activation sizes for a layer 2 part which activates on the 7/8 meter segments. The majority of activations is within 100 milliseconds of the average size of 1.52 seconds (about 6% difference in tempo); however, many occurrences also range from about 25% slower up to 25% faster).

refrain. By training the model on this song, we produced a 5 layer hierarchy, containing $\{3, 11, 11, 10, 9\}$ parts on layers $\{\mathscr{L}_1 \ldots \mathscr{L}_5\}$, respectively. We first observed the three parts forming the first layer. The part structures are depicted in Fig. 6.11.

As is evident in Fig.6.11, two parts P_1^1 and P_3^1 activate across the entire song, while one part activates only on the segments played in 7/8 meter. On layer $\mathscr{L}_2$, we can find five parts which cover 5–6 rhythmic events, explaining the 7/8 segments—as the 7/8 meter can be broken into 3/3 + 2/2 + 2/2 structure, these parts successfully find the downbeats of all three submeters.

The remaining six parts are the compositions of $\mathscr{L}_1$ parts, which cover both meters. Unfortunately, these parts also compose the majority of $\mathscr{L}_3$ parts and further. This is a consequence of the statistical nature of the model's learning. Namely, the parts covering both meters cover a greater portion of the events when compared to the parts covering only the 7/8 meter segments. Therefore, the greedy learning procedure favours such parts on the higher layers and tends to ignore the 7/8-meter candidates with smaller coverage.

Nevertheless, the model successfully shows the ability to distinguish rhythmic structures with different meters on $\mathscr{L}_1$ and $\mathscr{L}_2$ layers. Moreover, the song has a very uneven tempo due to the live performance and the style of trumpet playing. The variations are visible in Figure 6.12, which shows variations in the scale of activations of a $\mathscr{L}_2$ part, implying large differences in the timing of the events covered by this part. The relative encoding of the compositions enables the model to extract the rhythmic structures robustly in such circumstances.

6.4 Scalability

In order to evaluate how the model scales with the amount of training data, we trained 11 five layer models on datasets ranging from 400 to 350.000 events (onsets). The training dataset consisted of files from the ballroom dataset, augmented with 30 second clips of songs in a variety of popular music styles.

The results are shown in Table 6.1. The time needed to train a model grows linearly with the number of events. Training of a five layer compositional structure on a database of approx. 350.000 events takes 50 minutes on a single core CPU. The linear dependency is clearly visible in Fig. 6.13. Although a single core was used in our experiments, the model implementation can be highly parallel in most stages: during learning, generation of histograms, as well as candidate picking can both be parallelized, The inference process can also be parallelized, except for the inhibition mechanism.

The amount of parts per layer grows only slowly with larger training datasets. Due to the relative encoding of the learned structures, even small datasets can already produce parts general enough to cover a variety of input data. The first layer is always small (3-4 parts), while the number of parts on higher layers remains approximately constant (8-10 parts). Next to relative encoding, this is also likely due to the similar rhythmic structures used in the dataset — it mostly covers popular music genres. Using a more varied training dataset (e.g. music with very different metric structure) would likely increase the number of parts.

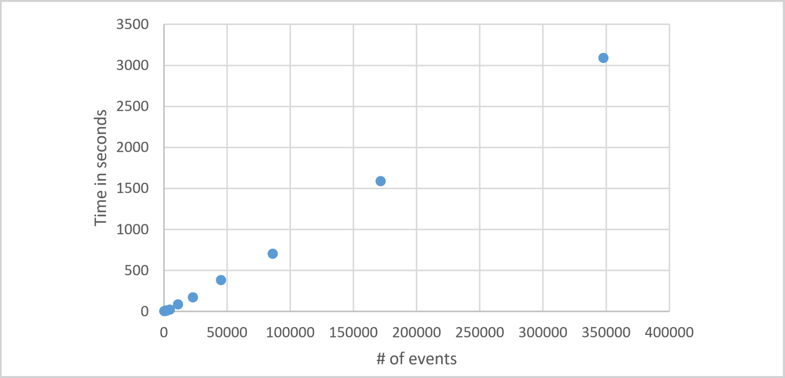


Figure 6.13

The graph shows the time needed to train a model (in seconds) in relation to the number of input events. The dependency is linear.

Table 6.1

The table summarizes the time needed and the number of learned parts when learning a five layer model with different number of input events. The left side of the table shows the number of music files and events in the input, along with the time needed to train a 5 layer hierarchy. On the right side, the number of parts per individual layer, along with the sum of all parts in the hierarchy, is displayed.

# of files	time (s)	events	$\mathscr{L}_1$	$\mathscr{L}_2$	$\mathscr{L}_3$	$\mathscr{L}_4$	$\mathscr{L}_5$	# of parts
2	3.77	389	2	6	9	10	9	36
4	5.16	660	3	10	9	9	10	41
8	10.43	1175	5	10	9	9	10	43
16	9.16	2307	3	10	9	8	9	39
32	20.13	4754	3	9	9	8	8	37
64	86.98	11097	4	9	10	8	9	40
128	171.00	22892	4	9	9	8	8	38
256	382.47	45229	4	10	9	9	9	41
512	704.29	86118	4	10	10	8	8	40
1024	1587.78	171585	4	10	10	9	9	42
2048	3092.72	347863	4	10	10	10	10	44

6.5 Discussion

The latest extension of the compositional hierarchical model for rhythm modeling was presented in this chapter. The part definition in the model was adjusted to sufficiently represent the rhythmic structures. Additionally, a different activation definition was proposed to efficiently encode the tempo changes in the input. The model was applied to the Ballroom dataset, commonly used for the genre classification, the beat tracking and the downbeat estimation tasks. A model was trained for each genre. Due to its transparency, the learned structures were clearly interpretable in the trained models. An analysis of the trained models showed the existence of structures resembling the specific rhythmic structures of individual genres.

Additionally, the model was applied to a live music example with meter changes. The observation of the learned concepts showed that a distinction among various meter segments can be made by an observation of part activations. Moreover, the model sufficiently overcame the problem of the varying tempo in the observed live performance. Due to the scaling component, the parts robustly activated on the tempo-varying rhythmic structures.

The initial results are encouraging and indicate the model's ability to model rhythm. The model does not incorporate any assumptions about specific meters and is therefore capable of analyzing different, also non-Western-influenced, rhythmic patterns. The robustness to tempo indicates a potential in modeling live audio. Further development and evaluation of this extension is anticipated, in which the model will be applied to the beat tracking and downbeat estimation tasks.

Conclusion

7

7.1 Overview

In this dissertation, we have presented the compositional hierarchical model for music processing. Based on the recent popularity of deep architectures, we proposed an alternative deep architecture, based on the principles of compositionality, transparency, unsupervised learning, relativity, and shareability of encoded structures. We described the model as a general learning and inference framework and applied it to several music information retrieval tasks.

The model was initially applied to tasks based on time-frequency representations of music signals: automated chord estimation and multiple fundamental frequency estimation. We trained several layers of the model, which was shown to successfully encode the concept of pitch, however was unable to encode more complex structures, such as intervals and chords, mainly due to the statistical dominance of higher harmonics. The model achieved good results for multiple fundamental frequency estimation - while it did not outperform the state of the art approaches on the established datasets containing well-defined timbres, it surpassed them on a dataset of field-recorded folk songs. In the latter, the songs were sung by amateur singers and recorded in every-day conditions with portable recording equipment. The model has shown its robustness and ability to perform in such situations. In addition, the model was shown to have low computational requirements for inference, as well as low memory footprint, making it suitable for real-time usage.

The model was subsequently applied to modeling melodic sequences in symbolic music representations. We evaluated it on the MIREX discovery of repeated themes and sections task. We showed that the model can unsupervisedly learn to encode melodic patterns in symbolic music and used its transparency to interpret the learned hierarchies and extract the learned patterns. While some of the more focused approaches achieved better overall results, the model was shown to be competitive and is currently the only deep architecture we are aware of to be applied to this type of tasks. We further improved the model's results with an improved pattern selection algorithm. To show the model's ability to perform as a feature extractor, we applied it to the tune family identification task. We have shown that activations of the model's parts can be used as features for classification of melodies into tune families, without explicitly encoding musicological know-how into the model or its features.

In our latest work we applied the model to modeling rhythm. Current results show that the model can successfully encode rhythmic structures independently of the underlying (and changing) tempo. Moreover, the model is agnostic to meter and can successfully extract rhythmic structures in different and also alternating meters.

7.2 Future work

Despite the amount of work, which has been put into the development and several implementations of this model, the model has several limitations, which are reflected in the results. The model can be unsupervisedly trained on small datasets with a small number of parameters. However, it has not, in its current implementation, significantly improved the current state of the art on the selected tasks. It performed well for multiple fundamental frequency estimation, where the model surpassed the compared approaches on the Slovenian folk song dataset. Results on many other tasks are satisfying but not the best (e.g. the discovery of repeated themes and sections), however we still managed to demonstrate the model's ability to perform in an unsupervised manner and without the need of domain-specific knowledge.

Future work will be directed to improving the model in several areas, which we outline below.

Considering the spectral representations, the future work includes an improved learning procedure in order to train models capable of encoding harmonic structures. In addition, magnitudes of harmonic series could be more explicitly encoded, which could improve the internal representation of the learned compositions for different instruments and improve overall results.

We will also focus on the symbolic music representations. We plan to include event duration into pattern selection and merging. We could also introduce pattern ranking and potentially include music theory rules. The model's output could further be optimized by supervised training of model parameters, especially the number of layers in the hierarchy and the layers in the model's output. However, a sufficiently large annotated dataset is needed for such optimization, significantly larger than the datasets currently used to evaluate the pattern discovery task. The model can also be used as an analytical tool. In this aspect, we are planning an experiment with multiple music theory experts and evaluate the model's transparent structure as a source for semi-automatic music analysis.

We also intend to extend the spectral model to encode long-term temporal dependencies of music events, thus encoding concepts such as melodic lines, chord progressions and rhythmic patterns. By combining symbolic and audio implementations of the proposed compositional hierarchical model, we aim to develop the model as a general purpose model for music information retrieval and music analysis. In this way, we will introduce a single model which covers different MIR tasks and is suitable for real-world applications.

One of the most promising future directions lays in our recent work with rhythm-related tasks. The initial results show that the model is capable of rhythmic-pattern extraction from live recordings and can discern between different meters and rhythmic patterns. We will apply the model to genre classification and downbeat tracking, especially focusing on non-mainstream music, which offers a variety of different rhythmic styles and meters.

7.3 *Principal Scientific Contributions*

In our work, we had followed the proposed dissertation topic, which served as our research plan. Specifically, this dissertation seeks to fulfill the following planned scientific contributions:

- *A short-time compositional hierarchical model featuring biologically-inspired mechanisms for music information retrieval.* The compositional hierarchical model has been developed and initially applied to automated chord estimation. Results were published in three conference and workshop publications [157, 179, 180].

- *Extension of the model to time-dependent music processing.* The model has been extended in several aspects to fulfill the requirements of this contribution. We first implemented the automatic gain control mechanism as a short-term time-dependent mechanism. We have evaluated the model for multiple fundamental frequency estimation and published the results in a scientific journal paper [158]. We have also developed the SymCHM modification of the model for discovery of melodic patterns. We have evaluated the model for discovery of repeated themes and sections [181] and published the results in two workshop papers [182, 183] and one scientific journal paper [184]. In its most recent development, the model was adjusted for rhythmic modeling of music.

- *Extension of the model for discriminative tasks.* The compositional hierarchical model was used for discriminative tasks in several different scenarios. In the spectral model, we introduced the *octave-invariant* layer, which was used for chord estimation [157]. In the multiple fundamental frequency task, the structure of parts was directly used to discriminate between pitches, while in tune family classification, activations of the model's parts were mapped to feature vectors that were used for classification [158].

Razširjeni povzetek

A

Motivacija S porastom globokih arhitektur, ki temeljijo na nevronskih mrežah, so se v zadnjem času bistveno izboljšali rezultati pri reševanju problemov na več različnih področjih. Zaradi popularnosti in uspešnosti takšnih globokih nevronsko-baziranih pristopov so bili drugi, predvsem kompozicionalni, pristopi odmaknjeni od središča pozornosti raziskav.

Kljub napredku pa trenutni globoki pristopi ne prinašajo popolne rešitve. Čeprav omogočajo nenadzorovano učenje in dosegajo zadovoljive rezultate pri različnih nalogah, jim manjka transparentnost, kar bi omogočilo vpogled v naučene koncepte. Vizualizacija naučenih konceptov trenutnih pristopov predstavlja netrivialen problem. Velikokrat se zato pristopi uporabijo kot črne škatle (angl. black box), ki sicer rešujejo nalogo, a jih je težko nadalje izboljšati in nadgrajevati. Prav tako so zaradi velike količine vozlišč in povezav v strukturah modelov za učenje potrebne velike podatkovne zbirke, ki jih je težko pridobiti. Med najpogostejšimi težavami pridobivanja zbirk so potencialni problemi z avtorskimi pravicami. Enako pomembna težava pa je tudi količina potrebnega časa za anotacijo zbirk. Anotiranje je pogosto subjektiven proces, ki za izravnavo pristranskosti zahteva več anotatorjev. Pri večini problemov potrebujemo anotatorje, ki so strokovnjaki v določeni domeni, na primer za glasbeno transkripcijo. Nenazadnje pa zbiranje anotacij zahteva veliko časa.

Seveda so v današnjem času globoke arhitekture učinkovito uporabljene za več različnih nalog, ki se nanašajo na razlikovanje med naučenimi koncepti. Takšni sistemi rešujejo vprašanje, ali opazovani vhod pripada eni ali drugi skupini. Nasprotno je težko uporabiti takšen model za odkrivanje zakonitosti – model bi moral izdelati lastno opažanje visokonivojskih abstraktnih pojmov, ki so prisotni na vhodu.

V pričujoči disertaciji se posvečamo vprašanju, ali je mogoče razviti globoko arhitekturo, ki bo presegla obstoječe probleme globokih arhitektur. S tem namenom se vračamo h kompozicionalnim modelom in predlagamo *kompozicionalni hierarhični model* kot alternativno globoko arhitekturo, ki ni osnovana na nevronskih mrežah.

Kompozicionalni Hierarhični Model V tem delu predstavljamo kompozicionalni hierarhični model za pridobivanje informacij iz glasbe. Z nenadzorovanim učenjem modela zgradimo hierarhično predstavitev konceptov, od enostavnih konceptov na najnižjem nivoju proti najkompleksnejšim konceptom na najvišjih nivojih. Ideja o takšni strukturi modela izvira iz raziskav na področju strojnega vida. Na slednjem sta Leonardis in Fidler [43, 153] predstavila koncept lHoP (angl. *learned Hierarchy of Parts*).

Njun model se lahko nauči hierarhične predstavitve objektov na slikah, začenši z enostavnimi gradniki na nizkih nivojih, ki jih združuje v kompleksnejše dele objektov na višjih nivojih. Model se uči na podlagi statistike pojavitev in ga je moč uporabiti kot robusten način za kategorizacijo objektov in druge sorodne probleme na področju računalniškega vida. V svojem delu bomo predstavili podoben koncept modela, ki bo izdelan specifično za področje pridobivanja informacij iz glasbe.

Ideja modela temelji na predpostavki, da lahko kompleksne sestavljene signale razdrobimo na enostavnejše gradnike – *dele*. Deli so lahko različno kompleksni in glede na kompleksnost tvorijo različne nivoje. Posamezne dele na višjih nivojih lahko tvorimo s kombiniranjem delov na nižjih nivojih in tako tvorimo kompozicionalni model. V glasbi je takšen pristop človeku intuitiven, saj so glasbeni dogodki tvorjeni na podoben način: akord je sestavljen iz vsaj teh tonov, posamezen ton pa iz več frekvenc. Posamezen del tako opisuje posamezne frekvence na nižjem nivoju, na višjih nivojih pa njegove tvorjene kombinacije – kompozicije – tvorijo kompleksnejše dogodke. Na enak način lahko modeliramo tudi vzorce v glasbi, sosledja tonskih višin in akordov. Celotna struktura modela je transparentna, saj lahko za vsak del pregledamo in interpretiramo njegovo vlogo.

Struktura modela Kot je prikazano na sliki 3.1, je model sestavljen iz začetnega nivoja $\mathscr{L}_0$ in več kompozicionalnih nivojev $\{\mathscr{L}_1, \dots, \mathscr{L}_\mathrm{N}\}$. Vsak kompozicionalni nivo $\mathscr{L}_n$ vsebuje množico delov $\{\mathrm{P}_1^n, \dots, \mathrm{P}_\mathrm{M}^n\}$, vsak del pa predstavlja kompozicijo delov s predhodnega nivoja $\mathscr{L}_{n-1}$ in hkrati tvori več delov na naslednjem nivoju $\mathscr{L}_{n+1}$.

Model naučimo nenadzorovano, kjer se na podlagi frekvence sopojavitev delov na vhodnem nivoju tvorijo deli na višjem nivoju v obliki kompozicij. Struktura posameznega dela je predstavljena v relativnem sistemu: vsaka nova kompozicija se tvori na podlagi relativnih razlik poddelov, ki kompozicijo tvorijo. Tako se lahko en tvorjen del pojavi na več različnih mestih, na primer: del, ki opisuje tonsko višino, se lahko v enem samem akordu, zaigranemu ob času t, pojavi trikrat.

Pojavitve delov poimenujemo *aktivacije*. Aktivacije predstavljajo pojavitve v obliki časa pojavitve, lokacije in magnitude, kot je predstavljeno v enačbi 3.2. Model na podlagi aktivacij tvori nove dele na višjem nivoju in njihovo sopojavitev sestavi v novo kompozicijo, pri čemer je struktura predstavljena relativno, kot je to zapisano v enačbi 3.1. Aktivacije delov se na višjih nivojih tvorijo na podlagi aktivacij poddelov na nižjih nivojih.

Kot vhod v model lahko podamo poljubno glasbeno predstavitev, ki vsebuje naslednje tri komponente: čas pojavitve dogodka, lokacijo dogodka in magnitudo dogodka, kot je zapisano v enačbi 3.5.

Učenje Model je zgrajen po nivojih z uporabo nenadzorovanega učenja na učni množici, začenši z nivojem $\mathscr{L}_1$. Učni proces lahko definiramo kot optimizacijski problem, kjer želimo tvoriti minimalno število delov (tj. novih kompozicij), ki bodo pojasnile čim večji delež informacij na vhodu. Učni proces temelji na statistiki pojavitev aktivacij in kombiniranjem sopojavitev slednjih.

Problem lahko formaliziramo tudi kot problem pokritja. Ker je ta problem NP-težek, smo razvili požrešno metodo, ki za pokritje preferira dele, ki pokrijejo več. Požrešna metoda v vsakem koraku izbere nov del iz množice kandidatov, ki jih predstavljajo na novo tvorjene kompozicije. Za dodani del velja, da pokrije čim večji del vhoda. V nadaljnjih korakih metoda dodaja nove dele, ki doprinesejo največ pokritja, pri čemer upošteva le dodano pokritje s strani na novo dodanega dela glede na predhodno pokritje vhoda s strani predhodno izbranih delov. Algoritem je predstavljen na sliki 3.4.

Biološko navdahnjeni mehanizmi Halucinacija omogoča nadomestitev izgubljene ali poškodovane informacije v glasbenem signalu in je implementirana v obliki aktivacije sestavljenega dela ne glede na morebitno pomanjkljivost vhoda. V primeru nepopolnega prileganja strukture nekega dela informaciji na vhodu se bo del aktiviral, v kolikor bo pokril dovolj velik delež informacije. Primer delovanja halucinacije na spektralni predstavitvi glasbe je prikazan na sliki 4.2.

Inhibicija je po drugi strani dejavnik uravnoteževanja, saj odstranjuje redundantne aktivacije na podoben način kot lateralna inhibicija, prisotna v človeškem slušnem sistemu. V primeru več aktivacij različnih delov, ki pokrivajo isto informacijo na vhodu, inhibicija odstrani tiste aktivacije, ki imajo nižjo magnitudo. Primer delovanja inhibicije na spektralni predstavitvi glasbe je prikazan na sliki 4.3.

Oba mehanizma na učinkovit način omogočata prečiščevanje hipotez in odstranitev šuma v modelu.

Relativnost in delitev struktur Predlagani model ima dve pomembni funkciji, ki ga ločujeta od drugih globokih arhitektur.

Relativnost struktur delov omogoča, da posamezni del predstavlja abstrakten visokonivojski koncept, ne glede na njegovo lokacijo v vhodnem signalu. Relativna percepcija se naravno pojavlja v procesu človeškega učenja. Je pomemben del abstrakcije in omogoča oblikovanje popolne percepcije ne glede na okolje. To zmanjšuje količino spomina, ki je potreben za shranjevanje naučenih konceptov in omogoča njihovo zanesljivo identifikacijo v predhodno neopaženih senzoričnih vhodih, kot so avdio signali slabše kvalitete in percepcija glasbe v hrupnem okolju.

Relativnost je neločljivo povezana z našim modelom in jo lahko opazimo v definicijah sestave delov in aktivacije, saj so deli relativni in predstavljajo abstraktne koncepte brez neposredne absolutne pozicije koncepta. Na primer, model tona G5 ne kodira eksplicitno, temveč kodira le koncept tona. Aktivacija dela podaja informacijo, kje in kdaj se v signalu pojavlja struktura opazovanega koncepta. Ker lahko do pojavitve pride na več lokacijah, ima lahko en del več aktivacij na različnih lokacijah.

Relativna narava delov omogoča tudi učinkovito delitev delov. Del na nivoju $\mathscr{L}_{n-1}$ je lahko poddel več kompozicij na nivoju $\mathscr{L}_n$. Posledično lahko posamezna dva ali več delov $\mathscr{L}_{n-1}$ sestavljata več različnih kompozicij $\mathscr{L}_n$ v različnih prostorskih kombinacijah.

Posledica relativnosti in deljivosti je zmožnost učinkovitega kodiranja kompleksnih konceptov. Kot primer: del, ki predstavlja koncept tonske višine, lahko zaradi deljivosti sestavlja več različnih kompozicij na višjem nivoju. To kodiranje je splošno, kompaktno in učinkovito, če upoštevamo alternativo kodiranja vseh tonskih višin na absoluten način. To je razvidno tudi pri analizi predlaganega modela, kjer se učna hierarhija z majhnim številom kompozicij izkaže kot robustna. Robustnost se izkazuje predvsem pri učinkovitem modeliranju glasbenih dogodkov v zvočnih signalih, ki so slabše, nestudijske, kvalitete in vsebujejo šum in hrup.

Kompozicionalni hierarhični model za časovno-frekvenčne predstavitve glasbe V tem poglavju smo predstavili dve opravili – avtomatsko ocenjevanje akordov in ocenjevanje osnovnih frekvenc v signalu.

Avtomatsko ocenjevanje akordov Sosledja akordov in melodije so temelj zahodne glasbe. Pogosto nosita opisa dovolj informacije za imenovanje pesmi, ne glede na pomanjkanje ostalih značilk. Avtomatsko ocenjevanje akordov lahko uporabimo za transkripcijo [76–79], klasifikacijo glasbe [80] in druga opravila. Ocenjevanje akordov je uporabno tudi za agregacijo informacij ali izbor metapodatkov za visokonivojsko ocenjevanje sosledij akordov [16, 81] in analizo vzorcev [3, 82].

Pogosto uporabljene značilke za avtomatsko ocenjevanje akordov so kromatski vektorji [76, 83] ali profili razredov tonskih višin (angl. *pitch class profile – PCP*) [84]. To so srednjenivojske predstavitve avdio signala, sestavljene iz 12 dimenzij, ki predstavljajo oktavno-invariantne tonske višine oziroma poltone znotraj ene oktave. Vsaka komponenta kromatskega vektorja se izračuna iz frekvenčnega spektra, preslikanega v eno oktavo. Ker kromatski vektorji vsebujejo delno informacijo o tonski višini, jih lahko uporabimo za klasifikacijo harmonij s pomočjo standardnih algoritmov strojnega učenja, npr. z metodo podpornih vektorjev. Vendar takšna klasifikacija ne povzema informacije o časovnem sosledju vektorjev, saj so slednji opazovani časovno neodvisno. Za časovno odvisno procesiranje se pogosto uporabljajo skriti Markovski modeli (angl. *hidden Markov model – HMM*), ki modelirajo kromatske vektorje kot izhode modela in ocenjene akorde kot skrita stanja.

Za opravilo avtomatskega ocenjevanja akordov smo zgradili tronivojski model. Model smo naučili na 88 klavirskih tonih in ga aplicirali na glasbeno zbirko skupine The Beatles.

Ocenjevanje osnovnih frekvenc v signalu V literaturi lahko zasledimo številne pristope k ocenjevanju osnovnih frekvenc in transkripciji glasbe [91–93], začenši v zgodnjih sedemdesetih letih. Nekateri pristopi uporabljajo spekter signala za ocenjevanje transkripcijskih hipotez [94, 95], drugi [96–98] se lotevajo problema transkripcije z opazovanjem signala kot kompozicije virov, kar je delno podobno predlaganemu pristopu. Pristopi so lahko usmerjeni na transkripcijo specifičnih inštrumentov [185] ali na simbolično transkripcijo, značilno za posamezen inštrument, kot je transkripcija prstnih redov za kitaro [103]. Z ozirom na človeško zaznavo so raziskovalci predlagali več pristopov [12, 186], ki poskušajo modelirati procesiranje človeškega slušnega sistema.

Za to opravilo smo uporabili transparentno strukturo modela. Model smo naučili na 88 klavirskih tipkah in ga aplicirali na več različnih zbirk podatkov. Poleg javno dostopne zbirke MAPS, ki se pogosto uporablja za evalvacijo pristopov pri ocenjevanju osnovnih frekvenc, smo predstavili svojo zbirko slovenske ljudske glasbe. Zbirka vsebuje 38 ljudskih pesmi, ki jih večglasno poje več amaterskih pevcev. Zbirka je posneta v vsakdanjih prostorih z osnovno produkcijsko opremo. Na tej zbirki smo evalvirali tudi druge pristope in pokazali, da se predlagani kompozicionalni hierarhični model zaradi svoje robustnosti odreže bolje. Prav tako smo analizirali hitrost delovanja in ugotovili, da je predlagani model hitrejši od drugih pristopov in je zato primeren za aplikacije v vgrajenih sistemih, mobilnih napravah in drugih podobnih, računsko manj zmožnih napravah.

Kompozicionalni hierarhični model za simbolne predstavitve glasbe

Odkrivanje ponavljajočih vzorcev je znan problem na različnih področjih, med drugim na področjih računalniškega vida (npr. [110]), bioinformatike (npr. [111]) in pridobivanja informacij iz glasbe. Čeprav problem izgleda enostaven in podoben na vseh naštetih področjih, se tako njegova definicija kot pristopi za odkrivanje vzorcev med področji bistveno razlikujejo. O pomembnosti ponavljanja v glasbi so razpravljali številni teoretiki glasbe in nedavno tudi raziskovalci, ki so razvili algoritme za polavtomatsko analizo glasbe, kot na primer Marsden [51]. V okviru skupnosti MIREX se je v zadnjem desetletju izoblikovalo več formaliziranih opravil, ki se posredno ali neposredno ukvarjajo z vzorci in strukturami v glasbi. Nekatera od teh opravil so strukturna segmentacija glasbe, simbolna melodična podobnost v glasbi in odkrivanje vzorcev.

Predlagani model smo dodatno nadgradili in prilagodili za delo s simbolnimi glasbenimi predstavitvami z namenom razširitve nabora opravil na področju pridobivanja informacij iz glasbe. To razširitev modela smo poimenovali SymCHM. Ker model vsebuje transparentno hierarhično strukturo, smo model aplicirali na problem odkrivanja vzorcev v simbolnih glasbenih predstavitvah. Zaradi transparentnosti strukture je model moč uporabiti za opravila odkrivanja, kar je izredno težko doseči pri drugih strukturah, ki temeljijo na nevronskih mrežah.

Model smo ocenili v okviru opravila MIREX in primerjali z drugimi pristopi. Za model smo predlagali dodatno izboljšavo SymCHMMerge, ki omogoča bolj prečiščene izhodne podatke modela in dodatno pripomore k boljšim rezultatom pri opravilu odkrivanja vzorcev.

Kompozicionalni hierarhični model za modeliranje ritma

Glavni aspekti glasbe so ritem, melodija in harmonija. Ritem je neposredno povezan s tempom; še več, ritem lahko vpliva in spremeni samo dojemanje tempa, ne da bi se slednji ob tem spremenil. Ritmični vzorci pomembno vplivajo na melodične in harmonične vidike glasbenega dela. S spremembo osnovnih ritmičnih vzorcev lahko dve različici iste pesmi pripadata različnim zvrstem in implicirata popolnoma različne plesne stile. Percepcija ritma je izredno zapleten koncept. Ritmične strukture predstavljajo osnovo za dojemanje strukture pesmi in služijo kot podlaga za segmentacijo in ponavljanje vzorcev. Tako kot pri harmoničnem in melodičnem zaznavanju, glasbeno znanje poslušalca bistveno pripomore k njihovemu zaznavanju in razumevanju glasbe.

V okviru skupnosti MIREX se je razvilo več opravil, povezanih z ritmom. Primeri takšnih opravil so razvrščanje v žanre, ocenjevanje tempa, sledenje osnovnih dob in ocenjevanje prvih dob v glasbi. Opravilo razvrščanja avdio žanrov je tesno povezano z ritmom, saj ritmični vzorci predstavljajo eno od ključnih značilnosti razlikovanja glasbenih zvrsti. Na primer, že leta 2004 se je Dixon et al.[145] soočil s problemom klasifikacije žanrske plesne glasbe z identificiranjem različnih vzorcev, ki opredeljujejo vsak glasbeni žanr. Avtorji so ocenili svoj pristop na podatkovni zbirki plesne glasbe Ballroom, ki zaobsega osem glasbenih zvrsti: jive, cha cha, quickstep, rumba, samba, tango, dunajski valček in angleški valček. Pokazali so, da so ritmični vzorci koristni za klasifikacijo žanrov. Naloga ocenjevanja tempa kot ena od prvih nalog v okviru skupnosti MIREX je tudi tesno povezana z ritmičnim vidikom glasbe. V zadnjih letih se je za ocenjevanje tempa pričelo uporabljati globoko učenje. Na primer, Böck et al.[146] so predlagali pristop, ki temelji na rekurenčnih nevronskih mrežah. Kot razširitev opravila ocenjevanja tempa je cilj sledenja osnovnim dobam identificirati osnovno ritmično strukturo v glasbi. Čeprav je opravilo dojemljivo tudi glasbeno neizobraženim in deluje relativno enostavno, trenutne F-mere najboljših pristopov na različnih podatkovnih zbirkah še vedno dosegajo le 0,6 (MIREX 2017). Zato še vedno ostaja veliko prostora za izboljšanje [147]. V zadnjem času se je na podlagi opravila sledenja osnovnih dob izoblikovalo opravilo ocenjevanja prvih dob (angl. *downbeat estimation*). Naloga slednjega je identifikacija prvih dob v ritmu. Da bi zmanjšali prevlado tričetrtinskih in štiričetrtinskih taktovskih načinov, ki so prevladujoči v zahodni glasbi, se za ocenjevanje delovanja algoritmov za to nalogo uporabljajo v zadnjem času številnejše podatkovne zbirke, ki vključujejo turško, kretsko in indijsko tradicionalno in ljudsko glasbo. Zaradi močne medsebojne povezanosti taktovskega načina, osnovnih dob in tempa vse več pristopov poskuša modelirati več kot en vidik ritma. Na primer, Krebs et al.[148] so predlagali sistem skritega Markovega modela, ki so ga uporabili za ocenjevanje osnovnih in prvih dob. Kot mnogi drugi so tudi ti raziskovalci ocenili delovanje svojega pristopa na podatkovni zbirki Ballroom.

Predlagani model smo dodatno razširili in aplicirali na simbolne ritmične predstavitve glasbe. Prvi rezultati so bili vzpodbudni in nakazujejo na zmožnost modeliranja ritma s predlaganim modelom. V model nismo vgradili nobenih specifičnih predispozicij za zahodno glasbo. Z modelom smo tudi izkazali robustnost pri delu na glasbi v živo in pokazali zmožnost razlikovanja med različnimi taktovskimi načini.

Zaključek V pričujoči doktorski disertaciji so tako podani naslednji izvirni prispevki k znanosti:

- *Kratkočasovni biološko navdahnjen kompozicionalni hierarhični model za pridobivanje informacij iz glasbe.* Razvit je bil kompozicionalni hierarhični model za procesiranje glasbenih signalov. Predlagani model predstavlja alternativo obstoječim pristopom, ki temeljijo na globokih arhitekturah, saj omogoča transparenten vpogled v procesiranje na vseh nivojih hierarhije. Model vsebuje več mehanizmov po zgledu človeškega slušnega sistema, kar se odraža v njegovi večji robustnosti. Model je bil ovrednoten na opravilih pridobivanja informacij iz glasbe na standardnih anotiranih podatkovnih bazah.

- *Razširitev modela za časovno odvisno procesiranje.* Razvit je bil kratkočasovni mehanizem samodejnega uravnavanja aktivacij kot osnova za dogodkovno procesiranje.

- *Razširitev modela za diskriminativna opravila.* Za posamezna opravila je bil vpeljan diskriminativni nivo, ki omogoča uporabo generativnega modela za diskriminativna opravila.

BIBLIOGRAPHY

[1] Downie JS, Ehmann AF, Bay M, Jones MC (2010) The Music Information Retrieval Evaluation eXchange: Some Observations and Insights in *Advances in Music Information Retrieval*, eds. A.A. W, Z.W. R. (Springer-Verlag, Berlin), pp. 93–115.

[2] Gelfand SA (2004) *Hearing: An introduction to psychological and physiological acoustics.* (CRC Press), p. 312.

[3] Milne AJ (2010) Tonal music theory: A psychoacoustic explanation? in *Proceedings of International Conference of Music Perception and Cognition.* (Seattle).

[4] Tirovolas AK, Levitin DJ (2011) music perception and cognition research from 1983 to 2010: a categorical and bibliometric analysis of empirical articles in Music Perception. *Music Perception: An Interdisciplinary Journal* 29(1):23–36.

[5] Amitay S, Irwin A, Moore DR (2006) Discrimination learning induced by training with identical stimuli. *Nature Neuroscience* 9(11):1446–1448.

[6] Peretz I, Coltheart M (2003) Modularity of music processing. *Nature Neuroscience* 6(7):688–691.

[7] Werner L, Fay RR, Popper AN, eds. (2012) *Human Auditory Development*, Springer Handbook of Auditory Research. (Springer New York, New York, NY) Vol. 42, p. 284.

[8] Lerdahl F, Jackendoff R (1983) *A generative theory of tonal music.* (Cambridge: MIT Press).

[9] McDermott JH, Oxenham AJ (2008) Music perception, pitch and the auditory system. *Current Opinion in Neurobiology* 1(18):452–463.

[10] de Cheveigne A (2002) YIN, a fundamental frequency estimator for speech and music. *The Journal of Acoustical Society of America* 111(4):1917–1930.

[11] Mauch M, Dixon S (2010) Simultaneous Estimation of Chords and Musical Context From Audio. *IEEE Transactions on Audio, Speech, and Language Processing* 18(6):1280–1289.

[12] Tolonen T, Karjalainen M (2000) A computationally Efficient Multipitch Analysis Model. *IEEE Transactions on Speech and Audio Processing* 8(6):708–716.

[13] Ryynänen MP, Klapuri AP (2008) Automatic Transcription of Melody, Bass Line, and Chords in Polyphonic Music. *Computer Music Journal* doi: 10.1162/comj.2008.32.3.72.

[14] Bittner RM, Justin S, Essid S, Bello JP (2015) Melody Extraction By Contour Classification in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Malaga), pp. 500–506.

[15] Harte C, Sandler M, Abdallah S, Gomez E (2005) Symbolic representation of musical chords: A proposed syntax for text annotations in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (London).

[16] Papadopoulos H, Peeters G (2007) Large-case Study of Chord Estimation Algorithms Based on Chroma Representation and HMM. *Content-Based Multimedia Indexing* 53-60.

[17] Sigtia S, Boulanger-Lewandowski N, Dixon S (2015) Audio Chord Recognition With A Hybrid Recurrent Neural Network in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. (Malaga), pp. 127–133.

[18] Korzeniowski F, Böck S, Krebs F, Widmer G (2017) Mirex submissions for chord recognition and key estimation 2017. *MIREX evaluation results*.

[19] Holzapfel A, Davies MEP, Zapata JR, Oliveira JL, Gouyon F (2012) Selective Sampling for Beat Tracking Evaluation. *IEEE Transactions on Audio, Speech, and Language Processing* doi: 10.1109/TASL.2012.2205244.

[20] Durand S, Bello JP, David B, Richard G (2015) No Title in *Acoustics, Speech and Signal Processing (ICASSP)*. pp. 409–413.

[21] Laurier C et al. (2009) Indexing music by mood: design and integration of an automatic content-based annotator. *Multimedia Tools and Applications* doi: 10.1007/s11042-009-0360-2.

[22] Pesek M, Strle G, Kavčič A, Marolt M (2017) The Moodo dataset: Integrating user context with emotional and color perception of music for affective music information retrieval. *Journal of New Music Research* doi: 10.1080/09298215.2017.1333518.

[23] Tkalčič M et al. (2017) A Research Tool for User Preferences Elicitation with Facial Expressions in *Proceedings of the Eleventh ACM Conference on Recommender Systems*. (ACM, Como, Italy), doi: 10.1145/3109859.3109978.

[24] Tzanetakis G, Cook P (2002) Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing* doi: 10.1109/TSA.2002.800560.

[25] Anglade A, Ramirez R, Dixon S (2009) Genre Classification Using Harmony Rules Induced from Automatic Chord Transcriptions in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. (Kobe), pp. 669–764.

[26] Lee J, Park J, Kim K, Nam J (2018) SampleCNN: End-to-End Deep Convolutional Neural Networks Using Very Small Filters for Music Classification. *Applied Sciences* doi: 10.3390/app8010150.

[27] Conklin D (2010) Discovery of distinctive patterns in music. *Intelligent Data Analysis* 14(5):547–554.

[28] Meredith D, Lemstrom K, Wiggins GA (2002) Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. *Journal of New Music Research* doi: 10.1076/jnmr.31.4.321.14162.

[29] Ren IY, Koops HV, Volk A, Swierstra W (2017) In Search of the Consensus Among Musical Pattern Discovery Algorithms in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. (Suzhou, China), pp. 671–678.

[30] Humphrey EJ, Bello JP, LeCun Y (2012) Moving beyond feature design: deep architectures and automatic feature learning in music informatics in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. (Porto).

[31] Rigaud F, Radenen M (2016) Singing Voice Melody Transcription using Deep Neural Networks in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. (New York), pp. 737–743.

[32] Jeong IY, Lee K (2016) Learning Temporal Features Using a Deep Neural Network and its Application to Music Genre Classification in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. (New York), pp. 434–440.

[33] Schluter J, Bock S (2013) Musical Onset Detection with Convolutional Neural Networks in *6th International Workshop on Machine Learning and Music, held in conjunction with the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ECML/PKDD 2013*.

[34] Battenberg E, Wessel D (2012) Analyzing Drum Patterns using Conditional Deep Belief Networks in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. pp. 37–42.

[35] Deng J, Kwok YK (2016) A Hybrid Gaussian-Hmm-Deep-Learning Approach for Automatic Chord Estimation with Very Large Vocabulary in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (New York), pp. 812–818.

[36] Bengio Y, Courville A, Vincent P (2013) Representation learning: a review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* doi: 10.1109/TPAMI.2013.50.

[37] Bilal A, Jourabloo A, Ye M, Liu X, Ren L (2018) Do Convolutional Neural Networks Learn Class Hierarchy? *IEEE Transactions on Visualization and Computer Graphics* doi: 10.1109/TVCG.2017.2744683.

[38] Zeiler MD, Fergus R (2014) Visualizing and Understanding Convolutional Networks in *Computer Vision – ECCV 2014 SE - 53*, Lecture Notes in Computer Science, eds. Fleet D, Pajdla T, Schiele B, Tuytelaars T. (Springer International Publishing) Vol. 8689, pp. 818–833.

[39] Simonyan K, Vedaldi A, Zisserman A (2013) Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *preprint.*

[40] Dosovitskiy A, Brox T (2016) Inverting visual representations with convolutional networks in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* pp. 4829–4837.

[41] Seyedhosseini M, Sajjadi M, Tasdizen T (2013) Image segmentation with cascaded hierarchical models and logistic disjunctive normal networks in *Proceedings of the IEEE International Conference on Computer Vision.* (NIH Public Access), doi: 10.1109/ICCV.2013.269.

[42] Lázaro-Gredilla M, Liu Y, Phoenix DS, George D (2016) Hierarchical compositional feature learning in *arxiv.org.* pp. 1–18.

[43] Leonardis A, Fidler S (2007) Towards scalable representations of object categories: Learning a hierarchy of parts. *Computer Vision and Pattern Recognition, IEEE* pp. 1–8.

[44] Tabernik D, Leonardis A, Boben M, Skočaj D, Kristan M (2015) Adding discriminative power to a generative hierarchical compositional model using histograms of compositions.

[45] Zhu L, Yuille AL (2006) A Hierarchical Compositional System for Rapid Object Detection in *Advances in Neural Information Processing Systems 18*, eds. Weiss Y, Schölkopf B, Platt JC. (MIT Press), pp. 1633–1640.

[46] Kortylewski A, Blumer C, Vetter T (2017) Greedy Compositional Clustering for Unsupervised Learning of Hierarchical Compositional Models.

[47] Topfer D, Spehr J, Effertz J, Stiller C (2015) Efficient Road Scene Understanding for Intelligent Vehicles Using Compositional Hierarchical Models. *IEEE Transactions on Intelligent Transportation Systems* doi: 10.1109/TITS.2014.2354243.

[48] Schenker H (1980) *Harmony.* (University of Chicago Press), p. 359.

[49] Hamanaka M, Hirata K, Tojo S (2006) Implementing "A Generative Theory of Tonal Music"†. *Journal of New Music Research* doi: 10.1080/09298210701563238.

[50] Hirata K, Tojo S, Hamanaka M (2007) Techniques for Implementing the Generative Theory of Tonal Music in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Vienna).

[51] Marsden A (2010) Schenkerian Analysis by Computer: A Proof of Concept. *Journal of New Music Research* doi: 10.1080/09298215.2010.503898.

[52] Farbood M (2010) Working memory and the perception of hierarchical tonal structures in *Proceedings of International Conference of Music Perception and Cognition.* (Seattle), pp. 219–222.

[53] Sapp CS (2005) Visual hierarchical key analysis. *Computers and Intertainment* 3(4):1–19.

[54] Woolhouse M, Cross I, Horton T (2006) The perception of non-adjecent harmonic relations in *Proceedings of International Conference on Music Perception and Cognition.* (Bologna).

[55] Balaguer-Ballester E, Clark NR, Coath M, Krumbholz K, Denham SL (2009) Understanding Pitch Perception as a Hierarchical Process with Top-Down Modulation. *PLoS Computational Biology* 4(3):1–15.

[56] Clarkson MG, Martin RL, Miciek SG (1996) Infants' Perception of Pitch: Number of Harmonics. *Infant behavior and development* 19(2):191–197.

[57] Felleman DJ, Van Essen DC (1991) Distributed Hierarchical Processing in the Primate Cerebral Cortex. *Cerebral Cortex* 1(1):1–47.

[58] Conklin D, Anagnostopoulou C (2001) Representation and Discovery of Multiple Viewpoint Patterns in *Proceedings of the 2001 International Computer Music Conference.* (Cuba), pp. 479–485.

[59] Conklin D, Bergeron M (2008) Feature Set Patterns in Music. *Computer Music Journal* doi: 10.1162/comj.2008.32.1.60.

[60] Conklin D (2006) Melodic analysis with segment classes. *Machine Learning* doi: 10.1007/s10994-006-8712-x.

[61] Wiggins GA, Forth J (2015) IDyOT: A Computational Theory of Creativity as Everyday Reasoning from Learned Information in *Computational Creativity Research: Towards Creative Machines.* (Atlantis Press, Paris), pp. 127–148.

[62] Rosenblatt F (1957) The Perceptron - a perceiving and recognizing automaton, (Cornell Aeronautical Laboratory, Inc.), Technical report.

[63] Rosenblatt F (1962) *Principles of neurodynamics: perceptrons and the theory of brain mechanisms.* (Spartan Books), p. 616.

[64] Werbos P (1990) Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE* doi: 10.1109/5.58337.

[65] Rumelhart DE, Hinton GE, Williams RJ (1986) Learning internal representations by error propagation in *Parallel Distributed Processing: Foundations*, eds. McClelland JL, Rumelhart DE. (MIT Press), pp. 318–362.

[66] Hinton GE (2007) Deep Belief Nets in *Tutorial - Proceedings of the NIPS.* (Vancouver, Canada), pp. 1–100.

[67] Hinton GE, Sejnowski TJ (1983) Optimal Perceptual Inference in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR).* pp. 448–453.

[68] Bengio Y, Lamblin P, Popovici D, Larochelle H (2007) Greedy Layer-Wise Training of Deep Networks in *Advances in Neural Information Processing Systems.* (MIT Press), pp. 153–160.

[69] Szegedy C et al. (2015) Going deeper with convolutions in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* (IEEE), doi: 10.1109/CVPR.2015.7298594.

[70] He K, Zhang X, Ren S, Sun J (2016) Deep Residual Learning for Image Recognition in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* (IEEE), doi: 10.1109/CVPR.2016.90.

[71] Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural computation* 9(8):1735–80.

[72] Goodfellow I et al. (2014) Generative Adversarial Nets.

[73] Vondrick C, Pirsiavash H, Torralba A (2016) Generating Videos with Scene Dynamics in *Advances in Neural Information Processing Systems 29*, eds. Lee DD, Sugiyama M, Luxburg UV, Guyon I, Garnett R. (Curran Associates, Inc.), pp. 613–621.

[74] Radford A, Metz L, Chintala S (2016) Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *Proceedings of the ICLR* doi: 10.1051/0004-6361/201527329.

[75] Downie JS (2008) The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology* 29(4):247–255.

[76] Mauch M, Dixon S (2008) A Discrete Mixture Model for Chord Labelling in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Philadelphia), Vol. 1, pp. 45–50.

[77] Mauch M, Dixon S, Harte C (2007) Discovering Chord Idioms Through Beatles and Real Book Songs in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Vienna).

[78] Papadopoulos H, Peeters G (2011) Joint Estimation of Chords and Downbeats From an Audio Signal. *IEEE Transactions on Audio, Speech, and Language Processing* 19(1):138–152.

[79] Smith JBL, Burgoyne JA, Fujinaga I, De Roure D, Downie JS (2011) Design and Creation of a Large-scale Database of Structural Annotations in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Miami), pp. 555–560.

[80] Ni Y, McVicar M, Santos-Rodriguez R, Bie TD (2012) Using Hyper-genre Training to Explore Genre Information for Automatic Chord Estimation in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Porto), pp. 109–114.

[81] Sheh A, Ellis D (2003) Chord segmentation and recognition using em-trained HMM in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Baltimore), pp. 183–189.

[82] Scholz R, Vincent E, Bimbot F (2009) Robust modeling of musical chord sequences using probabilistic N-grams in *Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP).* (IEEE), doi: 10.1109/ICASSP.2009.4959518.

[83] Müller M, Ewert S (2011) Chroma Toolbox: MATLAB Implementations for Extracting Variants of Chroma-Based Audio Features in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Miami), pp. 288–295.

[84] Gomez E, Herrera P (2004) Estimating the Tonality of Polyphonic Audio Files: Cognitive versus Machine Learning Modelling Strategies in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Barcelona), pp. 92–95.

[85] Bello JP, Pickens J (2005) A robust mid-level representation for harmonic content in music signals in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (London), pp. 304–311.

[86] Noland K, Sandler M (2006) Key Estimation Using a Hidden Markov Model in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Victoria).

[87] Boulanger-Lewandowski N, Bengio Y, Vincent P (2013) Audio chord recognition with recurrent neural networks in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).*

[88] Korzeniowski F, Widmer G (2016) Feature Learning for Chord Recognition: the Deep Chroma Extractor in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (New York), pp. 37–43.

[89] Korzeniowski F, Widmer G (2016) A fully convolutional deep auditory model for musical chord recognition in *IEEE International Workshop on Machine Learning for Signal Processing, MLSP.* (IEEE), Vol. 2016-Novem, doi: 10.1109/MLSP.2016.7738895.

[90] Benetos E, Weyde T (2015) Multiple-F0 estimation and note tracking for Mirex 2015 using a sound state-based spectrogram factorization model in *11th Annual Music Information Retrieval eXchange (MIREX'15).* (Malaga), pp. 1–2.

[91] Gerhard D (2003) Pitch Extraction and Fundamental Frequency: History and Current Techniques, (University of Regina, Saskatchewan, Canada, Regina), Technical report.

[92] Klapuri A, Davy M, eds. (2006) *Signal Processing Methods for Music Transcription.* (Springer, New York), p. 440.

[93] Klapuri AP (2004) Automatic Music Transcription as We Know it Today. *Journal of New Music Research* doi: 10.1080/0929821042000317840.

[94] Roebel A, Rodet X (2010) Multiple Fundamental Frequency Estimation and Polyphony Inference of Polyphonic Music Signals. *IEEE Transactions on Audio, Speech, and Language Processing* doi: 10.1109/TASL.2009.2030006.

[95] Pertusa A, Iñesta JM (2012) Efficient methods for joint estimation of multiple fundamental frequencies in music signals. *EURASIP Journal on Advances in Signal Processing* doi: 10.1186/1687-6180-2012-27.

[96] Dessein A, Cont A, Lemaitre G (2010) Real-time polyphonic music transcription with non-negative matrix factorization and beta-divergence in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* pp. 489–494.

[97] Grindlay G, Ellis DPW (2011) Transcribing Multi-Instrument Polyphonic Music With Hierarchical Eigeninstruments. *IEEE Journal of Selected Topics in Signal Processing* doi: 10.1109/JSTSP.2011.2162395.

[98] Smaragdis P, Brown J (2003) Non-negative matrix factorization for polyphonic music transcription in *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No.03TH8684).* (IEEE), doi: 10.1109/ASPAA.2003.1285860.

[99] Marolt M (2004) A Connectionist Approach to Automatic Transcription of Polyphonic Piano Music. *IEEE Transactions on Multimedia* doi: 10.1109/TMM.2004.827507.

[100] Weninger F, Kirst C, Schuller B, Bungartz HJ (2013) A discriminative approach to polyphonic piano note transcription using supervised non-negative matrix factorization in *Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP).* (Vancouver), pp. 6–10.

[101] Boulanger-Lewandowski N, Bengio Y, Vincent P (2012) Discriminative Non-Negative Matrix Factorization For Multiple Pitch Estimation in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Porto, Portugal), pp. 205–210.

[102] Vincent E, Bertin N, Badeau R (2010) Adaptive Harmonic Spectral Decomposition for Multiple Pitch Estimation. *IEEE Transactions on Audio, Speech, and Language Processing* doi: 10.1109/TASL.2009.2034186.

[103] Barbancho AM, Klapuri A, Tardon LJ, Barbancho I (2012) Automatic Transcription of Guitar Chords and Fingering From Audio. *IEEE Transactions on Audio, Speech, and Language Processing* doi: 10.1109/TASL.2011.2174227.

[104] Bock S, Schedl M (2012) Polyphonic Piano Note Transcription with Recurrent Neural Networks in *Proceedings of the 37th International Conference on Acoustics, Speech and Signal Processing (ICASSP).* pp. 121–124.

[105] Nam J, Ngiam J, Lee H, Slaney M (2011) A Classification-Based Polyphonic Piano Transcription Approach Using Learned Feature Representations in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Miami), pp. 175–180.

[106] Kelz R et al. (2016) On the Potential of Simple Framewise Approaches to Piano Transcription in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (New York), pp. 475–481.

[107] Emiya V, Badeau R, David B (2010) Multipitch Estimation of Piano Sounds Using a New Probabilistic Spectral Smoothness Principle. *IEEE Transactions on Audio, Speech, and Language Processing* doi: 10.1109/TASL.2009.2038819.

[108] Bittner RM, McFee B, Salamon J, Li P, Bello JP (2017) Deep Salience Representations for F0 Estimation in Polyphonic Music in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Suzhou, China), pp. 63–70.

[109] Hawthorne C et al. (2017) Onsets and Frames: Dual-Objective Piano Transcription.

[110] Campilho A, Kamel M, eds. (2012) *Image Analysis and Recognition*, Lecture Notes in Computer Science. (Springer Berlin Heidelberg, Berlin, Heidelberg) Vol. 7324.

[111] Coward E, Drabløs F (1998) Detecting periodic patterns in biological sequences. *Bioinformatics (Oxford, England)* 14(6):498–507.

[112] Collins T (2016) Discovery of Repeated Themes & Sections - MIREX Wiki.

[113] Collins T, Thurlow J, Laney R, Willis A, Garthwaite PH (2010) A Comparative Evaluation of Algorithms for Discovering Translational Patterns in Baroque Keyboard Works

in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Utrecht), pp. 3–8.

[114] Wang Ci, Hsu J, Dubnov S (2015) Music Pattern Discovery with Variable Markov Oracle: A Unified Approach to Symbolic and Audio Representations in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Malaga), pp. 176–182.

[115] Hsu JL, Liu CC, Chen AL (2001) Discovering nontrivial repeating patterns in music data. *IEEE Transactions on Multimedia* 3(3):311–325.

[116] Knopke I, Jürgensen F (2009) A system for identifying common melodic phrases in the masses of palestrina. *Journal of New Music Research* 38(2):171–181.

[117] Cambouropoulos E, Crochemore M, Iliopoulos CS, Mohamed M, Sagot MF (2005) A Pattern Extraction Algorithm for Abstract Melodic Representations that Allow Partial Overlapping of Intervallic Categories in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (London), pp. 167–174.

[118] Chiu SC, Shan MK, Huang JL, Li HF (2009) Mining polyphonic repeating patterns from music data using bit-string based approaches in *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on.* (IEEE), pp. 1170–1173.

[119] Meek C, Birmingham WP (2003) Automatic Thematic Extractor. *Journal of Intelligent Information Systems* doi: 10.1023/A:1023549700206.

[120] Barlow H, Morgenstern S (1975) *A dictionary of musical themes.* (Crown Pub).

[121] Rolland PY (1999) Discovering Patterns in Musical Sequences. *Journal of New Music Research* doi: 10.1076/0929-8215(199912)28:04;1-O;FT334.

[122] Owens T (1974) *Charlie Parker: Techniques of Improvisation*, Charlie Parker: Techniques of Improvisation. (University of California at Los Angeles.) No. let. 1.

[123] Cambouropoulos E (2006) Musical Parallelism and Melodic Segmentation. *Music Perception: An Interdisciplinary Journal* 23(3).

[124] Meredith D (2015) Music Analysis and Point-Set Compression. *Journal of New Music Research* doi: 10.1080/09298215.2015.1045003.

[125] Meredith D (2013) COSIATEC AND SIATEC-COMPRESS: PATTERN DISCOVERY BY GEOMETRIC COMPRESSION in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* pp. 1–6.

[126] Velarde G, Meredith D (2014) Submission to MIREX Discovery of Repeated Themes and Sections in *10th Annual Music Information Retrieval eXchange (MIREX'14).* (Taipei), pp. 1–3.

[127] Velarde G, Weyde T, Meredith D (2013) An approach to melodic segmentation and classification based on filtering with the Haar-wavelet. *Journal of New Music Research* doi: 10.1080/09298215.2013.841713.

[128] Lartillot O (2014) Submission to MIREX Discovery of Repeated Themes and Sections in *10th Annual Music Information Retrieval eXchange (MIREX'14).* (Taipei), pp. 1–3.

[129] Lartillot O (2014) In-depth motivic analysis based on multiparametric closed pattern and cyclic sequence mining in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Taipei), pp. 361–366.

[130] Ren IY (2016) Closed Patterns in Folk Music and Other Genres in *6th International Workshop on Folk Music Analysis.* (Dublin), pp. 56–58.

[131] Nieto O, Farbood M (2014) MIREX 2014 Entry: Music Segmentation Techniques And Greedy Path Finder Algorithm To Discover Musical Patterns in *10th Annual Music Information Retrieval eXchange (MIREX'14).* pp. 1–2.

[132] Nieto O, Farbood MM (2014) Identifying Polyphonic Patterns From Audio Recordings Using Music Segmentation Techniques in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Taipei), pp. 411–416.

[133] Bayard SP (1950) Prolegomena to a study of the principal melodic families of british-american folk song. *The Journal of American Folklore* 63(247):1–44.

[134] Mongeau M, Sankoff D (1990) Comparison of musical sequences. *Computers and the Humanities* 24(3):161–175.

[135] Bountouridis D, Van Balen J (2014) The cover song variation dataset.

[136] Walshaw C (2017) Tune classification using multilevel recursive local alignment algorithms in *Proceedings of the International Workshop on Folk Music Analysis (FMA2017)*. (Universidad de Malaga).

[137] van Kranenburg P, Janssen B, Volk A (2016) The meertens tune collections: The annotated corpus (mtc-ann) versions 1.1 and 2.0. 1.

[138] Bountouridis D, Brown DG, Wiering F, Veltkamp RC (2017) Melodic similarity and applications using biologically-inspired techniques. *Applied Sciences* 7(12):1242.

[139] Savage PE, Atkinson QD (2015) Automatic tune family identification by musical sequence alignment in *Proceedings of the 16th ISMIR Conference*. Vol. 163.

[140] Van Kranenburg P (2010) *A Computational Approach to Content-Based Retrieval of Folk Song Melodies*. No. June, p. 172.

[141] Gotoh O (1982) An improved algorithm for matching biological sequences. *Journal of molecular biology* 162(3):705–708.

[142] Velarde G, Weyde T, Meredith D (2013) Wavelet-filtering of symbolic music representations for folk tune segmentation and classification in *Proceedings of the Third International Workshop on Folk Music Analysis (FMA2013)*. p. 56.

[143] Schaal NK, Banissy MJ, Lange K (2015) The Rhythm Span Task: Comparing Memory Capacity for Musical Rhythms in Musicians and Non-Musicians. *Journal of New Music Research* doi: 10.1080/09298215.2014.937724.

[144] de Fleurian R, Blackwell T, Ben-Tal O, Müllensiefen D (2016) Information-Theoretic Measures Predict the Human Judgment of Rhythm Complexity. *Cognitive Science* doi: 10.1111/cogs.12347.

[145] Dixon S, Gouyon F, Widmer G (2004) Towards Characterisation of Music via Rhythmic Patterns in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. pp. 509–516.

[146] Böck S, Krebs F, Widmer G (2015) Accurate Tempo Estimation Based on Recurrent Neural Networks and Resonating Comb Filters in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. (Malaga), pp. 625–631.

[147] Dixon S (2007) Evaluation of the Audio Beat Tracking System BeatRoot. *Journal of New Music Research* doi: 10.1080/09298210701653310.

[148] Krebs F, Böck S, Widmer G (2013) Rhythmic pattern modeling for beat and downbeat tracking in musical audio in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. (Curitiba), pp. 1–6.

[149] Esparza TM, Bello JP, Humphrey EJ (2015) From Genre Classification to Rhythm Similarity: Computational and Musicological Insights. *Journal of New Music Research* doi: 10.1080/09298215.2014.929706.

[150] Holzapfel A (2015) Relation Between Surface Rhythm and Rhythmic Modes in Turkish Makam Music. *Journal of New Music Research* doi: 10.1080/09298215.2014.939661.

[151] London J, Polak R, Jacoby N (2016) Rhythm histograms and musical meter: A corpus study of Malian percussion music. *Psychonomic Bulletin & Review* doi: 10.3758/s13423-016-1093-7.

[152] Panteli M, Dixon S (2016) On the Evaluation of Rhythmic and Melodic Descriptors for Music Similarity in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. (New York), pp. 468–474.

[153] Fidler S, Boben M, Leonardis A (2009) Learning Hierarchical Compositional Representations of Object Structure in *Object Categorization: Computer and Human Vision*

Perspectives. (Cambridge University Press), pp. 196–215.

[154] Engell A et al. (2016) Modulatory Effects of Attention on Lateral Inhibition in the Human Auditory Cortex. *PLOS ONE* doi: 10.1371/journal.pone.0149933.

[155] Di Russo F, Spinelli D, Morrone M (2001) Automatic gain control contrast mechanisms are modulated by attention in humans: evidence from visual evoked potentials. *Vision Research* 41(19):2435–2447.

[156] Au WWL, Benoit-Bird KJ (2003) Automatic gain control in the echolocation system of dolphins. *Nature* doi: 10.1038/nature01727.

[157] Pesek M, Leonardis A, Marolt M (2014) A compositional hierarchical model for music information retrieval in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Taipei), pp. 131–136.

[158] Pesek M, Leonardis A, Marolt M (2017) Robust Real-Time Music Transcription with a Compositional Hierarchical Model. *PLoS ONE* doi: 10.1371/journal.pone.0169411.

[159] McFee B, Bello JP (2017) Structured training for large-vocabulary chord recognition in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Suzhou, China), pp. 188–194.

[160] Boulanger-Lewandowski N, Bengio Y, Vincent P (2013) High-dimensional sequence transduction in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing.* (IEEE), doi: 10.1109/ICASSP.2013.6638244.

[161] Lardeur M, Essid S, Richard G, Haller M, Sikora T (2009) Incorporating prior knowledge on the digital media creation process into audio classifiers in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on.* doi: 10.1109/ICASSP.2009.4959918.

[162] Mauch M, Ewert S (2013) The Audio Degradation Toolbox and its Application to Robustness Evaluation in *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR 2013).* pp. 83–88.

[163] Su L, Yang YH (2015) Escaping from the Abyss of Manual Annotation: New Methodology of Building Polyphonic Datasets for Automatic Music Transcription in *International Symposium on Computer Music Multidisciplinary Research.*

[164] Benetos E, Weyde T (2015) An efficient temporally-constrained probabilistic model for multiple-instrument music transcription in *16th International Society for Music Information Retrieval Conference*, eds. Mueller M, Wiering F. (ISMIR, Malaga, Spain), pp. 701–707.

[165] Mirex (2016) Multiple Fundamental Frequency Estimation & Tracking.

[166] Sigtia S, Benetos E, Dixon S (2016) An End-to-End Neural Network for Polyphonic Piano Music Transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* doi: 10.1109/TASLP.2016.2533858.

[167] Meredith D (2004) Method of computing the pitch names of notes in MIDI-like music representations.

[168] Cook N (1987) *A guide to musical analysis.* (Oxford University Press, Oxford, UK), p. 376.

[169] Žerovnik M (2017) Magistrsko delo: Odkrivanje vzorcev v večglasni glasbi z nenadzorovanim učenjem / Discovery of patterns in polyphonic music in an unsupervised manner, (University of Ljubljana, Faculty of Computer and Information Science), Technical report.

[170] Meredith D (2013) COSIATEC and SIATECompress: Pattern Discovery by Geometric Compression in *9th Annual Music Information Retrieval eXchange (MIREX'13).*

[171] Collins T (2013) JKU Patterns Development Database.

[172] Rowe R (2005) The Cognition of Basic Musical Structures. *Music Perception* doi: 10.1525/mp.2005.23.2.189.

[173] Savage PE, Atkinson QD (2015) Automatic tune family identification by musical sequence alignment in *Proceedings of the International Conference on Music Information Retrieval (ISMIR).* (Malaga, Spain), pp. 162–168.

[174] van Kranenburg P, de Bruin M, Volk A (2017) Documenting a song culture: the Dutch Song Database as a resource for musicological research. *International Journal on Digital Libraries* doi: 10.1007/s00799-017-0228-4.

[175] Van Kranenburg P, De Bruin M, Grijp L, Wiering F (2014) Meertens Tune Collections, (Meertens Institute, Amsterdam), Technical report.

[176] Jesser B, Deutsches Volksliedarchiv. (1991) *Interaktive Melodieanalyse : Methodik und Anwendung computergestuutzter Analysenverfahren in Musikethnologie und Volksliedforschung : typologische Untersuchung der Balladensammlung des DVA.* (P. Lang), p. 308.

[177] McKay C (2004) Masters thesis (McGill University).

[178] van Kranenburg P, Volk A, Wiering F (2013) A Comparison between Global and Local Features for Computational Classification of Folk Song Melodies. *Journal of New Music Research* doi: 10.1080/09298215.2012.718790.

[179] Pesek M, Marolt M (2013) Chord estimation using compositional hierarchical model in *6th International Workshop on Machine Learning and Music, held in conjunction with the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ECML/PKDD 2013*.

[180] Pesek M, Guna J, Leonardis A, Marolt M (2013) Visualization of a deep architecture using the compositional hierarchical model in *Proceedings of the 4th International Conference World Usability Day Slovenia 2013*. pp. 56–59.

[181] Pesek M, Medvešek U, Leonardis A, Marolt M (2015) SymCHM: a compositional hierarchical model for pattern discovery in symbolic music representations in *11th Annual Music Information Retrieval eXchange (MIREX'15)*. (Malaga), pp. 1–3.

[182] Pesek M, Marolt M, Leonardis A (2016) SymCHMMerge - hypothesis refinement for pattern discovery with a compositional hierarchical model in *MML 2016 : proceedings : held in conjunction with the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, (ECML/PKDD 2016)*. (Riva del Garda), pp. 46–50.

[183] Pesek M, Leonardis A, Marolt M (2016) Pattern discovery and music similarity with compositional hierarchical model in *Proceedings of the CogMIR workshop*, eds. Vempala N, Russo F. (New York, New York, USA), p. 8.

[184] Pesek M, Leonardis A, Marolt M (2017) SymCHM—An Unsupervised Approach for Pattern Discovery in Symbolic Music with a Compositional Hierarchical Model. *Applied Sciences* doi: 10.3390/app7111135.

[185] Marolt M (2002) Ph.D. thesis (University of Ljubljana, Faculty of computer and information science).

[186] Klapuri A (2005) A perceptually motivated multiple-F0 estimation method in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2005*. (IEEE), doi: 10.1109/ASPAA.2005.1540227.