Logic-based Modelling of Musical Harmony for Automatic Characterisation and Classification

Amélie Anglade

Thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
of the
University of London

School of Electronic Engineering and Computer Science
Queen Mary, University of London

January 2014

I, Amélie Anglade, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

Signature:

Date: January 29, 2014

Details of collaboration and publications:
All collaborations and earlier publications that have influenced the work and writing of this thesis are fully detailed in Section 1.4.

Abstract

Harmony is the aspect of music concerned with the structure, progression, and relation of chords. In Western tonal music each period had different rules and practices of harmony. Similarly some composers and musicians are recognised for their characteristic harmonic patterns which differ from the chord sequences used by other musicians of the same period or genre. This thesis is concerned with the automatic induction of the harmony rules and patterns underlying a genre, a composer, or more generally a 'style'.

Many of the existing approaches for music classification or pattern extraction make use of statistical methods which present several limitations. Typically they are black boxes, can not be fed with background knowledge, do not take into account the intricate temporal dimension of the musical data, and ignore rare but informative events. To overcome these limitations we adopt first-order logic representations of chord sequences and Inductive Logic Programming techniques to infer models of style.

We introduce a fixed length representation of chord sequences similar to n-grams but based on first-order logic, and use it to characterise symbolic corpora of pop and jazz music. We extend our knowledge representation scheme using context-free definite-clause grammars, which support chord sequences of any length and allow to skip ornamental chords, and test it on genre classification problems, on both symbolic and audio data. Through these experiments we also compare various chord and harmony characteristics such as degree, root note, intervals between root notes, chord labels and assess their characterisation and classification accuracy, expressiveness, and computational cost. Moreover we extend a state- of-the-art genre classifier based on low-level audio features with such harmony-based models and prove that it can lead to statistically significant classification improvements.

We show our logic-based modelling approach can not only compete with and improve on statistical approaches but also provides expressive, transparent and musicologically meaningful models of harmony which makes it suitable for knowledge discovery purposes.

To all women engineers,
in particular to those who inspired me,
but most importantly to all those to come.

Acknowledgements

As I am writing this acknowledgments section I realise that throughout the years of my PhD research and writing-up I have met and have been influenced by a great number of people without whom my research and life would have been dramatically different.

Thanks to my supervisor, Simon Dixon, for his guidance throughout the PhD, but most importantly for remaining calm and never losing faith in me. Thanks to Rafael Ramirez for sharing with me his expertise on Inductive Logic Programming applied to music, for clearing up the questions I had on this topic and for collaborating with me on what was then turned into two publications and forms the central part of my thesis.

Thanks to the Pattern Recognition and Artificial Intelligence Group of the University of Alicante for sharing their invaluable dataset, and to the Declarative Languages and Artificial Intelligence Group of the KU Leuven for swiftly and kindly answering all my queries about ACE-ILP, often with in-depth analyses of my application specific problems.

Thanks to Emmanouil Benetos, Matthias Mauch, Mathieu Barthet, Gyorgy Fazekas, Sefki Kolozali, Robert Macrae, Thierry Bertin-Mahieux, Jason Hockman, Mohamed Sordo, Òscar Celma, Paul Lamere, Ben Fields, Brian McFee, Claudio Baccigalupo and Norman Casagrande for helping me escape the lonely experience PhD research can sometimes be with our fulfilling and close collaborations, be they experiments, publications, development of the Hotttabs app or organisation of the f(MIR) and WOMRAD workshops.

Thanks to those whose feedback greatly influenced my work: Lesley Mearns for the insightful musicological discussions, and Claudio Baccigalupo for his handsome LaTeX template and for the interesting conversations on music recommendations we had together. Thanks to Mathieu Barthet, Matthew Davies, Sheila Stark and Johanna Brewer for their indispensable guidance while I was writing-up.

Thanks to my OMRAS2 collaborators with whom I enjoyed sharing ideas and projects: Ben Fields, Yves Raimond, Kurt Jacobson, Matthias Mauch, Robert Macrae, Chris Cannam, Christopher Sutton, Michael Casey, Tim Crawford, Dan Tidhar, Thomas Wilmering, Mathieu Barthet, Gyorgy Fazekas, Xue Wen, Mark Levy, Panos Kudumakis, Samer Abdallah,

Christophe Rhodes, Mark D'Inverno, Michaela Magas, Polina Proutskova.

Thanks to Mark Sandler and Mark Plumbley for welcoming me in the Centre for Digital Music and providing all the resources for me to thrive as a researcher. Thanks also to those co-workers I have not mentioned yet who made my time at QMUL an enjoyable one: Louis Matignon, Katy Noland, Andrew Robertson, Andrew Nesbit, Enrique Perez Gonzalez, Dan Stowell, Asteris Zacharakis, Magdalena Chudy, Martin Morrell, Michael Terrell, Alice Clifford, Josh Reiss, Anssi Klapuri, Daniele Barchiesi, Chris Harte, Aris Gretsistas, Maria Jafari, Adam Stark, Tim Murray Browne, Jean-Batiste Thiebaut, Nela Brown, Steve Welburn, Steven Hargreaves, Holger Kirchhoff, Peter Foster, Luis Figueira.

I would like to express my sincere gratitude to the established Music Information Retrieval researchers that made me feel part of the community, inspired me, encouraged me, mentored me, and even though I was only a PhD student, treated me as an equal. In no particular order: Òscar Celma, Paul Lamere, Norman Casagrande, Douglas Eck, Jean-Julien Aucouturier, George Tzanetakis, Frans Wiering, Emilia Gómez, Fabien Guyon, Xavier Serra, Masataka Goto, Juan Pablo Bello, Xavier Amatriain, Fabio Vignoli, Steffen Pauws.

Thanks to my amazing past and current managers and colleagues at SoundCloud and frestyl who over the past months gave me the space to complete my writing-up with flexible working hours and assignments.

Thanks to my family and my friends for their invaluable support. Thanks especially to my mother who so many times took on her some of my personal tasks to allow me to focus on my PhD. Thanks to my father, brother, grand-parents and family in-law for constantly checking on me and for tolerating my absences as I was writing-up. Thanks to Becky for being such an inspiring and supportive colleague and friend and for her and Ben's kind hospitality. Thank you also for being, like me, an active women in Science, Engineering and Technology with whom I could talk to about gender questions and simply share girly moments with during my PhD. For the same reason I would like to thank Nela, Magdalena, Claire and Rita as well.

Thanks to all those fellow volunteers I was so lucky to collaborate with over the past years, for their inspiringly untiring dedication and for proving me that engineers can contribute to making the world a better place.

My endless gratitude goes to my friend Benjamin whose unreserved coaching over the past year made the difference between a plan and its execution. Above all, I would like to thank my husband, Benoît, for his writing guidance and unconditional emotional support.

This work was supported by the EPSRC grants EP/E017614/1 and EP/E045235/1.

Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

Music like other online media is undergoing an information explosion. Massive online music stores such as the iTunes Store[1] or Amazon MP3[2], and their counterparts, the streaming platforms, such as Spotify[3], Rdio[4] and Deezer[5], offer more than 30 million[6] pieces of music to their customers, that is to say anybody with a smart phone. Indeed these ubiquitous devices offer vast storage capacities and cloud-based apps that can cater any music request. As Paul Lamere puts it[7]:

> *"we can now have a virtually endless supply of music in our pocket. The 'bottomless iPod' will have as big an effect on how we listen to music as the original iPod had back in 2001. But with millions of songs to chose from, we will need help finding music that we want to hear [...]. We will need new tools that help us manage our listening experience."*

Retrieval, organisation, recommendation, annotation and characterisation of musical data is precisely what the Music Information Retrieval (MIR) community has been working on for at least 15 years (Byrd and Crawford, 2002). It is clear from its historical roots in practical fields such as Information Retrieval, Information Systems, Digital Resources and Digital Libraries but also from the publications presented at the first *International Symposium on Music Information Retrieval* in 2000 that MIR has been aiming to build tools to help people to navigate, explore and make sense of music collections (Downie et al., 2009). That also includes analytical tools to support for instance computational musicology, in which the user is then an expert, a musicologist.

[1] http://www.apple.com/itunes/

[2] http://www.amazon.com/MP3-Music-Download/b?node=163856011

[3] https://www.spotify.com

[4] http://www.rdio.com/

[5] http://www.deezer.com/

[6] http://thenextweb.com/media/2013/11/06/deezer-debuts-new-mac-app-discovery-features-hits-5m-subscribers-12m-monthly-active-users

[7] in *Is that a million songs in your pocket, or are you just glad to see me?* posted on September 2, 2010 in *Music Machinery*: http://musicmachinery.com/2010/09/02/is-that-a-million-songs-in-your-pocket-or-are-you-just-glad-to-see-me/

1.1 Motivation

1.1.1 Music Characterisation

If MIR field remains largely application oriented, it however seems like the end user himself has been neglected (Schedl et al., 2013). We believe this is due to MIR approaches focusing often too much on the result, the prediction as the end goal. Retrieval methods are evaluated on their precision and recall. Classification techniques are judged on their accuracy. Recommendations are gauged on their mean absolute error. They are assessed upon their predictive capacities, while neglecting to tap into the descriptive power of the models they employ. If many of them are nonetheless based on acoustical or musicological concepts, the signal-based and statistical tools generally adopted to represent and model those concepts are often obfuscating the underlying musical phenomena which then remain invisible to the end-user. One example of this that we will describe further in Chapter 2 is the Bag-of-Features, the most popular approach to genre classification. The underlying properties of timbre, pitch or rhythm it uses are reduced to low-level representations through signal-based descriptors, and lose their temporal dimension as they are processed by statistical models. Such unexpressive black-boxes result in opaque predictions. This goes against what the Expert Systems (Swartout et al., 1991) and later the Recommender Systems (Herlocker et al., 2000; Sinha and Swearingen, 2002) communities suggested, which is to provide users with explanations for the predictions. As Tintarev and Masthoff (2007) summarise it in their survey of explanations in recommender systems:

> "*among other things, good explanations [for recommendations] could [and do (as shown in the survey)] help inspire user trust and loyalty, increase satisfaction, make it quicker and easier for users to find what they want, and persuade them*".

Accordingly, we focus in this work on adding description to prediction, by concentrating on music characterisation, which is an intrinsically transparent MIR task. What we try to characterise in the music is what we call *style*, the underlying common traits of a genre, a composer, a musical period, or even a user's preference.

1.1.2 The Expressive Power of Logic-based Modelling

Automatic characterisation of music requires machine learning modelling techniques to support the discovery and extraction of patterns from data, that is to say inductive learning. Since we are pursuing transparency we need tools that enable it. This is precisely what a logic-based representation can bring. Logic rules are written in a symbolic language that has the advantage of being human readable. Automatically extracted musical patterns expressed in logical formulae can be transmitted to musicologists who can in turn analyse them. In the musical domain we could obtain rules describing the relationship between musical phenomena and structured descriptions of local musical content. When using first-order (or relational) logic those descriptors can be not only low level concepts but also high level ones involving relations between individual data points. For example temporal relations between local musical events can easily be represented.

To extend this expressive power from the representation of the data to the representation of models and even to the inductive learning process discovering them, Inductive Logic Programming (ILP) is a fitting framework. This field, that we will describe in more details in Chapter 3, is at the intersection of Machine Learning and Logic Programming (Muggleton, 1991). It "*is concerned with inductive inference. It generalizes from individual instances/observations in the presence of background knowledge, finding regularities/hypotheses about yet unseen instances*" (Džeroski and Lavrac, 2001a). Inductive inference learns relations from a training dataset to be applied on new data points. The use of prior knowledge is one of the distinctive strengths of ILP. Moreover the expression of this background information is quite natural thanks again to the use of a relational representation. It is also compact as only a fraction of it has to be explicitly expressed and the rest can be derived as part of the mining process.

This short presentation of the expressive power of logic modelling triggers our first research question:

RQ1: How can logic and in particular ILP support music characterisation? How can musical properties be represented and extracted?

1.1.3 The Characterisation Power of Harmony

Having found a tool for characterisation, we need now to find a set of features from which patterns can be extracted which capture an important aspect of music. We could focus on some characterising aspects such as rhythm, melody or timbre but this thesis will limit its

scope to another one, harmony. Harmony is a high level descriptor of music focusing on the structure, progression, and relation of chords. In Western tonal music, to which this thesis will limit its scope, harmony can be used to characterise a musical period (e.g. Baroque music) or a musical genre. Indeed, as depicted by Piston (1987), each period had different rules and practices of harmony. Some harmonic patterns forbidden in a period became common practices afterwards: for instance the tritone was considered as *diabolus in musica* until the early 18th century and became later on a key component of the tension/release mechanism of the tonal system. "Modern" musical genres are also characterised by typical chord sequences: if the pop-rock tunes mainly follow the tonic-subdominant-dominant chord sequence, jazz standards usually follow more complex chord progressions. Similarly some composers, musicians and bands are recognised for their characteristic harmonic patterns which differ from the chord sequences used by other musicians of the same period or genre.

Despite its richness, harmony is a concept that can be understood not only by experts but also amateur musicians, through simplified notations such as lead sheets or tabs containing chord charts. The availability of guitar tabs all over the internet and their success with the amateur guitarists attests a popular interest and understanding of at least basic harmonic concepts.

Hence we believe harmony has a good discriminative (can distinguish between genres) and expressive power (can be understood by experts and amateurs). This paragraph leads to our second research question:

RQ2: Is it possible to leverage harmony's descriptive and expressive power to characterise music automatically? To what extent can harmony be used by itself to characterise styles?

1.2 Research Goal

Our working hypothesis is that we can combine ILP and harmony to build characterisation models of musical styles. As mentioned above a widely spread way of thinking of harmony is as sequences of chords. There is obviously more to harmony than sequences of chords, and we will describe more harmony concepts such as tonality in Chapter 2, but at its core harmony describes chords and temporal patterns between them. Sequences imply temporal

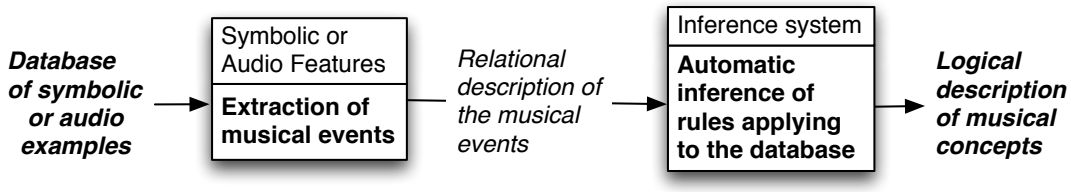

Figure 1.1: Logic-based framework for musical rule induction.

relations, which can be represented in ILP. This brings us to the next research question:

RQ3: Which representation schemes are suited to represent harmony and in particular chord sequences in ILP?

The idea in this work is to create high-level music descriptors based on logical expressions. High-level harmony descriptors are closer to the description of music given by musicologists than the low-level signal-based descriptors commonly used in MIR which are unintelligible and can not be presented to non-technical users. By use of ILP on a learning database representing a particular style we derive logical rules describing this style. Therefore the goal of this thesis is to build a harmony- and logic-based reasoning framework for music characterisation.

An illustration of this framework is given in Figure 1.1. It takes as input a database of examples to characterise. Notice that these examples can either be audio signals or symbolic examples. They are then analysed by a musical event extractor using either symbolic or audio features. Finally the relational description of the examples resulting from this analysis is given to an inference system which derives musical rules that cover the examples. To implement this framework we need to answer the question:

RQ4: Which induction algorithms can best extract logic rules from Harmony?

1.3 Contributions

This thesis describes the development, tuning and testing of such a harmony- and logic-based framework for musical style characterisation. The main contributions can be found in Chapters

4, 5 and 6. To build our framework we answer our research questions with an explorative approach. Throughout the thesis, we experiment in five main directions:

- representation of chord sequences
- ILP induction method
- granularity of the musical styles to characterise (looking at different levels of a genre taxonomy)
- different datasets representing these styles
- symbolic and audio domains

The first representation scheme for chord sequences is presented in Chapter 4. It consists of using fixed-length blocks of chord progressions of length four. We introduce in that chapter the idea of expressing various chord properties to describe the chord progressions: root note, bass note, chord category, root interval, bass interval and degree. If in Chapter 5 we use the same chord properties, we however exchange the chord progression representation for a more flexible one. We relax the constraints to allow for chord sequences of any length and add the concept of gap to skip non-characterising chords. Inspired by biological studies, this is implemented with a context-free definite-clause grammar and a difference-list representation.

For each induction method we exploit an existing ILP piece of software. In Chapter 4 we base the research on Aleph, an inverse entailment ILP tool. Our experiments show our implementation scales to datasets of musicologically meaningful sizes while keeping computation short and lightweight. However Aleph can not infer rules from flexible length chord progressions. As a consequence, in Chapter 5, we move to TILDE, an ILP decision tree induction software. We consider both single tree models and random forests.

We explore characterisation of an artist, in this case actually a band, The Beatles, thanks to Harte's dataset (2010), containing transcriptions of the 180 songs featured on all their 12 studio albums (Section 4.2). We also differentiate genres with several datasets:

- transcriptions of 244 Jazz standards from the Real Book (various, 2004), representing Jazz – presented in Section 4.2
- the *Perez-9-genres* dataset consisting of 856 pieces of music representing three main genres (popular, jazz and academic music), that are further separated into 9 subgenres (pop, blues and Celtic; pre-bop, bop and bossanova; Baroque, Classical and Romantic periods) (Pérez-Sancho, 2009) – presented in Section 5.2.3

- the GTZAN database which contains 1000 audio recordings equally distributed across 10 music genres (Tzanetakis and Cook, 2002) – presented in Section 6.3
- the dataset created for the ISMIR 2004 Genre Classification Contest which covers seven genre classes with varying numbers of audio recordings per classe (ISMIR, 2004), from which we use 447 pieces – presented in Section 6.3.

Additionally we develop techniques to cover both symbolic and audio data types. Symbolic data in our case comes in several formats: the Real Book and the *Perez-9-genres* datasets are encoded in the Band in a Box file format, while the Beatles dataset is in a Resource Description Framework (RDF) format. They are both described in more details in Section 2.2.2. To extend our framework from symbolic to audio data, which are more readily available and are the main target of industrial applications, we use a chord transcription algorithm from Mauch (2010). We investigate in Chapter 5 the use on audio data of ILP models trained on symbolic files, but obtain better results with models trained on audio files. The audio datasets we run experiments on are: *Perez-9-genres*, GTZAN and ISMIR 2004 Genre Classification Contest.

From the experiments, the following research question arises:

RQ5: How can characterisation accuracy be evaluated?

Our first attempt at answering it is to examine the transparent models resulting from the experiments where we confirm that the logical rules in them agree with generally accepted musicological findings. This work can be read in Section 4.3 and 5.3.5. If Computational Musicology is not the focus of this thesis, this analysis shows automatic rule generation assisting musicology could still be an application of our research. We contribute to the field not only the methods, but also the sets of transparent and human readable rules generated from our experiments that characterise various styles.

In order to get a quantifiable measure of accuracy we pursue in Chapter 5 the neighbouring task of genre classification, focusing on extracting transparent models of classification to still allow for characterisation. This task of genre classification has the advantage of having established measures of success, due to its clear binary classifying decision (matching predicted labels with ground truth labels). Moreover as described in more details in Section 2.3.2 the genre classification task has received a lot of interest from the MIR community, which allows us to test our classification methods on multiple existing and well studied datasets. In Chapter 5 we compare our transparent approach to classification with some of the many statistical

methods applied on the same datasets. We show that our level of accuracy is equivalent to those other methods.

Finally in Chapter 6, in order to improve on a state of the art genre classifier we pair our harmony models with more traditional low level signal-based descriptors, resulting in a meta-classifier which obtains statistically significantly better results than the same framework without our contributed models.

In a nutshell, this thesis work brings the following novel points to the state-of-the-art:

- **A variable- and arbitrary-length representation of chord sequences** which enriches the fixed-length approaches (n-grams or similar) employed up to now for their representation. The original use of context-free definite-clause grammars is the key ingredient for such a flexible representation.
- **A modelling of gaps of unspecified length in between harmonic sequences of interest** to skip ornamental and passing chords, which we were the first to introduce. We capture gaps of flexible length thanks to the recursive power of Inductive Logic Programming.
- **The proof of the appropriateness and usefulness of the combination of degree and chord category in automatic characterisation and classification of style**. We show that despite the complexity of such models including both a function relative to the tonality (degree) and the chords internal structure (category), they obtain statistically significantly better results when used for genre classification. This composite representation also result in less complex models requiring smaller computation times. Finally we also show that it is also more musicologically meaningful than using each component independently.
- **The incorporation of harmony models into state-of-the-art signal-based genre classifiers**. Where current genre-classification approaches suffer from a semantic gap due to acoustical low-level features, ours includes a temporal modelling of harmony to retain musical meaning. This results in a statistically significant increase in genre classification accuracy of up to 2.5%.
- **Automatically generated datasets of harmony rules characterising several genres and styles for further musicological studies**. Our experiments produce sets of rules for all characterisation and classification problems we study. These are available upon request to musicologists and other researchers, either in a human-readable format or as logic programs.

1.4 Related Publications by the Author

We list below those of our publications that have influenced the work and writing of this thesis. When they were the results of collaborations, the contributions of the co-authors are also specified accordingly.

Simon Dixon was the author's supervisor and provided guidance, support and feedback for the entire duration of her thesis. Rafael Ramirez acted as her local supervisor for the 2-month research exchange she spent at the Music Technology Group (Universitat Pompeu Fabra, Barcelona) in January-March 2009 and for the subsequent publications.

Peer-Reviewed Conference Papers

(Anglade and Dixon, 2008a) – Amélie Anglade and Simon Dixon. **Characterisation of Harmony with Inductive Logic Programming**. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, pages 63–68, Philadelphia, PA, USA, 2008. The poster presenting this publication at the ISMIR 2008 conference also received the Best Poster Award at the London Hopper Colloquium for women in computing research, British Computer Society Headquarters, London, 2009.

(Anglade and Dixon, 2008b) – Amélie Anglade and Simon Dixon. **Towards Logic-based Representations of Musical Harmony for classification, Retrieval and Knowledge Discovery**. In *Proceedings of the 1st International Workshop on Machine Learning and Music 2008 (MML), co-located with the 25th International Conference on Machine Learning (ICML), the 24th Conference on Uncertainty in Artificial Intelligence (UAI) and the 21st Annual Conference on Learning Theory (COLT)*, pages 11–12, Helsinki, Finland, 2008.

(Anglade et al., 2009b) – Amélie Anglade, Rafael Ramirez and Simon Dixon. **First-order Logic Classification Models of Musical Genres Based on Harmony**. In *Proceedings of the 6th Sound and Music Computing Conference (SMC)*, pages 309–314, Porto, Portugal, 2009.

(Anglade et al., 2009c) – Amélie Anglade, Rafael Ramirez and Simon Dixon. **Genre**

classification using harmony rules induced from automatic chord transcriptions. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, pages 669–674, Kobe, Japan, 2009.

(Barthet et al., 2011) – Mathieu Barthet, Amélie Anglade, Gyorgy Fazekas, Sefki Kolozali and Robert Macrae. **Music Recommendation for Music Learning: Hotttabs, a Multimedia Guitar Tutor**. In *Proceedings of the 2nd Workshop on Music Recommendation and Discovery (WOMRAD), co-located with the 5th ACM Recommender Systems Conference (ACM RecSys)*, pages 7–13, Chicago, IL, USA, 2011.
This publication describes a web application developed at the London Music Hack Day 2009 and Barcelona Music Hack Day 2009. All co-authors have built or provided pieces (algorithms or infrastructure) of the application and worked on their integration together. The author's work lies mainly in the design, implementation and refinement of the guitar tab clustering component, as well as the conceptualisation of the original idea for the web application and contribution to the definition and refinement of its functionalities.

Journal Article

(Anglade et al., 2010) – Amélie Anglade, Emmanouil Benetos, Matthias Mauch and Simon Dixon. **Improving Music Genre Classification Using Automatically Induced Harmony Rules**. *Journal of New Music Research*, 39(4):349–361, 2010.
Emmanouil Benetos and the author equally collaborated on this work. Emmanouil Benetos provided his prior implementations of the signal-based features and statistical classification algorithms, while the author contributed her harmony-based ILP classification models. The integration of those components as well as the experimental design, execution and interpretation were the fruit of their collaboration. Matthias Mauch contributed the chord transcription algorithm and ran the transcription tasks.

Other Publications

(Anglade et al., 2009a) – Amélie Anglade, Rafael Ramirez and Simon Dixon. **Computing genre statistics of chord sequences with a flexible query system**. *Demo at the SIGMUS Symposium*, 2009[8].

(Dixon et al., 2011) – Simon Dixon, Matthias Mauch and Amélie Anglade. **Probabilistic**

[8] http://www.sigmus.jp/SIG/sig200911listofdemos-e.html

and Logic-Based Modelling of Harmony. In Ystad, S., Aramaki, M., Kronland-Martinet, R., and Jensen, K., editors, *Exploring Music Contents, 7th International Symposium, CMMR 2010*, volume 6684 of *Lecture Notes in Computer Science*, pages 1–19. Springer Berlin Heidelberg, 2011.
This publication provides a summary and comparison of Matthias Mauch's and the author's thesis research.

1.5 Thesis Outline

Chapter 1: Introduction
In this chapter we describe the context and motivation for this work. The research goal and research questions are presented and the thesis contributions are discussed.

Chapter 2: Background and Related Work in Music Information Retrieval
In this chapter we define music theory and in particular harmony-related concepts that will be used in the remainder of the thesis. We also review prior work in the field of Music Information Retrieval focusing on music characterisation and classification.

Chapter 3: Background and Related Work in Inductive Logic Programming
This chapter covers the theory, background as well as related work in Inductive Logic Programming. A review of related logic-based approaches to musical tasks concludes the chapter.

Chapter 4: Automatic Characterisation of the Harmony of Song Sets
Based on work also published (Anglade and Dixon, 2008a) and (Anglade and Dixon, 2008b).
In this chapter we present a style characterisation approach using chord sequences of fixed-length as representation scheme, and Aleph as ILP induction algorithm. We study and characterise a band, the Beatles represented by their entire studio albums, and a genre, Jazz, represented by a set of pieces from the Real Book. A qualitative analysis of the extracted rules, as well as a comparison with a statistical study of the same corpora is provided. We conclude with an analysis and description of the constraints and limitations of this approach.

Chapter 5: Automatic Genre Classification

Extends work published in (Anglade et al., 2009b) and (Anglade et al., 2009c).

In this chapter we move to the task of genre classification, and use the TILDE algorithm which is designed for building decision tree models. We introduce a new Context-Free Definite-Clause Grammar scheme to represent harmony sequences of any length including gaps. We report on the results of classification experiments on 9 genres, comparing the discriminative power of several chord properties but also of single decision trees vs. random forests. We also perform a comparison of our method with another study on the same dataset using a statistical approach. We conclude with a musicological analysis of some of the extracted rules.

Chapter 6: Improving on State-of-the-art Genre Classification

Based on work also published in (Anglade et al., 2010).

In this chapter we present a new meta-classification framework extending a state-of-the-art statistical genre classifier based on timbral features with our harmony models. The latter are the first-order random forests built in the previous chapter using the best representation scheme and trained on audio data. Tests on two new genre datasets (not used in our previous experiments) indicate that the proposed harmony-based rules combined with the timbral descriptor-based genre classification system lead to significantly improved genre classification rates.

Chapter 7: Conclusions

This chapter provides a summary of our findings and discusses future research topics and applications of this work.

CHAPTER 2

BACKGROUND AND RELATED WORK IN MUSIC INFORMATION RETRIEVAL

2.1

Introduction

In this chapter we review background music theory concepts as well as related work in the field of Music Information Retrieval. We begin by providing music theory definitions which are essential to the understanding of our work. We particularly have a close look at harmony-related concepts, terms and notation conventions which we will later use. We go on to summarise and discuss MIR research to date covering the two tasks we are interested in, namely automatic characterisation and music genre classification. We provide a description of those tasks and how they relate to each other, a review of state-of-the-art approaches to tackle them as well as a discussion of their limitations and recent endeavours to overcome them.

2.2

Musical Concept Definitions

The work done in this thesis is based on music theory concepts that we define here. We will limit those definitions to the scope of this work, which as seen in Chapter 1 does not have musicological goals but instead borrows useful concepts from musicology for music information retrieval purposes – some of which might be of interest to musicologists too. Particularly, if our work spans three genres – classical, jazz and popular music – which have their own independent bodies of musicological research, it is outside the scope of this work to explain in detail how their views of the following concepts differ and instead we will focus on their common traits.

The *pitch* is the perceptual interpretation of the frequency of a sound wave. It consists of a pitch-class, notated with the 7 pitch labels of the diatonic scale (A, B, C, D, E, F, G) which can all be altered with one or several sharps (♯) or flats (♭), and the octave in which the pitch is found. In this work we will often omit the octave and treat pitch and pitch-class as synonyms. Similarly for a *musical note*, which consists of a pitch and a duration, we will often omit the duration and use it to refer to its pitch-class. If in music *tone* is usually the audio instantiation of a note, here we will only employ that term to refer to the intervals called *semitone* and *(whole)*

tone. A semitone is the distance between adjacent notes in the chromatic scale, as on the piano keyboard, or also the distance between a note and the same note altered by one sharp (raising by a semitone) or one flat (lowering by a semitone). A tone equals two semitones.

2.2.1 Harmony-Related Terms

The main musical concept addressed throughout this work is *harmony*. In music, harmony is concerned with the study of chords and of their structures, progressions and relations. In Western Music, harmony is one of the fundamental elements of music, clearly as important as melody and rhythm to which it is linked, the three supporting each other. We use the most generally accepted definition for a *chord*: the simultaneous combination of 3 or more notes. We will exceptionally extend this definition to some 2-note chords when it is clear from their context that the third note is implied. Another important building block of harmony is the *interval*, "the distance between two notes" as Piston (1987) defines it. When the two tones are sounded simultaneously, he calls it a *harmonic interval*, which he uses to provide another definition of the chord: "the combination of two or more harmonic intervals". A (non-exhaustive) list of the most important harmonic intervals is provided in Table 2.1.

Table 2.1: List of the most important harmonic intervals. 4 and 5 in the pitch names symbolise the (fourth and fifth) octave, where the fourth octave extends from middle C up to, but not including, the next C.

Name	diatonic steps	semitones	Examples
Perfect unison	0	0	C4-C4
Minor second	1	1	C-D♭, B-C, E-F
Major second	1	2	C-D
Minor third	2	3	C-E♭, A-C, E-G
Major third	2	4	C-E
Diminished fourth	3	4	C-F♭, C♯-F
Perfect fourth	3	5	C-F
Augmented fourth	3	6	C-F♯, F-B
Diminished fifth	4	6	C-G♭, B-F
Perfect fifth	4	7	C-G
Augmented fifth	4	8	C-G♯, C♭-G
Minor sixth	5	8	C-A♭, A-F
Major sixth	5	9	C-A
Minor seventh	6	10	C-B♭, A-G
Major seventh	6	11	C-B
Perfect Octave	7	12	C4-C5
Minor ninth	8	13	C4-D♭5, B4-C5
Major ninth	8	14	C4-D5
Perfect eleventh	10	17	C4-F5
Major thirteenth	12	21	C4-A5

In this thesis we will focus on a limited number of chord types, or as we will refer to them,

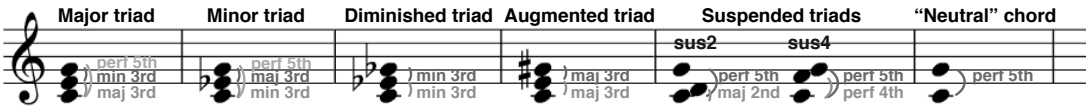

Figure 2.1: Types of triads (and neutral chord) used in this work.

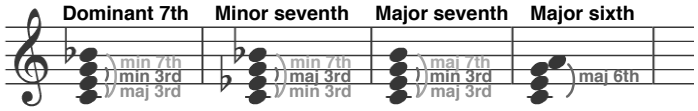

Figure 2.2: Types of seventh and major sixth chords used in this work.

chord categories. They are defined and characterised by the number of notes and specific (up to octave equivalence) intervals they contain. We will work with triads, seventh chords and major sixth chords. *Triads* are chords of three notes that are separated by 2 intervals of a third. Figure 2.1 illustrates the types of triads that will be used in the next chapters: major, minor, diminished, augmented, suspended, neutral (when the third is neither present nor suspended). A *seventh chord* is a triad with an additional note a third above the top note, which is also a seventh above the root note. The *root note* is the note used to name the chord, and the lowest note in that chord when it is played in its root position, i.e. when all notes are separated by thirds. The chord is *inverted* when its notes are reorganised. The lowest note in an inverted chord is called the *bass note*. Figure 2.2 shows the types of seventh chords that will be used in this thesis: dominant 7th, minor 7th, major 7th. Finally we also use the *major sixth chord*, also known in classical music as an *added sixth chord* and in popular music as a *sixth chord*, which is a triad with an added sixth interval from the root. It is included in our study due to its extensive use in modern popular music. It can also be found in Figure 2.2. Hence we only consider *tertian* chords, i.e. chords which are built of superimposed thirds, their inversions and chords based on tertian chords where some intervals are omitted (such as the 3rd in the "neutral" triad), replaced with other intervals (e.g. suspended triads where the 3rd is replaced with a 2nd or a 4th) or added (such as the added 6th in the major sixth chord). Moreover it is noticeable that chords with larger intervals than the 7th (such as 9th, 11th and 13th chords) have the same functions as the 7th chords they contain and extend, which is why we omit them from our vocabulary of chords for our experiments and instead focus on the underlying seventh and triad chords. We acknowledge though that it is a simplification and at the perception level these more complex chords would still sound different and can be used by composers and musicians to add harmonic colours to their music.

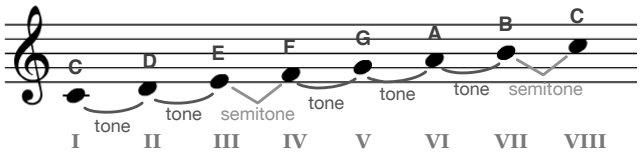

Figure 2.3: Example of the major mode: C Major scale, with pitch labels and degrees.

Another essential concept in harmony is *tonality*, the hierarchical organisation of pitches in a scale and around a tonal centre called the *tonic*. In harmony analysis the seven pitches of a *diatonic scale* are identified with roman numerals which correspond to their position in the scale, also called their *degree*, the tonic being the first degree (I). The full list of degrees and their names is provided in Table 2.2. Each chord can then be characterised by its type, its root note or degree of its root note and potentially its inversion or bass note. When degrees are used we talk about *Roman numeral analysis* which allows to identify patterns across music pieces with different tonal centres. The tonic together with the type of scale, the *mode*, are grouped into the *key*. Although some of the genres studied for this work might occasionally use different modes, we will limit the modes used to those of common practice: major and minor. This has the advantage of providing a common referential in which we can compare various genres that have all been represented or transcribed into those two modes. Studies comparing genres sharing other modes could however employ those as well or instead. Major and minor modes are characterised by the respective positions of the tones and semitones in the diatonic scale which are illustrated in Figures 2.3 and 2.4. The tuning system assumed to be underlying all pieces studied in this work is the *12-tone equal temperament*, in which the octave is divided in 12 equal semitones, or chromatic intervals, allowing for *transposition* to other keys: movement of the tonal centre and all pitches while keeping the intervals between all notes identical. When a change of tonal centre is only temporary and occurs inside a piece of music then we talk about *modulation*.

Table 2.2: Diatonic scale degrees.

Degree	Name
I	Tonic
II	Supertonic
III	Mediant
IV	Subdominant
V	Dominant
VI	Submediant
VII	Leading note

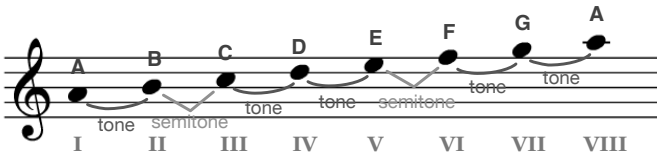

Figure 2.4: Example of the minor mode: A (natural) minor scale, with pitch labels and degrees.

Finally higher level concepts of harmony that will be mentioned in this thesis are *cadence* and *harmonic rhythm*. A harmonic cadence is a chord sequence that acts as a punctuation mark at the end of a musical phrase or a musical piece. Specific examples of cadences will be introduced in later chapters when they will be needed or identified. Harmonic rhythm refer to either the specific rhythm (original definition) or more generally the tempo at which the chord changes are happening in a piece of music.

2.2.2 Chord Notations

To study and analyse the harmony of a piece of music one looks at all the chords it contains and labels them. In classical music all the individual notes are usually provided in the score, without explicit notation of their harmonic function, and it is the task of the musician or musicologist to group them together into chords and then perform harmonic analysis to identify degrees, types and inversions of the chords, with notation conventions such as classical Roman numeral analysis. In popular music and jazz however it is more common to directly represent the chords in a shorthand fashion without specifying the individual notes, and root note is often preferred over degree. Those shorthand labels are explicit so that they can be played at sight. They juxtapose root note, chord type and inversion (preceded by a forward slash: "/"). Such jazz/pop/rock shorthand chord labels are found in lead sheets (e.g. on top of lyrics) and real or fake books for instance. The various chord syntaxes mentioned here are illustrated in Figure 2.5 (taken from (Harte et al., 2005) where they are also described in more detail).

The datasets we use for our experiments were provided by their creators in what can be considered as standard formats now in the Music Information Retrieval community. The first format is the Band in a Box file format. Band in a Box[1] is an accompaniment software for musicians who need a computer generated "band" to play along with them. The general interface displays a list of bars in which the user can type in chords (annotated in a jazz/pop/rock shorthand fashion). More parameters allow the user to define the tempo, the style, the repetitions, etc. Thus, a Band in a Box file contains a user supplied list of

[1] http://www.pgmusic.com/products_bb.htm

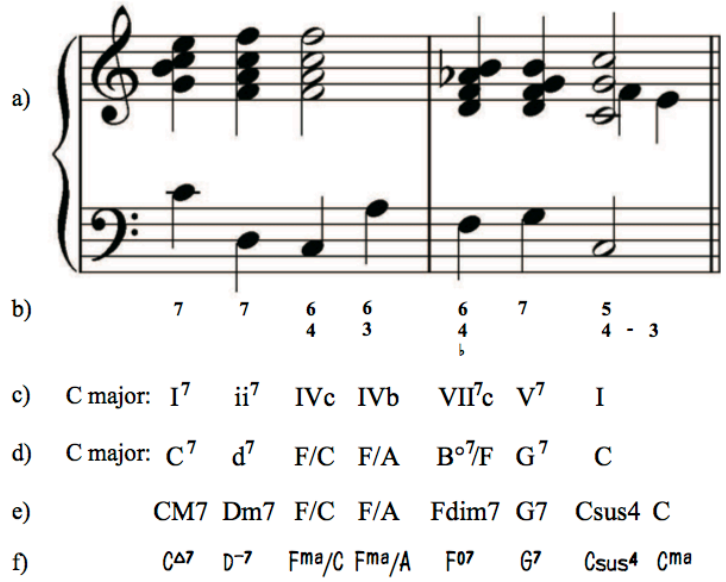

Figure 2.5: A short extract of music in C major with different harmony notations: a) Musical score b) Figured bass, c) Classical Roman numeral, d) Classical letter, e) Typical Popular music guitar style, f) Typical jazz notation. From (Harte et al., 2005).

pairs $(time, chord)$ and as such can be seen as a simplified music score. The second format is the chord notation introduced by Harte et al. (2005) which was then integrated into the Music Ontology (Raimond et al., 2007) as the Chord Ontology[2] to allow for structured RDF (Resource Description Framework) descriptions of harmonic events. In this representation each harmonic event (or chord) is associated with a start time, an end time and a web identifier from which one can retrieve an RDF description of the chord. As shown in Figure 2.6, in the Chord Ontology each RDF description of a chord contains in turn none (if the chord is unknown), one or several of the following:

- the root note of the chord,
- the bass note of the chord,
- the component intervals of the chord (*additive description*), or a base chord (i.e. maj, 7, sus4, etc.) and optionally the intervals from that base chord that are not contained in the current chord (*subtractive description*).

All datasets used in our experiments will be pre-processed with specific parsers or converters extracting from the aforementioned formats the chord types and transforming them into jazz/pop/rock shorthand representations of the chord types. The shorthand notations for the

[2] http://motools.sourceforge.net/chord/

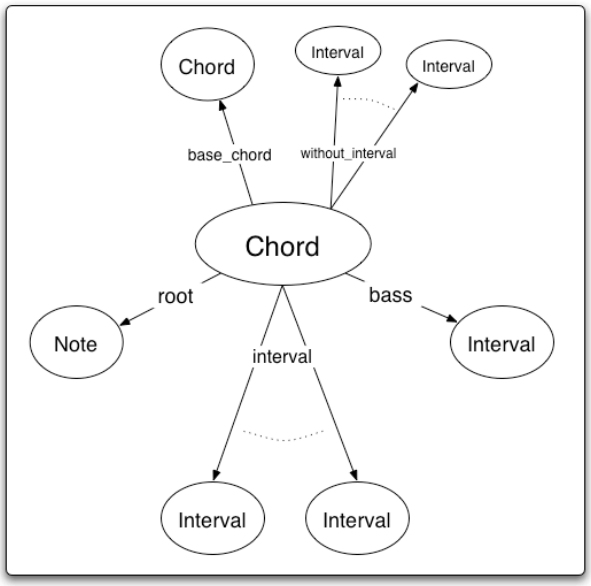

Figure 2.6: Model of a chord in the Chord Ontology. From `http://motools.sourceforge.net/chord/` licensed by Christopher Sutton, Yves Raimond and Matthias Mauch under Creative Commons Attribution 1.0 Generic.

chord categories used in this work are shown in Table 2.3. All parsers and converters will be described and introduced in later chapters, together with the datasets they will be used on.

Table 2.3: Shorthand notations of main chord categories as used in this work.

Full chord name	frequently abbreviated as	shorthand label
major triad	major chord	maj
minor triad	minor chord	min
diminished triad	diminished chord	dim
augmented triad	augmented chord	aug
suspended (second or fourth) triad	sus chord	sus
“neutral” triad	neutral chord	neut
dominant seventh	dominant chord or dominant 7th	dom or 7
major seventh	major 7th	maj7
minor seventh	minor 7th	min7
major sixth	major 6th	maj6

In this section we describe the Music Information Retrieval state-of-the-art approaches to the tasks of characterisation and classification and how they relate to each other.

2.3.1 Characterisation

The discussion of music theoretical terms in the previous section is important because as Piston (1987) describes it in the introduction of his book, *Harmony*, *"a secure grounding in [music] theory is [...] a necessity [...], since it forms the basis for any intelligent appraisal of individual styles of the past or present"*, suggesting that harmony is one of the areas of music theory that allows to identify both *common practices* and *individual styles*. The act of finding and *"describ[ing] the distinctive nature or features of"* an item or concept is what the Oxford Dictionary defines as *characterisation*. Because of its descriptive nature, in Music Information Retrieval, the task of automatic characterisation lies at the border with computational (ethno)musicology. For instance it is for its descriptive power that Taminau et al. (2009) employ the rule learning technique Descriptive Subgroup. It enables them to discover in a dataset of folk tunes both subgroups and interpretable rules describing them. The latter are of great importance for ethnomusicologists. Characterisation studies are also conducted on composers (van Kranenburg, 2006), genres (Pérez-Sancho, 2009) and on musical corpora representing or exhibiting specific styles. For instance in search of chord idioms, Mauch et al. (2007) made an inventory of chord sequences present in the Real Book (a corpus representing an entire genre, Jazz) and in the Beatles' studio albums (a corpus representing a specific band). Their approach is entirely statistical and resulted in an exhaustive list of chord sequences together with their relative frequencies.

Additionally the methods employed for characterisation often belong to the pattern recognition domain, since, as van Kranenburg (2006) describes it, it fits within Meyer's theory of musical style which states that *"style is a replication of patterning, whether in human behavior or in the artifacts produced by human behavior, that results from a series of choices made within some set of constraints"* (Meyer, 1989). McKay and Fujinaga (2007) for instance have developed an entire computer-based framework implementing state-of-the-art Pattern Recognition and Data Mining techniques. It is meant to be used by musicologists for exploratory analysis of large amount of data and considering many musical aspects (or features) at a time. Furthermore for accuracy reasons many of those pattern recognition studies are conducted on symbolic data, extracted from scores or score-like data (e.g. extracted from audio by mean of transcription techniques), such as in (Pérez-Sancho, 2009) where naïve Bayes and n-grams models are first tested on symbolic melodic and harmonic data and then extended to audio

data using polyphonic transcription and chord recognition algorithms respectively.

However if what we mean precisely by characterisation in this work is the descriptive analysis whose goal is the (human-readable) description itself, it is important to notice that characterisation techniques are often used to another end. Indeed the result of a characterisation process, the description itself can be seen and used as model of the style it represents. Cope (1996), after modelling characteristic compositional traits of various classical composers, automatically composed new musical works in their styles with impressive results. Similarly, after characterising Johann Sebastian Bach's fugue compositions and those of 9 other composers (his son and students) using 20 features mostly focusing on polyphonic characteristics, van Kranenburg (2006) use a Fisher-transformation to project the multi-dimensional representations of the fugues onto a 2 dimensional space where the compositions of each composer are expected to form a separate cluster. As seen in this work a few of Bach's disputed compositions actually cluster closer to other composers which means that this characterisation technique is a useful tool for discussions of authorship attribution. Finally, the most common application of characterisation is certainly to the task of classification.

2.3.2 Automatic Genre Classification

In Music Information Retrieval, classification consists in the automatic tasks of learning and assigning labels to pieces of music. Because genre is a characteristic of music that has been historically and widely used to organise music, and even though it is a "ill-defined" concept that even experts would not agree on (Aucouturier and Pachet, 2003), music genre classification – also sometimes called music genre recognition – has been one of the earliest and most widely investigated MIR tasks (Lee et al., 2009; Sturm, 2012). Most of the works in music genre classification to date focus on the task of assigning a single label to each piece of music – which we will also limit this work to – but some work on multi-label and multi-domain approaches to this problem have also been published (Lukashevich et al., 2009). Other MIR classification tasks include mood and emotion classification which are outside the scope of this work. Automatic tagging – cf. (Bertin-Mahieux et al., 2010) for an overview – is a larger problem that we will also not describe here.

The topic of music genre classification itself being so popular we can not possibly cover the entire music genre recognition literature but we refer the reader to surveys such as the one from Scaringella et al. (2006), as well as the thorough review work from Sturm (2012) who, even though he focuses on evaluation of music genre recognition algorithms, references all publications up to December 2012 on this topic.

Bag-of-Frames / Bag-of-Features Approach

The majority of genre classification systems are signal-based – cf. (Scaringella et al., 2006) for an overview of these systems – and most of them are based on the so-called "Bag-of-Frames" or "Bag-of-Features" (BOF) approach. It proceeds as follows:

1. Each class is represented by several audio examples.
2. For each of these examples, the acoustic signal is cut into short overlapping frames (typically ∼50 ms frame with an overlap of 50%).
3. For each frame a feature vector is computed (typically spectral features such as Mel Frequency Cepstrum Coefficients).
4. These feature vectors are given to a statistical classifier (e.g. Gaussian Mixture Models) which models the global distributions or average values of these vectors over the whole piece or passage for each class. Interestingly such distributions have not only been used to separate the examples into classes but also to compute similarity measures between examples for tasks such as retrieval or recommendation (Aucouturier and Pachet, 2008), making the background literature on music genre classification and music similarity indissociable.

Typically the features used in the BOF are low level descriptors of music, focusing mostly on timbral texture (Aucouturier and Pachet, 2004), rhythmic content (Gouyon and Dixon, 2005), pitch content (melody or harmony) (Gómez, 2006) or, as suggested by Tzanetakis and Cook (2002), and in accordance with the modular architecture of music processing in the human brain pointed out by the neuroscientists Peretz and Coltheart (2003), a combination of the three (Basili et al., 2004; Berenzweig et al., 2004; Cano et al., 2005; Shen et al., 2006).

One interesting example of the use of the BOF is the Extractor Discovery System (EDS), an expert system developed at Sony CSL (Zils, 2004). Its particularity lies in optimising combinations of signal processing features with genetic programming. It is able to distinguish sounds with different timbres, even when they are played on the same instrument with only slight modifications of timbre (Roy et al., 2007), to learn subjective measures such as the perceived intensity (Zils and Pachet, 2003) and to build a classifier "modelling [urban sounds] to near-perfect precision", but fails in classifying polyphonic music with the same precision (Aucouturier et al., 2007).

It has indeed been suggested by many that the BOF presents a glass-ceiling, or in other words a maximum accuracy that can not be surpassed, even when optimising its various steps

(e.g. design of the signal-based features, feature selection, etc.) (Aucouturier and Pachet, 2004). Other reported and connected limitations of the BOF include the creation of false positive hubs (Aucouturier and Pachet, 2008) and false negative orphans (Pampalk, 2006), respectively abnormally similar and dissimilar to any other piece. Explanations for these behaviours such as the curse of dimensionality – the feature space being high-dimensional – (Karydis et al., 2010), as well as fixes for them e.g. using mutual proximity (Flexer et al., 2012) have also been provided by the community.

If those shortcomings are purely statistical, the MIR community has also criticised the BOF approach for ignoring the musical nature and properties of the content it classifies. For instance Aucouturier and Pachet (2008) explain that most of the time there is no direct mapping between the acoustical properties and mental representation of a musical entity, such as genre or mood. They also point out that contrary to the bag-of-frames assumption the contribution of a musical event to the perceptual similarity is not proportional to its statistical importance – rare musical events can even be the most informative ones to determine its genre (Aucouturier et al., 2007). Moreover the bag-of-frames approach ignores the temporal organisation of the acoustic signal. Indeed rarely does time modelling go beyond delta features – comparing values of the current frame with those of the preceding one. And yet when comparing pieces from similar genres or passages of a same song it is crucial in the retrieval process to use sequences and not only average values or global statistical distributions of features over a whole passage or piece (Casey and Slaney, 2006). In summary, the BOF approach, based on low-level signal-based content descriptors, lacks high-level, contextual concepts which are equally important for the human perception and characterisation of music genres (McKay and Fujinaga, 2006).

Combining Low and Higher Level Descriptors/Features

Thus, recently, several attempts have been made to use, or integrate with state-of-the-art low-level audio features, such higher-level or contextual features, including: long-time audio features (Meng et al., 2005), statistical (Lidy et al., 2007) or distance-based (Cataltepe et al., 2007) symbolic features, text features derived from song lyrics (Neumayer and Rauber, 2007), cultural features or contextual features extracted from the web (Whitman and Smaragdis, 2002), social tags (Chen et al., 2009) or combinations of several of these high-level features (McKay and Fujinaga, 2008).

Using Sequences

When dealing with symbolic data, the Bag-of-Frames approach can obviously not be applied. However, as Hillewaere et al. (2009) explain, much of the work on genre classification of symbolic musical data uses *global features*, i.e. features at the level of the entire piece of music. They nonetheless show that models using *event features*, i.e. features representing the pieces of music as sequences of events, outperform global feature models. In their case the models employed with such event features are n-grams, and also their own multiple viewpoint model. Pérez-Sancho et al. (2009) also employ n-grams to represent melodic and harmonic sequences and perform genre classification on symbolic data. They also prove that the same sequence-based approach can be applied to audio data (Pérez-Sancho et al., 2010).

Harmony-Based Approaches to Music Genre Classification

Although some harmonic (or chord) sequences are famous for being used by a composer or in a given genre, harmony is scarcely found in the automatic genre recognition literature as a means to that end. Pérez-Sancho et al. (2008) investigated whether stochastic language models of harmony including naïve Bayes classifiers and 2-, 3- and 4-grams could be used for automatic genre classification on both symbolic and audio data. They reported better classification results when using a richer vocabulary (i.e. including seventh chords), reaching 3-genre classification accuracies on symbolic data of 86% with naïve Bayes models and 87% using bi-grams (Pérez-Sancho et al., 2009). To deal with audio data generated from MIDI they used a chord transcription algorithm and obtain accuracies of 75% with naïve Bayes (Pérez-Sancho, 2009) and 89% when using bi-grams (Pérez-Sancho et al., 2010). Earlier attempts at using harmony include Tzanetakis et al. (2003), who introduced pitch histograms as a feature describing the harmonic content of music. Statistical pattern recognition classifiers were trained to extract the genres. Classification of audio data covering 5 genres yielded recognition rates around 70%, and for audio generated from MIDI files rates reached 75%. However this study focused on low-level harmony features. Only a few studies have considered using higher-level harmonic structures, such as chord progressions, for automatic genre recognition. In (Shan et al., 2002), a frequent pattern technique was used to classify sequences of chords into three categories: Enya, Beatles and Chinese folk songs. The algorithm looked for frequent sets, bi-grams and sequences of chords. A vocabulary of 60 different chords was extracted from MIDI files through heuristic rules: major, minor, diminished and augmented triads as well as dominant, major, minor, half and fully diminished seventh chords. The best two way classifications were obtained using sequences with accuracies between 70% and

84%. Lee (2007) considered automatic chord transcription based on chord progression. He used hidden Markov models on audio generated from MIDI and trained by genre to predict the chords. It turned out he could not only improve chord transcription but also estimate the genre of a song. He generated 6 genre-specific models, and although he tested the transcription only on the Beatles' songs, frame rate accuracy reached highest level when using blues- and rock-specific models, indicating that models could be used to identify genres.

2.4 Conclusions

In this chapter we have provided definitions and context information for the music theory and musicological concepts and terms we will be using in the following chapters. Harmony being the domain we have decided to explore as a high-level descriptor of music we have taken the time to define and explain the terms and notations associated with it, including in particular chord symbols and notation conventions. We also reviewed related work on style characterisation and music genre classification which are the tasks we will explore using harmony only in Chapters 4 and 5 respectively. We saw that what we defined as characterisation, the task of analysing and extracting patterns of interest in pieces of music representing a unified musical style, has not only been explored by the MIR community as a task in itself for its musicological and ethnomusicological applications, but has also been used as an intermediate step in retrieval and identification tasks. The most popular of such tasks is the extensively studied problem of music genre classification which we have also reviewed and for which we have also described the most commonly employed approaches. Many of those in fact build models which are black-boxes (due to the low-level signal based-features they employ) and ignore high-level and temporal musical properties of the items they classify. We saw that promising solutions in music genre classification have employed either high-level and contextual features or sequences of musically meaningful events. It is at the intersection of these that harmony-based music genre classification approaches lie. It is clear from our literature review that most of those harmony-based methods use n-grams or other statistical sequential models of fixed-length. If such sequential harmony models have shown to have a distinctive characterisation power which we shall build upon in this thesis, much could be

done to better capture of the essence of harmony. One limitation of these models is their lack of flexibility, which we address by applying techniques from another domain to both represent and infer harmony-based models for characterisation and classification: Inductive Logic Programming, which we will now review in Chapter 3.

CHAPTER 3

BACKGROUND AND RELATED WORK IN INDUCTIVE LOGIC PROGRAMMING

3.1 Introduction

Inductive Logic Programming (ILP) is a field at the intersection of Machine Learning and Logic Programming (Muggleton, 1991). It is a technique that learns from examples (i.e. which induces general rules from specific observations). Based on a first-order logic framework it permits to express concepts that might not be formulated in a traditional attribute-value framework (Lavrac and Džeroski, 1994). Moreover it supports background knowledge and can handle imperfect data. At first, ILP was restricted to binary classification tasks but recently it has been adapted to many more data mining tasks. Finally ILP has already been successfully used for knowledge discovery and to build expert/reasoning systems in various engineering and research domains including MIR.

3.2 A Definition of ILP

To define what Inductive Logic Programming is we first describe the tasks of inductive concept learning (without and with background knowledge) and relational learning.

3.2.1 Inductive Concept Learning

Given a universal set of objects U, a concept C is a subset of objects in U ($C \subseteq U$). The problem of inductive concept learning can be defined as follows: given instances and non-instances of C, find a hypothesis able to tell for each $x \in U$ whether $x \in C$.

To perform an inductive concept learning task one needs to specify a language of examples L_E which defines the space of instances considered (i.e. U) and a language of concept description L_H which defines the space of hypotheses considered. If an example e expressed in L_E is an instance of the concept C then e is a positive example of C otherwise it is said to be a negative example of C. A coverage relation between L_H and L_E, $covers(H, e)$ needs also to be specified. It returns $true$ when the example e belongs to the concept defined by the hypothesis H and

$false$ otherwise. We can then define a new relation:

$$covered(H, E) = \{e \in E | covers(H, e) = true\}$$

which returns the set of examples E which are covered by H. So the problem of inductive concept learning can be reformulated as follows: given a set of examples E containing positive (set E^+) and negative (set E^-) examples of a concept C expressed in a given language of examples L_E, find a hypothesis H described in a given language of concept description L_H such that:

- every positive example $e \in E^+$ is covered by H (completeness): $covered(H, E^+) = E^+$
- no example $e \in E^-$ is covered by H (consistency): $covered(H, E^-) = \emptyset$

3.2.2 Inductive Concept Learning with Background Knowledge

When a concept learner has also access to prior knowledge, this prior knowledge is called *background knowledge*.

The task of inductive concept learning with background knowledge is described as follows: given a set of examples E and background knowledge B, find a hypothesis H described in a given language of description L_H such that it is complete and consistent with respect to the set of examples E and the background knowledge B ($covered(B, H, E^+) = E^+$ and $covered(B, H, E^-) = \emptyset$).

Notice that the $covers$ relation is extended as follows: $covers(B, H, e) = covers(B \cup H, e)$.

3.2.3 Relational Learning

One class of learning systems is the class of relational learners. They deal with structured concepts and structured objects defined in terms of their components and relations among them. These relations constitute the background knowledge.

The languages of examples and of concept description used by relational learners are typically subsets of first-order logic. When the hypothesis language used by a relational learner is the language of logic programs (Lloyd, 1987) it is called an *inductive logic programming* system. It turns out that in most ILP systems not only the hypotheses are expressed in logic program form but also the examples and the background knowledge (with additional restrictions for each of the languages).

So in the case of ILP the coverage relation can be written: $covers(B, H, e) \equiv B \wedge H \models e$ where $\models$ stands for logical implication or entailment.

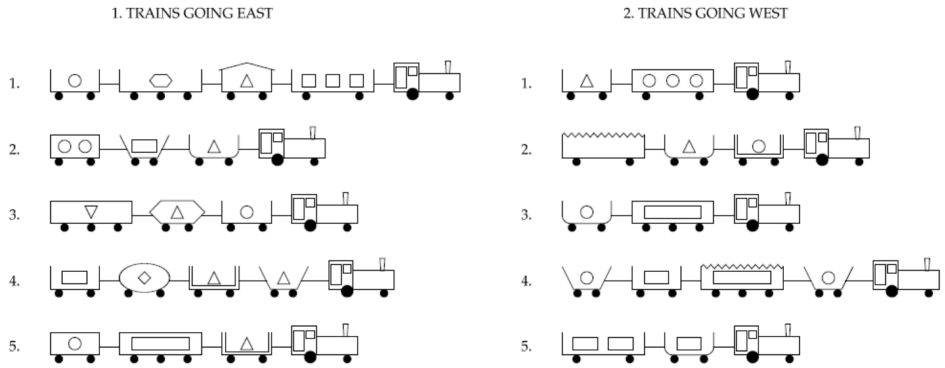

Figure 3.1: Michalski's train problem. From (Michalski, 1980).

3.2.4 A Simple Inductive Logic Programming Problem

To understand in more detail how ILP works we illustrate its principle using a simple and well-known relational learning problem: Michalski's train challenge (Michalski, 1980).

Descriptions are provided for ten trains, five eastbound trains and five westbound trains. Each description contains information about the number of carriages of a train, the length of each carriage (which can be long or short), the roof of each carriage (open or closed roof), the number of wheels each carriage has, and the loads carried or not in each carriage (information about their presence and about their shapes). The description of these ten trains is illustrated in Figure 3.1. The challenge consists in finding a way to generalise from the examples and automatically distinguish the eastbound trains from the westbound trains (binary classification problem).

An ILP system can learn a rule that defines what is an eastbound train. In ILP the positive examples are described as Prolog facts, so in this problem the positive examples can be expressed as follows[1]:

```
eastbound(eastTrain1).
eastbound(eastTrain2).
eastbound(eastTrain3).
eastbound(eastTrain4).
eastbound(eastTrain5).
```

Similarly the negative examples (i.e. facts that are false) are:

```
eastbound(westTrain1).
eastbound(westTrain2).
```

[1]the predicates employed in this example are the ones used by Srinivasan (2003) to express the same problem.

```
eastbound(westTrain3).
eastbound(westTrain4).
eastbound(westTrain5).
```

Then we need to store the descriptions of each train in our background knowledge. For instance the description of the first eastbound train in Prolog facts can be expressed as follows:

```
has_car(eastTrain1,car_11).
has_car(eastTrain1,car_12).
has_car(eastTrain1,car_13).
has_car(eastTrain1,car_14).
short(car_12).
closed(car_12).
long(car_11).
long(car_13).
short(car_14).
open_car(car_11).
open_car(car_13).
open_car(car_14).
shape(car_11,rectangle).
shape(car_12,rectangle).
shape(car_13,rectangle).
shape(car_14,rectangle).
load(car_11,rectangle,3).
load(car_12,triangle,1).
load(car_13,hexagon,1).
load(car_14,circle,1).
wheels(car_11,2).
wheels(car_12,2).
wheels(car_13,3).
wheels(car_14,2).
```

Notice that the background knowledge is not limited to facts and can contain rules. For instance imagine that we want to add information about the carriages' positions in the train in terms of which carriage follows which other carriage. We could add the following facts:

```
succ(car_12,car_11).
```

```
succ(car_13,car_12).
succ(car_14,car_13).
```

where succ(X,Y) means X follows Y.

Using the predicates from the positive examples, negative examples and background knowledge the ILP system can then generate the following hypothesis which covers all the positive examples and none of the negative examples:

```
eastbound(A) :- has_car(A, B), short(B), closed(B).
```

which says that an eastbound train always has a carriage which is short and closed.

3.3 ILP Techniques and Frameworks

Let us go back to the basic ILP problem of relational rule induction. To induce a hypothesis H which is complete and consistent with respect to a set of examples E and background knowledge B without enumerating all the possible results, several ILP techniques have been developed including least general generalisation, inverse resolution, inverse entailment. It is beyond the scope of this thesis to enumerate and explain all the possible ILP techniques to search the space of clauses. For a good overview and description of these techniques we refer the reader to (Džeroski et al., 2000).

The goal of this thesis is not to develop a new ILP technique or framework. That is why we looked at established ILP frameworks, starting from the in-depth comparison provided in (Maclaren, 2003, Section 3.2). After testing we selected Aleph (Srinivasan, 2003) for its usability, responsiveness of its user community and existing examples of its use in MIR tasks. We later moved on to TILDE as we wanted to perform classification. TILDE not only is a classification algorithm, it also benefits from a very active and responsive maintenance team which constantly optimises its performance. The few other candidates were rejected for their non-maintained state, lack of support or poor performance.

Aleph (used in Chapter 4) is based on inverse entailment. Inverse entailment consists in selecting an uncovered example, saturating it (i.e. looking for all the facts that are true about this example using the example itself and the background knowledge) to obtain a bottom clause (the disjunction of all the facts found in the saturation phase) and searching the space of clauses

that subsumes this bottom clause in a top-down manner starting from the shortest clauses. The clause that covers the maximum number of positive examples and the minimum number of negative examples (i.e. which maximises a score function based on the number of positive and negative examples covered by this clause) is kept as a hypothesis. The examples covered by this hypothesis are removed and the next uncovered example is selected to be saturated, and so on until no uncovered example is left. Finally Aleph returns a set of hypotheses that covers all the positive examples. Note that like most of the recent ILP systems, Aleph is able to handle noise and imperfect data. One of the parameters the user can change is the noise level, which is the amount of negative examples that can be covered by a hypothesis.

In order to build classification model, we use in Chapter 5 TILDE. It is a first order logic extension of the C4.5 decision tree induction algorithm (Quinlan, 1993). Like C4.5 it is a top-down decision tree induction algorithm. The difference is that at each node of the trees conjunctions of literals are tested instead of attribute-value pairs. At each step the test (i.e. conjunction of literals) resulting in the best split of the classification examples is kept. As explained in (Blockeel and De Raedt, 1998) "*the best split means that the subsets that are obtained are as homogeneous as possible with respect to the classes of the examples*". By default TILDE uses the information gain-ratio criterion (Quinlan, 1993) to determine the best split. TILDE builds first-order logic decision trees expressed as ordered sets of rules (or Prolog programs). For an example illustrating the induction of a classification tree from a set of examples covering three musical genres, we refer the reader to the Figure 5.2 in Chapter 5.

3.4 Relational Data Mining

If relational rule induction was the first and is still the most common task of ILP, it is no longer restricted to it. The ILP approach has been extended to most data mining tasks. For each data mining technique using a propositional approach a relational approach using first-order logic has been suggested and classified under the umbrella term *Relational Data Mining* (Džeroski and Lavrac, 2001b). Note that there is a trade-off between the expressiveness of first-order logic and computational complexity of the algorithms using such an approach. This explains why these relational data mining techniques were successfully developed only recently. Džeroski (2006) gives an overview of all the relational data mining techniques one can now

use. Among them:

- induction of relational classification rules – with the ICL software (Van Laer and De Raedt., 2001),
- relational classification using nearest-neighbors – with RIBL (Emde and Wettschereck, 1996) and RIBL2 (Horváth et al., 2001; Kirsten et al., 2001),
- relational decision trees – TILDE (Blockeel and De Raedt, 1998),
- first-order random forests (Van Assche, 2008) – also implemented in TILDE,
- relational regression trees and rules – TILDE, S-CART (Kramer, 1996) and RIBL2,
- relational clustering (Kirsten et al., 2001)
- frequent pattern discovery (Dehaspe, 1999),
- discovery of relational association rules (Dehaspe and Toivonen, 1999, 2001).

3.5 Applications of ILP

3.5.1 A Tool Used in Many Disciplines

ILP has been successfully used for knowledge discovery and to build expert/reasoning systems in various engineering and research domains. For instance it has been used to learn rules for early diagnosis of rheumatic diseases (using examples and background knowledge provided by an expert), to design finite element meshes (by constructing rules deciding appropriate mesh resolution, a decision usually made by experts), to predict protein secondary structure, to design drugs (by finding structure-activitiy relations of the chemical components), to learn diagnosis rules from qualitative models. For a detailed description of these examples we refer the reader to (Lavrac and Džeroski, 1994) and (Bratko and Muggleton, 1995). It is also extensively used in Natural Language Processing (Džeroski et al., 2000; Claveau et al., 2003).

3.5.2 Musical Applications of ILP

Not surprisingly, ILP and similar inductive logic approaches have also been successfully used on musical data.

Widmer worked on identifying relevant rules of expressive performance from MIDI recordings of W.A. Mozart's sonatas performed by different pianists on a Bösendorfer SE290 computer-monitored grand piano (Widmer et al., 2003). Because it was not possible to build completely discriminative models, which would mean that the artists who perform are "perfectly consistent and predictable" (Widmer, 2001), he developed the PLCG (for Partition Learn Cluster Generalize) algorithm, an inductive rule learning system which builds partial models, i.e. models that explain only the examples that can be explained (Widmer, 2003). The target was to learn local rules (i.e. for each note) concerning the tempo (*accelerando* or *ritardando*), dynamics (*crescendo* or *diminuendo*) and articulation properties (*staccato*, *legato* or *portato*) of the note. To illustrate each concept, positive examples were given to the system and they were also used as negative examples of the competing classes. The background knowledge was fed with descriptions of each note containing information about intrinsic properties (e.g. as duration, metrical position) and information about the context of the note (such as the interval between a note and its predecessor, and the duration of surrounding notes). The PLCG algorithm extracted 17 expressive performance rules (2 for *accelerando*, 4 for *ritardando*, 3 for *crescendo*, 3 for *diminuendo*, 4 for *staccato*, 1 for *legato*) among which some were surprising but nevertheless relevant performance rules, such as:

> "*Given two notes of equal duration followed by a longer note, lengthen the note (i.e., play it more slowly) that precedes the final, longer one, if this note is in a metrically weak position [...]; none of the existing theories of expressive performance were aware of this simple pattern*".

In a similar study, Dovey (1995) analysed and extracted rules from piano performances of Rachmaninoff recorded in the 1920's on an Ampico Recording Piano. For that he used the PROGOL ILP system.
His work was extended by Van Baelen and De Raedt (1996) who used both Ampico recordings and MIDI performance data analysed using the Sound Harmony Melody Rhythm and Growth (SHMRG) model (LaRue, 1970). With additional context information (i.e. more background knowledge containing also non-local rules) coming from the analysis of the MIDI pieces they obtained better rules of performance regularities than Dovey's and used them to predict the

performance of each note. These predictions were then encoded as MIDI information. A listening analysis of these files showed that at expressive performance was not modelled well at a global level, but at a local level, some bars were actually very well interpreted by the automatic system.

But Inductive Logic Programming has not only been employed for musical performance analysis. Morales (1997) implemented a pattern-based first-order inductive system called PAL to learn counterpoint rules. The system looks for patterns in the notes, described by their pitch (including octave) and voice, using background knowledge restricted to the classification of intervals between pairs of notes into perfect or imperfect consonant, and dissonant, valid and invalid intervals. PAL was fed with a small number of examples of the four counterpoint rules of the first species and was able to induce those rules automatically.

The most recent work using ILP for MIR is Ramirez's. His first ILP based application learns rules in popular music harmonisation using Aleph (Ramirez, 2003). The rules were constructed at a bar level (and not at a note level) to capture chord patterns. The structure (i.e. musical phrases) of the songs given as examples was manually annotated, which provided the system with a rich background knowledge containing not only local but also global information. The system proved to be capable of inducing very simple and very general rules. But the fact that manually annotated data is necessary limits the scalability of such a system.

Later on, Ramirez et al. (2004) studied Jazz performance but starting from audio examples this time. Monophonic recordings of jazz standards were automatically analysed, extracting low level descriptors (instantaneous energy and fundamental frequency), performing some note segmentation and using those results to compute note descriptors. The positive and negative examples given to the ILP system (Aleph) were these automatically extracted note descriptors. Ramirez et al. were interested in differences between the score indication and the actual interpretation of a note. So they asked the system to induce rules related to the duration transformation (lengthen, shorten or same) of a note, its onset deviation (advance, delay, or same), its energy (soft, loud and same) and note alteration which refers to alteration of the score melody by adding or deleting notes (consolidation, ornamentation and none). The background knowledge was composed of information about the neighbouring notes and the Narmour group(s), i.e. basic melodic structural units based on the Implication-Realisation model of Narmour (1990), to which each note belongs. The tempo of the performance was also given to the ILP system in order to study if it had an influence on the performance rules.

Some rules induced by the system turned out to have a high coverage which confirmed the presence of pattern in jazz expressive performance.

Finally, following Van Baelen and De Raedt's idea, Ramirez and Hazan (2006) implemented a framework which analyses classical violin performance by means of both an ILP technique (the relational decision tree learner called TILDE) and a numerical method. Another component of this system then uses these results to synthesise expressive performances from unexpressive melody descriptions.

3.6 Conclusions

In this chapter we have provided an introduction to Inductive Logic Programming and its core concepts through definitions and a simple example. We have also chosen and described two ILP techniques that we will use in our experiments. We finally reviewed applications of ILP, emphasising on its use in MIR.

We have seen that harmony has been already modelled with ILP with promising results. Additionally the numerous studies on musical performance with ILP allowed us to compare and identify interesting practices and algorithms. Building on the experience gathered by Ramirez et al. we will combine harmony and the ILP systems Aleph and TILDE in our own experiments in Chapters 4 and 5.

CHAPTER 4

AUTOMATIC CHARACTERISATION OF THE HARMONY OF SONG SETS

4.1 Introduction

In this chapter we present our first attempt at describing sets of songs using harmony-based representation and relational induction of logical rules. The starting point of this first approach is a paper by Mauch et al. (2007) in which the authors study two distinct corpora of two distinct genres which might still exhibit shared harmony practices. The two genres are British pop, represented by The Beatles, and jazz represented by a set of songs from the Real Book songs. We extract their respective most common chord sequences using a statistical approach. We present here our own analysis of the exact same symbolic corpora which is in our case entirely based on Inductive Logic Programming and compare the two approaches, stating how our methodology overcomes theirs. In Section 4.2 we explain our methodology to automatically extract logical harmony rules from manually annotated chords. In Section 4.3 the details and results of our automatic analysis of the Beatles and Real Book with ILP are presented. As in the next chapters the primary focus is on methodology and knowledge representation, rather than on the presentation of new musical knowledge extracted by the system. However we qualitatively evaluate the characterisation power of our methodology by performing a short musicological analysis of the harmony rules we automatically extracted. We conclude with an analysis and description of the constraints and limitations of the specific Inductive Logic Programming software used in this study, Aleph, and explaining how that led us to experiment with other knowledge representations and ILP induction techniques and software (Section 4.4).

4.2 Methodology

As seen in Section 2.3.1, in search of chord idioms, Mauch et al. (2007) made an inventory of chord sequences present in a subset of the Real Book and in The Beatles' studio albums. Their approach is entirely statistical and resulted in an exhaustive list of chord sequences together with their relative frequencies. To compare the results of our relational methodology

with their results obtained with a statistical method, we examine RDF (Resource Description Framework) descriptions of the two manually annotated collections they use:

- Harte's transcriptions of the 180 songs featured on the Beatles' studio albums[1] containing a total of 14,132 chords (Harte, 2010),
- transcriptions of 244 Jazz standards from the Real Book[2] containing 24,409 chords (various, 2004).

These transcriptions constitute a compact symbolic representation of the songs: the chords are manually labelled in a jazz/pop/rock shorthand fashion (explained in more details in Section 2.2.2) and their start and end times are provided.

The steps to extract harmony rules from these songs transcriptions are summarised as follows: First the RDF representation of the harmonic events is pre-processed and transcribed into a logic programming format that can be understood by an Inductive Logic Programming system. This logic programming representation is passed to the ILP software Aleph (Srinivasan, 2003) which induces the logical harmony rules underlying the harmonic events.

4.2.1 Harmonic Content Description

The RDF files describing the Beatles and Real Book songs we study contain a structured representation of the harmonic events based on the Music Ontology (Raimond et al., 2007) as described in Section 2.2.2.

We implemented an RDF chord parser to transcribe RDF chord representation into Prolog files that can be directly given as input to Aleph. For each of these chords it extracts the root note, bass note, component intervals (extracted from the additive or subtractive description of the chord), start time and end time from the RDF description. It then computes the chord category and degree (if key is given) of a chord and the root interval and bass interval between two consecutive chords.

As we do not know in which octaves the root and bass notes are (since this is not relevant to our harmony analysis), we chose to measure all intervals upwards, i.e. assuming that the second note always has a higher pitch than the first one. For instance the interval between C and Bb is a minor seventh (and not a downward major second). Similarly the interval between G and C is a perfect fourth (and not a downward perfect fifth). This choice guarantees that we consistently measure intervals and can find interval patterns in the chord sequences.

[1] these RDF files are available at `http://isophonics.net/content/reference-annotations-beatles`
[2] available at `http://www.omras2.org/chordtranscriptions`

For this study we limit the chord categories (or chord types) to 'major', 'minor', 'augmented', 'diminished', 'suspended', 'dominant', 'neutral' (when the 3rd is neither present nor suspended) and 'unknown' (for every chord that does not belong to the previous categories). For each chord, the intervals are analysed by the RDF chord parser which then assigns the chord to one of these categories. First it reduces the chord to a 7th chord and checks if this reduced chord is a dominant 7th, in which case the chord is labeled 'Dominant'. Otherwise the chord is reduced to a triad and the type of this triad is kept as the chord category.

The degrees are computed by our RDF chord parser using the current key. Key information was added by hand when available. We only had access to tonality information for the Beatles, so no degree details were added for the Real Book songs. For the Beatles we performed two studies: one without degree over the whole set of songs and one with degree in which only the songs where there was no key modulation were kept. In this second study we also filtered out the songs which were not tonal songs (i.e. songs that were not following major or minor scales) which yielded a remaining dataset of 73.9% of the Beatles' songs.

Our sole interest is in sequences of chords between which there is a harmonic modification (i.e. at least the root, bass or chord category differs from one chord to the next one). Although harmonic rhythm is important (cf. Section 2.2.1) we do not take it into account in this work.

4.2.2 Rule Induction with ILP

We restrict our focus to chord sequences of length 4 as in Mauch et al.'s study (2007). A four-chord sequence is a typical phrase length for the studied corpora. This choice is also the result of an empirical process: we also studied shorter sequences, but the results consist of only a few rules (25 for the Beatles and 30 for the Real Book) with high coverage and little interest (such as '2 consecutive major chords' covering 51% of the Beatles chord sequences of length 2). For longer sequences, the extracted patterns are less general, i.e. have a smaller coverage and thus are less characteristic of the corpus. The concept we want to characterise is the harmony of a set of songs e.g. all the Beatles songs, all the Real Book songs. Therefore the positive examples given to the ILP system are all the chord sequences of length 4 (predicate `chord_prog_4/4`) found in such a set of songs. These chord sequences overlap: from a chord sequence of length n, with $n \geq 4$ we extract $n - 4 + 1$ overlapping chord sequences of length 4. For instance the Aleph file containing all the positive examples for the Beatles looks like this:

```
chord_prog_4(chord1_1_1,chord1_1_2,chord1_1_3,chord1_1_4).
chord_prog_4(chord1_1_2,chord1_1_3,chord1_1_4,chord1_1_5).
chord_prog_4(chord1_1_3,chord1_1_4,chord1_1_5,chord1_1_6).
```

```
chord_prog_4(chord1_1_4,chord1_1_5,chord1_1_6,chord1_1_7).
chord_prog_4(chord1_1_5,chord1_1_6,chord1_1_7,chord1_1_8).
...
chord_prog_4(chord1_1_56,chord1_1_57,chord1_1_58,chord1_1_59).
chord_prog_4(chord1_1_57,chord1_1_58,chord1_1_59,chord1_1_60).
chord_prog_4(chord1_2_1,chord1_2_2,chord1_2_3,chord1_2_4).
chord_prog_4(chord1_2_2,chord1_2_3,chord1_2_4,chord1_2_5).
...
chord_prog_4(chord1_14_110,chord1_14_111,chord1_14_112,chord1_14_113).
chord_prog_4(chord2_1_1,chord2_1_2,chord2_1_3,chord2_1_4).
...
chord_prog_4(chord12_12_77,chord12_12_78,chord12_12_79,chord12_12_80).
```

Where `chordX_Y_Z` means the `Z`th chord in the `Y`th song of the `X`th album. These are Prolog atoms which uniquely identify each chord in each song. Hence all chord sequences of length 4 starting from any position of any song from any album of The Beatles (at least all those in our annotated corpus) are listed in this file.

The background knowledge is composed of the descriptions of all the chords previously derived by the RDF chord parser. So for each of those uniquely identified chords a full description of its attributes is stored in the background knowledge, in the following format:

```
chord(chord1_1_1).
has_category(chord1_1_1,maj).
has_root(chord1_1_1,[e,n]).
has_bassNote(chord1_1_1,[e,n]).
startingTime(chord1_1_1,2.612267).
has_degree(chord1_1_1,[1,n]).
rootInterval(chord1_1_1,chord1_1_2,[4,n]).
bassInterval(chord1_1_1,chord1_1_2,[4,n]).
pred(chord1_1_1,chord1_1_2).
```

This code says that the first chord of the first song of the first album of the Beatles is a chord (first line), which is a major chord (second line), whose root note is E (third line) and bass note is E (fourth line). That chord starts 2.612267 seconds into the song (fifth line) and it is on the tonic, i.e. degree I (sixth line). Then the next lines describe the connections between that first chord and the second one: there is a perfect fourth between the root notes of `chord1_1_1`

and `chord1_1_2` (seventh line) and similarly for the bass notes (eighth line) and `chord1_1_1` precedes `chord1_1_2` (ninth line).

In the ILP system we use to induce harmony rules, Aleph (Srinivasan, 2003), we can either provide negative examples of a concept (in our case, chord progressions of length 4 from another set of songs not included in the current one) or force Aleph to explain the positive examples using a well-designed negative example (we will refer to this mode as the *one negative example mode*). In the latter case our negative example consists of the first chord sequence of our corpus in which we exchanged the position of the first and second chords as shown below:

```
chord_prog_4(chord1_1_2,chord1_1_1,chord1_1_3,chord1_1_4).
```

It is a valid negative example because in our background knowledge the position of each uniquely identified individual chord relative to the other chords is specified, using the predicate `pred/2`. So it is impossible for the second chord in the first song of the corpus (`chord1_1_2`) to precede the first one (`chord1_1_1`). We found out that by limiting the set of negative examples to this very simple one we obtained a more complete set of rules than when using the *positive examples only mode* of Aleph which randomly generates a limited number of negative examples.

To generate hypotheses Aleph uses inverse entailment (cf. Section 3.3 for more details). It consists of selecting an uncovered example, saturating it to obtain a bottom clause and searching the space of clauses that subsumes this bottom clause in a top-down manner starting from the shortest clauses. The clause that is kept as a hypothesis is the one that maximises the evaluation function, which in our case is the default Aleph evaluation function called 'coverage' and equal to $P - N$, where P, N are the number of positive and negative examples covered by the clause. The examples covered by the found hypothesis are removed and the next uncovered example is selected to be saturated, and so on until no uncovered example is left. Finally Aleph returns a set of hypotheses that covers all the positive examples. The set of generated rules depends on the order in which the examples are selected by Aleph (which is the order in which the examples are given to Aleph). So the resulting set of rules is only one of the sets of rules that could be induced from the set of examples. However since Aleph looks for the most general rules at each step, the final set of rules is a sufficient description of the data (it explains all chord sequences) and is non-redundant (no subset of the rules explains all the chord sequences). This minimal sufficient description of a data set could be very useful for classification purposes since only a few characteristics need to be computed to classify a new example. This is one of the advantages of our method against the purely statistical method employed by Mauch et al. (2007) which only computes the frequencies of each chord sequence and does not try to build a sufficient model of the corpora.

To obtain meaningful rules we also constrain Aleph to look for a hypothesis explaining the chord progressions only in terms of specific root note progressions (`root_prog_4/8`), bass note progressions (`bassNote_prog_4/8`), chord category progressions (`category_prog_4/8`), root interval progressions (`rootInterval_prog_3/7`), bass interval progressions (`bassInterval_prog_3/7`) and degree progressions (`degree_prog_4/8`). The following lines of code in the background knowledge file tell Aleph that `chord_prog_4/4` can only be described with the six predicates aforementioned:

```
:-determination(chord_prog_4/4,root_prog_4/8).
:-determination(chord_prog_4/4,bassNote_prog_4/8).
:-determination(chord_prog_4/4,category_prog_4/8).
:-determination(chord_prog_4/4,rootInterval_prog_3/7).
:-determination(chord_prog_4/4,bassInterval_prog_3/7).
:-determination(chord_prog_4/4,degree_prog_4/8).
```

Additionally each of these six predicates is described, again in the background knowledge file, with the predicates used to describe the individual or pairs of chords:

```
category_prog_4(Chord1,Chord2,Chord3,Chord4,Cat1,Cat2,Cat3,Cat4):-
    pred(Chord1,Chord2),pred(Chord2,Chord3),pred(Chord3,Chord4),
    has_category(Chord1,Cat1),has_category(Chord2,Cat2),
    has_category(Chord3,Cat3),has_category(Chord4,Cat4).

root_prog_4(Chord1,Chord2,Chord3,Chord4,Root1,Root2,Root3,Root4):-
    pred(Chord1,Chord2),pred(Chord2,Chord3),pred(Chord3,Chord4),
    has_root(Chord1,Root1),has_root(Chord2,Root2),
    has_root(Chord3,Root3),has_root(Chord4,Root4).

bassNote_prog_4(Chord1,Chord2,Chord3,Chord4,BassNote1,BassNote2,BassNote3,BassNote4):-
    pred(Chord1,Chord2),pred(Chord2,Chord3),pred(Chord3,Chord4),
    has_bassNote(Chord1,BassNote1),has_bassNote(Chord2,BassNote2),
    has_bassNote(Chord3,BassNote3),has_bassNote(Chord4,BassNote4).

degree_prog_4(Chord1,Chord2,Chord3,Chord4,Degree1,Degree2,Degree3,Degree4):-
    pred(Chord1,Chord2),pred(Chord2,Chord3),pred(Chord3,Chord4),
```

```
    has_degree(Chord1,Degree1),has_degree(Chord2,Degree2),
    has_degree(Chord3,Degree3),has_degree(Chord4,Degree4).

rootInterval_prog_3(Chord1,Chord2,Chord3,Chord4,RootInterval1,RootInterval2,
RootInterval3):-
    rootInterval(Chord1,Chord2,RootInterval1),
    rootInterval(Chord2,Chord3,RootInterval2),
    rootInterval(Chord3,Chord4,RootInterval3).

bassInterval_prog_3(Chord1,Chord2,Chord3,Chord4,BassInterval1,BassInterval2,
BassInterval3):-
    bassInterval(Chord1,Chord2,BassInterval1),
    bassInterval(Chord2,Chord3,BassInterval2),
    bassInterval(Chord3,Chord4,BassInterval3).
```

4.3 Experiments and Results

4.3.1 Independent Characterisation of The Beatles and Real Book Chord Sequences

We run two experiments. In the first experiment we want to characterise the chord sequences present in the Beatles' songs and compare them to the chord sequences present in the Real Book songs. Therefore we extract all the chord sequences of length 4 in the Beatles' tonal songs with no modulation (10,096 chord sequences), all the chord sequences of length 4 in all the Beatles' songs (13,593 chord sequences) and all the chord sequences of length 4 from the Real Book songs (23,677 chord sequences). Then for each of these sets of chord sequences we induce rules characterising them using the *one negative example mode* in Aleph. It is important to realise that we run our experiments on all chord sequences of each group without considering the individual songs they are extracted from anymore.

Our system induces sets of 333 and 267 rules for each of the Beatles collections (all chord sequences in tonal songs with no modulation, all chord sequences in all songs) and a set of

646 rules for the Real Book. The positive coverage of a rule is the number of positive examples covered by this rule. We want to consider only the patterns characteristic of the corpus, i.e. the ones occurring in multiple songs. For that we leave out the rules with a too small coverage (smaller than 1%). The top rules for our first experiment are shown in Tables 4.1 and 4.2. For analysis purposes they have been re-ordered by decreasing coverage.

For readability we show here a compact representation of the body of rules:

- degrees are represented with roman numerals,
- "/ " precedes a bass note as in jazz chord notation,
- the intervals between roots (written first) or bass notes of the chords (following a "/") are put on top of the arrows,
- a bullet symbolises the absence of information about some characteristics of the chord.

In accordance with Mauch et al.'s conclusions (2007), some patterns extracted in these experiments are very common pop and jazz harmonic patterns. For instance, the Beatles rule with the highest coverage (more than a third of the chord sequences) is a sequence of 4 major chords. The minor chord is the second most frequent chord category in the Beatles and the dominant chord ranks quite low in the chord category rules (rule 25). For the Real Book, the rule with the highest coverage is a sequence of three perfect fourth intervals between chord roots. An interpretation of this rule is the very common jazz progression ii-V-I-IV. Another common jazz chord progression, I-VI-II-V (often used as a "turnaround" in jazz), is captured by rule 8 in Table 4.2. Moreover the dominant chord is the most frequent chord category in the Real Book which clearly distinguishes the jazz standards of the Real Book from the pop songs of the Beatles.

Note that due to the fact that the chord sequences overlap and due to the cyclic nature of some of the pop and jazz songs, many rules are not independent. For instance rules 2, 3, 6 and 7 in Table 4.1 can represent the same chord sequence maj-maj-maj-min repeated several times.

Moreover we can also derive rules that make use of degree information. For this we constrain Aleph to derive rules about the intervals between the chord roots associated with chord category in order to capture harmonic patterns which can then be interpreted in term of scale degrees. The top root interval and category rules for each corpus are presented in Tables 4.3 and 4.4. Furthermore, since we have key information for some of the Beatles songs we can actually obtain degree rules for them and an analysis of the degree rules allows us to

Table 4.1: Beatles harmony rules whose coverage is larger than 1%. C_1 and C_2 represent the positive coverage over all the Beatles songs and over the Beatles tonal songs with no modulation respectively. "perfU" means perfect unison.

Rule	C_1	C_2
1. maj → maj → maj → maj	4752 (35%)	3951 (39%)
2. maj → maj → maj → min	632 (4.65%)	431 (4.27%)
3. min → maj → maj → maj	628 (4.62%)	448 (4.44%)
4. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf5th}} \bullet \xrightarrow{\text{perf4th}} \bullet$	586 (4.31%)	-
5. $\bullet \xrightarrow{/_{\text{perfU}}} \bullet \xrightarrow{/_{\text{perfU}}} \bullet \xrightarrow{/_{\text{perfU}}} \bullet$	584 (4.30%)	-
6. maj → min → maj → maj	522 (3.84%)	384 (3.80%)
7. maj → maj → min → maj	494 (3.63%)	363 (3.60%)
8. $\bullet \xrightarrow{/_{\text{perf5th}}} \bullet \xrightarrow{/_{\text{perf4th}}} \bullet \xrightarrow{/_{\text{perf5th}}} \bullet$	463 (3.41%)	346 (3.43%)
9. maj → maj → min → min	344 (2.53%)	217 (2.15%)
10. $\bullet \xrightarrow{\text{perfU}} \bullet \xrightarrow{\text{perfU}} \bullet \xrightarrow{\text{perfU}} \bullet$	336 (2.47%)	237 (2.38%)
11. min → min → maj → maj	331 (2.44%)	216 (2.14%)
12. maj → min → min → maj	308 (2.27%)	197 (1.95%)
13. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{maj2nd}} \bullet \xrightarrow{\text{perf4th}} \bullet$	260 (1.91%)	209 (2.07%)
14. $\bullet \xrightarrow{\text{maj2nd}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet$	251 (1.85%)	195 (1.93%)
15. $/_A \rightarrow /_A \rightarrow /_A \rightarrow /_A$	-	176 (1.74%)
16. min → maj → min → maj	232 (1.71%)	167 (1.65%)
17. min → min → min → min	226 (1.66%)	104 (1.03%)
18. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet$	219 (1.61%)	146 (1.45%)
19. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf5th}} \bullet$	216 (1.59%)	165 (1.63%)
20. maj → min → maj → min	212 (1.56%)	157 (1.56%)
21. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{maj2nd}} \bullet$	211 (1.55%)	160 (1.58%)
22. min → maj → maj → min	205 (1.51%)	132 (1.31%)
23. min → min → min → maj	204 (1.50%)	113 (1.12%)
24. maj → min → min → min	203 (1.49%)	119 (1.18%)
25. maj → dom → maj → maj	200 (1.47%)	174 (1.72%)
26. maj → maj → dom → maj	192 (1.41%)	170 (1.68%)
27. $\bullet \xrightarrow{\text{perf5th}} \bullet \xrightarrow{\text{min7th}} \bullet \xrightarrow{\text{perf5th}} \bullet$	188 (1.38%)	-
28. maj → maj → maj → dom	187 (1.38%)	166 (1.64%)
29. dom → maj → maj → maj	183 (1.35%)	153 (1.52%)
30. min → min → maj → min	176 (1.29%)	86 (0.85%)
31. $\bullet \xrightarrow{\text{perfU}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf5th}} \bullet$	172 (1.27%)	-
32. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perfU}} \bullet \xrightarrow{\text{perf4th}} \bullet$	169 (1.24%)	112 (1.11%)
33. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf5th}} \bullet \xrightarrow{\text{perf5th}} \bullet$	163 (1.20%)	152 (1.51%)
34. min → maj → min → min	163 (1.20%)	92 (0.91%)
35. $\bullet \xrightarrow{\text{perf5th}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet$	160 (1.18%)	-
36. dom → dom → dom → dom	147 (1.08%)	110 (1.09%)
37. $\bullet \xrightarrow{/_{\text{perf5th}}} \bullet \xrightarrow{/_{\text{perf5th}}} \bullet \xrightarrow{/_{\text{min7th}}} \bullet$	142 (1.04%)	132 (1.31%)
38. I → V → IV → I	-	111 (1.10%)
39. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perfU}} \bullet \xrightarrow{\text{perfU}} \bullet$	138 (1.02%)	88 (0.87%)
40. $\bullet \xrightarrow{\text{perfU}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perfU}} \bullet$	138 (1.01%)	100 (0.99%)
41. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perfU}} \bullet \xrightarrow{\text{perf5th}} \bullet$	135 (0.99%)	112 (1.11%)
42. $\bullet \xrightarrow{\text{perf5th}} \bullet \xrightarrow{\text{perfU}} \bullet \xrightarrow{\text{perf4th}} \bullet$	114 (0.84%)	103 (1.02%)

Table 4.2: Real Book harmony rules whose coverage is larger than 1%. C is the positive coverage.

Rule	C
1. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet$	1861 (7.86%)
2. min → dom → min → dom	969 (4.09%)
3. min → dom → maj → min	727 (3.07%)
4. dom → min → dom → min	726 (3.07%)
5. min → min → min → min	708 (2.99%)
6. dom → dom → dom → dom	674 (2.85%)
7. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perfU}} \bullet$	615 (2.60%)
8. $\bullet \xrightarrow{\text{maj6th}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet$	611 (2.58%)
9. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf5th}} \bullet \xrightarrow{\text{perf4th}} \bullet$	608 (2.57%)
10. dom → min → dom → maj	594 (2.51%)
11. dom → maj → min → dom	586 (2.47%)
12. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perfU}} \bullet \xrightarrow{\text{perf4th}} \bullet$	579 (2.45%)
13. $\bullet \xrightarrow{/\text{maj6th}} \bullet \xrightarrow{/\text{perf4th}} \bullet \xrightarrow{/\text{perf4th}} \bullet$	547 (2.31%)
14. maj → min → dom → maj	478 (2.02%)
15. $\bullet \xrightarrow{/\text{maj7th}} \bullet \xrightarrow{/\text{perf4th}} \bullet \xrightarrow{/\text{perf4th}} \bullet$	477 (2.01%)
16. $\bullet \xrightarrow{/\text{perf4th}} \bullet \xrightarrow{/\text{maj6th}} \bullet \xrightarrow{/\text{perf4th}} \bullet$	440 (1.86%)
17. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{maj6th}} \bullet$	436 (1.84%)
18. min → dom → maj → dom	424 (1.79%)
19. min → min → dom → maj	413 (1.74%)
20. $\bullet \xrightarrow{\text{perfU}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet$	395 (1.67%)
21. $\bullet \xrightarrow{\text{maj2nd}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet$	366 (1.55%)
22. $\bullet \xrightarrow{/\text{perfU}} \bullet \xrightarrow{/\text{perfU}} \bullet \xrightarrow{/\text{perfU}} \bullet$	358 (1.51%)
23. dom → maj → min → min	357 (1.51%)
24. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{maj2nd}} \bullet$	351 (1.48%)
25. maj → min → min → dom	317 (1.34%)
26. maj → min → dom → min	300 (1.27%)
27. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{maj2nd}} \bullet \xrightarrow{\text{perf4th}} \bullet$	292 (1.23%)
28. min → min → min → dom	290 (1.22%)
29. min → dom → min → min	288 (1.22%)
30. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf5th}} \bullet$	272 (1.15%)
31. $\bullet \xrightarrow{/\text{aug4th}} \bullet \xrightarrow{/\text{perf4th}} \bullet \xrightarrow{/\text{perf4th}} \bullet$	272 (1.15%)
32. min → dom → maj → maj	272 (1.15%)
33. min → min → dom → min	267 (1.13%)
34. dom → maj → dom → maj	251 (1.06%)
35. dom → dom → min → dom	247 (1.04%)
36. $\bullet \xrightarrow{/\text{perf5th}} \bullet \xrightarrow{/\text{perf4th}} \bullet \xrightarrow{/\text{perf5th}} \bullet$	245 (1.03%)
37. dom → min → dom → dom	241 (1.02%)
38. $\bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{maj7th}} \bullet \xrightarrow{\text{perf4th}} \bullet$	240 (1.01%)
39. $\bullet \xrightarrow{\text{perf5th}} \bullet \xrightarrow{\text{perf4th}} \bullet \xrightarrow{\text{perf4th}} \bullet$	238 (1.01%)
40. maj → dom → min → dom	236 (1.00%)

Table 4.3: Beatles root interval and chord category rules (whose coverage is larger than 1%) and the associated degree and chord category rules. C_1 and C_2 represent the positive coverage over all the Beatles songs and over the Beatles tonal songs with no modulation respectively.

	Rule	C_1	C_2
1.	maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj	3.13%	3.79%
	I maj $\rightarrow$ IV maj $\rightarrow$ I maj $\rightarrow$ IV maj	-	2.47%
	V maj $\rightarrow$ I maj $\rightarrow$ V maj $\rightarrow$ I maj	-	1.00%
2.	maj $\overset{\text{perf5th}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj	2.94%	3.61%
	IV maj $\rightarrow$ I maj $\rightarrow$ IV maj $\rightarrow$ I maj	-	2.43%
	I maj $\rightarrow$ V maj $\rightarrow$ I maj $\rightarrow$ V maj	-	0.84%
3.	maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj	1.38%	1.75%
	I maj $\rightarrow$ IV maj $\rightarrow$ V maj $\rightarrow$ I maj	-	1.59%
4.	maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj	1.21%	1.47%
	IV maj $\rightarrow$ V maj $\rightarrow$ I maj $\rightarrow$ IV maj	-	1.15%
5.	maj $\overset{\text{perf5th}}{\longrightarrow}$ maj $\overset{\text{min7th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj	1.04%	1.28%
	I maj $\rightarrow$ V maj $\rightarrow$ IV maj $\rightarrow$ I maj	-	0.69%
	IV maj $\rightarrow$ I maj $\rightarrow$ bVII maj $\rightarrow$ IV maj	-	0.52%
6.	maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj	0.93%	1.11%
	V maj $\rightarrow$ I maj $\rightarrow$ IV maj $\rightarrow$ V maj	-	1.03%
7.	maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj	0.91%	1.09%
	V maj $\rightarrow$ I maj $\rightarrow$ IV maj $\rightarrow$ I maj	-	0.83%

Table 4.4: Top ten Real Book harmony rules when considering root interval progressions and chord category progressions. C is the positive coverage.

Rule	C
1. maj $\overset{\text{maj6th}}{\longrightarrow}$ min $\overset{\text{perf4th}}{\longrightarrow}$ min $\overset{\text{perf4th}}{\longrightarrow}$ dom	190 (0.80%)
2. dom $\overset{\text{perf4th}}{\longrightarrow}$ min $\overset{\text{perf4th}}{\longrightarrow}$ dom $\overset{\text{perf4th}}{\longrightarrow}$ maj	176 (0.74%)
3. min $\overset{\text{perf4th}}{\longrightarrow}$ dom $\overset{\text{perf4th}}{\longrightarrow}$ min $\overset{\text{perf4th}}{\longrightarrow}$ dom	174 (0.73%)
4. min $\overset{\text{perf4th}}{\longrightarrow}$ min $\overset{\text{perf4th}}{\longrightarrow}$ dom $\overset{\text{perf4th}}{\longrightarrow}$ maj	171 (0.72%)
5. min $\overset{\text{perf4th}}{\longrightarrow}$ dom $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{maj6th}}{\longrightarrow}$ min	170 (0.72%)
6. dom $\overset{\text{perfU}}{\longrightarrow}$ min $\overset{\text{perf4th}}{\longrightarrow}$ dom $\overset{\text{perf4th}}{\longrightarrow}$ maj	133 (0.56%)
7. maj $\overset{\text{maj2nd}}{\longrightarrow}$ min $\overset{\text{perf4th}}{\longrightarrow}$ dom $\overset{\text{perf4th}}{\longrightarrow}$ maj	126 (0.53%)
8. min $\overset{\text{perf4th}}{\longrightarrow}$ dom $\overset{\text{perf5th}}{\longrightarrow}$ min $\overset{\text{perf4th}}{\longrightarrow}$ dom	124 (0.52%)
9. min $\overset{\text{perfU}}{\longrightarrow}$ min $\overset{\text{perfU}}{\longrightarrow}$ min $\overset{\text{perfU}}{\longrightarrow}$ min	124 (0.52%)
10. dom $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{maj6th}}{\longrightarrow}$ min $\overset{\text{perf4th}}{\longrightarrow}$ min	121 (0.51%)

match each root interval rule (with no tonal centre information) with the degree rules which are covered by it. The result of this matching process between degree and root interval rules is presented in Table 4.3 (top rules only). So for instance in Table 4.3 the instances of the root interval rule 5:

$$\text{maj} \overset{\text{perf5th}}{\longrightarrow} \text{maj} \overset{\text{min7th}}{\longrightarrow} \text{maj} \overset{\text{perf5th}}{\longrightarrow} \text{maj}$$

are for 54% of them instances of the degree rule:

$$\text{I maj} \rightarrow \text{V maj} \rightarrow \text{IV maj} \rightarrow \text{I maj}$$

and for 41%, instances of the degree rule:

$$\text{IV maj} \rightarrow \text{I maj} \rightarrow \text{bVII maj} \rightarrow \text{IV maj}$$

4.3.2 Characterisation of The Beatles vs. Real Book Songs

For the second experiment we want to know the Beatles chord sequences that are not present in the Real Book. Aleph is provided with all the Beatles chord sequences of length 4 as positive examples and all the Real Book chord sequences of length 4 as negative examples. It returns 1679 rules which characterise all the chord sequences that only appear in the Beatles songs. The top ten rules are shown in Table 4.5. Some of these rules are correlated. For instance the 3 chord cyclic pattern I-IV-V-I-IV-V-I..., very common in the early compositions of the Beatles (see for instance the song *Please Please Me* of the album *Please Please Me*), is covered by rules 1, 2 and 4. Similarly the cyclic pattern I-V-IV-I-V-IV-I... is covered by rules 3, 7 and 8. Note also that the "back and forth" pattern between the first and fourth degree or between the fifth and first degree mentioned by Mauch et al. (2007) and identified in rule 1 of Table 4.3 appears in rules 5 and 10 (and also to some extent in rule 9) of Table 4.5.

As in the previous experiment we also try to characterise the chord sequences in terms of root intervals and chord categories and obtain a set of 1520 rules. The top ten rules are shown in Table 4.6. The first seven rules were also in Table 4.5 and have been interpreted above. Additionally rule 8 in Table 4.6, can be interpreted as the so-called *fifties progression* (I)-vi-IV-V-I, where the first tonic is missing due to the 4-chord length constraint when building the rules. The I-vi-IV-V-I turnaround is extensively used in Western popular music, for instance in our case in the chorus of the Beatles' "Happiness Is a Warm Gun". Rule 9, which we interpret as IV-V-I-V, can be seen as a variant on the last four chords of a eight-bar blues: I-V-I-V. Finally rule 10 is again related to the cyclic progression I-IV-V-I-IV-V-I... mentioned in the previous paragraph.

Table 4.5: Top ten Beatles harmony rules when the Real Book is taken as the source of negative examples. C is the positive coverage.

Rule	C
1. maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj	188 (1.38%)
2. maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj	165 (1.21%)
3. maj $\overset{\text{perf5th}}{\longrightarrow}$ maj $\overset{\text{min7th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj	141 (1.04%)
4. maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj	126 (0.93%)
5. A maj $\rightarrow$ D maj $\rightarrow$ A maj $\rightarrow$ D maj	114 (0.84%)
6. maj$/_A \rightarrow$ maj$/_A \rightarrow$ maj$/_A \rightarrow$ maj$/_A$	110 (0.81%)
7. maj $\overset{\text{min7th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj	108 (0.79%)
8. maj $\overset{\text{perf5th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj $\overset{\text{min7th}}{\longrightarrow}$ maj	102 (0.75%)
9. maj $\overset{\text{perfU}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj	99 (0.73%)
10. D maj $\rightarrow$ G maj $\rightarrow$ D maj $\rightarrow$ G maj	92 (0.68%)

Table 4.6: Top ten Beatles root interval and chord category rules when the Real Book is taken as the source of negative examples. C represents the positive coverage over all the Beatles songs.

Rule	C
1. maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj	188 (1.38%)
2. maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj	165 (1.21%)
3. maj $\overset{\text{perf5th}}{\longrightarrow}$ maj $\overset{\text{min7th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj	141 (1.04%)
4. maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj	126 (0.93%)
5. maj $\overset{\text{min7th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj	108 (0.79%)
6. maj $\overset{\text{perf5th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj $\overset{\text{min7th}}{\longrightarrow}$ maj	102 (0.75%)
7. maj $\overset{\text{perfU}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj	99 (0.73%)
8. min $\overset{\text{min6th}}{\longrightarrow}$ maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj	63 (0.46%)
9. maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perf5th}}{\longrightarrow}$ maj	57 (0.42%)
10. maj $\overset{\text{maj2nd}}{\longrightarrow}$ maj $\overset{\text{perf4th}}{\longrightarrow}$ maj $\overset{\text{perfU}}{\longrightarrow}$ maj	54 (0.40%)

4.3.3 Considerations About the Size of the Corpora and the Computation Time

Such an ILP approach has never been applied on such a scale: we dealt with data-sets a musicologist would typically be interested in studying (unified corpora of songs commonly accepted as representative of a composer/band/genre).

Although ILP systems are usually known to be resource intensive, the computation time of the ILP system was not a limiting factor in this case. Aleph computed all the rules in less than a minute on a regular desktop computer. We see our framework as a useful tool for musicologists since manual harmonic annotation and analysis of a whole musical corpus

can take several years of musicological work[3] whereas the automatic extraction of the chord progression patterns using ILP takes only seconds, allowing the user to concentrate on the interpretation of the results.

4.4 Discussion and Conclusions

If the computation time of our models was fast with the representation scheme described in this chapter, it however was not true for the less specific representation paradigms we first tried feeding Aleph with. In ILP both the concept and the vocabulary to describe it (or background knowledge) need to be defined in advance. Thus, two different vocabularies result in different descriptions of the concept. The vocabulary described in this chapter – which is the result of an iterative process, during which we manually refined the vocabulary further at each step until the resulting rules were meaningful and the computation time was reasonable – is very specific: the concept, "chord sequence (of length 4) in a corpus" (`chord_prog_4/4`), can be described only in terms of "chord category sequence of length 4" (`category_prog_4/8`), "root interval sequence of length 3" (`rootInterval_prog_3/7`), etc. The major restriction of that representation is that the length of the rules is necessarily fixed (always to the same length $n = 4$ throughout the whole set of rules or model). Ideally we want to be able to have patterns of all lengths in our models, e.g. some of length 4, some of length 3, some of length m, whatever fits best the data (and not whatever works best for the induction system). We tried to overcome this restriction in our earlier attempts by using only low level concepts, e.g. using independent descriptions of each chord, linked only by a "predecessor" predicate, without any sequence-related predicates in our vocabulary. But it failed to provide meaningful descriptions of the target concept in a reasonable amount of time (or at all). As we were refining the vocabulary we felt we were inevitably reducing the problem to a pattern matching task and not to a pattern discovery task as we had intended, allowing us to validate or refute hypotheses about the concept but not to make any really novel (knowledge) discoveries. In other words, with our restricted vocabulary in this approach we did not make use of the recursive power of ILP as it turned out to be too computationally expensive. That led us to move away from Aleph to a computationally more

[3] for instance Alan Pollack's analysis of the full corpus of the Beatle's songs took over 10 years to complete: `http://www.icce.rug.nl/~soundscapes/DATABASES/AWP/awp-notes_on.shtml`

powerful ILP system called TILDE (Blockeel and De Raedt, 1998). Additionally since TILDE builds classification models, the evaluation of their accuracy allowed us to quantitively assess the performance of our approach. We also changed our representation scheme, allowing to describe the concepts with patterns of flexible length and introducing the possibility of having gaps in the harmony patterns. Those experiments are described in the next chapter.

CHAPTER 5

AUTOMATIC GENRE CLASSIFICATION

5.1 Introduction

In Chapter 4 we qualitatively evaluated the characterisation power of our harmony-based models by examining them and describing their musicological relevance. Unfortunately this method does not scale when comparing multiple representations and learning algorithms. In this chapter we therefore perform classification in order to quantitatively evaluate our models. We also introduce a new flexible harmonic representation paradigm based on context-free definite-clause grammars in the form of variable- and arbitrary-length chord sequences containing gaps (Section 5.2.1). To support classification and more demanding computations we use a new induction system, TILDE, which builds ILP decision-trees and random forests (Section 5.2.2). With an additional dataset, the *Perez-9-genres* Corpus (Section 5.2.3), we not only study symbolic but also audio data. To that end a chord transcription step is performed (Sections 5.2.4 and 5.2.5). We run experiments on several subsets of the dataset requiring advanced statistical comparison tools (Section 5.3.1). In these experiment we compare several representation schemes for the harmonic steps (Section 5.3.2), as well as evaluate the performance of our method on symbolic and audio data (Section 5.3.3). We experiment further by extending our decision tree induction approach to random forests (Section 5.3.4). Finally, we inspect and conduct a musical analysis of some the rules extracted from our decision-tree models (Section 5.3.5).

5.2 Learning Harmony Rules

5.2.1 Representing Harmony with Context-Free Definite-Clause Grammars

Characteristic harmony patterns or rules often relate to chord progressions, i.e. sequences of chords. However, not all chords in a piece of music are of equal significance in harmonic patterns. For instance, ornamental chords (e.g. passing chords) can appear between more relevant chords. Moreover, not all chord sequences, even when these ornamental chords are

removed, can be typical of the genre of the piece of music they are part of: some common chord sequences are found in several genres, such as the perfect cadence (moving from the fifth degree to the first degree) which is present in all tonal Classical music periods, jazz, pop music and numerous other genres. Thus, the chord sequences to look for in a piece of music as hints to identify and characterise its genre are sparse, can be punctuated by ornamental chords, might be located anywhere in the piece of music, and additionally, they can be of any length. Our objective is to describe these distinctive harmonic sequences of a style. To that end we adopt a context-free definite-clause grammar representation which proved to be useful for solving a structurally similar problem in the domain of biology: the logic-based extraction of patterns which characterise the neuropeptide precursor proteins (NPPs), a particular class of amino acid sequences (Muggleton et al., 2001). Indeed NPPs share common characteristics with musical pieces (represented as chord sequences): these sequences are highly variable in length, they tend to show almost no overall sequence similarity and the class (NPPs or non-NPPs in the case of amino acids sequences, musical genres in the case of pieces of music) to which a given sequence belongs is not always clear (some NPPs have not yet been discovered and experts can disagree on the genre of a given piece). Both because of these similarities in the data and because context-free definite-clause grammars can be induced using Inductive Logic Programming, we choose to adopt this representation scheme.

In this formalism we represent each piece of music as the list or sequence of chords it contains and each genre as a set of music pieces. We then look for a set of harmony rules describing characteristic chord sequences present in the musical pieces of each genre. These rules define a Context-Free Grammar (CFG). In the linguistic and logic fields, a CFG can be seen as a finite set of rules which describes a set of sequences. Because we are only interested in identifying the harmony sequences characterising a genre, and not in building a comprehensive chord grammar, we use the concept of 'gap' (of unspecified length) between sub-sequences of interest to skip ornamental chords and non-characteristic chord sequences in a musical piece, as done by Muggleton et al. (2001) when building their grammar to describe NPPs. Notice that like them, to automate the process of grammar induction we also adopt a Definite Clause Grammar (DCG) formalism to represent our Context-Free Grammars as logic programs, and use Inductive Logic Programming (ILP), which is concerned with the inference of logic programs (Muggleton, 1991).

We represent our DCGs using the *difference-list* representation, and not the DCG representation itself, as this is what TILDE, the inference system we use, returns (more details in the following section). In our formalism the terminal symbols are the chords labelled in a

jazz/pop/rock shorthand fashion (e.g. G7, D♭, BM7, F#m7, etc.). Properties of the chords are described using predicates (i.e. operators which return either *true* or *false*). In the difference-list representation these predicates take at least two arguments: an input list, and an output list. The predicate and the additional arguments (if there are any) apply to the difference between the input list and the output list (which could be one or several elements). For instance, `degree(1,[cmaj7,bmin,e7],[bmin,e7],cmajor)` says that in the key of C major (last argument, `cmajor`) the chord Cmaj7 (difference between the input list `[cmaj7,bmin,e7]` and the output list `[bmin,e7]`) is on the tonic (or first degree, `1`).

The predicates that can be used by the system for rule induction are defined in the background knowledge:

- for each chord in a chord sequence its root note is identified using the `rootNote/4` predicate: `rootNote(Root,InputList,OutputList,Key)` (the key which is not needed to define the root note is included in this predicate in order to define the degree in the `degree/4` predicate, cf. Table 5.1);
- the root interval between two chords is defined using the `rootInterval/3` predicate: `rootInterval(Interval,InputList,OutputList)`;
- degrees are expressed with the `degree/4` predicate which definition is based on the `rootNote/4` predicate (cf. Table 5.1): `degree(Degree,Root,InputList,OutputList,Key)`;
- chord categories (e.g. min, 7, maj7, dim, etc.) are identified using the `category/3` predicate: `category(ChordCategory,InputList,OutputList)`;
- degrees and categories are united in a single predicate `degreeAndCategory/5`: `degreeAndCategory(Degree,ChordCategory,InputList,OutputList)`;
- the `gap/2` predicate matches any chord sequence of any length, allowing to skip uninteresting subsequences (not characterised by the grammar rules) and to handle large sequences for which otherwise we would need larger grammars and more training data: `gap(InputList,OutputList)`.

Figure 5.1 illustrates how a piece of music, its chords and their properties are represented in our formalism, when using only the `degreeAndCategory/5` and `gap/2` predicates (other predicates from our formalism could be used in a similar way).

Additionally notice that some of those predicates are defined from others, and these definitions are also provided in the background knowledge. Table 5.1 provides a snippet of such a background knowledge.

Table 5.1: Background knowledge used in the first-order logic decision tree induction algorithm.

```
% Notation conventions:
% - Sharps are represented by 's', while flats are represented by 'b'
% - Chord categories are those used in jazz shorthand notation, e.g. min is minor, maj is major
% - Chord symbols are represented by the root note, any accidental affecting the root note,
% and the chord category, aggregated together in one term without space or symbols in between,
% e.g. csmin represents a C# minor chord
% - As it is the case in jazz shorthand notation major is omitted in a chord name,
% e.g. c represents a C major chord
% - To distinguish the terms used for the chords and those for the root notes,
% an underscore '_' is used between the note and its potential accidental,
% e.g. c_ is the note C, c_s is the note C#
% - Tonalities are represented by the juxtaposition of the tonic and the mode in full,
% e.g. cmajor is the tonality C major
% - Degrees are represented by a number which similarly to the root notes can be affected by
% a accidental attached to it with an underscore,
% e.g. 1_ is the tonic
% - Intervals are represented by abbreviations of their quality (min, maj, etc.) followed by
% an underscore and an abbreviation of their diatonic number (sec, third, etc.),
% e.g. maj_sec is a major second

rootNote(c_,[c|T],T,Key).
rootNote(c_,[cmin|T],T,Key).
rootNote(c_s,[cs|T],T,Key).
rootNote(c_s,[csmin|T],T,Key).
…
category(min,[cmin|T],T).
category(maj,[c|T],T).
category(min,[csmin|T],T).
category(maj,[cs|T],T).
…
degree(1_,A,B,cmajor) :- rootNote(c_,A,B,cmajor).
degree(1_s,A,B,cmajor) :- rootNote(c_s,A,B,cmajor).
…
degreeAndCategory(Deg,Cat,A,B,Key) :- degree(Deg,A,B,Key), category(Cat,A,B).

rootInterval(min_sec,A,B):- rootNote(c_,A,B,_), rootNote(d_b,B,C,_).
rootInterval(maj_sec,A,B):- rootNote(c_,A,B,_), rootNote(d_,B,C,_).
…
gap(A,A).
gap([_,A],B) :- gap(A,B).
```

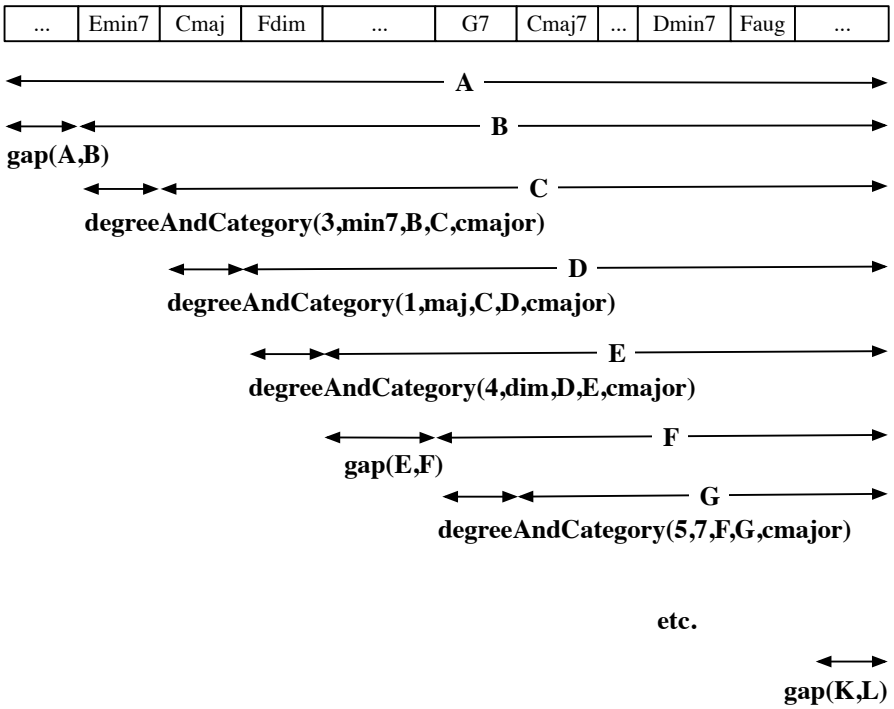

Figure 5.1: A piece of music (i.e. list of chords) assumed to be in C major, and its Definite Clause Grammar (difference-list Prolog clausal) representation.

5.2.2 Learning Algorithm

To induce the harmony grammars we apply the ILP decision tree induction algorithm TILDE (Blockeel and De Raedt, 1998) described in chapter 3. Each tree built by TILDE is an ordered set of rules which is a genre classification model (i.e. which can be used to classify any new unseen musical piece represented as a list of chords) and describes the characteristic chord sequences of each genre in the form of a grammar. The system takes as learning data a set of triples ($chord\_sequence$, $tonality$, $genre$), $chord\_sequence$ being the full list of chords present in a musical piece, $tonality$ being the global tonality of this piece and $genre$ its genre.

TILDE does not build sets of grammar rules for each class but first-order logic decision trees expressed as ordered sets of rules (or Prolog programs). Each rule is simply a program following the path from the root of the tree to one of its leaves. Each tree covers one classification problem (and not one class), so in our case rules describing harmony patterns of a given genre coexist with rules for other genres in the same tree (or set of rules). That is why the ordering of the rules we obtain with TILDE is an essential part of the classification: once a rule describing genre g is fired on an example e then e is classified as a piece of genre g and the following rules in the grammar are not tested over e. Thus, the rules of a model can not be used independently from each other.

In the case of genre classification, the target predicate given to TILDE, i.e. the one we want to find rules for, is `genre/4`, where `genre(G,A,B,Key)` means the piece `A` (represented as its full list of chords) in the tonality `Key` belongs to genre `G`. The output list `B` (always an empty list), is necessary to comply with the difference-list representation. The training dataset provided to TILDE contains several values for `G` (all the genres of the current classification problem) and several examples (i.e. input list `A`) per genre. We constrain the system to use at least two consecutive `degree` or two `degreeAndCategory` predicates or one `rootInterval` predicate between any two `gap` predicates. This guarantees that we are considering local chord sequences of at least length 2 (but also larger) in the pieces of music. Here is an example in Prolog notation of a grammar rule built by TILDE for Classical music (extracted from an ordered set containing rules for several genres):

```
genre(classical,A,Z,Key) :-
        gap(A,B), degreeAndCategory(2,7,B,C,Key),
        degreeAndCategory(5,maj,C,D,Key),
        gap(D,E), degreeAndCategory(1,maj,E,F,Key),
        degreeAndCategory(5,7,F,G,Key), gap(G,Z).
```

Which can be translated as : *"Some Classical music pieces contain a dominant 7th chord on the supertonic (II) followed by a major chord on the dominant, later (but not necessarily directly) followed by a major chord on the tonic followed by a dominant 7th chord on the dominant"*.
Or: *"Some Classical music pieces can be modelled as: ... II7 - V ... I - V7 ..."*.
Thus, complex rules combining several local patterns (of any length greater than or equal to 2) separated by gaps can be constructed with this formalism.

A simple example illustrating the induction of a decision tree in TILDE for a 3-genre classification problem is provided in Figure 5.2.

For each classification task we perform a 5-fold cross-validation (except where explicitly stated otherwise). We adopt the best minimal coverage of a leaf learned from previous experiments: we constrain the system so that each leaf in each constructed tree covers at least five training examples. By setting this TILDE parameter to 5 we avoid any overfitting – as a smaller number of examples for each leaf means a larger number of rules and more specific rules – and in the same time it is still reasonable given the size of the dataset – a larger value would have been unrealistically too large for the system to learn any tree, or would have required a long computation time for each tree.

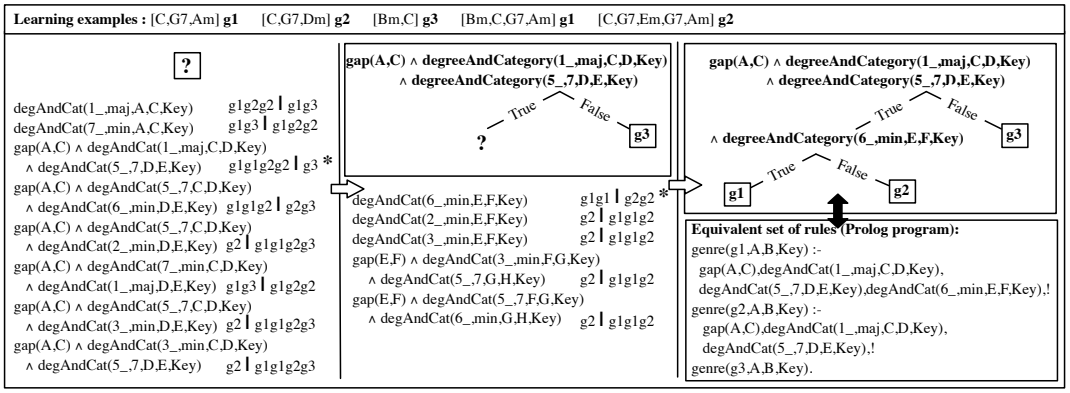

Figure 5.2: Schematic example illustrating the induction of a first-order logic tree for a 3-genre classification problem. The three genres (g1, g2 and g3) are illustrated with the 5 learning examples at the top. At each step the partial tree (top) and each literal (or conjunction of literals) considered for addition to the tree (bottom) are shown together with the split resulting from the choice of this literal (e.g. g1g1g2|g2 means that two examples of g1 and one of g2 are in the left branch and one example of g2 is in the right branch). The literal resulting in a the best split is indicated with an asterisk. The final tree and the equivalent ordered set of rules (or Prolog program) are shown on the right. The key is C Major for all examples. For space reasons `degAndCat` is used to represent `degreeAngCategory`.

5.2.3 Dataset

The data used to train our harmony-based genre classifier has been collected, annotated and provided by the Pattern Recognition and Artificial Intelligence Group of the University of Alicante. It consists of a collection of Band in a Box files (format as described in Section 2.2.2) covering three genres – popular, jazz, and "academic" music and has sometimes been referred to as the *Perez-9-genres* Corpus (Pérez-Sancho, 2009). We use our own internal Band in a Box converter[1] to extract the lists of shorthand chord symbols from this proprietary format.

In the *Perez-9-genres* dataset the popular music set contains pop, blues, and Celtic (mainly Irish jigs and reels) music; jazz consists of a pre-bop class grouping swing, early, and Broadway tunes, bop standards, and bossanovas; and "academic"[2] music consists of Baroque, Classical and Romantic period music. All the categories were defined by music experts who also collaborated in the task of assigning meta-data tags to the files and rejecting outliers. The total amount of pieces used for our experiments is 856 (Baroque period 56; Classical period 50; Romantic period 129; pre-bop 178; bop 94; bossanova 66; blues 84; Celtic 99; pop 100).

The *Perez-9-genres* dataset also contains 856 audio files generated from the Band in a Box

[1] developed by Simon Dixon and Matthias Mauch, with some help from Bas de Haas (from Universiteit Utrecht)

[2] although "Classical" would be a more appropriate name for this genre we kept the name "academic" to match the original name given by the dataset's authors because it has been used in several other publications including (Pérez-Sancho et al., 2009), (Pérez-Sancho, 2009) and (Pérez-Sancho et al., 2010). This also avoids confusion with the Classical period (1750-1820).

files according to the following workflow: the Band in a Box software allows for "exportation with arrangement" of its files into MIDI files that were then synthesised into audio files using the TIMIDITY++[3] software. Both the MIDI and the audio files generated in the process contain interpretations (i.e. expressive performances) of the original pieces thanks to the Band in a Box 'style' parameter (that can take on hundreds of values) controlling the arrangement, rhythmic interpretation, etc. of each piece of music.

Additionally since Band in a Box exports the repetitions in the MIDI files too, these are also played in the audio files generated with TIMIDITY++. To match the symbolic and audio file, our internal converter can also detect repetitions in the Band in a Box file format and write the duplicated chord sequences in the symbolic files. Hence all files, symbolic and audio, contain repetitions in our experiments.

The classification tasks that we are interested in are relative to the three main genres of this dataset: academic, jazz and popular music. For all our experiments we consider each time the 3-way classification problem and each of the 2-way classification problems. In addition we also study the 3-way classification problem dealing with the popular music subgenres (blues, Celtic and pop music). We do not work on the academic subgenres and jazz subgenres as these two datasets contain unbalanced subclasses. Because of this last characteristic removing examples to get the same number of examples per class would lead to poor models built on too few examples. Moreover resampling can not be used as TILDE automatically removes identical examples. Finally for comparison purposes we also consider the 9-way classification problem dealing with all the subgenres at once.

5.2.4 Chord Transcription Algorithm

To extract the chords from the synthesised audio dataset an automatic chord transcription algorithm is needed. We use an existing automatic chord labelling method, which can be broken down into two main steps: generation of a beat-synchronous chromagram and an additional beat-synchronous bass chromagram, and an inference step using a musically motivated dynamic Bayesian network (DBN). The following paragraphs provide an outline of these two steps. Please refer to (Mauch, 2010, Chapters 4 and 5) for details.

The chroma features are obtained using a prior approximate note transcription based on the non-negative least squares method (NNLS). First a log-frequency spectrogram (similar to a constant-Q transform) is calculated, with a resolution of three bins per semitone. As is

[3] http://timidity.sourceforge.net

frequently done in chord- and key- estimation (e.g. Harte and Sandler, 2005), this spectrogram is adjusted to compensate for differences in the tuning pitch. The tuning is estimated from the relative magnitude of the three bin classes. Using this estimate, the log-frequency spectrogram is updated by linear interpolation to ensure that the centre bin of every note corresponds to the fundamental frequency of that note in equal temperament. The spectrogram is then updated again to attenuate broadband noise and timbre. To determine note activation values a linear generative model is assumed in which every frame Y of the log-frequency spectrogram can be expressed approximately as the linear combination $Y \approx Ex$ of note profiles in the columns of a dictionary matrix E, multiplied by the activation vector x. Finding the note activation vector that approximates Y best in the least-squares sense subject to $x \geq 0$ is called the non-negative least squares problem (NNLS). A semitone-spaced note dictionary with exponentially declining partials, and the NNLS algorithm proposed by Lawson and Hanson (Lawson and Hanson, 1974) to solve the problem and obtain a unique activation vector are used. For treble and bass chroma mapping different profiles are chosen: the bass profile emphasises the low tone range, and the treble profile encompasses the whole note spectrum, with an emphasis on the mid range. The weighted note activation vector is then mapped to the twelve pitch classes C,...,B by summing the values of the corresponding pitches. In order to obtain beat times an existing automatic beat-tracking method (Davies et al., 2009) is used. A beat-synchronous chroma vector can then be calculated for each beat by taking the median (in the time direction) over all the chroma frames whose centres are situated between the same two consecutive beat times.

The two beat-synchronous chromagrams are now used as observations in the DBN, which is a graphical probabilistic model similar to a hierarchical hidden Markov model. The DBN jointly models metric position, key, chords and bass pitch class, and parameters are set manually according to musical considerations. The most likely sequence of hidden states is inferred from the beat-synchronous chromagrams of the whole piece using the BNT[4] implementation of the Viterbi algorithm (Rabiner, 1989). The method detects the 24 major and minor keys and 121 chords in 11 different chord categories: major, minor, diminished, augmented, dominant 7th, minor 7th, major 7th, major 6th, and major chords in first and second inversion, and a 'no chord' type. The chord transcription algorithm correctly identifies 80% (correct overlap, Mauch, 2010, Chapter 2) of the chords in the MIREX audio chord recognition dataset and was rated first in its category in the 2009 and 2010 MIREX evaluations.

[4] http://code.google.com/p/bnt/

5.2.5 Data Post-Processing

To match the symbolic data (i.e. coming from the Band in a Box files) to the audio transcriptions we can get from the chord transcription algorithm, the extensive set of chord categories found in the Band in a Box dataset is reduced to eight categories before training: major, minor, diminished, augmented, dominant 7th, minor 7th, major 7th, major 6th. This reduction is done by mapping each category to the closest one in term of both number of intervals shared and musical function. Similarly, for the audio transcription dataset, since chord inversions and "no chord" are not used in the Band in a Box dataset we replace after transcription the major chords in first and second inversion with major chords and the sections with no chords are simply ignored. Finally in both datasets repeated chords are merged to a single instance of the chord.

5.3 Experiments and Results

In this section we describe the set of experiments we performed on the *Perez-9-genres* dataset, with the goal to assess the best knowledge representation scheme in various conditions. We also compare our results with related work performed on the same dataset (Pérez-Sancho, 2009). Notice that the results presented here are better than those we initially reported in (Anglade et al., 2009b) and (Anglade et al., 2009c). The differences can be explained by the use of 10-fold cross-validation in the experiments reported here vs. 5-fold cross-validation then, but also a richer chord vocabulary (allowing diminished and augmented triads) both in symbolic and audio data due to the use of a new, more accurate, chord transcription algorithm from Mauch (2010), when we previously used the one from Gómez (2006).

5.3.1 Statistical Considerations

In the following sections we will compare several approaches to the problem of genre classification (changing the knowledge representation for instance). As such we are performing a multi-classifier comparison, which is analogous to comparing multiple algorithms over multiple datasets, the null hypothesis being that they all perform equally well on those datasets. Arguably we only use one dataset, however since we are considering several classification tasks

with different classes (3-main-genre, 2-main-genre, 9-subgenre and one of the 3-subgenre problems) we can still consider those are separate datasets due to the difference in focus when building the classification rules for each one of them (leading to different rules themselves). The musicological interest in looking at sub classification tasks is the following: in the case of the 2-genre classification tasks for instance we are looking at core differences between genres, while the 3-main-genre and 9-subgenre classification rules will highlight distinguishing features of each genre (in comparison to all others). It is nonetheless noticeable that these datasets are not entirely independent, but can be considered as such, which allows us to apply statistical significance tests (described below) and get some indication of the significance of the results.

The widely used t-test (Student, 1908), as described by Dietterich (1998), "has an elevated probability of type I error", usually higher than the target level, even when used in conjunction with 10-fold cross-validation. Furthermore as explained in (Demšar, 2006) we do not fall into the trap of performing all pairwise t-tests. Indeed this implies running a large number of tests which increases the chances of rejecting the null hypothesis due to random chance, a phenomenon known as the *multiplicity effect*. As suggested by Salzberg (1997), the Bonferroni correction would be a way to deal with this but Demšar (2006) warns against its conservatism and weakness and recommends "more powerful specialized procedures" for comparing multiple classifiers over multiple datasets, such as the repeated-measures ANOVA (Fisher, 1956) and the Friedman test (Friedman, 1937).

If ANOVA is the most commonly used statistical test for comparing more than 2 algorithms, it was pointed out by Demšar (2006) that it is based on assumptions that are usually violated when considering classification: assumptions of normality of the distribution and sphericity, which "due to the nature of the learning algorithms and data sets [...] cannot be taken for granted". He recommends to use instead the non-parametric Friedman test, which by definition does not rely on a particular data distribution.

Additionally, once the null-hypothesis is rejected by the Friedman test, a post-hoc test can be applied to identify which pairwise differences between classifiers performances are statistically significant. In our case we do not have a control classifier, to which all other classifiers can be compared, so we need a post-hoc test to compare all classifiers to each other. Demšar (2006) suggests the Nemenyi test (Nemenyi, 1963) to perform all pairwise comparisons. But as pointed out by García and Herrera (2008), this test is "very conservative and it may not find any difference in most of the experimentations". This is why we here follow García and Herrera's suggestion and use the most powerful of the post-hoc tests that they compare in their paper: Bergmann and Hommel procedure (Bergmann and Hommel, 1988).

García and Herrera also provide a script[5] that we will use which computes Friedman's and Iman-Davenport's statistics as well as adjusted p-values based on the post-hoc test. In statistics, a p-value indicates if a statistical test is significant by comparing it to a significance level α, to which it needs to be smaller. The smaller the more statistically significant. Adjusted p-values (Wright, 1992) are modified p-values so that the multiple comparisons are taken into account, reflecting the process of the post-hoc test it depends on itself. Similarly to p-values, adjusted p-values can then be directly compared to the significance level α. We will use $\alpha = 0.05$ except when stated otherwise.

In each of the following experiments we perform 10-fold cross-validation. The folds were separately built with a script performing random selection and not directly using TILDE. That allowed us to save those folds and always use the same ones across all experiments. Note however that we started off building separate sets of folds for the symbolic data and for the audio data, as those experiments were not supposed to be linked and it was sometimes difficult to match files between datasets. So the folds for those two kinds of experiments do not match. We have nonetheless later on built a second set of folds from the audio data that matches the folds of the symbolic data, referred to as "Audio Examples by Symbolic Folds". This set of folds was used when evaluating the symbolic models on audio data (cf. Section 5.3.3) hence making sure none of the example musical pieces used for training the symbolic models were present (in their audio representation) in the evaluation (or test) folds.

In the following section the tables contain, for each knowledge representation and for each classification problem, the average and standard deviation over the 10 folds of the following measures: accuracy, run time, number of nodes in the tree, number of literals in the tree. In some cases (identified by '(*)' in the tables) computing models for some folds was taking an extremely long time and the computation was stopped. The previous measures were then averaged and their standard deviation were computed over the folds for which models were built, and the number of folds on which these measures were computed is indicated in parentheses.

5.3.2 Choosing the Knowledge Representation

As described in Section 5.2.1 we consider several knowledge representation schemes: harmonic steps are represented as sequences of intervals between root notes, sequences of degrees, or sequences of degrees and chord categories. These 3 representations will be referenced in the rest of this chapter as *Root Interval*, *Degree*, *Degree & Category* respectively.

[5] `http://sci2s.ugr.es/keel/multipleTest.zip`

To compare these 3 representations we first run all 6 classification problems described in Section 5.2.3 on clean data, i.e. on the manual chord transcriptions contained in the Band in a Box files. The results of those experiments on symbolic data are shown in Table 5.2.

Table 5.2: Classification results on symbolic data using 10-fold cross-validation. b is the baseline. Values are average ± standard deviation over the folds. Highest accuracy is shown in **bold**. (*): Experiments stopped after one of the iterations (folds) run for too long; these results were averaged over a few folds (exact number of folds is provided) only and are only given as indications.

	Root Interval	Degree	Deg. & Cat.
3 main genres			
Accuracy (b = 0.395)	0.706 ± *0.047*	0.695 ± *0.060*	**0.834** ± *0.038*
Runtime (in CPU seconds)	141,429 ± *108,452*	20,610 ± *36,681*	3,744 ± *683*
# nodes in the tree	33.7 ± *4.4*	38.8 ± *8.1*	26.9 ± *3.1*
# literals in the tree	55.5 ± *8.6*	116.4 ± *24.3*	80.3 ± *8.8*
academic/jazz			
Accuracy (b = 0.590)	0.845 ± *0.041*	0.846 ± *0.025*	**0.949** ± *0.016*
Runtime (in CPU seconds)	2,256 ± *166*	419 ± *53*	544 ± *88*
# nodes in the tree	11.3 ± *2.2*	19.3 ± *3.0*	8.3 ± *1.1*
# literals in the tree	17.9 ± *3.3*	57.9 ± *9.1*	24.7 ± *3.0*
academic/popular			
Accuracy (b = 0.545)	0.750 ± *0.067*	0.795 ± *0.049*	**0.855** ± *0.073*
Runtime (in CPU seconds)	59,629 ± *105,742*	689 ± *339*	896 ± *322*
# nodes in the tree	17.3 ± *3.4*	15.0 ± *3.1*	11.2 ± *1.3*
# literals in the tree	29.9 ± *5.9*	45.0 ± *9.4*	33.6 ± *4.0*
jazz/popular			
Accuracy (b = 0.547)	0.819 ± *0.045*	0.801 ± *0.035*	**0.895** ± *0.033*
Runtime (in CPU seconds)	5,806 ± *9,499*	450 ± *126*	1,470 ± *272*
# nodes in the tree	17.7 ± *4.1*	13.5 ± *2.5*	12.5 ± *1.5*
# literals in the tree	31.6 ± *6.3*	40.5 ± *7.4*	37.5 ± *4.5*
blues/Celtic/pop			
Accuracy (b = 0.355)	0.709 ± *0.100*	0.730 ± *0.060*	**0.762** ± *0.062*
Runtime (in CPU seconds)	479 ± *56*	115 ± *142*	189 ± *29*
# nodes in the tree	10.1 ± *1.6*	13.5 ± *2.7*	16.1 ± *1.9*
# literals in the tree	18.1 ± *3.1*	40.5 ± *8.1*	47.7 ± *5.1*
9 subgenres	(1 fold)	(4 folds)	
Accuracy (b = 0.208)	(0.447*)	(0.437* ± *0.055**)	**0.542** ± *0.054*
Runtime (in CPU seconds)	(905,792*)	(46,266* ± *74,871**)	80,505 ± *51,237*
# nodes in the tree	(64.0*)	(63.8* ± *6.9**)	68.5 ± *3.9*
# literals in the tree	(97.0*)	(191.3* ± *20.8**)	201.3 ± *11.5*

For all classification problems the Degree & Category representation obtains the highest accuracy. Furthermore we perform a statistical significance analysis. García and Herrera's script provides the following average ranking values for each knowledge representation: 2.5 (Root Interval), 2.5 (Degree) and 1 (Degree & Category). Friedman's statistics is $\chi_F^2 = 9.0$ from which Iman-Davenport's statistics is computed: $F_F = 15.0$. With 3 algorithms and 6 datasets F_F is distributed according to the F distribution with $3-1=2$ and $(3-1)(6-1)=10$

degrees of freedom. The critical value of $F(2, 10)$ for $\alpha = 0.05$ is 4.10 so we can reject the null hypothesis that all algorithms are equivalent and can proceed with the post-hoc Bergmann and Hommel test.

García and Herrera's script computes the following adjusted p-values based on Bergmann and Hommel's procedure: 0.028 (Root Interval vs. Degree & Category), 0.028 (Degree vs. Degree & Category) and 1.0 (Root Interval vs. Degree). So the two first hypotheses are rejected since their adjusted p-values are below the statistical significance level $\alpha = 0.05$, meaning that Degree & Category perform statistically significantly better than the Root Interval and Degree representations.

Additionally if we look at the runtimes for each representation and classification problem, we can see that there is no fixed time cost. The figures in Table 5.2 are only examples of what the runtimes could be, as the sometimes large standard deviation values attest (especially for the 9-genre classification problem). What this shows is that the computation time of a model is very much dependant on the specific examples and the order in which they are processed by the system. However notice that in average the Degree & Category models are often computed more quickly than the Root Interval representation, while computation times are comparable for Degree and Degree & Category. The Degree & Category representation is also the only one for which all folds in the 9-genre classification problem were processed fully, and the other representations yield to very lengthy computation times for that same problem and had to be stopped.

Moreover as the number of nodes and literals present in a tree gives an estimation of its complexity, we can compare the complexity of the models resulting from the various representations. As can be seen in Table 5.2, all models need a few number of nodes and literals per tree: in average 14 nodes and 35 literals for the 2-class problems, 33 nodes and 80 literals for the 3-main-genre problem, and 65 nodes and 163 literals for the 9-class problem. Additionally all representation schemes on symbolic data result in similarly simple trees. This is interesting as the Degree & Category representation contains more information than the other two (the category), so the rules are inherently more specific which could result in models containing more rules to cover all possible cases. But this is not the case, suggesting that this representation fits the data more closely.

We obtain respectively 83.4% (academic/jazz/popular), 94.9% (academic/jazz), 85.5% (academic/popular), 89.5% (jazz/popular), 76.2% (blues/Celtic/pop) and 54.2% (9 subgenres) accuracy. The confusion matrices for the 6 classification tasks when using the Degree & Category representation are shown in Tables 5.3 and 5.4. The best results are obtained when

trying to distinguish jazz from another genre (academic or popular). The biggest difficulty that appears in both the 3-class task and the 2-class tasks is to distinguish academic music from popular music. Indeed the harmony of these two genres can be very similar, whereas jazz music is known for its characteristic chord sequences, very different from other genres' harmonic progressions. A close look at the 9-subgenre classification results in Table 5.4 show that classification errors between popular and academic music (and vice-versa) can be further refined as classification errors between Romantic and pop music.

Table 5.3: Confusion matrices (test results aggregated over all folds) for all main-genre classification problems using the Degree & Category representation on the symbolic dataset.

Real/Predicted	academic	jazz	popular	Total
academic	189	3	43	235
jazz	3	296	39	338
popular	29	25	229	283
Total	221	324	311	856
academic	221	14		235
jazz	15	323		338
Total	236	337		573
academic	192		43	235
popular	32		250	282
Total	224		293	517
jazz		321	17	338
popular		48	232	280
Total		369	249	618

Let us summarize our findings and compare the 3 knowledge representation schemes we evaluated here:

- The root interval representation is the simplest to implement as it does not require key estimation. However it is ambiguous; one unique root interval covers several degree sequences. This ambiguity might be the reason for the greater complexity and computation time of the models using it (as seen before) since the harmony sequences can not fully be captured by this representation.

- The Degree representation is one step closer to solving the ambiguity, since the degree is explicitly stated. However since the tonality is not stated in the models, the problem of distinguishing between minor and major thirds is still there; chord categories on every degree differ in minor and major keys. So this representation is still ambiguous while it nonetheless requires key estimation, so additional information or computation.

Table 5.4: Confusion matrices (test results aggregated over all folds) for 9-subgenre and popular subgenres classification problems using the Degree & Category representation on the symbolic dataset.

Real/Pred.	Baroq.	Class.	Rom.	pre-bop	bop	bossa.	blues	Celtic	pop	Total
Baroque	10	11	23	0	0	0	1	3	8	56
Classical	3	24	16	0	0	0	1	4	2	50
Romantic	11	18	68	1	1	0	4	10	16	129
pre-bop	0	0	1	144	13	16	0	0	4	178
bop	1	0	1	32	24	8	3	12	13	94
bossanova	0	0	1	24	17	19	0	2	3	66
blues	0	0	4	3	5	0	59	7	6	84
Celtic	0	0	5	0	3	0	1	81	9	99
pop	3	1	16	5	9	4	4	23	35	100
Total	28	54	135	209	72	47	73	142	96	856
blues							66	1	16	83
Celtic							1	81	17	99
pop							11	21	68	100
Total							78	103	101	282

- Finally the Degree & Category representation is the only non-ambiguous representation of all three. Indeed the category captures the difference between the minor and major triads. It also allows to capture chords that do not belong to the main key (e.g. due to local modulations) which would not be possible with the Degree representation, even when adding the main key to the Degree models. Additionally there is no limit to the number of chord categories one can use with the Degree & Category representation, while the Degree representation does not make any distinction between chords of different category types on the same degree (e.g. maj and maj7 chords).

So for both musicological and statistical reasons the Degree & Category representation seems to be the best one to tackle the problem of genre classification.

To conclude this section we compare the classification results obtained with our context-free definite-clause grammar models and those obtained by the creators of the dataset, namely Pérez-Sancho et al. who applied n-gram classifiers on the same data (cf. Section 2.3.2 for more details on their approach). We use the results presented in (Pérez-Sancho et al., 2010) focusing here on the "groundtruth classification" results they obtained on symbolic data. Their "4-note Degree chords" correspond to our Degree & Category representation. They obtain their best results with 2-grams with 87% accuracy on the 3-class problem and 51% accuracy on the 9-class problem respectively. Since we do not have access to their specific results per fold we

can not use the McNemar test which would be the most precise test to compare the results of two classifiers on the same dataset. The Wilcoxon signed-ranks test requires to compare the algorithms on more than 5 datasets, and we only have 2, the 3-class problem and the 9-class problem. Additionally since our classifier outperforms theirs on the 9-class problem but theirs outperforms ours on the 3-class problem, neither system appear to be generally superior. If the performance of the two approaches are equivalent, we argue that our representation is richer and more flexible, allowing for longer patterns of chord sequences potentially containing gaps and having variable length (with no upper limit on the length). On the other hand Pérez-Sancho et al.'s n-gram representation only allows for patterns of size n and below to be taken into account (without gaps) in one model. We show that this enables us to discover and represent interesting chord patterns that could not be captured with an n-gram representation. Examples of such chord patterns are provided and discussed in Section 5.3.5.

5.3.3 From Symbolic Data to Automatic Chord Transcriptions

For such a method to be usable in more situations we need to be able to analyse audio data as well as symbolic data. In this section we study whether the approach previously tested on symbolic data performs equally well on automatically transcribed chords from audio data. We first directly apply to the automatic chord transcriptions the models trained on symbolic data from Section 5.3.2. The results can be found in Table 5.5.

Notice that the unique symbolic model for Root Interval and 9 genres (due to large computation times only one fold was computed then, cf. Table 5.2) required too many resources to evaluate its coverage on audio data and we had to cancel its evaluation. Looking at this model closely one explanation for this would be that contrary to more complex models (i.e. models containing more nodes and literals overall) which successfully run, this one starts with very specific rules which means a lot of literals to evaluate against each new input, when more complex models would often successfully classify a new input with one of the simple rules it starts with. One possible way to solve this issue would be to recompute the model(s) changing the order of the training examples, since the order of the rules directly depends on it.

Overall the symbolic models still perform higher than the baseline on audio data but there is a clear degradation in accuracy compared to the results they obtain on symbolic data. This is probably due to the noisy data resulting from the automatic transcription process. The best knowledge representation still seems to be Degree & Category in average but this time the Degree representation often performs equally well or even better, such as in the case of

Popular subgenres. This is not surprising since the Degree & Category representation requires a correct estimation of both the Degree (or chord root note) and the chord category while the Degree representation only requires the former making the data used by this representation less likely to be noisy and hence more likely to match the rules of its symbolic-trained models.

The statistical significance analysis using García and Herrera's script provides the following average ranking values: 2.6 (Root Interval), 1.8 (Degree) and 1.6 (Degree & Category). Friedman's statistic is $\chi_F^2 = 2.8$ from which Iman-Davenport's statistic is computed: $F_F = 1.56$. With 3 algorithms and 5 datasets (since there is no complete comparison of the 3 knowledge representations on 9 subgenres) F_F is distributed according to the F distribution with $3-1=2$ and $(3-1)(5-1)=8$ degrees of freedom. The critical value of $F(2,8)$ for $\alpha = 0.05$ is 4.46 so we can not reject the null hypothesis that all algorithms are equivalent. None of the knowledge representation schemes outperforms the others when applying the symbolic models on audio data.

Table 5.5: Classification results of models trained on symbolic data when tested on audio data. All symbolic models previously built during the 10-fold cross-validation experiments of Table 5.2 are tested on audio data matching each time the 10th testing fold. b is the baseline. Values are average ± standard deviation over the folds. Highest accuracy is shown in **bold**. (*): Experiments stopped after one of the iterations (folds) run for too long; these results were averaged over a few folds (exact number of folds is provided) only and are only given as indications.

	Root Interval	Degree	Deg. & Cat.
3 main genres Accuracy (b = 0.414)	0.515 ± *0.050*	0.536 ± *0.054*	**0.653** ± *0.057*
academic/jazz Accuracy (b = 0.621)	0.563 ± *0.039*	0.615 ± *0.087*	**0.642** ± *0.066*
academic/popular Accuracy (b = 0.571)	0.650 ± *0.106*	0.648 ± *0.065*	**0.701** ± *0.085*
jazz/popular Accuracy (b = 0.553)	0.725 ± *0.063*	**0.747** ± *0.052*	0.570 ± *0.049*
blues/Celtic/pop Accuracy (b = 0.368)	0.543 ± *0.100*	**0.706** ± *0.070*	0.684 ± *0.084*
9 subgenres Accuracy (b = 0.216)	–	(4 folds) (0.283* ± *0.037**)	**0.335** ± *0.087*

Another approach one can take is to train new models directly on audio data. The results of such an approach are provided in Table 5.6. The accuracies are higher than when using symbolic models on audio data, which was predictable as systematic errors of the transcription algorithm are now probably captured by such models. Once more the Degree & Category gets better results on average than the other knowledge representations, and when not at least similar results as the best approach. However a statistical significance analysis (using García

and Herrera's script) reaches similar conclusions as for the symbolic models on audio data. The average ranking values are 2.1 (Root Interval), 2.4 (Degree) and 1.5 (Degree & Category). Friedman's statistic is $\chi_F^2 = 2.10$ from which Iman-Davenport's statistic F_F is equal to 1.06. The critical value of $F(2, 8)$ for $\alpha = 0.05$ being 4.46 we can not reject the null hypothesis that all algorithms are equivalent. However the computation times for the Degree & Category models are often lower than for the other models. Likewise the complexity of the audio models using this representation is often lower than the other ones, making it still the most interesting representation.

This approach does not compare though with the n-gram approach used by Pérez-Sancho et al. (2010) which reaches higher accuracies when trained and used on audio data: 82% accuracy (3 genres) and 58% accuracy (9 subgenres) using their 4-chord Degree representation, whereas we obtain 70.4% (3 genres) and 46.3% (9 subgenres) accuracy. Notice that they use a different chord recognition algorithm (supporting different chord categories) making those results not directly comparable.

In general it is important to notice that in our method the chord transcription algorithm employed, its coverage and accuracy do affect the classification accuracy and content of the classification models. As chord recognition algorithms will become more accurate and will cover more chord categories, so will classification models built with our approach, and their content, exempt of noise, will tend to be more musicologically meaningful.

5.3.4 Towards Ensemble Methods

We investigate in this section if an ensemble method using several trees, namely Random Forests (Breiman, 2001), would improve the results over single decision trees. Random Forests consist of several decisions trees, each constructed from a sample of the training dataset and a new randomly restricted feature set for each node in each tree. No pruning of the individual trees is performed, and the random forest prediction is obtained by taking the mode vote of all trees. Breiman reports "significant improvements in classification accuracy" when using Random Forests over using single trees. As seen in Section 3.4, Van Assche (2008) developed a Relational Learning version of (First-Order) Random Forests which has been integrated with TILDE in the ACE-ilProlog system. We use this implementation, set the number of trees to 30 and the query sampling probability to 0.25 in all experiments, and obtain the results shown in Table 5.7.

As expected Random Forest models are computationally more expensive than single-tree models to the extent that most Root Interval models and some of the Degree models could

Table 5.6: Classification results of models trained and tested on audio data using 10-fold cross-validation. b is the baseline. Values are average ± standard deviation over the folds. Highest accuracy is shown in **bold**. (*): Experiments stopped after one of the iterations (folds) run for too long; these results were averaged over a few folds (exact number of folds is provided) only and are only given as indications.

	Root Interval	Degree	Deg. & Cat.
3 main genres	(4 folds)		
Accuracy (b = 0.413)	(0.670* ± *0.036**)	0.675 ± *0.048*	**0.704** ± *0.045*
Runtime (in CPU seconds)	(94,204* ± *113,713**)	108,205 ± *203,027*	8,116 ± *5,731*
# nodes in the tree	(28.0* ± *2.4**)	56.3 ± *4.5*	47.9 ± *5.1*
# literals in the tree	(45.0* ± *2.9**)	168.9 ± *13.4*	143.2 ± *15.4*
academic/jazz			
Accuracy (b = 0.621)	0.784 ± *0.060*	0.780 ± *0.041*	**0.893** ± *0.062*
Runtime (in CPU seconds)	12,030 ± *23,496*	188 ± *20*	1,646 ± *171*
# nodes in the tree	17.2 ± *4.2*	24.6 ± *1.6*	16.1 ± *2.4*
# literals in the tree	28.3 ± *7.3*	73.8 ± *4.7*	48.3 ± *7.3*
academic/popular			
Accuracy (b = 0.571)	0.737 ± *0.064*	**0.739** ± *0.047*	0.732 ± *0.077*
Runtime (in CPU seconds)	66,502 ± *450,151*	4,004 ± *4,245*	3,581 ± *4,998*
# nodes in the tree	21.0 ± *3.2*	27.4 ± *5.2*	22.6 ± *1.9*
# literals in the tree	34.4 ± *5.5*	82.2 ± *15.7*	67.2 ± *5.7*
jazz/popular			
Accuracy (b = 0.552)	**0.808** ± *0.057*	0.807 ± *0.040*	**0.808** ± *0.046*
Runtime (in CPU seconds)	7,458 ± *3,573*	244 ± *21*	2,468 ± *234*
# nodes in the tree	15.6 ± *4.8*	29.7 ± *6.0*	22.6 ± *3.5*
# literals in the tree	28.4 ± *8.3*	89.1 ± *18.0*	67.2 ± *10.0*
blues/Celtic/pop			
Accuracy (b = 0.368)	0.658 ± *0.094*	0.621 ± *0.112*	**0.662** ± *0.063*
Runtime (in CPU seconds)	920 ± *216*	1,640 ± *3,677*	237 ± *20*
# nodes in the tree	15.0 ± *2.1*	16.8 ± *3.2*	16.2 ± *2.1*
# literals in the tree	27.1 ± *3.5*	50.4 ± *9.7*	48.4 ± *6.2*
9 subgenres			
Accuracy (b = 0.216)	(–)	0.390 ± *0.029*	**0.463** ± *0.049*
Runtime (in CPU seconds)	(–)	28,782 ± *79,044*	7,974 ± *540*
# nodes in the tree	(–)	79.9 ± *5.7*	80.2 ± *7.2*
# literals in the tree	(–)	239.7 ± *17.1*	237.8 ± *21.5*
Average ranking			

Table 5.7: Classification results of Random Forests models trained and applied on symbolic data, trained on symbolic data and applied audio data, and trained on audio data and applied on audio data. 10-fold cross-validation is used in all cases with matching symbolic and audio folds. The number of trees is equal to 30, and query sampling probability is equal to 0.25 for each model. b is the baseline. Values are average ± standard deviation over the folds. Highest accuracy is shown in **bold**. (*): Experiments stopped after one of the iterations (folds) run for too long; these results were averaged over a few folds (exact number of folds is provided) only and are only given as indications. (−): Experiments stopped during the first iteration due to long computation times and for which we have no results.

	Root Interval	Degree	Deg. & Cat.
Symbolic models			
3 main genres	(3 folds)		
Accuracy (b = 0.395)	(**0.727*** ± *0.063**)	0.763 ± *0.046*	**0.877** ± *0.038*
Runtime (in CPU seconds)	(478,395* ± *37,709**)	41,436 ± *13,379*	38,433 ± *3,392*
# nodes in the tree	(610.0* ± *4.5**)	709.2 ± *30.2*	535.1 ± *9.2*
# literals in the tree	(1,042.3* ± *0.5**)	2,127.6 ± *90.6*	1,583.7 ± *23.8*
9 subgenres			
Accuracy (b = 0.208)	(–)	(–)	**0.596** ± *0.071*
Runtime (in CPU seconds)	(–)	(–)	89,154 ± *18,938*
# nodes in the tree	(–)	(–)	1,124.9 ± *24.7*
# literals in the tree	(–)	(–)	3,310.3 ± *69.2*
Symbolic models on audio			
3 main genres	(3 folds)		
Accuracy (b = 0.414)	(**0.628*** ± *0.085**)	0.622 ± *0.063*	**0.628** ± *0.060*
9 subgenres			
Accuracy (b = 0.216)	(–)	(–)	**0.388** ± *0.058*
Audio models			
3 main genres			
Accuracy (b = 0.413)	(–)	0.719 ± *0.039*	**0.761** ± *0.042*
Runtime (in CPU seconds)	(–)	58,437 ± *37,464*	43,647 ± *3,158*
# nodes in the tree	(–)	986.4 ± *37.8*	887.6 ± *30.9*
# literals in the tree	(–)	2,959.2 ± *113.3*	2,650.8 ± *90.9*
9 subgenres			
Accuracy (b = 0.216)	(–)	0.439 ± *0.062*	**0.494** ± *0.056*
Runtime (in CPU seconds)	(–)	175,566 ± *191,545*	55,148 ± *1,292*
# nodes in the tree	(–)	1,319.7 ± *39.1*	1,353.2 ± *25.7*
# literals in the tree	(–)	3,959.1 ± *117.4*	4,033.2 ± *79.5*

not be computed in a reasonable amount of time (see empty cells in Table 5.7). All Degree & Category models were computed with computation times varying between 1.1 and 10 times the computation time of single-tree models on the same classification problems. When results for several knowledge representations are available, here again Degree & Category outperforms the others in accuracy and computation time, even if no in-depth statistical significance analysis can be performed due to the limited number of results for the other representations.

A statistical significance analysis of the single-tree and Random Forest accuracies on all classification problems available (3- and 9-genres; symbolic models, symbolic models on audio, and audio models) using the Degree & Category knowledge representation allows us to further refine our conclusions. Since we are only comparing two algorithms, we use the Wilcoxon signed-ranked test, which like the Friedman test is a non-parametric test. We compute the differences in performances of the two approaches in Table 5.8 and rank them from the smallest to the largest difference (ignoring the signs). The sum of the ranks of the positive differences is $R^+ = 2 + 3 + 4 + 5 + 6 = 20$ and the sum of the ranks of the negative differences is $R^- = 1$. According the table of critical values for the Wilconxon test for a confidence level of $\alpha = 0.05$ and 6 datasets the difference between the classifiers is significant if the smaller of these sums is equal to 0, or in other words if one of the algorithms always outperforms the other, which is not the case. So the results obtained with the Random Forests are marginally, but not significantly, higher than those with singles trees. Additionally the parameters used here, i.e. number of trees in a random forest and query sampling probability were not tuned and simply chosen by hand, due to the computational overheads of running multiple tuning experiments on separate tuning datasets. Hence we expect we could obtain even better results by automatically fine-tuning those parameters.

Table 5.8: Comparison of classification accuracies for Single Tree (SG) and Random Forest (RF) models with Degree & Category representation. r is the rank.

Classification problem	Baseline	ST	RF	difference	r
Symbolic models – 3 main genres	0.395	0.834	**0.877**	+0.043	3
Symbolic models – 9 subgenres	0.208	0.542	**0.596**	+0.054	5
Symbolic models on audio – 3 main genres	0.414	**0.653**	0.628	−0.025	1
Symbolic models on audio – 9 subgenres	0.216	0.335	**0.388**	+0.053	4
Audio models – 3 main genres	0.414	0.704	**0.761**	+0.057	6
Audio models – 9 subgenres	0.216	0.463	**0.494**	+0.031	2

Finally we evaluate how the Random Forest approach compares to the n-gram approach employed by Pérez-Sancho et al. (2010). We already know from Section 5.3.2 that we can not use the McNemar test on the Pérez-Sancho et al.'s test results. Furthermore we can only

compare the two approaches on 4 experiments, symbolic and audio models applied to the 3-genre and 9-subgenre datasets, meaning that we can not apply the Wilcoxon signed-ranked test. But we can compute the signed differences in accuracy between the algorithms on each dataset. Accuracies and signed differences can be found in Table 5.9. The Random Forest approach outperforms the n-gram one on symbolic data, but the opposite is true on audio data, with the largest difference each time being 8.6% on the 9-genre problem.

Table 5.9: Comparison of classification accuracies for our Random Forest (RF) models and Pérez-Sancho et al.'s n-gram models

Classification problem	Baseline	n-grams	RF	difference
Symbolic models – 3 main genres	0.395	0.870	**0.877**	+0.007
Symbolic models – 9 subgenres	0.208	0.510	**0.596**	+0.086
Audio models – 3 main genres	0.414	**0.820**	0.761	−0.059
Audio models – 9 subgenres	0.216	**0.580**	0.494	−0.086

However the Random Forest approach seems the most promising one involving relational learning and a logic-based representation.

5.3.5 Examples of Rules

In this section we study the rules and models we get from these experiments and describe the general musical patterns they exhibit. Our conclusions are drawn from all the models generated, i.e. we consider the five models resulting from the 5-fold cross-validation generated for each classification task.

We provide here human-readable versions of the rules where:

- each chord is represented by its juxtaposed degree and category,
- "..." represents a gap (of any length per definition of a gap in this study).

Since the classification models are trees (or ordered sets of rules) a rule in itself can not perform classification both because of having a lower accuracy than the full model and because the ordering of rules in the model is important to the classification (i.e. some rule might never be used on some example because one of the preceding rules in the model covers this example). Hence for each of the following example rules only the local coverage is shown. This is the coverage provided in the model, and represents the coverage of the rule on the remaining examples once all examples covered by previous rules in the model are removed.

In all models the academic music is characterised by rules that establish the tonality: the tonic (first degree of the scale), the dominant (fifth degree) or quite often both of these

degrees are present in a vast majority of the academic rules. As explained by Piston (1987) tonic and dominant (together with subdominant, fourth degree of the scale) "are called the tonal degrees of the scale, since they are the mainstay of the tonality". If all 3 main genres are characterised by the V-I cadence in music theory, in our rules, this pattern associated with triads distinguishes well academic from jazz. Examples of academic rules of this kind include:

[local coverage: *academic=131/131=1; jazz=0*]
*... **Imaj V7 Imaj** ...*

or more simply:

[local coverage: *academic=126/127=0.99; jazz=1/127=0.01*]
*... **Vmaj Imaj** ...*

which both contain the perfect cadence (V-I), and:

[local coverage: *academic=27/27=1; jazz=0*]
*... **Vmaj Imin** ...*

which is the perfect cadence in minor keys.
Some of the jazz rules use the dominant and the tonic as well but with more complex and colourful chords than the academic rules which often only use triads:

[local coverage: *jazz=156/161=0.97; academic=0; popular=5/161=0.03*]
*... **V7 Imaj6** ...*

Notice that the above example is almost a perfect cadence, except for the chord category of the tonic chord (which should be major or minor triad). Because of the major sixth chord, we do not know which cadence we are looking at: it could be interpreted as V7-Imaj or as V7-VImin, both equally plausible. The extra note is adding more possibilities (to the chords that could come next), turning the chord sequence into a more unstable one with multiple functions, which is typical of jazz music.
Even when they do not convey ambiguity the chords used in the jazz rules are more "colourful", like in the following rules where the minor (respectively major) 7th chord is used on the tonic

(instead of a simple minor or major triad):

[local coverage: *jazz=27/27=1; academic=0; popular=0*]
*... **V7 Imin7** ...*

[local coverage: *jazz=35/37=0.95; academic=0; popular=2/37=0.05*]
*... **V7 Imaj7** ...*

The jazz rules are less about the tonality and more about what can be called harmonic "colour": they contain a larger variety of chord categories including a lot of seventh (i.e. minor seventh, major seventh and dominant seventh) chords, less common triads (such as augmented and diminished triads) but also major sixth and some (seventh) chords that, even though there are no 9th chords in our vocabulary, can be interpreted as 9th chords on another degree than the one they are officially attached to. For instance in the following rule:

[local coverage: *jazz=4/6=0.67; academic=0; popular=2/6=0.33*]
*... **IVmin7 bVII7** ...*

for instance in C major, the bVII7 chord is BbDFAb which can also be interpreted as a rootless minor ninth chord on the dominant (Vmin9) borrowed from the parallel minor (i.e. C minor when in C major): (G)BbDFAb. This chord progression (IVmin7 - bVII7 going to I) is known in jazz as the backdoor progression.
Other typical jazz harmonic patterns also appear in these models. For instance the tritone substitution – a principle according to which a dominant 7th chord may be replaced by another dominant 7th chord whose root is a tritone away from its root – is most frequently used in the context of IImin7-V7-I which becomes IImin7-bII7-I, a pattern that we found in several models:

[local coverage: *jazz=5/5=1; academic=0; popular=0*]
*... **IImin7 bII7** ...*

[local coverage: *jazz=5/5=1; academic=0; popular=0*]
*... **bII7 Imaj7** ...*

Popular music harmony tends to have simpler harmonic rules, or even tends to be defined by the absence of the other rules that characterise academic music and jazz. This is natural as melody is predominant in this style.
The system is also able to find rules spanning longer time periods that a human might not spot easily, due to the arbitrary-sized gaps they contain:

[local coverage: *academic=89/90=0.99; jazz=0; popular=1/90=0.01*]
*... **II7 Vmaj** ... **Imaj V7** ...*

which is a modulation to the dominant, followed eventually by a return to the tonic key.
On subgenres the system extracts rules such as the following characterising blues:

[local coverage: *blues=38/40=0.95; Celtic=0; pop=2/40=0.05*]
*... **I7 IV7** ...*

which is part of a typical 12-bar blues progression.

Working from audio data, even though the transcriptions are not fully accurate, the classification and rules still capture the same general trends as for symbolic data.

5.4 Conclusions

In this chapter we showed that it is not only possible to automatically discover patterns in chord sequences which characterise a corpus of data but also to use such models as classifiers. This was made possible by the ILP decision tree induction algorithm TILDE which we used to build both single decision trees and random forests. We presented a new and more expressive representation scheme of harmonic sequences based on context-free definite-clause grammars that allowed chord sequences of any length to coexist in the same model, allowed uninteresting sequences to be skipped with the use of gaps, but also allowed context information, such as key, to be captured. Our experiments, on several 2-, 3- and 9-

genre classification problems, found that the more musically meaningful Degree & Category representation gave better classification results than using root intervals, or degree alone, while not increasing the complexity of the models or the computation times which remain low. All results are significantly above the baseline, but performance clearly decreases for more difficult tasks. The results using transcription from audio data were encouraging in that although some information was lost in the transcription process, the classification results remained well above the baseline, and thus this approach is still viable when symbolic representations of the music are not available. On both symbolic and audio data the best results were reached with Random Forests which outperform the single decision trees and came on par with a n-gram approach applied to the same dataset. Our logic-based models however were more flexible and could discover and capture complex chord patterns that n-grams could not, or that even humans might not spot easily.

Perfect classification is not to be expected from harmony data, since other aspects of music such as instrumentation (timbre), rhythm and melody are also involved in defining and recognising musical genres. Additionally the audio used in these experiments was synthesised from MIDI, which might have an effect on the classification results. Nevertheless, the positive results of the experiments described in this chapter encouraged further experiments that will tackle those two limitations. In the next chapter we will integrate a state-of-the-art genre classification system, employing signal-based timbre, rhythm and melody features, with the current harmony-based classification approach, and test whether the addition of a harmony feature could improve genre classification on real (and not synthesised) audio data.

CHAPTER 6

IMPROVING ON STATE-OF-THE-ART GENRE CLASSIFICATION

6.1

Introduction

In this chapter we take a different approach and instead of testing if harmony can be used alone to model genres as we did in Chapter 5, we consider it as being only one feature of a larger classification framework. We propose the combination of a state-of-the-art statistical genre classifier based on timbral features with harmony- and logic-based random forest models resulting from the work presented in Chapter 5, in an effort to improve on genre classification performance using the chord sequences as an additional source of data. To our knowledge no attempt to integrate signal-based features with high-level harmony descriptors has been made in the literature.

The outline of the chapter is as follows. In Section 6.2, a standard state-of-the-art classification system, its signal-based features, together with the fusion procedure to integrate the harmony models are described. Section 6.3 presents the datasets used for both training and testing of the framework together with data post-processing. Section 6.4 assesses and discusses the performance of the proposed fused classifier against the standard classifier. Conclusions are drawn in Section 6.5.

6.2

Combining Audio and Harmony-based Classifiers

In this section, we describe the standard state-of-the-art classification system we employed in our experiments in combination with the harmony-based classifier previously described. Its extracted features are listed in Section 6.2.1, the feature selection procedure is described in Section 6.2.2 and finally the fusion procedure is explained and the employed machine learning classifiers are presented in Section 6.2.3.

6.2.1 Feature Extraction

In feature extraction, a set of vectors is computed to describe aspects of an audio recording (Tzanetakis and Cook, 2002). Extracting features is the first step in pattern recognition systems, since any classifier can be applied afterwards. In most genre classification experiments the extracted features belong to 3 categories: timbre, rhythm, and melody (Scaringella et al., 2006). For our experiments, the feature set proposed in (Benetos and Kotropoulos, 2010) was employed, which contains timbral descriptors such as energy and spectral features, as well as pitch-based and rhythmic features, thus being able to accurately describe many aspects of the audio signal. The complete list of extracted features can be found in Table 6.1.

Table 6.1: Extracted Features

Feature	**# Values per segment**
Short-Time Energy (STE)	$1 \times 4 = 4$
Spectral Centroid (SC)	$1 \times 4 = 4$
Spectral Rolloff Frequency (SRF)	$1 \times 4 = 4$
Spectral Spread (SS)	$1 \times 4 = 4$
Spectral Flatness (SF)	$4 \times 4 = 16$
Mel-frequency Cepstral Coefficients (MFCCs)	$24 \times 4 = 96$
Spectral Difference (SD)	$1 \times 4 = 4$
Bandwidth (BW)	$1 \times 4 = 4$
Auto-Correlation (AC)	13
Temporal Centroid (TC)	1
Zero-Crossing Rate (ZCR)	$1 \times 4 = 4$
Phase Deviation (PD)	$1 \times 4 = 4$
Fundamental Frequency (FF)	$1 \times 4 = 4$
Pitch Histogram (PH)	$1 \times 4 = 4$
Rhythmic Periodicity (RP)	$1 \times 4 = 4$
Total Loudness (TL)	$1 \times 4 = 4$
Specific Loudness Sensation (SONE)	$8 \times 4 = 32$
Total number of features	**206**

The feature related to the audio signal energy is the Short-Time Energy (STE). Spectral descriptors of the signal are the Spectral Centroid (SC), Spectral Rolloff Frequency (SRF), Spectral Spread (SS), Spectral Flatness (SF), Mel-frequency Cepstral Coefficients (MFCCs), Spectral Difference (SD) – also known as spectral flux – and Bandwidth (BW). Temporal descriptors include the Auto-Correlation (AC), Temporal Centroid (TC), Zero-Crossing Rate (ZCR), and Phase Deviation (PD). As far as pitch-based features are concerned, the Fundamental Frequency (FF) feature is computed using maximum likelihood harmonic matching, while the Pitch Histogram (PH) describes the amplitude of the maximum peak of the folded histogram (Tzanetakis et al., 2003). The Rhythmic Periodicity (RP) feature

was proposed in (Pampalk et al., 2004). Finally, the Total Loudness (TL) feature and the Specific Loudness Sensation (SONE) coefficients are perceptual descriptors which are based on auditory modeling.

All in all, 206 feature values are extracted for each sound recording. For the computation of the feature vectors, the descriptors are computed on a frame basis and their statistical measures are employed in order to result in a compact representation of the signal characteristics. To be specific, their mean and variance are computed along with the mean and variance of the first-order frame-based feature differences over a 1 sec texture window. The same texture window size was used for genre classification experiments in (Tzanetakis and Cook, 2002). Afterwards, the computed values are averaged for all the segments of the recording, thus explaining the factor 4 appearing in Table 6.1. This is applied for all extracted features apart from the AC values and the TC, which are computed for the whole duration of the recording. In addition, it should be noted that for the MFCCs, 24 coefficients are computed over a 10 msec frame (which is a common setting for audio processing applications), while 8 SONE coefficients are computed over the same duration – which is one of the recommended settings in (Pampalk et al., 2004).

6.2.2 Feature Selection

Although the 206 extracted features are able to capture many aspects of the audio signal, it is advantageous to reduce the number of features through a feature selection procedure in order to remove any feature correlations and to maximise classification accuracy in the presence of relatively few samples (Scaringella et al., 2006). One additional motivation behind feature selection is the need to avoid the so-called curse of dimensionality phenomenon (Burred and Lerch, 2003).

In this work, the selected feature subset is chosen as to maximise the inter/intra class ratio (Fukunaga, 1990). The aim of this feature selection mechanism is to select a set of features that maximises the sample variance between different classes and minimises the variance for data belonging to the same class, thus leading to classification improvement. The branch-and-bound search strategy is employed for complexity reduction purposes, being also able to provide the optimal feature subset. In the search strategy, a tree-based structure containing the possible feature subsets is traversed using depth-first search with backtracking (van der Hedjen et al., 2004).

For our experiments, several feature subsets were created, containing $\Theta = \{10, 20, \ldots, 100\}$ features. In Table 6.2, the subset for 10 selected features is listed, where it can be seen that

Table 6.2: The subset of 10 selected features.

No.	Selected Feature
1.	Variance of 1st order difference of 7th SONE
2.	Variance of BW
3.	Mean of SD
4.	Variance of PH
5.	Mean of 7th MFCC
6.	Variance of 5th MFCC
7.	Mean of SS
8.	Variance of 1st order difference of 9th MFCC
9.	Variance of FF
10.	Variance of 1st order difference of 1st SONE

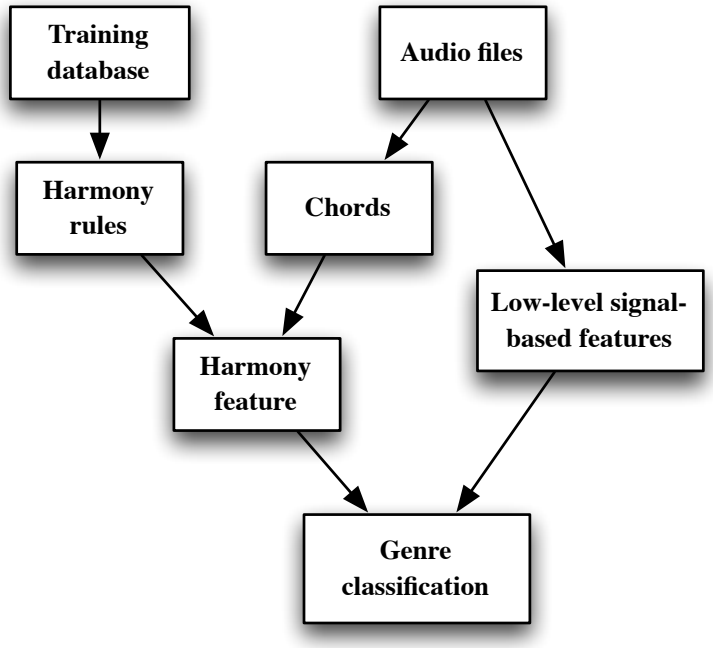

Figure 6.1: Block diagram of the genre classifier

the MFCCs and the SONE coefficients are some of the most discriminative features.

6.2.3 Classification System

Figure 6.1 represents the steps that are performed to build our genre classification system. The proposed classifier combines the extracted and selected features presented in Sections 6.2.1 and 6.2.2 with the output of the harmony-based classifier described in Chapter 5. Considering the extracted feature vector for a single recording as $\mathbf{v}$ (with length Θ) and the respective output of the harmony-based classifier as $r = 1, \ldots, C$, where C is the number of genre classes, a combined feature vector is created in the form of $\mathbf{v}' = [\mathbf{v}\ r]$. Thus, the output of the harmony-

based classifier is treated as an additional feature used, along with the extracted and selected audio features, as an input to the learning phase of the overall genre classifier.

Two machine learning classifiers were employed for the genre classification experiments, namely Multilayer Perceptrons (MLPs) and Support Vector Machines (SVMs). For the MLPs, a 3-layered perceptron with the logistic activation function was utilized, while training was performed with the back-propagation algorithm with learning rate equal to 0.3 for 500 training epochs, with momentum equal to 0.1. A multi-class SVM classifier with a 2nd order polynomial kernel with unit bias/offset was also used (Schölkopf et al., 1999). The experiments with the aforementioned classifiers were conducted on the training matrix $\mathbf{V}' = [\mathbf{v}'_1\ \mathbf{v}'_2\ \cdots\ \mathbf{v}'_M]$, where M is the number of training samples.

6.3 Datasets

To run our final genre classification from audio experiments we decided to use two datasets common employed in the literature for genre classification experiments. Firstly, the GTZAN database was used, which contains 1000 audio recordings distributed across 10 music genres, with 100 recordings collected for each genre (Tzanetakis and Cook, 2002). All recordings are single channel, are sampled at 22.05 kHz rate and have a duration of approximately 30 sec. The second dataset that was used was created for the ISMIR 2004 Genre Classification Contest (ISMIR, 2004). It covers 7 genre classes with varying number of audio recordings per classes. The duration of the recordings is also not constant, ranging from 19 seconds to 14 minutes. The recordings were sampled at 22kHz rate and were converted from stereo to mono.

As for training the harmony features we chose to use the *Perez-9-genres* Corpus that was used in Chapter 5, as we already knew what sort of models we could get from it and how well they were performing genre classification by themselves. As shown in our previous experiments presented in Section 5.3.3, audio-trained models outperformed symbolic-trained models when used on audio data, so we used the synthesised audio version of the *Perez-9-genres* Corpus.

From those three datasets we extracted and grouped classes so that we could have a common taxonomy over all datasets made of 3 classes: classical, jazz/blues and rock/pop. From the 10 genre classes of the GTZAN dataset, 3 were selected for the experiments, namely the

classical, jazz, and pop classes (with 100 recordings each). In the ISMIR04 dataset 3 were used: classical (319 recordings), jazz/blues (26 recordings), and pop/rock (102 recordings).

Finally we re-organised the *Perez-9-genres* Corpus into the following three classes, in order to train our harmony-based classifier on classes that match the testing dataset's classes: classical (the full classical dataset from the *Perez-9-genres* Corpus, i.e. all the files from its 3 sub-classes), jazz/blues (a class grouping the blues and the 3 jazz subgenres from the *Perez-9-genres* Corpus) and pop (containing only the pop sub-class of the popular dataset from the *Perez-9-genres* Corpus). Thus we do not use the celtic subgenre.

6.4 Experiments

6.4.1 Training Results

We first simultaneously train our random forest classifier and estimate the best results it could obtain on clean and accurate transcriptions by performing a 5-fold cross-validation on the restricted and re-organised symbolic and synthesised audio dataset we created from the *Perez-9-genres* Corpus (cf. Section 6.3). The resulting confusion matrices are given in Table 6.3 and Table 6.4. The columns correspond to the predicted music genres and the rows to the actual ones. The average accuracy is 84.8% for symbolic data, and 79.5% for the synthesised audio data, while the baseline classification accuracies are 55.6% and 58%, when attributing the most probable genre to all the songs. The classifiers detects the classical and jazz/blues classes very well but only correctly classifies a small number of pop songs. We believe that this is due to the shortage of pop songs in our training dataset, combined with the unbalanced number of examples in each class: the jazz set is twice as large as the classical set which in turn is twice as large as the pop set. Performance of these classifiers on real audio data will be presented in Section 6.4.2.

6.4.2 Testing on Real Audio Data

Secondly the harmony-based classifier (trained on both the re-organised symbolic and synthesised audio *Perez-9-genres* datasets) was tested on the two audio datasets. The results are shown in Tables 6.5, 6.6, 6.7 and 6.8. For the GTZAN dataset, the classification accuracy

Table 6.3: Confusion matrix (cumulative results over the 5 folds of the cross-validation) for the harmony-based classifier applied on the classical-jazz/blues-pop restricted and re-organised version of the *Perez-9-genres* Corpus (symbolic dataset).

Real/Predicted	classical	jazz/blues	pop	Total
classical	218	15	1	234
jazz/blues	9	407	2	418
pop	26	61	13	100
Total	253	483	16	752

Table 6.4: Confusion matrix (cumulative results over the 5 folds of the cross-validation) for the harmony-based classifier applied on the classical-jazz/blues-pop restricted and re-organised version of the *Perez-9-genres* Corpus (synthesised audio dataset).

Real/Predicted	classical	jazz/blues	pop	Total
classical	181	20	1	202
jazz/blues	34	373	1	408
pop	31	57	5	93
Total	246	450	7	703

using the harmony-based classifier is 41.67% (symbolic training) and 44.67% (synthesised audio training), while for the ISMIR04 dataset it is 57.49% (symbolic training) and 59.28% (synthesised audio training). These results are not very impressive: on the ISMIR04 dataset they are lower than the baseline (71.36%), while for the GTZAN dataset they are just above the baseline (33.3%). This however does not mean that as an additional feature in a larger classifier, our harmony models can not help improve the classification accuracy. This is what we will investigate in the remainder of this chapter.

Like for synthesised audio in Chapter 5, when tested on real audio data, the classifiers trained on symbolic data obtain worse results than the ones trained on synthesised audio data. Given these results we will use the classifier trained on synthesised audio data in the experiments merging the harmony-based and the audio feature based classifiers.

Table 6.5: Confusion matrix for the harmony-based classifier trained on symbolic data and applied on the GTZAN dataset.

Real/Predicted	classical	jazz	pop	Total
classical	38	47	15	100
jazz	19	72	9	100
pop	24	61	15	100
Total	81	180	39	300

Then experiments using the SVM and MLP classifiers with 5x5-fold cross-validation were performed using the original extracted audio feature vector $\mathbf{v}$ which does not include the output

Table 6.6: Confusion matrix for the harmony-based classifier trained on synthesised audio data and applied on the GTZAN dataset.

Real/Predicted	classical	jazz	pop	Total
classical	59	39	2	100
jazz	21	70	9	100
pop	22	73	5	100
Total	102	182	16	300

Table 6.7: Confusion matrix for the harmony-based classifier trained on symbolic data and applied on the ISMIR04 dataset.

Real/Predicted	classical	jazz/blues	pop/rock	Total
classical	207	34	78	319
jazz/blues	8	10	8	26
pop/rock	47	15	40	102
Total	262	59	126	447

Table 6.8: Confusion matrix for the harmony-based classifier trained on synthesised audio data and applied on the ISMIR04 dataset.

Real/Predicted	classical	jazz/blues	pop/rock	Total
classical	233	61	25	319
jazz/blues	9	16	1	26
pop/rock	27	59	16	102
Total	269	136	42	447

of the harmony-based classifier. First these classifiers were tested on the synthesised *Perez-9-genres* Corpus which is described in Section 5.2.3. The full set of 206 audio features was employed for classification. For the SVM, classification accuracy is 95.56%, while for the MLP classifier, the classification accuracy is 95.67%. While classification performance appears to be very high compared to the harmony-based classifier for the same data, it should be stressed that the *Perez-9-genres* dataset consists of synthesised MIDI files. These files use different sets of synthesised instruments for each of the 3 genres, which produce artificially high results when a timbral feature-based classifier is employed.

Finally, experiments comparing results of the SVM and MLP classifiers with and without the output of the harmony-based classifier (trained on synthesised audio data) were performed with the various feature subsets on the SVM and MLP classifiers using 5x5-fold cross-validation. The average accuracy achieved by the classifiers using 5x5-fold cross-validation for the various feature subset sizes using the GTZAN dataset is shown in Figure 6.2, while the average accuracy for the ISMIR04 dataset is shown in Figure 6.3. In Table 6.9 the best

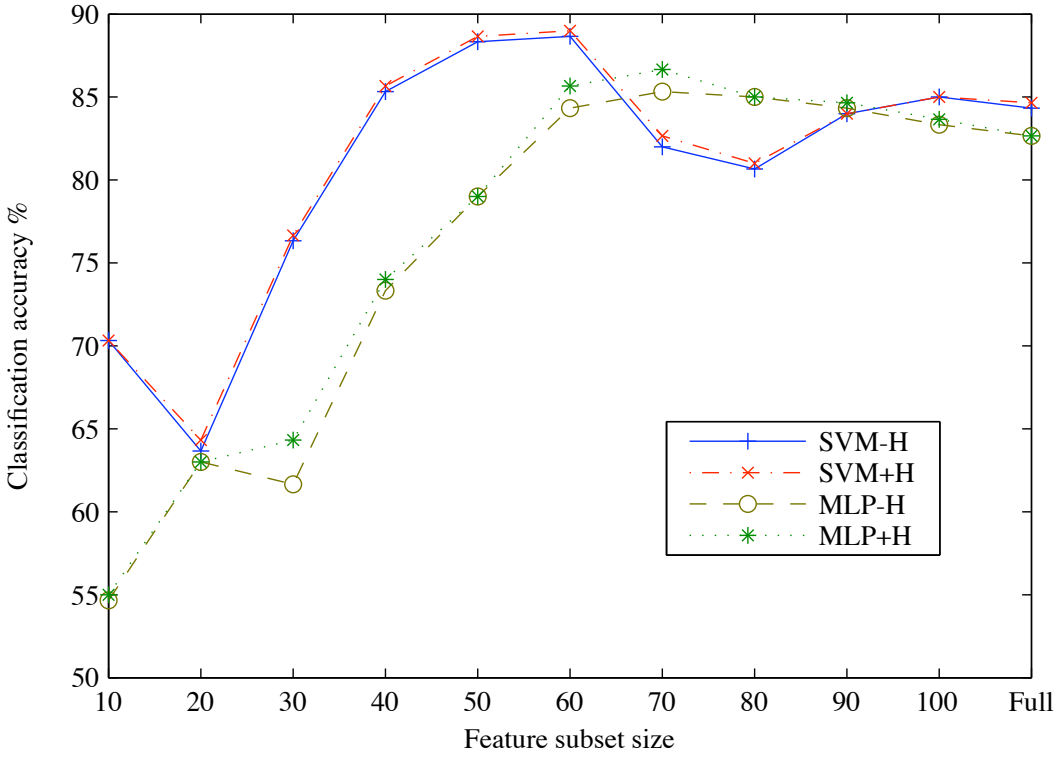

Figure 6.2: Classification accuracy for the GTZAN dataset using various feature subsets.

accuracy achieved for the various feature subsets and classifiers is presented. The SVM-H and MLP-H classifiers stand for the standard feature set $\mathbf{v}$ (without harmony), while the SVM+H and MLP+H classifiers stand for the feature set $\mathbf{v}'$ (with harmony).

Table 6.9: Best mean accuracy achieved by the various classifiers for the GTZAN and ISMIR04 datasets using 5x5-fold cross-validation.

Classifier	GTZAN Dataset	ISMIR04 Dataset
SVM-H	88.66% (60 Features)	93.77% (70 Features)
SVM+H	91.13% (50 Features)	95.30% (80 Features)
MLP-H	87.19% (60 Features)	91.45% (90 Features)
MLP+H	87.53% (60 Features)	91.49% (Full Feature Set)

For the GTZAN dataset, the highest classification accuracy is achieved by the SVM+H classifier using the 50 feature subset, reaching 91.13% accuracy. The MLP classifiers fall behind the SVM classifiers for most feature subsets, apart from the subsets containing 70, 80, or 90 features. For the ISMIR04 dataset, the highest accuracy is also achieved by the SVM+H classifier, reaching 95.30% classification accuracy, for the 80 feature subset. The SVM-H classifier reaches 93.77% for the same subset. In most cases, the SVM+H and MLP+H classifiers display increased classification rates over the SVM-H and MLP-H

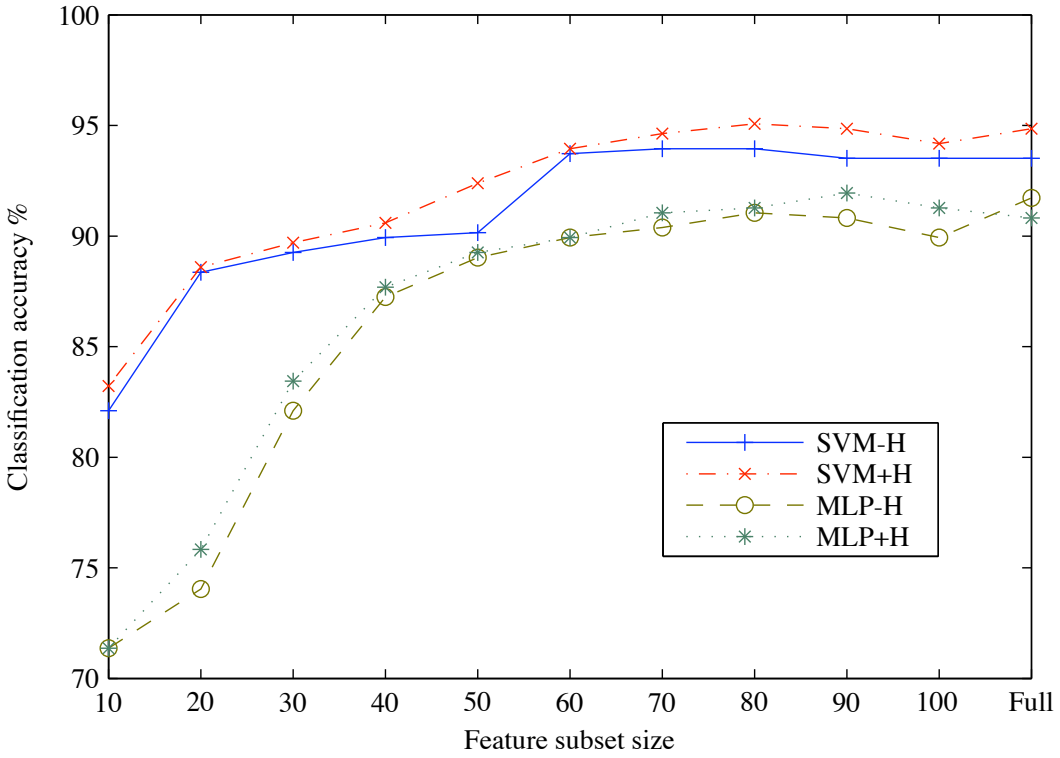

Figure 6.3: Classification accuracy for the ISMIR04 dataset using various feature subsets.

classifiers, respectively. There are however some cases where the classification rate is identical, for example for the MLP classifiers using the 60 features subset for the ISMIR04 dataset.

In order to compare the performance of the employed feature set with other feature sets found in the literature, the extracted features from the MARSYAS (Tzanetakis, 2007) toolbox were employed, which contain the mean values of the spectral centroid, spectral rolloff, spectral flux, and the mean values of 30 MFCCs for a 1sec texture window. Results on genre classification using the MARSYAS feature set with 5x5-fold cross-validation on both datasets and using the same classifiers (SVM, MLP) and their respective settings can be seen in Table 6.10, where it can be seen that for the MLP classifier, the classification accuracy between the MARSYAS feature set and the employed feature set is roughly the same for both datasets. However, when the SVM classifier is used, the employed feature set outperforms the MARSYAS features by at least 3% for the GTZAN case and 4% for the ISMIR04 set. It should be noted however that no feature selection took place for the MARSYAS features.

Insight into the performance of the best cases of the various classifiers using both datasets is obtained from the confusion matrices determined for each classifier run of the 5-fold cross-validation. The confusion matrices using the best SVM-H and SVM+H classifiers for the GTZAN and ISMIR04 datasets are presented in Tables 6.11, 6.12, 6.13 and 6.14. For the

Table 6.10: Mean accuracy achieved by the various classifiers for the GTZAN and ISMIR04 datasets, using the MARSYAS feature set and 5x5-fold cross-validation.

Classifier	GTZAN	ISMIR04
SVM	85.66%	91.51%
MLP	85.00%	91.96%

GTZAN dataset most misclassifications occur for the pop class, in both cases. However, the SVM+H algorithm rectifies some misclassifications of the pop class compared to the SVM-H classifier. For the ISMIR04 dataset, most misclassifications occur for the jazz/blues class for both classifiers. Even for the SVM+H classifier, when taking normalized rates, the jazz/blues class suffers the most, having only 63.58% correct classification rate. It should be noted though that the SVM+H classifier has 6 more jazz/blues samples correctly classified compared to the SVM-H one. The classical class on the other hand, seems largely unaffected by misclassifications.

Table 6.11: Confusion matrix for one 5-fold cross validation run of the SVM-H classifier applied on the GTZAN dataset using the 60 selected features set.

Real/Predicted	classical	jazz	pop	Total
classical	97	3	0	100
jazz	8	91	1	100
pop	3	19	78	100
Total	108	113	79	300

Table 6.12: Confusion matrix for one 5-fold cross validation run of the SVM+H classifier applied on the GTZAN dataset using the 50 selected features set.

Real/Predicted	classical	jazz	pop	Total
classical	97	3	0	100
jazz	8	90	2	100
pop	3	10	87	100
Total	108	103	89	300

Table 6.13: Confusion matrix for one 5-fold cross validation run of the SVM-H classifier applied on the ISMIR04 dataset using the 70 selected features set.

Real/Predicted	classical	jazz/blues	pop/rock	Total
classical	319	0	0	319
jazz/blues	10	11	5	26
pop/rock	12	1	89	102
Total	341	12	94	447

Table 6.14: Confusion matrix for one 5-fold cross validation run of the SVM+H classifier applied on the ISMIR04 dataset using the 80 selected features set.

Real/Predicted	classical	jazz/blues	pop/rock	Total
classical	317	0	2	319
jazz/blues	6	17	3	26
pop/rock	7	3	92	102
Total	330	20	97	447

Concerning the statistical significance of the improvement in results due to the use of the proposed feature vector $\mathbf{v}'$ compared to the performance of the standard feature vector $\mathbf{v}$, the McNemar test (McNemar, 1947) was employed, which is applied to 2x2 contingency tables for a single classifier run. We consider the cases exhibiting the highest classification rates, as shown in Table 6.9. For the GTZAN dataset, the SVM-H classifier using the 60 feature set is compared against the SVM+H classifier using the 50 feature set. For the ISMIR04 dataset, the SVM-H classifier using 70 features is compared against the SVM+H classifier using 80 features. The contingency tables for the GTZAN and ISMIR04 datasets are respectively:

$$\begin{bmatrix} 264 & 10 \\ 2 & 24 \end{bmatrix} \text{ and } \begin{bmatrix} 416 & 10 \\ 3 & 18 \end{bmatrix} \tag{6.1}$$

The binomial distribution is used to obtain the McNemar test statistic, where for both cases the null hypothesis (the difference between the two classifiers is insignificant) is rejected with 95% confidence.

6.4.3 Discussion

This improvement of the classification results might come as a surprise when one considers that the harmony-based classifier by itself does not perform sufficiently well on audio data. However, harmony is only one dimension of music which despite being relevant for genre identification can not capture by itself all genres' specificities. We believe that the classification improvement lies in the fact that it covers an aspect of the audio-signal (or rather of its musical properties) that the other (low-level) features of the classifier do not capture.

In order to justify that the combination of several features improves classification accuracy even when each feature's classification accuracy is lower than the baseline, the mean of the 5th MFCC was employed as an example feature. 5-fold cross-validation experiments were performed on the GTZAN and ISMIR04 datasets based on this single feature using SVMs. Results indicated that classification accuracy for the GTZAN dataset was 31.33%, while for the ISMIR04 dataset it was 71.36%, both of which are below the baseline. However the feature,

being one of the selected ones, when combined with several other features manages to report a high classification rate as shown in Section 6.4.2. Thus, the inclusion of the output of the harmony-based classifier, while being lower than the baseline by itself, still manages to provide improved results when combined with several other descriptors. In order to compare the addition of the harmony-derived classification to the feature set with an additional feature, the Total Loudness (TL) was added into the SVM-H classifier using the 70 features subset (TL is not included in the set). Using the GTZAN dataset for experiments, classification accuracy for the 70 features subset is 82%, while adding the TL feature it increased by 0.66%, where the performance improvement is lower compared to the harmony-based classifier addition (which was 1.66%).

6.5 Conclusions

In this chapter, an approach for automatic music genre classification was proposed, combining low-level features with a first-order logic random forest based on chord transitions and built using the Inductive Logic Programming algorithm TILDE. Three-class genre classification experiments were performed on two commonly used datasets, where an improvement was reported for both cases when the harmony-based classifier was combined with a low-level feature set, using support vector machines and multilayer perceptrons. The combination of these low-level features with the harmony-based classifier produces improved results despite the fact that the classification rate of the harmony-based classifier is not sufficiently high by itself. For both datasets when the SVM classifier was used, the improvement over the standard classifier was found to be statistically significant when the highest classification rate is considered. All in all, it was shown that the combination of high-level harmony features with low-level features can lead to genre classification accuracy improvements and is a promising direction for genre classification research.

CHAPTER 7

CONCLUSIONS

In this chapter we summarise the experiments and findings described throughout the thesis (Section 7.1). We then share our conclusions and answers to our research questions (Section 7.2), and go on to suggest directions for future research to expand on this work (Section 7.3).

7.1 Summary

In this thesis we have explored how to characterise music leveraging the expressiveness and characterisation power of both logic and harmony. Throughout our experiments we developed, refined and tested a harmony-based Inductive Logic Programming framework for characterisation and classification of musical styles.

We first experimented in Chapter 4 with the concept of a logic-based representation of harmony combined with logical inference of characterisation models on symbolic data using as inference system Aleph (Srinivasan, 2003).

Our objective was to characterise the harmony of the 180 songs featured on the Beatles' studio albums (Harte's transcriptions) and of 244 jazz standards from the Real Book. We analysed manually annotated chord data available in Resource Description Framework format giving us access to the root, bass and component intervals of the chords and keys. We pre-processed these data to obtain chord category (e.g. major, minor, suspended), degree (e.g. tonic, dominant) and intervals between chords, before passing them to Aleph which extracted the underlying harmony rules. For computational reasons we focused on characterising only the chord sequences of length 4 in these corpora.

We ran comparison experiments using the chord sequences of one corpus as positive examples and those of the other as negative examples. We also independently characterised each dataset, but instead of the built-in *positive examples only mode* (the system generates random negative examples) we preferred the *one negative example mode* that we developed, providing to the system one impossible example as the only negative example. Aleph models the positive examples in a sufficient and non-redundant way. This is one of the advantages of our method against the purely statistical one employed on the same datasets by Mauch et al. (2007).

Varying the form of the rules we extracted with this framework a total of 3667 harmony rules characterising the Real Book songs and the Beatles music. Encouragingly some of these

rules cover well-known pop or jazz harmonic patterns such as the I-IV-I-V pattern (for the Beatles) and ii-V-I-IV and the "turnaround" pattern I-VI-II-V (for the Real Book). It was also possible to identify patterns specific to the Beatles such as the extensive use of major chords (the predominant maj-maj-maj-maj pattern) and the cyclic patterns (e.g. I-IV-V-I-IV...) characterising their early compositions.

Focusing on fixed-length chord sequences constrained the search space, leading to short computation times, but also meant that the patterns were limited in both size and content. However our attempts at using a less constraining representation with independent descriptions of each chord, linking the chords only with a 'predecessor' predicate proved to be impossible in a reasonable amount of time with Aleph.

To overcome this knowledge representation difficulty, in Chapter 5 we moved on to a context-free definite-clause (CFDC) grammar representation. We encoded pieces of music as lists of chords, adopted a difference-list representation, and combined them with the concept of gaps of unspecified length between sub-sequences of interest. As a consequence we obtained a new flexible representation scheme that allows us to look at patterns of chord sequences of any length located anywhere in the music and even themselves containing gaps. In Chapter 4 the evaluation of the performance of the characterisation was qualitative. In Chapter 5 classification replaced pure characterisation to serve as a quantifiable measure of our method. We applied the ILP decision tree induction algorithm TILDE, which builds first-order decision trees which are equivalent to ordered sets of rules. As such CFDC grammars can be represented in this formalism and TILDE used to induce CFDC grammars.

The classification data consisted of 856 pieces of music in both symbolic and synthesised audio format containing three genres further subdivided into nine subgenres (*Perez-9-genres* Corpus). On these we considered the 3-way classification problem and all the 2-way classification problems on the main genres. In addition we studied the 3-way classification problem dealing with popular music subgenres (blues, Celtic and pop music) and the 9-way classification problem dealing with all the subgenres at once. We adapted our algorithm from the symbolic to the audio domain using a state-of-the-art chord transcription algorithm (Mauch, 2010).

A first round of experiments tested all the harmonic chord characteristics defined in Chapter 4. They showed that the representation merging degree and chord category performs genre classification statistically significantly better than the other representations. In all tasks it got a higher accuracy, correctly classifying 83.4% in the main genres case when the other

harmonic representations only reached around 70%. In the most difficult case of the 9-subgenre classification, it was the only one to even successfully run all cross-validation folds, while still scoring much better than the baseline (54.2% vs. 20.8%). More importantly, despite being a composite descriptor, it built less complex models in shorter computation times, leading us to conclude that it best captures the harmonic properties of genres. We also showed that these findings agree with musicological considerations.

The second round of experiments showed that audio data classification, although less performant due to transcription inaccuracies still reaches satisfying levels. Applying the models trained on symbolic data to audio displayed poor results, where models specifically trained on audio returned 70.4% accuracy on the 3 genre classification, while reaching 46.3% on the 9 subgenre one.

While the first experiments were using single decision trees, we then explored random forests to improve classification results. With 30 trees without further tuning, we obtained higher results : on symbolic data 87.7% (3 genres) and 59.6% (9 subgenres), and on synthesised audio data 76.1% and 49.4%.

Finally we compared the random forest implementation against the state-of-the-art n-gram algorithms of the dataset creators. We found equivalent results, outperforming them on symbolic data where the n-grams scored 87% (3 genres) and 51% (9 genres), while the opposite was true on synthesised audio data: 82% and 58% for the n-grams.

After showing similar performance, we detailed several rules highlighting the added value of our approach over n-grams. For instance we found simple and well-known rules such as the backdoor progression (IVmin7 - bVII7 going to I) known in jazz, but also some spanning longer sequences not easily spotted by human beings due to their internal gaps (e.g. II7 - Vmaj ... Imaj - V7).

In Chapter 6 we first took our random forest models from Chapter 5 trained on the *Perez-9-genre* Corpus and tested them on real (i.e. not synthesised) audio files from the GTZAN and ISMIR 2004 Genre Classification Contest datasets. On real audio as well, the synthesised audio trained models outperformed the symbolic trained models. And yet tests of the synthesised audio models on real audio data yielded unimpressive results: 44.67% (GTZAN) and 59.28% (ISMIR 2004) on a three-way classification involving the genres classical, jazz/blues and pop/rock.

Harmony is only one dimension of music, and hence cannot capture all genres' specificities. However it covers some that are left out by established low level features usually employed

in state-of-the-art genre classifiers. That is why we integrated our harmony and logic based random forest models in such a state-of-the-art genre classifier. The original classifier follows the typical bag-of-features paradigm. It describes the audio content by means of 206 timbral, pitch-based and rhythmic features, on which we performed feature selection. The harmony random forests models were integrated as an additional feature after this selection. We worked with two established statistical classifying algorithms: Multilayer Perceptrons and Support Vector Machines. This new enriched classifier showed on both datasets and both algorithms an increased accuracy compared to the regular one, even reaching statistical significance in the SVM case: from 88.66% (without harmony) to 91.13% (with harmony) on the GTZAN, and from 93.77% (without harmony) to 95.30% (with harmony) on ISMIR 2004. The performance improvement turned out to be three times higher than when extending the feature set by one signal-based feature.

7.2 Conclusions and Answers to Research Questions

As this summary shows, we have investigated and answered our research questions across multiple chapters. We provide here the conclusions we reached for each one of them, also highlighting the core contributions of this thesis.

RQ1: How can logic and in particular ILP support music characterisation? How can musical properties be represented and extracted?

In ILP knowledge representation is relational. One of the key relational concepts in music is time, for instance scores express, among other things, the temporal chaining between musical events. ILP can capture their succession, using for instance predecessor and successor predicates (specifying that a term follows another) or lists which are intrinsically ordered. ILP enables then a representation of music close to the musical score. Furthermore, logical truths in the studied domain can be encoded as background knowledge in ILP, and as part of the mining process they are applied to the data. This can be seen as a data preprocessing or analysis step to extract a higher level understanding of the data. It is akin to a musicological analysis of scores in our case. In the specific case of the work of this thesis, the score is a list of chords (similar to chords in a lead sheet or tab), the background

knowledge contains chords properties such as root and bass note, category and degree as well as relations between successive chords such as interval between their root notes. Therefore in the induction phase, the ILP system, reasoning on these basic facts, extracts higher-level harmonic models based on chord properties instead of the bare chords themselves.

RQ2: Is it possible to leverage harmony's descriptive and expressive power to characterise music automatically? To what extent can harmony be used by itself to characterise styles?

Based on the observations made in Chapter 1 and the statistical work of Mauch et al. (2007), we reduced harmony to its most descriptive component: chord sequences. We experimented with several chord properties to represent those chord sequences. We discovered that the one combining both degree and chord category led to the best classification accuracy, the least complex models (fewer rules) and the fastest computation times. All this was achieved despite it being the richer representation (composite characteristic). This result was a notable discovery of this thesis as tonality (necessary to compute a chord degree) is very seldom employed in harmony-based genre classification.

As for characterising styles, our thesis work determined that, on symbolic or synthesised audio data, harmony-only models achieve results comparable to a state-of-the-art n-gram approach. However these models do not perform adequately on real audio data. Nevertheless, this thesis work introduces the idea of combining ILP generated harmony models as additional features to standard signal-based and statistical genre classifiers. We furthermore proved the additional high-level dimension statistically significantly improves the classification accuracy.

RQ3: Which representation schemes are suited to represent harmony and in particular chord sequences in ILP?

The harmony representation established in this work is our most noteworthy contribution. Unlike common representations of chord progressions, we produced a variable- and arbitrary-length representation of chord sequences, interspersed with gaps of unspecified length. We implemented a Context-Free Definite-Clause grammar representation which we integrated in the ILP software TILDE through a difference list paradigm. The gaps allowed us to capture musicological patterns spanning across an entire piece of music, while non fixed lengths added more freedom in the discovery of meaningful patterns. This scheme not only reproduces the human harmonic pattern analysis, but potentially extends its reach with patterns harder to identify manually.

RQ4: Which induction algorithms can best extract logic rules from Harmony?

As far as fixed length chord progressions are concerned, the Inverse Entailment software Aleph achieved fast lightweight computation but did not scale to the richer variable-length representations. The decision-tree induction algorithm TILDE overcame this difficulty by computationally supporting our CFDC grammar design. The most satisfying results, competing with statistical algorithms, were obtained when taking advantage of multiple decision trees at once in the ensemble method Random Forests.

RQ5: How can characterisation accuracy be evaluated?

We approached the evaluation both qualitatively and quantitatively. This work delivered a succinct musical analysis of the generated rules, identifying well known chord sequences and patterns in various genres for all experiments supporting our approach. In doing so we made available several databases of rules, both for fixed- and variable-length sequences. Generated on three datasets, this type of database could help musicologists in their harmony analysis of genres and composition styles.

Classification, as a binary task, provided a quantitative measure of the discriminative power of our characterisation. We performed extensive numerical studies evaluating the rigour, the limitations and the accuracy of our approach, providing, we hope, a solid basis for future research building on our findings.

7.3 Directions for Future Research

During the course of this thesis work, several research ideas for extensions and applications of this work have emerged. We share here a selection of them as potential directions for future research.

7.3.1 Further Developments of our Approach

There are various ways in which our harmony and ILP based approach to music characterisation could be expanded or further refined.

One shortcoming of the Random Forests is their complexity. The models comprising

several decisions trees are more difficult to comprehend than single decision trees (which can be represented as simple sets of rules). Hence if by using them we increased the classification accuracy of our models we also reduced their characterisation power. The difficulty of getting a human-readable description of the models when involving more than one tree has been addressed by Van Assche and Blockeel (2007). Using a method that they called RISM (standing for Relational Interpretable Single Model) they successfully approximated an ensemble of trees with a single interpretable tree capturing the same information, "*retaining some of the accuracy gains of the ensemble, while providing a model that keeps most of the comprehensibility of a single model directly learned on the data*". We could not test the RISM algorithm as it was not yet publicly available at the time but believe it is a promising technique for turning our more accurate random forests into interpretable models.

Automatically generated rules from examples may still not yet be on the same foot as rules tailored by experts, closer to a musician's understanding of harmony. We believe that extending the simple representation of harmony as a list of chord symbols, with the introduction of deeper harmonic concepts such as voice leading, would help capture other important aspects of harmonic style. It seems to us that inductive approaches like ours are still using simplistic representations of harmony, while the neighbouring domain of computational models of harmony applies their rich expert models to small hand-picked examples. Examples of such rich models can be found in the seminal work of Steedman (1984) as well as in more recent studies such as (Bergeron, 2010; de Haas, 2012; Mearns, 2013; Granroth-Wilding, 2013). A long-term plan would be to merge the strong points of both these branches of research.

Moreover harmony is only one dimension of music. As we have shown studying the case of harmony, ILP relational representation makes it possible to capture the temporality of musical events as a score would do, while the background knowledge inherent to ILP systems ensures that a pre-analysis of the musical data is performed and that the final models incorporate higher-level musical properties. The same approach could be applied to other musicological concepts requiring a preprocessing step consisting of a musical analysis of score-like data, such as rhythmic, melodic or musical structure concepts. Exploring those other musical facets, as well as combining several of them in such a framework could lead to powerful characterisation models. We imagine such a multi-faceted ILP system could induce rules such as this one:
IF 12_bar_structure(X) AND blues_scale(X) AND syncopation(X) THEN blues(X)
where X is a piece of music.

Chapter 6 showed the advantages of combining a logic-based approach with a statistical one. We believe that combinations of relational and statistical or probabilistic approaches

could be further explored in order to model for instance uncertainty or simultaneity. Hence one could extend our genre classification framework to perform multi-label classification. We would recommend tapping into the fields of Statistical Relational Learning (Getoor and Taskar, 2007) and Probabilistic Inductive Logic Programming (De Raedt et al., 2008), which have been concerned with the development of such hybrid induction methods, to replace our traditional relational decision trees and random forests.

Furthermore the MIR community has recently been focusing on large-scale datasets to ensure algorithms realistically scale to the size of commercial music databases (Bertin-Mahieux et al., 2011). In our case, we could not only extend our work to experiments on large-scale audio datasets, but could also like Macrae (2012) tap into "*the data deluge that is the web*" by collecting online large quantities of free crowd-sourced tabs and lead sheets, and running large-scale tests of our characterisation approach on them.

7.3.2 Potential Applications of our Work

We envision several applications for our work, some of which build on initial experiments we performed ourselves.

Exploiting the flexible chord progression representation we introduced in this thesis, we have already demonstrated a simple query-by-chord-sequence system at the SIGMUS workshop 2009 in Tokyo (Anglade et al., 2009a). Implemented in Prolog it tapped directly into the list representations of the pieces of music and the background knowledge built for our 9 genre experiments and was deployed on the *Perez-9-genre* dataset. Here is a description of that query system from (Anglade et al., 2009a):

> *"Through a friendly interface the user can query chord sequences of any length, specifying for each chord either the root note, the category, the degree or a combination of these. One extreme but admissible example of chord sequence to query is: [A7 IV B Vmin]. Sequences containing gaps such as [Amin G7 ... C9 ... Emaj7] are also allowed. Depending on the task the coverage of a chord sequence on each genre/subgenre or on only one given genre can be looked at. The name of the music pieces and the precise location of the chord sequence in the piece can be displayed if required. The user can also specify if he wants to consider only pieces in a given mode (major or minor)."*

We believe such a query system could become an assisting tool for musicologists who would want to explore a dataset and collect statistics and examples of chord patterns they have in mind.

Similarly the human-readable sets of rules generated by our experiments in Chapter 4, 5 and 6 could be further analysed by musicologists to identify and discover chord sequences characterising the Beatles and the various genres we have studied. A small project could consist in building an interface to facilitate their browsing.

Moreover, as harmonic modelling has proven to be useful for search and retrieval purposes (Pickens et al., 2002), we could further extend the query system presented above to support querying by symbolic or audio examples, placing our own models of harmony at the core of the retrieval process. We imagine the resulting meta-query system could then support queries going both directions (from examples to characteristic chord sequences and vice versa) but also retrieving similar music pieces based on their harmonic properties.

Lastly we believe our approach could have several automatic recommendation applications. Indeed as pointed out in Section 1.1.1, the transparency and expressiveness of first-order logic is very suitable for automatic recommender systems. Moreover harmony can be an interesting and innovative approach to recommend music.

One obvious way of using harmony in recommender systems would be to recommend chord progressions to a composer who wants to write in some particular style. As they are now our models of genres are not generative since they contain gaps of unspecified length. They can however be used to support the composition process by identifying parts of characteristic chord progressions and suggesting the chords that would complement them to the composer.

Another recommendation application would be to embed a chord progression clustering algorithm in an e-learning program for guitarists. As a side research project, we have contributed to Hotttabs, a multimedia guitar tutor (Barthet et al., 2011). It makes use of the vast amount of guitar tabs and lead sheets available online to provide guitarists with scores of recent popular songs. One of its key features is for each song to retrieve many tablatures and cluster them by difficulty, making it easier for users to pick one that matches their skills or one that would help them learn new, more difficult chord progressions. As it is now the clustering is only based on the size of the chord vocabulary on each tablature. We would like to extend it to clusters of tabs that share similar chord sequences, allowing the user to easily identify differences in styles, such as the ones found in cover songs or arrangements to other musical genres, expanding their guitar learning experience.

Finally since harmony captures some elements of style we believe that even for a casual user who does not necessarily know much about harmony, browsing, recommending and playlisting music collections based on their frequent chord sequences could be a new and interesting way

of discovering or rediscovering music.

Bibliography

Anglade, A., Benetos, E., Mauch, M., and Dixon, S. (2010). Improving music genre classification using automatically induced harmony rules. *Journal of New Music Research*, 39(4):349–361. [↪ pages 24 and 26]

Anglade, A. and Dixon, S. (2008a). Characterisation of harmony with inductive logic programming. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, pages 63–68, Philadelphia, PA. [↪ pages 23 and 25]

Anglade, A. and Dixon, S. (2008b). Towards logic-based representations of musical harmony for classification, retrieval and knowledge discovery. In *Proceedings of the 1st International Workshop on Machine Learning and Music (MML), co-located with the 25th International Conference on Machine Learning (ICML), the 24th Conference on Uncertainty in Artificial Intelligence (UAI) and the 21st Annual Conference on Learning Theory (COLT)*, pages 11–12, Helsinki, Finland. [↪ pages 23 and 25]

Anglade, A., Ramirez, R., and Dixon, S. (2009a). Computing genre statistics of chord sequences with a flexible query system. Demo at the SIGMUS Symposium. [↪ pages 24 and 119]

Anglade, A., Ramirez, R., and Dixon, S. (2009b). First-order logic classification models of musical genres based on harmony. In *Proceedings of the 6th Sound and Music Computing Conference (SMC)*, pages 309–314, Porto, Portugal. [↪ pages 23, 26, and 79]

Anglade, A., Ramirez, R., and Dixon, S. (2009c). Genre classification using harmony rules induced from automatic chord transcriptions. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR 2009)*, pages 669–674, Kobe, Japan. [↪ pages 23, 26, and 79]

Aucouturier, J.-J., Defreville, B., and Pachet, F. (2007). The bag-of-frames approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music. *Journal of the Acoustical Society of America*, 122(2):881–891. [↪ pages 37 and 38]

Aucouturier, J.-J. and Pachet, F. (2003). Representing musical genre: A state of the art. *Journal of New Music Research*, 32(1):83–93. [↪ page 36]

Aucouturier, J.-J. and Pachet, F. (2004). Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences*, 1(1):1–13. [↪ pages 37 and 38]

Aucouturier, J.-J. and Pachet, F. (2008). A scale-free distribution of false positives for a large class of audio similarity measures. *Pattern recognition*, 41(1):272–284. [↪ pages 37 and 38]

Barthet, M., Anglade, A., Fazekas, G., Kolozali, S., and Macrae, R. (2011). Music recommendation for music learning: Hotttabs, a multimedia guitar tutor. In *Proceedings of the 2nd Worskshop on Music Recommendation and Discovery (WOMRAD), co-located with the 5th ACM Recommender Systems Conference (ACM RecSys)*, pages 7–13, Chicago, IL, USA. [↪ pages 24 and 120]

Basili, R., Serafini, A., and Stellato, A. (2004). Classification of musical genre: a machine learning approach. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, pages 505–508, Barcelona, Spain. [↪ page 37]

Benetos, E. and Kotropoulos, C. (2010). Non-negative tensor factorization applied to music genre classification. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(8):1955–1967. [↪ page 99]

Berenzweig, A., Logan, B., Ellis, D. P., and Whitman, B. (2004). A large-scale evaluation of acoustic and subjective music-similarity measures. *Computer Music Journal*, 28(2):63–76. [↪ page 37]

Bergeron, M. (2010). *Structured Polyphonic Patterns*. PhD thesis, City University London, London, UK. [↪ page 118]

Bergmann, B. and Hommel, G. (1988). Improvements of general multiple test procedures for redundant systems of hypotheses. In Bauer, P., Hommel, G., and Sonnemann, E., editors, *Multiple Hypothesenprüfung / Multiple Hypotheses Testing*, volume 70 of *Medizinische Informatik und Statistik*, pages 100–115. Springer Berlin Heidelberg. [↪ page 80]

Bertin-Mahieux, T., Eck, D., and Mandel, M. I. (2010). Automatic tagging of audio: The state-of-the-art. In Wang, W., editor, *Machine Audition: Principles, Algorithms and Systems*, chapter 14, pages 334–352. IGI Publishing. [↪ page 36]

Bertin-Mahieux, T., Ellis, D. P., Whitman, B., and Lamere, P. (2011). The million song dataset. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*, pages 591–596, Miami, FL, USA. [↪ page 119]

Blockeel, H. and De Raedt, L. (1998). Top down induction of first-order logical decision trees. *Artificial Intelligence*, 101(1-2):285–297. [↪ pages 48, 49, 68, and 74]

Bratko, I. and Muggleton, S. (1995). Applications of Inductive Logic Programming. *Communications of the ACM*, 38(11):65–70. [↪ page 49]

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32. [↪ page 88]

Burred, J. J. and Lerch, A. (2003). A hierarchical approach to automatic musical genre classification. In *Proceedings of the 6th International Conference on Digital Audio Effects (DAFx)*, pages 8–11, London, UK. [↪ page 100]

Byrd, D. and Crawford, T. (2002). Problems of music information retrieval in the real world. *Information Processing and Management*, 38(2):249–272. [↪ page 15]

Cano, P., Koppenberger, M., and Wack, N. (2005). Content-based music audio recommendation. In *Proceedings of the 13th annual ACM International Conference on Multimedia (ACM MM)*, pages 211–212, Singapore. [↪ page 37]

Casey, M. and Slaney, M. (2006). The importance of sequences in musical similarity. In *Proceedings of the 2006 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5–8, Toulouse, France. [↪ page 38]

Cataltepe, Z., Yaslan, Y., and Sonmez, A. (2007). Music genre classification using MIDI and audio features. *EURASIP Journal on Advances in Signal Processing*, 2007:036409. [↪ page 38]

Chen, L., Wright, P., and Nejdl, W. (2009). Improving music genre classification using collaborative tagging data. In *Proceedings of the 2nd ACM International Conference on Web Search and Data Mining (WSDM)*, pages 84–93, Barcelona, Spain. [↪ page 38]

Claveau, V., Sébillot, P., Fabre, C., and Bouillon, P. (2003). Learning semantic lexicons from a part-of-speech and semantically tagged corpus using inductive logic programming. *Journal of Machine Learning Research*, 4:493–525. [↪ page 49]

Cope, D. (1996). *Experiments in musical intelligence*. AR Editions Madison, WI. [↪ page 36]

Davies, M. E. P., Plumbley, M. D., and Eck, D. (2009). Towards a musical beat emphasis function. In *Proceedings of the 2009 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 61–64, New Paltz, NY, USA. [↪ page 78]

de Haas, B. (2012). *Music information retrieval based on tonal harmony*. PhD thesis, Utrecht University, Utrecht, The Netherlands. [↪ page 118]

De Raedt, L., Frasconi, P., Kersting, K., and Muggleton, S. H., editors (2008). *Probabilistic inductive logic programming, Theory and Applications*, volume 4911 of *Lecture Notes in Artificial Intelligence*. Springer Berlin Heidelberg. [↪ page 119]

Dehaspe, L. (1999). Frequent pattern discovery in first-order logic. *AI Communications*, 12(1-2):115 – 117. [↪ page 49]

Dehaspe, L. and Toivonen, H. (1999). Discovery of frequent datalog patterns. *Data Mining and Knowledge Discovery*, 3(1):7–36. [↪ page 49]

Dehaspe, L. and Toivonen, H. (2001). *Relational Data Mining*, chapter Discovery of Relational Association Rules, pages 189–212. In Džeroski and Lavrac (2001b). [↪ page 49]

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30. [↪ page 80]

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1923. [↪ page 80]

Dixon, S., Mauch, M., and Anglade, A. (2011). Probabilistic and logic-based modelling of harmony. In Ystad, S., Aramaki, M., Kronland-Martinet, R., and Jensen, K., editors, *Exploring Music Contents, 7th International Symposium, CMMR 2010*, volume 6684 of *Lecture Notes in Computer Science*, pages 1–19. Springer Berlin Heidelberg. [↪ page 24]

Dovey, M. J. (1995). Analysis of Rachmaninoff's piano performances using inductive logic programming. In Lavrac, N. and Wrobel, S., editors, *Machine Learning: ECML-95*, volume 912 of *Lecture Notes in Computer Science*, pages 279–282. Springer Berlin Heidelberg. [↪ page 50]

Downie, J. S., Byrd, D., and Crawford, T. (2009). Ten years of ISMIR: reflections on challenges and opportunities. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, pages 13–18, Kobe, Japan. [↪ page 15]

Džeroski, S. (2006). From inductive logic programming to relational data mining. In Fisher, M., Hoek, W., Konev, B., and Lisitsa, A., editors, *Logics in Artificial Intelligence*, volume 4160 of *Lecture Notes in Artificial Intelligence*, pages 1–14. Springer Berlin Heidelberg. [↪ page 48]

Džeroski, S., Cussens, J., and Manandhar, S. (2000). An introduction to inductive logic programming and learning language in logic. In Cussens, J. and Džeroski, S., editors, *Learning Language in Logic*, volume 1925 of *Lecture Notes in Artificial Intelligence*, pages 3–35. Springer Berlin Heidelberg. [↪ pages 47 and 49]

Džeroski, S. and Lavrac, N. (2001a). *Relational Data Mining*, chapter An Introduction to Inductive Logic Programming, pages 48–73. In Džeroski and Lavrac (2001b). [↪ page 17]

Džeroski, S. and Lavrac, N., editors (2001b). *Relational Data Mining*. Springer Berlin Heidelberg. [↪ pages 48, 125, 126, 128, and 133]

Emde, W. and Wettschereck, D. (1996). Relational instance-based learning. In Kaufmann, M., editor, *Proceedings of the 13th International Conference on Machine Learning (ICML)*, pages 122–130, Bari, Italy. [↪ page 49]

Fisher, R. A. (1956). *Statistical Methods and Scientific Inference*. Oliver & Boyd, Edinburgh. [↪ page 80]

Flexer, A., Schnitzer, D., and Schlüter, J. (2012). A MIREX meta-analysis of hubness in audio music similarity. In *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR)*, pages 175–180, Porto, Portugal. [↪ page 38]

Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701. [↪ page 80]

Fukunaga, K. (1990). *Introduction to Statistical Pattern Recognition*. Academic Press Inc., San Diego, CA. [↪ page 100]

García, S. and Herrera, F. (2008). An extension on "Statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research*, 9:2677–2694. [↪ pages 80, 81, 82, 83, and 87]

Getoor, L. and Taskar, B. (2007). *Introduction to statistical relational learning*. The MIT press. [↪ page 119]

Gómez, E. (2006). *Tonal Description of Music Audio Signals*. PhD thesis, Universitat Pompeu Fabra, Barcelona, Spain. [↪ pages 37 and 79]

Gouyon, F. and Dixon, S. (2005). A review of automatic rhythm description systems. *Computer Music Journal*, 29(1):34–54. [↪ page 37]

Granroth-Wilding, M. (2013). *Harmonic Analysis of Music Using Combinatory Categorial Grammar*. PhD thesis, University of Edinburgh, Edinburgh, UK. [↪ page 118]

Harte, C. (2010). *Towards Automatic Extraction of Harmony Information from Music Signals*. PhD thesis, Queen Mary, University of London, London, UK. [↪ pages 20, 55, and 112]

Harte, C. and Sandler, M. (2005). Automatic chord identifcation using a quantised chromagram. In *Proceedings of the 118th Audio Engineering Society Convention*, Barcelona, Spain. [↪ page 78]

Harte, C., Sandler, M., Abdallah, S., and Gómez, E. (2005). Symbolic representation of musical chords: a proposed syntax for text annotations. In *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pages 66–71, London, UK. [↪ pages 32 and 33]

Herlocker, J. L., Konstan, J. A., and Riedl, J. (2000). Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work (CSCW)*, pages 241–250, Philadelphia, PA, USA. [↪ page 16]

Hillewaere, R., Manderick, B., and Conklin, D. (2009). Global feature versus event models for folk song classification. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR)*, pages 729–733, Kobe, Japan. [↪ page 39]

Horváth, T., Wrobel, S., and Bohnebeck, U. (2001). Relational instance-based learning with lists and terms. *Machine Learning*, 43(1-2):53–80. [↪ page 49]

ISMIR (2004). ISMIR audio description contest. `http://ismir2004.ismir.net/ISMIR_Contest.html`. [↪ pages 21 and 102]

Karydis, I., Radovanovic, M., Nanopoulos, A., and Ivanovic, M. (2010). Looking through the "glass ceiling": A conceptual framework for the problems of spectral similarity. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 267–272, Utrecht, The Netherlands. [↪ page 38]

Kirsten, M., Wrobel, S., and Horváth, T. (2001). *Relational Data Mining*, chapter Distance Based Approaches to Relational Learning and Clustering, pages 213–232. In Džeroski and Lavrac (2001b). [↪ page 49]

Kramer, S. (1996). Structural regression trees. In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI)*, pages 812–819, Portland, OR, USA. [↪ page 49]

LaRue, J. (1970). *Guidelines for Style Analysis*. W. W. Norton & Co. [↪ page 50]

Lavrac, N. and Džeroski, S. (1994). *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, New York. [↪ pages 43 and 49]

Lawson, C. L. and Hanson, R. J. (1974). *Solving Least Squares Problems*, chapter 23. Prentice-Hall. [↪ page 78]

Lee, J. H., Jones, M. C., and Downie, J. S. (2009). An analysis of ISMIR proceedings: patterns of authorship, topic and citation. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR)*, pages 57–62, Kobe, Japan. [↪ page 36]

Lee, K. (2007). A system for automatic chord transcription using genre-specific hidden markov models. In *Proceedings of the 5th International Workshop on Adaptive Multimedia Retrieval (AMR)*, pages 134–146, Paris, France. [↪ page 40]

Lidy, T., Rauber, A., Pertusa, A., and Iñesta, J. M. (2007). Improving genre classification by combination of audio and symbolic descriptors using a transcription system. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, pages 61–66, Vienna, Austria. [↪ page 38]

Lloyd, J. (1987). *Foundations of Logic Programming*. Springer Berlin, 2nd edition. [↪ page 44]

Lukashevich, H. M., Abeßer, J., Dittmar, C., and Großmann, H. (2009). From multi-labeling to multi-domain-labeling: A novel two-dimensional approach to music genre classification. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR)*, pages 459–464, Kobe, Japan. [↪ page 36]

Maclaren, H. (2003). *A Divide and Conquer Approach to Using Inductive Logic Programming for Learning User Models*. PhD thesis, University of York. [↪ page 47]

Macrae, R. (2012). *Linking Music Metadata*. PhD thesis, Queen Mary, University of London, London, UK. [↪ page 119]

Mauch, M. (2010). *Automatic Chord Transcription from Audio Using Computational Models of Musical Context*. PhD thesis, Queen Mary, University of London, London, UK. [↪ pages 21, 77, 78, 79, and 113]

Mauch, M., Dixon, S., Harte, C., Casey, M., and Fields, B. (2007). Discovering chord idioms through Beatles and Real Book songs. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, pages 255–258, Vienna, Austria. [↪ pages 35, 54, 56, 58, 61, 65, 112, and 116]

McKay, C. and Fujinaga, I. (2006). Musical genre classification: is it worth pursuing and how can it be improved? In *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, pages 101–106, Victoria, Canada. [↪ page 38]

McKay, C. and Fujinaga, I. (2007). Style-independent computer-assisted exploratory analysis of large music collections. *Journal of Interdisciplinary Music Studies*, 1(1):63–85. [↪ page 35]

McKay, C. and Fujinaga, I. (2008). Combining features extracted from audio, symbolic and cultural sources. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, pages 597–602, Philadelphia, PA, USA. [↪ page 38]

McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157. [↪ page 109]

Mearns, L. (2013). *The Computational Analysis of Harmony in Western Art Music*. PhD thesis, Queen Mary, University of London, London, UK. [↪ page 118]

Meng, A., Ahrendt, P., and Larsen, J. (2005). Improving music genre classification by short-time feature integration. In *In Proceedings of the 30th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 5, pages v/497 – v/500, Philadelphia, PA, USA. [↪ page 38]

Meyer, L. (1989). *Style and Music: Theory, History, and Ideology*. University of Chicago Press. [↪ page 35]

Michalski, R. S. (1980). Pattern recognition as rule-guided inductive inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(4):349–361. [↪ page 45]

Morales, E. F. (1997). PAL: A pattern-based first-order inductive system. *Machine Learning*, 26:227–252. [↪ page 51]

Muggleton, S. (1991). Inductive Logic Programming. *New Generation Computing*, 8(4):295–318. [↪ pages 17, 43, and 71]

Muggleton, S., Bryant, C. H., Srinivasan, A., Whittaker, A., Topp, S., and Rawlings, C. (2001). Are grammatical representations useful for learning from biological sequence data? - a case study. *Journal of Computational Biology*, 8(5):493–522. [↪ page 71]

Narmour, E. (1990). *The analysis and cognition of basic melodic structures: The implication-realization model*. University of Chicago Press. [↪ page 51]

Nemenyi, P. (1963). *Distribution-free multiple comparisons*. PhD thesis, Princeton University, Princeton, NJ, USA. [↪ page 80]

Neumayer, R. and Rauber, A. (2007). Integration of text and audio features for genre classification in music information retrieval. In *Proceedings of the 29th European Conference on Information Retrieval (ECIR)*, pages 724–727, Rome, Italy. [↪ page 38]

Pampalk, E. (2006). *Computational Models of Music Similarity and their Application in Music Information Retrieval*. PhD thesis, Vienna University of Technology, Vienna, Austria. [↪ page 38]

Pampalk, E., Dixon, S., and Widmer, G. (2004). Exploring music collections by browsing different views. *Computer Music Journal*, 28(2):49–62. [↪ page 100]

Peretz, I. and Coltheart, M. (2003). Modularity of music processing. *Nature Neuroscience*, 26(7):688–691. [↪ page 37]

Pérez-Sancho, C. (2009). *Stochastic language models for music information retrieval*. PhD thesis, Universidad de Alicante, Alicante, Spain. [↪ pages 20, 35, 39, 76, and 79]

Pérez-Sancho, C., Rizo, D., and Iñesta, J. M. (2008). Stochastic text models for music categorization. *Lecture Notes in Computer Science*, 5342:55–64. [↪ pages 39 and 85]

Pérez-Sancho, C., Rizo, D., and Iñesta, J. M. (2009). Genre classification using chords and stochastic language models. *Connection Science*, 21(2 & 3):145–159. [↪ pages 39 and 76]

Pérez-Sancho, C., Rizo, D., Iñesta, J. M., Ponce de León, P. J., Kersten, S., and Ramirez, R. (2010). Genre classification of music by tonal harmony. *Intelligent Data Analysis*, 14:533–545. [↪ pages 39, 76, 85, 86, 88, and 91]

Pickens, J., Bello, J. P., Monti, G., Crawford, T., Dovey, M., Sandler, M., and Byrd, D. (2002). Polyphonic score retrieval using polyphonic audio queries: A harmonic modeling approach. In *Journal of New Music Research*, pages 140–149. [↪ page 120]

Piston, W. (1987). *Harmony*. Norton, W. W. & Company, Inc., 5th edition. [↪ pages 18, 29, 35, and 93]

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc. [↪ page 48]

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286. [↪ page 78]

Raimond, Y., Abdallah, S., and Sandler, M. (2007). The Music Ontology. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, Vienna, Austria. [↪ pages 33 and 55]

Ramirez, R. (2003). Inducing musical rules with ILP. In *Proceedings of the 19th International Conference on Logic Programming (ICLP)*, volume 2916 of *Lecture Notes in Computer Science*, pages 502–504. Springer. [↪ page 51]

Ramirez, R. and Hazan, A. (2006). A tool for generating and explaining expressive music performances of monophonic jazz melodies. *International Journal on Artificial Intelligence Tools*, 15(4):673–691. [↪ page 52]

Ramirez, R., Hazan, A., Gómez, E., and Maestre, E. (2004). Understanding expressive transformations in saxophone jazz performances using inductive machine learning. In *Proceedings of the 1st Sound and Music Computing Conference (SMC)*, Paris, France. [↪ pages 51 and 52]

Roy, P., Pachet, F., and Krakowski, S. (2007). De l'intérêt des features analytiques : une étude pour la classification des sons de pandeiro. In *Actes des Journées d'Informatique Musicale (JIM)*, Lyon, France. [↪ page 37]

Salzberg, S. L. (1997). On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, 1(3):317–328. [↪ page 80]

Scaringella, N., Zoia, G., and Mlynek, D. (2006). Automatic genre classification of music content: a survey. *IEEE Signal Processing Magazine*, 23(2):133–141. [↪ pages 36, 37, 99, and 100]

Schedl, M., Flexer, A., and Urbano, J. (2013). The neglected user in music information retrieval research. *Journal of Intelligent Information Systems*, 41(3):523–539. [↪ page 16]

Schölkopf, B., Burges, C. J. C., and Smola, A. J. (1999). *Advances in Kernel Methods: Support Vector Learning*. MIT Press, Cambridge, MA, USA. [↪ page 102]

Shan, M.-K., Kuo, F.-F., and Chen, M.-F. (2002). Music style mining and classification by melody. In *Proceedings of 2002 IEEE International Conference on Multimedia and Expo (ICME)*, volume 1, pages 97–100. [↪ page 39]

Shen, J., Shepherd, J., and Ngu, A. H. (2006). InMAF: Indexing music databases via multiple acoustic features. In *Proceedings of the 2006 ACM International Conference on Management of Data (ACM SIGMOD)*, pages 778–780, Chicago, Illinois, USA. [↪ page 37]

Sinha, R. and Swearingen, K. (2002). The role of transparency in recommender systems. In *2002 Conference on Human Factors in Computing Systems (CHI)*, pages 830–831, Minneapolis, Minnesota, USA. [↪ page 16]

Srinivasan, A. (2003). The Aleph Manual (version 4). `http://www.cs.ox.ac.uk/activities/machlearn/Aleph/aleph.html`. [↪ pages 45, 47, 55, 58, and 112]

Steedman, M. J. (1984). A generative grammar for jazz chord sequences. *Music Perception*, pages 52–77. [↪ page 118]

Student (1908). The probable error of a mean. *Biometrika*, 6(1):1–25. [↪ page 80]

Sturm, B. (2012). A survey of evaluation in music genre recognition. In *10th International Workshop on Adaptive Multimedia Retrieval (AMR)*, Copenhagen, Denmark. [↪ page 36]

Swartout, W., Paris, C., and Moore, J. (1991). Explanations in knowledge systems: design for explainable expert systems. *IEEE Expert*, 6(3):58–64. [↪ page 16]

Taminau, J., Hillewaere, R., Meganck, S., Conklin, D., Nowé, A., and Manderick, B. (2009). Descriptive subgroup mining of folk music. In *Proceedings of the 2nd International Workshop on Machine Learning and Music (MML)*, pages 1–6, Bled, Slovenia. [↪ page 35]

Tintarev, N. and Masthoff, J. (2007). A survey of explanations in recommender systems. In *Proceedings of the Workshop on Recommender Systems and Intelligent User Interfaces, co-located with the 23rd IEEE International Conference on Data Engineering*, pages 801–810, Istanbul, Turkey. [↪ page 16]

Tzanetakis, G. (2007). Marsyas: a case study in implementing music information retrieval systems. In Shen, J., Shepherd, J., Cui, B., and Liu, L., editors, *Intelligent Music Information Systems: Tools and Methodologies*, pages 31–49. Idea Group Reference. [↪ page 107]

Tzanetakis, G. and Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302. [↪ pages 21, 37, 99, 100, and 102]

Tzanetakis, G., Ermolinskyi, A., and Cook, P. (2003). Pitch histograms in audio and symbolic music information retrieval. *Journal of New Music Research*, 32(2):143–152. [↪ pages 39 and 99]

Van Assche, A. (2008). *Improving the Applicability of Ensemble Methods in Data Mining*. PhD thesis, Katholieke Universiteit Leuven. [↪ pages 49 and 88]

Van Assche, A. and Blockeel, H. (2007). Seeing the forest through the trees: Learning a comprehensible model from an ensemble. In *Proceedings of The 18th European Conference on Machine Learning (ECML)*, volume 4701 of *Lecture Notes in Computer Science*, pages 418–429. Springer-Verlag. [↪ page 118]

Van Baelen, E. and De Raedt, L. (1996). of piano performances using inductive logic programming. In *Proceedings of the 6th International Workshop in Inductive Logic Programming (ILP)*, volume 1314 of *Lecture Notes in Artificial Intelligence*, pages 55–71. Springer-Verlag. [↪ pages 50 and 52]

van der Hedjen, F., Duin, R. P. W., de Ridder, D., and Tax, D. M. J. (2004). *Classification, Parameter Estimation and State Estimation*. Wiley, London, UK. [↪ page 100]

van Kranenburg, P. (2006). Composer attribution by quantifying compositional strategies. In *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, pages 375–376, Victoria, Canada. [↪ pages 35 and 36]

Van Laer, W. and De Raedt., L. (2001). *Relational Data Mining*, chapter How to Upgrade Propositional Learners to First Order Logic: A Case Study, pages 235–261. In Džeroski and Lavrac (2001b). [↪ page 49]

various (2004). *The Real Book*. Hal Leonard Corporation, 6th edition. [↪ pages 20 and 55]

Whitman, B. and Smaragdis, P. (2002). Combining musical and cultural features for intelligent style detection. In *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, pages 47–52, Paris, France. [↪ page 38]

Widmer, G. (2001). Discovering strong principles of expressive music performance with the PLCG rule learning strategy. In *Proceedings of the 12th European Conference on Machine Learning (ECML)*, volume 2167 of *Lecture Notes in Artificial Intelligence*, pages 552–563. Springer-Verlag. [↪ page 50]

Widmer, G. (2003). Discovering simple rules in complex data: A meta-learning algorithm and some surprising musical discoveries. *AI*, 146(2):129–148. [↪ page 50]

Widmer, G., Dixon, S., Goebl, W., Pampalk, E., and Tobudic, A. (2003). In search of the Horowitz factor. *AI Magazine*, 24(3):111–130. [↪ page 50]

Wright, S. P. (1992). Adjusted p-values for simultaneous inference. *Biometrics*, 48(4):1005–1013. [↪ page 81]

Zils, A. (2004). *Extraction de descripteurs musicaux: une approche évolutionniste*. PhD thesis, University of Paris 6, Paris, France. [↪ page 37]

Zils, A. and Pachet, F. (2003). Extracting automatically the perceived intensity of music titles. In *Proceedings of the 6th International Conference on Digital Audio Effects (DAFX)*, pages 47–50, London, UK. [↪ page 37]

Colophon

This thesis was typeset using the LaTeX typesetting system created by Leslie Lamport and a memoir class.
The body text is set 10pt on a 369pt measure with Minister Std Light designed by Carl Albert Fahrenwaldt.
Other font families include Inconsolata, designed by Raph Levien, and Helvetica.