UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA


# MACHINE LEARNING TECHNIQUES FOR MUSIC INFORMATION RETRIEVAL

Gonçalo Caetano Marques

Tese orientada pelo Prof. Dr. Thibault Langlois,
especialmente elaborada para a obtenção do grau de doutor em Informática
(especialidade Engenharia Informática)

2014

# Abstract

The advent of digital music has changed the rules of music consumption, distribution and sales. With it has emerged the need to effectively search and manage vast music collections. Music information retrieval is an interdisciplinary field of research that focuses on the development of new techniques with that aim in mind.

This dissertation addresses a specific aspect of this field: methods that automatically extract musical information exclusively based on the audio signal. We propose a method for automatic music-based classification, label inference, and music similarity estimation. Our method consist in representing the audio with a finite set of symbols and then modeling the symbols time evolution. The symbols are obtained via vector quantization in which a single codebook is used to quantize the audio descriptors. The symbols time evolution is modeled via a first order Markov process. Based on systematic evaluations we carried out on publicly available sets, we show that our method achieves performances on par with most techniques found in literature.

We also present and discuss the problems that appear when computers try to classify or annotate songs using the audio as the only source of information. In our method, the separation of quantization process from the creation and training of classification models helped us in that analysis. It enabled us to examine how instantaneous sound attributes (henceforth features) are distributed in term of musical genre, and how designing codebooks specially tailored for these distributions affects the performance of ours and other classification systems commonly used for this task. On this issue, we show that there is no apparent benefit in seeking a thorough representation of the feature space. This is a bit unexpected since it goes against the assumption that features carry equally relevant information loads and somehow capture the specificities of musical facets, implicit in many genre recognition methods.

Label inference is the task of automatically annotating songs with semantic words - this tasks is also known as autotagging. In this context, we illustrate the importance of a number of issues, that in our perspective, are often overlooked. We show that current techniques are fragile in the sense that small alterations in the set of labels may lead to dramatically different results. Furthermore, through a series of experiments, we show that autotagging systems fail to learn tag models capable to generalize to datasets of different origins. We also show that the performance achieved with these techniques is not sufficient to be able to take advantage of the correlations between tags.



**Keywords:** music information retrieval, machine learning, signal processing, spectral similarity, genre classification, autotaggings.

# Resumo

A revolução digital mudou as regras do consumo, distribuição, armazenamento e venda de música. Com ela surgiu a necessidade de poder eficazmente pesquisar e gerir coleções contendo um vasto número de músicas. A recolha de informação musical (do inglês *Music Information Retrieval* - MIR) é uma área de investigação dedicada ao desenvolvimento de novas técnicas de abordagem desse problema. Esta área abrange temas tão diversos como o processamento de sinais e de fala, aprendizagem automática, musicologia, psicologia, sociologia, entre tantos outros.

Muito destes temas estão fora do âmbito desta dissertação, e nesta tese cingimos-nos ao estudo e à análise de métodos computacionais para extracção de características musicais em que o sinal áudio é a única fonte de informação disponível.

Propomos um novo método para classificação automática de músicas. Adicionalmente mostramos que o nosso método é adaptável a outros problemas, nomeadamente na etiquetação automática de sinais musicais, e medição de semelhança entre músicas. No nosso trabalho foi dado ênfase a duas tarefas em particular, o reconhecimento automático de géneros musicais e a etiquetação automática de músicas. Estas duas tarefas têm sido alvo de variados estudos na comunidade científica MIR, e são analisadas em detalhe nesta dissertação. No entanto, foram também testadas outras tarefas em particular, a identificação automática de artista e a geração de listagens de músicas (i.e. *playlists*).

A nossa abordagem consiste em representar o áudio como um conjunto finito de símbolos e modelar a evolução temporal dos mesmos. Os símbolos são obtidos por meio de quantificação vectorial, em que uma única tabela de codificação (*codebook*) é usada para quantificar os descritores do áudio. Este é um processo não supervisionado, no sentido em que nesta fase os sinais musicais não estão associados a nenhuma classe ou etiqueta. A evolução temporal dos símbolos é modelada através de um processo de Markov de primeira ordem. Os modelos de Markov são estimados de um modo supervisionado sendo construído um modelo por classe, treinado com os símbolos resultantes das músicas que pertencem a essa mesma classe. Um dos benefícios da nossa abordagem está na separação do processo de quantificação do treino dos modelos de Markov. Isto permitiu-nos testar diferentes técnicas de quantificação e de modelos temporais, e diferentes combinações das mesmas.

Avaliamos o desempenho do nosso método nos resultados que obtivemos em bases de dados musicais, disponibilizadas por investigadores da área. Os resultados obtidos mostram que o desempenho do nosso método é comparável à maioria das técnicas propostas na literatura que

utilizaram os mesmos conjuntos de dados. No entanto realçamos que recentemente surgiram alguns métodos com melhores resultados que os nossos.

Parte do trabalho apresentado nesta dissertação foi dedicado a analisar os problemas que surgem quando computadores tentam automaticamente reconhecer géneros ou atribuir etiquetas a músicas, baseados apenas no sinal de áudio.

No que diz respeito ao reconhecimento automático de géneros musicais, analisamos a influência de várias representações discretas do áudio. As representações foram obtidas com métodos de quantificação vectorial através de um único codebook, abordagem que é igualmente usada no método proposto nesta dissertação. O processo de geração do codebook foi feito de modo a tirar partido de certas características dos dados. No processo de construção do codebook, exploramos a inclusão de mais ou menos informação, selecionando os dados mais representativos através de técnicas de k-médias ou de modelos probabilísticos dos dados, ou através de seleção aleatória dos mesmos. Investigamos igualmente a geração de codebooks, restringindo os dados de treino a um único género musical. Através de testes sistemáticos, medimos o impacto destas abordagens em vários métodos de classificação. Para além do método proposto, foram testadas outras duas técnicas de classificação frequentemente utilizadas no contexto de reconhecimento automático de géneros, donde salientamos o algoritmo do vizinho mais próximo e máquinas de suporte vectorial. Os resultados obtidos com estes classificadores mostram que representações discretas dos dados feitas de um modo aleatório ou não supervisionado, são tão eficientes como as geradas de modo metódico e supervisionado. Esta observação vai contra o pressuposto assumido por muitas técnicas utilizadas para o reconhecimento de géneros musicais, em que a estratégia da construção dos modelos de classificação é baseada em distribuições globais de descritores condicionadas às classes a que eles pertencem.

O objectivo da etiquetagem automática é caracterizar o sinal musical com informação textual referente a conceitos pré-definidos. Esta tarefa é conceptualmente diferente do problema de classificação em que as classes são mutuamente exclusivas. Em etiquetagem, várias etiquetas podem ser associadas à mesma instância. Neste dissertação analisamos em detalhe as métricas utilizadas para aferir o desempenho de sistemas de etiquetagem automática de músicas. É de notar que nesta tarefa, o processo de avaliação é mais complicado do que no caso normal de classificação, visto estar condicionado a vários factores que podem ter uma influência significativa nos resultados. Como exemplo cito que os valores médios referentes às métricas usadas podem ser obtidos de três maneiras distintas, por música, por etiqueta, ou de uma forma global. Quando aplicadas ao mesmo parâmetro, estas três maneiras the obter médias produzem valores significativamente diferentes entre si. Além disso, existem duas sub-tarefas associadas ao processo de etiquetagem, a anotação e a recolha. Em anotação o objectivo é atribuir a cada música etiquetas que a caracterizem. Em recolha, o processo consiste, a partir de uma etiqueta, obter as músicas que lhe estejam associadas. Todas estes condicionantes dificultam a comparação

de desempenhos. Estas questões são conhecidas, mas existem outras que têm sido descuradas, e que em nossa opinião devem ser tidas em conta. Uma delas tem haver com o número de etiquetas usadas no processo de avaliação. A análise que fizemos mostra que, para a mesma base de dados, a utilização de sub-conjuntos de etiquetas, uma prática comum na área, produz resultados muito diferentes. Outra questão é a capacidade de generalização destes sistemas. Em testes que levamos a cabo com o nosso e outro sistema (que ficou em primeiro lugar na tarefa de etiquetagem automática na principal competição da área de MIR) mostram que estes métodos não conseguem aprender modelos de etiquetas com capacidade para generalizar a dados de origens diferentes. Outra questão tem haver com a correlação entre etiquetas. É natural que certas etiquetas estejam associadas a outras, como etiquetas de instrumentação e etiquetas de género musical. Algumas abordagens tentam tirar partido deste facto e usam um segundo nível de processamento para "corrigir" a saída dos sistemas de etiquetagem. No entanto, mostramos que o desempenho dos sistemas actuais não é suficientemente elevado para beneficiar desta estratégia.

Outro assunto abordado nesta dissertação é a formação de *hubs*. Hubs são pontos que aparecem na lista de vizinhos mais próximos de um grande número de outros pontos. Este é um efeito que se faz sentir quando a dimensão dos dados é elevada. No contexto de sistemas de recomendação musical, hubs são músicas que são recomendadas frequentemente sem razão aparente. Este fenómeno tem sido observado por vários investigadores, e tem um efeito nocivo nestes sistemas. Os nossos estudos sobre hubs ainda se encontram numa fase inicial, mas os resultados obtidos mostram que em dados de alta dimensão, o desempenho de métodos de classificação fica comprometido, mesmo em casos considerados triviais.

**Palavras Chave:** recolha de informação musical, aprendizagem automática, processamento de sinal, semelhança espectral, classificação em géneros musicais, etiquetagem automática.

# Acknowledgements

First of all, I want to thank Professor Thibault Langlois. As my thesis supervisor, he gave me the opportunity to work with him as a partner, and the freedom to pursue my personal research interests. Furthermore, his advise was wise and practical, and he helped me overcome some difficulties that surface during the writing of this dissertation. The work here presented is a result of working shoulder to shoulder with Professor Thibault Langlois, and for that I am deeply grateful to him.

I also want to thank the researchers that collaborated we me in the publications that resulted from this thesis: Fabien Gouyon, Marcos Domingues, Miguel Lopes and Mohamed Sordo. A special thanks goes to Fabien Gouyon, and the Sound and Music Computing Group at INESC Porto. As its coordinator, Fabien organized meetings in Porto, where we held informal but inspirational discussions. Fabien also made possible my trip to the Pontifícia Universidade Católica do Paraná, in Curitiba Brasil, where I did research work in "Grupo de Pesquisas em Descoberta do Conhecimento e Aprendizagem de Máquina". For this, I also thank Alessandro Koerich for receiving me kindly and making me feel at home.

Among my colleagues at ISEL, a special thanks goes to João Beleza for his support, for his unconditional help, and for the discussions we held on matters that were not his research area. I want to thank Pedro Mendes Jorge – he was there for me when I needed his help. João and Pedro: I am very lucky to work with you guys! I also want to thank Arnaldo Abrantes for his support during this arduous journey.

Finally, I want to thank my family for enduring my sour moods during the writing of this thesis. You are my center and my sanity.

This research was supported by the PROTEC grant SFRH/BD/50118/2009

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Motivation

The advent of digital music changed the rules of music consumption, distribution and sales. Private digital collections started with a few compact disks, then migrated to thousands files stored on hard disk and music players. About ten years ago, the first on-line music store (Apple i-Tunes) opened. Nowadays there are hundreds of such on-line media stores and the bulk of their sales are rivaling the (once) traditional way of acquiring music. Furthermore, ubiquitous Internet connectivity makes it possible to access thousands of Internet radios along with a array of music services and collections anchored in "the cloud".

With the capacity of accessing vast music collections, has emerged the need to create new tools that can search and manage efficiently music databases, and recommend music of interest to the end user. These tools are the focus of Music Information Retrieval (MIR). MIR is a relatively young field of research that started to take its first steps about a decade and a half ago, and since then, it has received an ever-increasing attention from academia and the music industry communities. Music information retrieval is a highly diversified field with many facets that encompass (but are not limited to) psychological or physiological aspects of how humans perceive music, cultural, sociological and historical elements, knowledge and understanding of the music creation process, etc. Many of these subjects are beyond the scope of this thesis. This thesis focuses on automatic music recommendation techniques that rely uniquely on information extracted from the audio signal - these are known as content-based methods. The aim is to make the computer act as a musical expert, organize and recommend music according to user's taste, mood or other relevant criteria. The ultimate goal of these methods is to create systems capable of "listening" and "understanding" music on their own, but the aim of having a machine that perceives music as humans do, is still an utopia. Music is intrinsically related to human

perception and cognition. The composer Milton Babbitt [Babbitt, 1965] proposed the separation of music in three domains: acoustic (or physical), auditory (or perceived) and graphemic (or notated). Wiggins [Wiggins, 2009] uses this representation to eloquently describe Music as "an abstract and intangible cultural construct, which does not have physical existence in itself, but which is described by all three of these representational domains; in a sense, something like the notion of the Platonic ideal, but without actually existing in the real world". Wiggins further argues that the fundamental source of music, without which it cannot exist, is the mind. In other words, music lives primarily in auditory domain, and the other two, the audio signal and notation are applied to or derived from a cognitive process. Therefore, in order for computers to perceive and understand music as we do, they would have, to a certain extent, to think and feel as we do – a scenario that is still rooted in the realm of science fiction.

The problem can be simplified if we settle for music recommendation systems that are able to meet the demands of our musical needs - and not necessarily perceive music as humans do. In this aspect, substantial progress has been made the last decade, particularly in what concerns context-based methods. These methods, unlike content-based ones, derive recommendations from information inserted by human annotations (or their behavior, e.g., song ratings, general preferences, etc). Some important advances have also been made by content-based approaches, specially at identifying music sub-components like estimation of tempo or detecting note onsets. However identifying broad musical characteristics such as genre, mood, or the degree of similarity between songs, have proven to be a challenging task. To a certain extent, this was expected since these high-level musical facets are intrinsically associated with social, cultural, and personal interpretations of music – and it is questionable that such constructs can be inferred looking solely at the audio. What was a bit surprising, though, was the fact that the first content-based techniques achieved relatively good results (well above random), and they did this by disregarding the temporal information in the musical signals. Undeniably, the temporal dynamics are an important part of how we perceive music, and ignoring them is somewhat counter intuitive. Some attempts have been made to explicitly model temporal information (e.g. using hidden Markov Models), but they have encountered limited success. Nowadays, methods that ignore the temporal information are the standard way to infer genre, mood, and other high-level musical concepts. Nevertheless, these techniques have reached a limit in performance that seems hard to break, and in this dissertation we address some of the possible causes for this performance boundary.

## 1.2 Thesis Scope and Contributions

In this thesis, we propose a new approach that can be used for music-based classification, label inference, and music similarity estimation. Our strategy consist in representing the audio with a finite set of symbols and then modeling the symbols time evolution. The symbols are obtained via vector quantization in which a single codebook is used to quantize the audio descriptors. The symbols time evolution is modeled via a first order Markov process. Unimaginatively, we called our approach Vector Quantization Markov Models (VQMMs). One of the benefits of our approach is that the quantization process and the Markov models training are two self contained parts. The first is not supervised while the second is. This allows the use of different quantification schemes and time modeling techniques – we tested several quantization and time-modeling methods in the course of our research.

Our method achieves performances on par with most techniques found in literature, based on evaluations we carried out on publicly available sets, although a few works report better results[1]. Beyond that, the VQMM were a useful tool for analyzing issues specific to genre classification and label inference tasks (also known as *autotagging*).

The separation of the feature space partitioning from the creation and training of classification models enabled us to explicitly characterize distinct partition, and examine how features associated with different musical facets populate this space. This was particularly helpful in the analysis of low-level features and their impact on genre classification systems. On this issue, we show that there is no apparent benefit in seeking a thorough representation of this feature space. This is a bit unexpected since it goes against the assumption that features carry equally relevant information loads and somehow capture the specificities of musical facets, implicit in many genre recognition methods. This work resulted in two publications [Marques et al., 2010, 2011b].

In the context of autotagging, we illustrate the importance of a number of issues, poorely reported in literature. We show that current autotagging techniques are fragile in the sense that small alterations in the set of labels may lead to dramatically different results. Furthermore, through a series of experiments, we show that ours and another autotagging method (that obtained the top performances on a competition held annually be the MIR community) fail to learn tag models capable to generalize to datasets of different origins. We also show that the performance achieved with these techniques is not sufficient to be able to take advantage of the correlations between tags. This findings were published in [Marques et al., 2011a].

We also present our studies on the subject of hub formation. This is a phenomenon that occurs in high dimensional spaces where certain data points keep appearing as nearest neighbors

[1] This will be addressed in Section 3.4.2

of a large number of other points; this can have a negative impact on a wide range of music similarity techniques. Our investigation on this matter is still uncompleted, but the results obtained so far show that even in simple classification scenarios, where points are distributed amongst tight clusters, can become problematic in high dimensional spaces.

### 1.2.1 List of Contributions

The main contributions of this dissertation are the following:

1. Propose a new method for content-based audio music classification. The approach is based on a two step procedure. The first step is an unsupervised process and consists of a quantization of the features used to characterize audio signals. The second step is a supervised one, where models are learned based on the first order transitions of the discrete symbols obtained in the first step.

2. Study the influence of distinct partitions of the audio features with different genre-related information loads, and their influence on the overall performance of several genre classification algorithms.

3. Analyze the performances metrics used to evaluate autotagging systems and show a number of limitations associated with their use. Study the generalization capacity of different autotagging systems. Study the benefits of using a second processing stage in autotagging systems to exploit tag correlations, in order to enhance the quality of the tag attributions.

### 1.2.2 List of Publications

The research presented in this thesis has resulted in the collaborative publications listed below. We include with the reference of each paper, the number of citations returned by Google Scholar[2].

- G. Marques, M. Domingues, T. Langlois, and F. Gouyon. Three current issues in music autotagging. In *Proc. of the $12^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Maimi, U.S.A., 2011a
  (17 citations)
- G. Marques, T. Langlois, F. Gouyon, M. Lopes, and M. Sordo. Short-term feature space and music genre classification. *Journal of New Music Research*, 40(2):127–137, 2011b
  (13 citations)

[2] scholar.google.com (visited September 2014)

- G. Marques, M. Lopes, M. Sordo, T. Langlois, and F. Gouyon. Additional evidence that common low-level features of individual audio frames are not representative of music genre. In *Proc. of the $7^{th}$ Sound and Music Computing Conf. (SMC)*, Barcelona, Spain, 2010
(16 citations)
- T. Langlois, T. Chambel, E. Oliveira, P. Carvalho, G. Marques, and A. Falcão. Virus: Video information retrieval using subtitles. In *Proceedings of the 14th International Academic MindTrek Conference: Envisioning Future Media Environments*, MindTrek, pages 197–200, New York, NY, USA, 2010. ACM
(10 citations)
- T. Langlois and G. Marques. A music classification method based on timbral features. In *Proc. of the $10^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Kobe, Japan, 2009a
(29 citations)
- T. Langlois and G. Marques. Automatic music classification using a hierachical clustering and a language modeling approach. In *Proc. of the $1^{st}$ Int. Conf. on Advances in Multimedia (MMEDIA)*, Colmar, France, 2009b
(10 citations)
- G. Marques and T. Langlois. A language modeling approach for classification of audio music. In *DMIN*, Las Vegas, U.S.A., 2009
(0 citations)
- G. Marques and T. Langlois. A similarity measure for music signals. In *Proc. of the $10^{th}$ Int. Conf. on Enterprise Information Systems (ICEIS)*, Barcelona, Spain, 2008
(0 citations)

## 1.3 Organization of the Thesis

This thesis is structured as follows: Chapter 2 introduces and formalizes some important aspects of content-based music similarity measures, and the common tasks that are used as proxy for audio-similarity. It describes audio-based features used in these scenarios, and reviews the most relevant works in this area, with special focus on genre classification and autotagging methods. Chapter 3 presents our approach, the VQMMs, and how it can be adapted to deal with audio-based classification, label inference and music similarity estimation. It gives a broad evaluation of our method for several classification tasks and on a variety of music datasets and compares it to the state of the art. Chapter 4 addresses the problem of using low-level descriptors to infer high-level musical concept such as genre, and the limitations inherent with

this approach. Chapter 5 addresses the issue of automatically annotating a music piece and investigates several issues associated with autotagging. Chapter 6 presents our current work, discusses future research directions, and concludes this thesis.

# Chapter 2

# Content-Based Music Information Retrieval

This chapter is an introduction to content-based methods. Here we present the music and signal processing terminology and concepts necessary to understand content-based MIR, and we review the state of the art, paying special attention to genre classification and autotagging systems. We start in Section 2.1 with a brief overview of music information retrieval. We also give a brief description of context-based methods, which are the standard methodology used to ascertain similarity measures, and examine the differences between these and content-based methods, which are the focus of this thesis. Section 2.2 describes the audio features used in content-based MIR. We briefly describe the fundamentals of digital audio signal processing, in particular the time/frequency domain representations which are the basis of most of the low-level MIR features, including the ones used in our work. In Section 2.3 we focus on music classification tasks, since in a machine learning context, audio-based music similarity is often replaced by classification tasks, in which specific labels are assigned to each song based on genre, artist, mood, etc. In music classification scenarios each song comes with one or more labels, and the process of training classification systems corresponds to finding the mapping between features and labels. When songs are represented by a single vector, this mapping can be estimated with fairly standard machine learning techniques. This is the setting for the majority of music classification systems, which are reviewed in Section 2.3.1, with a special emphasis on genre classification. The reasons for the special status that genre classification has received from the MIR community are also presented (the problem of genre classification is further explored in Chapter 4). In Section 2.3.2, we address the problem of autotagging, a fairly new MIR topic compared to its classification counterparts, that has emerged from the unmoderated, unstructured, and collaborative processes of the social networking era we have recently entered. (the problem of autotagging is further explored in Chapter 5). This chapter finalizes with

some practical considerations and issues that can affect the performance of MIR content-based recommendation systems.

## 2.1 MIR - A Brief Overview

### 2.1.1 MIR in Academia

The first publications mentioning music information retrieval date back half a century [Kassler, 1966, Lincoln, 1967], but for decades the area did not received much attention. Nowadays, things have changed considerably, and the research interest has expanded significantly, in part due to the success of digital media and new ways of acquiring, listening and processing music collections. MIR research efforts have encompassed several areas such as musicology, music perception, cognition, and computer science and engineering domains such as signal and image processing, machine learning, automatic speech recognition, and text retrieval. MIR interdisciplinary character and the connection to neighboring fields has given the possibility to transfer and reuse methodologies that are common practice in other areas, and therefore, theses synergies have helped jump-start a considerable amount MIR research topics. Concurrently, the challenges that MIR faces have also contributed to the state of the art of other fields such as content-based image and multimedia retrieval [Weninger et al., 2012]. For a comprehensive overview of the MIR research, we refer to [Casey et al., 2008a, Klapuri and Davy, 2006, Orio, 2006].

The ISMIR (International Society for Music Information Retrieval[1]) conference is the hub point for MIR research. ISMIR is an annual conference that started in the 2000, and since then, the statistics on attendance and participation have grown considerably [Downie et al., 2009], manifesting the success and the increasing interest on this area. Held in conjunction with the ISMIR conference is the MIREX (Music Information Retrieval eXchange[2]) meeting. This is a community-based initiative that provides a framework for formal evaluation of MIR systems using centralized tasks, datasets, platforms, and evaluation methods [Cunningham et al., 2012, Downie, 2008], and every year new tasks are evaluated allowing for knowledge to be gained on the limits of current algorithms and techniques. To get an idea of the current focus of MIR research, we show in Table 2.1 a list of evaluation tasks proposed for the MIREX competition since its creation in 2005. It is noticeable the strong bias towards audio-based approaches. Only three tasks, Symbolic Genre Classification, Symbolic Key Finding, and Symbolic Melodic Similarity are strictly in the symbolic domain. Other three, Query by Tapping,

[1] www.ismir.net/

[2] www.music-ir.org/mirex

- Audio Artist Identification
- Audio Beat Tracking
- Audio Classical Composer Identification
- Audio Chord Detection
- Audio Cover Song Identification
- Audio Downbeat Estimation
- Audio Drum Detection
- Audio Genre Classification
- Audio Key Detection
- Audio Melody Extraction
- Audio Mood Classification
- Audio Music Similarity and Retrieval
- Audio Onset Detection
- Audio Tag Classification
- Audio Tempo Estimation
- Discovery of Repeated Themes & Sections
- Multiple Fundamental Freq. Estimation & Tracking
- Music Structure Segmentation
- Query by Tapping
- Query by Singing/Humming
- Real-time Audio to Score Alignment (Score Following)
- Symbolic Genre Classification
- Symbolic Key Finding
- Symbolic Melodic Similarity

Table 2.1: List of MIR tasks proposed for the MIREX evaluations (from 2005 to 2014).

Query by Singing/Humming, and Score Following, although included in the audio-based class, are actually hybrid tasks since they try to match the audio input against symbolic representations. All other tasks are audio-based ones; according to Downie [Downie, 2008] there are three major reasons for the predominance of audio-based approaches. The first is that many MIR researchers come from signal processing background (e.g. electric engineering, acoustics, speech processing, etc). The second is that symbolic music representations, not restricted to the genre "classical" are quite rare, while music in its audio form is quite easy to obtain. The third is that dealing with music in its audio form requires less music-specific knowledge than with its symbolic form.

Among the twenty four tasks listed in Table 2.1, almost half are dedicated to low-level music sub-components such as onset, chord and drum detections, fundamental frequency and tempo estimations, and beat tracking. These are important because they can be used to extract higher level musical facets such as melody, harmonic progression and so on, which are needed for useful MIR systems. For example, the objective of onset techniques is to locate musically significant events in the audio. Once these are resolved satisfactorily, it can help the performances of other tasks like drum detection, audio segmentation, and tempo estimation and many others. Only four are Information Retrieval tasks in the strict sense (e.i. a set of candidates are returned for a given query): music similarity either symbolic or audio-based, and the queries by singing/humming and tapping. Six are structured within the supervised learning of train/test (or cross-validation) paradigms well known to the machine learning community: artist or composer identification, genre, mood and tag classification. These tasks are closely related to the concept of music similarity, a topic of paramount importance in MIR. Typically, music collections are organized according to a set of subcategories as genre, artist, mood, and so forth. These categories reflect, to certain extent, music that is considered similar, particularly when music is grouped by genre [Casey et al., 2008a, Fu et al., 2011, Gjerdingen and Perrott, 2008]. Methods

that produce a similarity score between music pieces are at the core of management and recommendation systems. Nevertheless the notion of music similarity not an unproblematic one since it is a subjective and multi-faceted concept. For example, Casey [Casey et al., 2008a] breaks down audio similarity-based retrieval systems into several levels of specificity. At the high-specificity end are tasks that require to match exact audio contents, as in audio fingerprinting, a task that seeks to identify particular recordings (see for e.g. [Wang, 2003]). At the low end are tasks such as genre, mood or artist identification, that seek to find high-level music characteristics but not the same audio contents. It must be noted that high specificity tasks are usually solved using sequence of features, while tasks with low specificity ignore the signals temporal information. Similar observations can be made about low-level music sub-component tasks: the approaches used to solve them are usually quite different from the ones used for music classification. It is undeniable that many low and mid-level MIR sub-components are useful for establishing measures of similarity, but no satisfactory way of integrating all these music elements in similarity-based retrieval system has yet been found. The divide between methods that are designed for low and mid-level tasks and high-level ones is also referred to as the *semantic gap* [Celma et al., 2006b] and it identifies a conceptual and methodological problem well known in the MIR community, that inherently limits the performance of algorithms tailored for high-level musical concept extraction. This limit in performance is also known as the *glass ceiling* [Aucouturier and Pachet, 2004], and finding the reasons for this limitation and, more importantly, how to overcome it has been an active subject of research (for an overview see [Aucouturier, 2006, Casey et al., 2008b, Celma and Serra, 2008, Gouyon et al., 2008, Pampalk, 2006b, Seyerlehner, 2010, Widmer et al., 2008] and references within), including part of the work presented in this thesis [Marques et al., 2010, 2011a,b].

### 2.1.2 Context-Based versus Content-Based Methods

From a simplistic perspective, music similarity can be regarded as calculating a suitable distance measurement between a query song and a set of potential candidates. This distance can be obtained using information extracted from the context, content, or both - contextual methods, typically based on user scores and social tags, are the most common approach [Bogdanov et al., 2011, Casey et al., 2008a]. Contextual information can be rich and expressive, so in many scenarios this source of information is sufficient. Nevertheless, context-based systems rely on annotations introduced by expert groups or by users and this information may not be available and may contain erroneous data. A particular case where the absence of contextual data is inevitable is when new music material is released. This is commonly referred to as the *cold start problem*. Another limitation of context-based systems is the *long-tail problem* [Anderson, 2006]. This problem arises for unknown artists or songs, where barely any information

can be found. Only music pieces with sufficient information are recommended and the long-tail material is overlooked[3]. For a review of context-based methods, we refer to [Knees and Schedl, 2013, Schedl and Knees, 2009]. Alternatively, musical information extracted directly from the audio content can help overcome the problems of context-based recommendation systems, since they can process any music piece, regardless of their popularity or maturity. Content-based music similarity methods are the focus of this thesis, but they too have serious limitations. Cultural and contextual aspects cannot be captured looking at the audio alone. Also it is hard to determine what content-based systems are really doing. Are they indeed extracting relevant musical information from the audio, or are they doing something else? Another difficult question is how to evaluate the performance of this systems, which has been addressed in [Craft et al., 2007, McKay and Fujinaga, 2006, Sturm, 2012a, Wiggins, 2009]. In summary, content-based methods have not yet reached the performance of its context-based counterparts, mainly due to the difficulty of characterizing and extracting the intricacies and dimensions of musical properties (and perceived stimuli) solely based on the audio signal. Nevertheless, the potential of content-based systems is undeniable, and they can become powerful tools for complementing context-based recommendation systems, for aiding human annotators, and for many other MIR tasks .

## 2.2 Music Content Descriptors

### 2.2.1 Music Terms and Dimensions

Most of the techniques in MIR are based on a number of music concepts and their effectiveness depends on how well these concepts are modeled. In this Section we give a short introduction of some of these basic concepts. Before discussing the terms and facets used in musical information retrieval, it is necessary to address the perceptual attributes of sound. Sound can be characterized with three subjective attributes [Moore, 1995, Rossing, 1990]:

**Pitch** The pitch is the human attribute that allows sounds to be ordered on a frequency scale, ranging from low or deep to high or acute sounds. The pitch is closely related to the *fundamental frequency* (or $F0$), which is its physical counterpart. The fundamental frequency is defined for periodic or quasi periodic signals and is the inverse of one period. Pitch is particularly relevant in the context of musical sounds, since most instruments (with the exception of percussive ones) produce periodic vibrations that result in sounds that are a combination of different frequencies, multiples of $F0$.

[3] This effect is also known as the *popularity bias*

**Loudness** Loudness is how humans perceive the amplitude level of sound, and thus the energy present in the signal. In the context of music processing, it is common to express the level of sound in decibels, which results from applying a logarithmic scale to the mean squared power of the signal, mainly to deal with the wide variety of dynamic ranges involved. Loudness is sometimes referred as intensity.

**Timbre** Timbre is the perceptual quality that allows to discriminate between different sounds with the same pitch and loudness. For example, the same note played with the same intensity by two different instruments (*e.g.* a flute and a cello) are easily distinguishable by humans.

The connection of these three attributes to its physical counterpart is a non-trivial one, and constitutes some of the fundamental aspects of psycho-acoustics[4]. Nevertheless, the first two, pitch and loudness, can be reasonably approximated by the fundamental frequency and the signal's energy. On the other hand, timbre does not have a simple physical counterpart, nor can it be easily encoded into a single scalar value. Timbre is multi-faceted characteristic, related to the perception many aspect of music, such as instrumentation, playing techniques, acoustic surroundings, recording and broadcast environments, etc. It is therefore not surprising that the definition of timbre is a negative one: what is not pitch nor loudness. It depends mainly on the spectral energy distribution of sound and its time evolution.

Apart from the three sound attributes just described, there are other concepts and terms that are commonly used to describe the characteristics and peculiarities of music language. Music can be defined as the relationships between individual sound events, silences, and larger entities composed of these. In addition, different sounds can be played simultaneously, and this relationship is also a relevant perceptual factor. This is a broad perspective of looking at music that characterizes it along two main dimensions, an horizontal and a vertical one, the first containing the temporal information, and the other the co-occurrence of sounds in a particular time instance. The words "horizontal" and "vertical" are due to the representation used in Western music scores, where the time is associated with the horizontal axes, and where simultaneous tones are vertically aligned in the score. The terminology commonly used to describe music is also influenced by a Western-based perspective; for instance, the concept of *tonality* or *harmony* are examples of that fact. Next we give a short lists of music terms and dimensions that are useful for the characterization of musical works. The purpose is to introduce some of the nomenclature used in the MIR community. We should add that our music analysis is heavily biased towards a signal processing perspective, and in our approach to the problem of audio-based music similarity, we considered music signals mainly as "just signals"

[4]Psycho-acoustics is the science that studies the perception of sound, or in other words, the relations between acoustic stimuli and the resulting (subjective) sensations in humans.

(although we built upon the knowledge acquired on other research areas such as automatic speech recognition). Therefore, the following definitions may not correspond and may not take into account some relevant aspects considered from the point of view of a musicologist (for a thorough analysis see for example [Orio, 2006, Randel, 1986]).

- **Rhythm** in music is the placement of sounds in time. This is one of the primal elements in music. Rhythm can exist without *melody* (e.g. taiko drums performances), but melody cannot exist without rhythm. Furthermore, in music that has both melody and harmony, the rhythmic structure cannot be dissociated from them [Britannica, 2014]. Rhythm is often identified with some of its constituents such as *metre* (or meter) and *tempo*. Metre is the pattern comprised of basic temporal units, called the *beats*, grouped into regular measures (or bars), and tempo is the pace of the fundamental beat, that is the speed which a musical work is played (measured in beats per minute).

- **Tonality** is the principle of organizing a music composition around a central note (called the tonic). Tonality is closely related to the concept of *key*, and sometimes these two terms are used interchangeably. The key is the name of the chord that plays a central role in a musical work.

- **Melody** is a succession of music tones implying a rhythmically ordered arrangement from pitch to pitch. Melodies often consist of one or more musical phrases or motifs, and are usually repeated throughout a composition.

- **Harmony**, in broad terms, refers to the sound of two or more notes or chords heard simultaneously. In Western music, harmony also refers to systems of rules that allows or forbids relationships between chords. The concept of harmony is based on certain tone relationships that are accepted almost reflexively by the human auditory system. As opposed to the melody which is considered an horizontal aspect, harmony is a vertical aspect of music.

- **Orchestration** is the art arranging and combining instruments and their capabilities of producing different timbres (or colors) in a musical work. Orchestration also refers to process of adapting an existing music piece to another medium, in particular adapting it to be played by an orchestra.

- **Structure** pertains to how musical work is divided and the temporal arrangement and repetitions of the different sections in it. In particular, in Western popular and rock music, the songs typically consists of distinguishable sections such as *intro*, *verse*, *bridge*, *chorus* and *outro*. A particular structure may consists of one or more repetitions of a verse and chorus.

### 2.2.2 Feature Hierarchy

In order to implement content-based music classification and similarity metrics, it is necessary to extract from the audio signal a set of suitable features. In a very broad sense, a feature is a compact description of a particular information present in the music signal. In content-based MIR, a feature is any piece of information related to the music that is representative and meaningful, and that can be extracted or predicted from the raw audio. What is "representative" or "meaningful" in music piece is a subjective concept and depends on the interpretation that the same piece has for different users and/or in different situations. Therefore, it is important to tackle the question of which features mean what to which users. One commonly adopted route, is to differentiate broad feature categories into three hierarchically structured description layers, each layer with different levels of abstraction [Celma and Serra, 2008, Gouyon et al., 2008]: low-level, mid-level, and high-level descriptors (or features).

Low-level descriptors are those who can be directly calculated from the audio signal, or derived after some signal transformation such as the Fourier or Wavelet transforms. This class of features are also denominated *signal-centered* descriptors, and the majority are related to the spectral content of the signal. A more detailed review of some of these features, mainly the ones used in this thesis (and also commonly used in many other works) will be given in the next section. Mid-level features are related to a higher plane of music characterization, and include musical dimensions such as tonality, instrumentation, rhythm, melody, to name just a few. Generally these features require a inference or generalization step from the low-level descriptors, for example, through statistical signal processing or machine learning techniques. This features are also referred as *object-centered* descriptors. Finally, high-level descriptors pertain to semantic aspects of music such as mood, genre, or similarity, that are intrinsically related to the "interpretation" or "understanding" of music. These also require an induction or generalization step, but unlike mid-level features where the induction operation is data-centered, the high-level inference is user-centered. In other words, to infer semantic aspects such as genre or mood, one needs to analyze the relations between low and mid-level features and user generated labels of songs in terms of those semantic characteristics. For this reason, high-level features are also referred as *user-centered* descriptors.

Low-level features are the starting point for large majority of the methods presented in literature, they are easy to extract and have shown good performances on wide range of classification tasks. A variety of studies, more or less systematic have been made on audio features. For instance in [Herre et al., 2001, Herrera et al., 2002], and also under the CUIDADO project [Peeters and Rodet, 2004, Vinet et al., 2002], some frame-based timbral descriptors were proposed, which are now common choices of many authors, including ours. Tonal related features were analyzed in [Gómez and Herrera, 2006], and in [Ellis, 2007] tonal and timbral

characteristics are used jointly, while in [Gouyon, 2005a] the focus was on rhythmic features. These and many other works (see for example [Jensen, 2009, Klapuri and Davy, 2006] and references within) were dedicated to more or less fine tune audio characteristics and/or enhancing signal aspects connected to our own musical perception. The broad categories which low-level features are typically associated with, are related to three musical dimensions: timbral, tonal, and rhythmic. In our work, the feature we used are predominantly timbral related features, although we also tested some tonal-related ones, namely the Pitch Class Profile coefficients. These are all calculated on a frame by frame basis and are directly related to the spectrogram of the signal (with the exception of the zero crossing rate). Next we give a brief overview of the signal processing concepts necessary to calculate these features, before presenting the details on how to compute the features we used in our work.

### 2.2.3 Low-Level Descriptors

A wide range of low-level features can be computed from audio signals, most of which have its origins in the well-established research fields of signal processing and automatic speech recognition. In order to extract a set of features from a music signal, the audio has to be converted to a digital format. From a practical point of view, the vast majority (if not all) of the signals analyzed are already in a digital format, and due to storage limitations, a significant percentage of the digital signals are encoded through some lossy compression scheme, such as MP3[5]. In this situations, the signals are often decoded into a more manageable format: the overwhelming choice being mono raw audio in 16-bits Pulse Code Modulation (PCM). Also, a re-sampling step is commonly performed to ensure that all the signals have the same sampling rate. The sampling rate can vary from 5 to 44.1 kHz, but often a low sampling frequency (typically from 5 to 22 kHz) is chosen to reduce the amount of audio data. A comprehensive analysis of signal representation and modeling techniques is beyond the scope of this thesis, (for that, the reader can consult [Oppenheim et al., 1997, Papoulis, 1977, Pereira, 2009, Rabiner and Schafer, 2010]), and from this point on, we will assume that music signals are already in PCM mono format. Next, we give a non-exhaustive review of common features used in audio music signal description, with special focus on the ones used in this thesis.

[5]MPEG-1 or MPEG-2 audio layer III. MPEG, the Moving Pictures Expert Group, sets standards for audio and video compression and transmission. MP3 is an audio-specific format that uses lossy compression and was designed to greatly reduce the amount of data required to represent an audio signal.

### Frequency Representations

Spectral analysis is the process of characterizing time-domain signals in terms of individual frequency components. Any real, time-dependent signal can be converted into a spectral domain representation, and back to its original form without any loss of information, via Fourier transforms[6]. The spectrum is the variations of the signal's amplitude and phase versus frequency, which in many cases, is a simpler and more intuitive description than time-domain counterpart, amplitude versus time. Spectral representations are particularly useful for audio and music signals because they highlight distinct characteristics of sound, such as harmonics, notes, or instruments timbral signatures. It is therefore unsurprising that the vast majority of low-level music features are derived from frequency-domain representations. The starting point is to calculate the spectrum of the audio piece being analyzed. This is easily done using Fast Fourier Transform (FFT) algorithms, which are efficient implementations of the Discrete Fourier Transform (DFT). The DFT is a specific transformation used in Fourier analysis, and it projects a discrete-time signal into a discrete-frequency domain representation (and back via the inverse DFT - iDFT). The DFT is ideal for processing digital information because of its fast implementations and because in both domains have discrete representations. From a mathematical point of view, this transformation is only exact for periodic discrete-time signals, but it also an important analysis tool for finite-length discrete-time signals which have a continuous spectral representation. In this case, using the DFT can be interpreted as sampling the continuous spectrum at regularly spaced frequency intervals, which can be made arbitrarily small by padding the finite sequence with zeros. Formally the DFT of a discrete-time, periodic signal $x[n]$ with a period of $N$ samples long is:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j\pi nk/N} \qquad n, k = 0, \ldots, N-1 \tag{2.1}$$

where $j = \sqrt{-1}$ and $X[k]$ is composed of $N$ complex value coefficients, which represent the magnitude and phase in each of the linearly-spaced $k$ values (frequency bins). $X[k]$ is usually presented separately as the magnitude spectrum, $|X[k]|$, and phase spectrum $\phi[k]$:

$$\begin{aligned} X[k] &= \mathrm{Re}\left(X[k]\right) + j\mathrm{Im}\left(X[k]\right) \\ &= |X[k]|\, e^{\,j\,\phi[k]} \\ \text{with:} \quad |X[k]| &= \sqrt{\mathrm{Re}\left(X[k]\right)^2 + \mathrm{Im}\left(X[k]\right)^2} \\ \phi[k] &= \arctan \frac{\mathrm{Re}\left(X[k]\right)}{\mathrm{Im}\left(X[k]\right)} \end{aligned} \tag{2.2}$$

[6] Strictly speaking, the Fourier transform is applicable to integrable functions, $f(x)$, that satisfy $\int_{-\infty}^{+\infty} |f(x)| dx < \infty$, and are Lebesgue measurable on the real line.

In practical terms, the magnitude spectrum carries most of the relevant information, and unless one is interested in finding specific time-occurrences such as in onset detection tasks, the phase is typically ignored. Furthermore, for the case of real-value signals such as music piece, the magnitude is an even, periodic function and half of the frequency coefficients can be discarded.

### Time-Frequency Representations

The DFT is an essential tool to convert a time signal into a frequency domain representation, but looking at a signal only in terms of frequency is not sufficient. This is because music signals are not stationary: they are comprised of a combination sound events localized in time, such as notes or arrangements, which also have a specific frequency signatures. The DFT gives us the spectral content of the entire signal, and the frequency characteristics of individual portions of the signal and the evolution between different sound events are lost. The solution is to cut the audio into a series of short duration segments, or frames, and look at the progression of the frequency spectrum of the individual segments. This is called the Short-Time Fourier Transform (STFT), and its magnitude squared yields the spectrogram: the starting point for most time-frequency representations. Formally, the STFT of any given frame of a signal $x[n]$, can be expressed in terms of a product of that signal with a window function which selects the samples in that frame:

$$X[k,m] = \sum_{n} x[n]w[n-m]e^{-j\pi nk/N} \qquad n,k=0,\ldots,N-1 \tag{2.3}$$

where $m$ is the time index of the frame and $w[n]$ is the window function of length $N$. This equation gives the discrete frequency representation of a (small) segment of the signal $x[n]$. To obtain the spectrogram, this calculation is repeated until the whole signal is covered. The window function has a direct impact on the STFT, and there are limitations and tradeoffs in resolution that are inherent to the choice of the shape and the size $w[n]$. In equation 2.3, the signal $x[n]$ is multiplied by the window function $w[n]$, which in the frequency domain, is equivalent to the convolution of the individual Fourier spectra of the signal, $X[k]$, and the window, $W[k]$. The convolution of the two spectra can create frequency components in $X[k,m]$ that were not present in the original signal. This effect is known as spectral leakage which is a smearing of the frequency components. There many window functions with different shapes (rectangular, Hanning, Hamming, Kaiser, Gaussian, Blackman-Harris, etc) that try to mitigate the spectral leakage, by redistributing its effects to frequency areas that do the least harm. The choice of the window function depends on the application, and in MIR the most common choice is the Hanning window. Throughout this thesis we also use the Hanning window as the standard window function:

$$w[n] = \frac{1}{2}\left(1 - \cos\left(\frac{2\pi(n-1)}{N}\right)\right) \tag{2.4}$$

Another factor that has a direct impact on the STFT is the length of the window function. A wider window gives a better frequency resolution, but a poorer time resolution, and a narrower window has the opposite effect. This time-frequency trade-off is depicted in Figure 2.1, where several spectrograms of two audio signals are calculated with different time windows. The spectrograms corresponding to the larger windows have sharper horizontal lines, but a poorer time resolution which is noticeable in the blurred note onsets[7]. Once again, this is due to the non-stationary nature of music signals. The resolution of the spectrogram is also affected by the hop size: the number of samples skipped between two consecutive frames. The smaller the hop size, the smoother the spectrogram resolution, but usually a hop size of half or a whole frame is chosen to save processing and memory resources.

The spectrogram models the frequency distribution of a signal as a function of time. This representation is particularly useful for characterizing music signals since it brings fourth localized sound events, such as notes, repetitions, etc, that otherwise would be unavailable in either time or frequency domain representations alone. Nevertheless, the frequency resolution of the spectrogram defined by the STFT is a linear one, and does not correspond to the way humans perceive sound and music, nor does it reflect the semitone scale used in Western musical notation. Many studies in psycho-acoustics have shown that the human auditory system has a logarithmic frequency resolution. To account for this fact, and based on experiments on the human hearing, many psycho-acoustic scales have been proposed, such as Mel [Stevens et al., 1937], Bark [Zwicker, 1961, Zwicker and Terhardt, 1980], Equal Relative Bandwidth [Moore, 1995], and others, the Mel scale being the overwhelming choice for music similarity and classification methods. The process of mapping the linear frequency spectrum obtained via STFT into one of the psycho-acoustic scales is closely related to the topic of auditory filter banks, which are non-uniform bandpass filters designed to imitate the human auditory system. A new frequency representation can be obtained by calculating the spectral energy in each filter band. This way a closer correspondence to human auditory system is obtained, and the number of spectral coefficients per frame is reduced to the number of bands. Another related frequency scale commonly used in MIR scenarios is the cent scale. This scale also has a logarithmic frequency resolution, but is inspired from a musical perspective rather than psycho-acoustic one.

Next, we give a brief description of the features used in this work, which are also commonly used in classification and similarity estimation tasks. We first present a set of descriptors derived from the Mel scale: the Mel-Frequency Cepstral Coefficients (MFCCs). We then analyze other descriptors that complement the information of the MFCCs, most of which are based on the spectral magnitude of the audio signal, and are calculated on a frame by frame basis.

[7]This relationship is similar to the Heisenberg uncertainty principle in quantum physics which states that it is impossible to measure both the exact position and the exact momentum of a particle at the same time. The more precisely one of the quantities is measured, the less precisely the other is known.

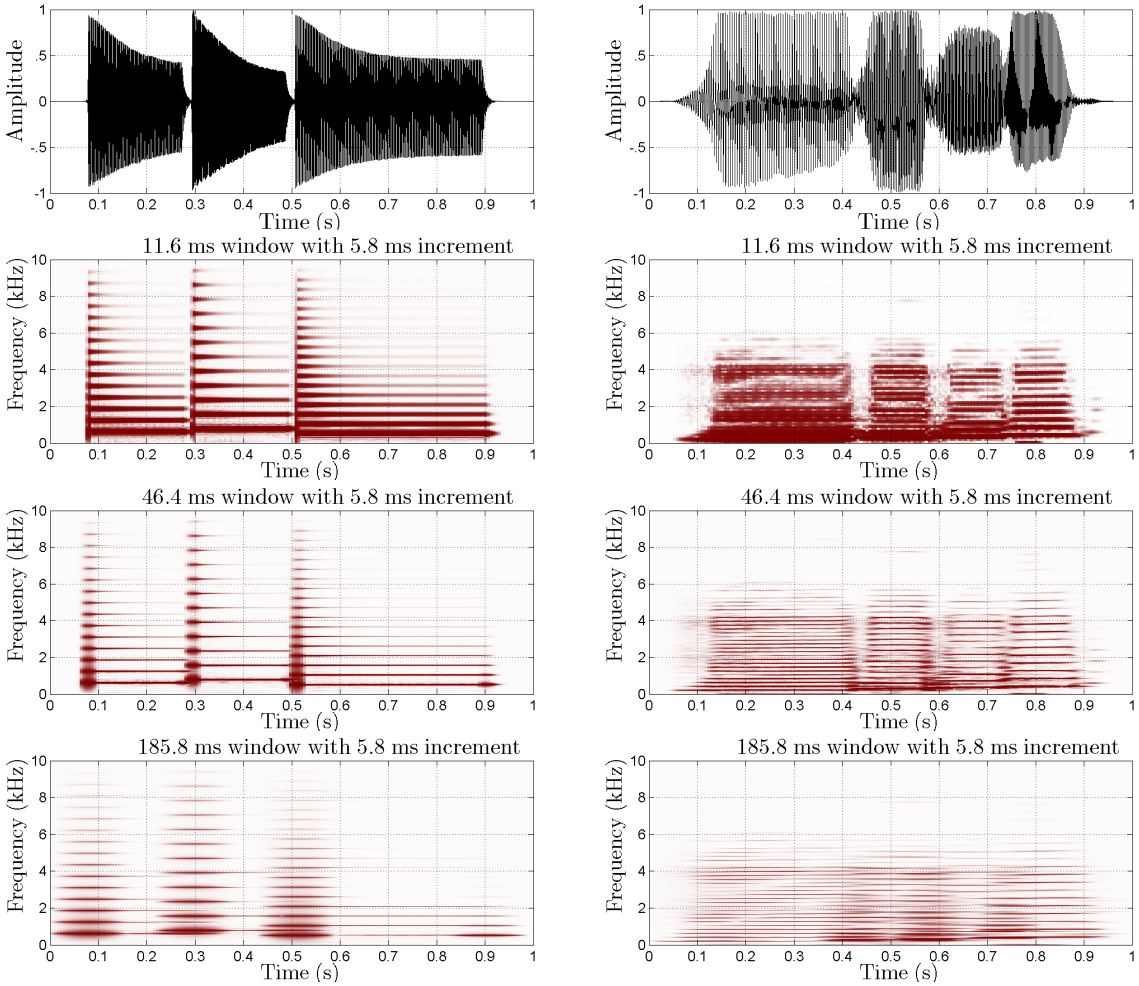



Figure 2.1: Two audio signals (first row) and corresponding spectrograms (bottom rows). The left column consists of a three tone synthesized signal, while the right one is a four note saxophone signal. The spectrograms were obtained using three different frame sizes, with a fixed increment of $\approx$ 6ms. The frame sizes of the last two rows are 4 and 16 times larger then the second one. As we move down, the frequency resolution increases (horizontal lines become sharper), at the expense of the time resolution. This is particularly noticeable in the last row, where note onsets are severely blurred.

### Mel Frequency Cepstral Coefficients

Mel Frequency Cepstral Coefficients are among the most common features used to characterize audio signals, in particular speech and music signals (see e.g [Rabiner and Juang, 1993]). The MFCCs are calculated on a frame by frame basis and are closely related to the spectral envelope of the audio frames. The spectral envelope is the shape of the power spectrum, and is generally assumed as an relevant characteristic, since sounds with similar spectral envelopes are usually perceived as similar. The MFCCs are obtained by filtering the power spectrum of the frames by a series of band-pass filters, which simulate the auditory response of the human hearing. The filters are positioned at regular intervals on the Mel-frequency scale, which corresponds to logarithmic-spaced intervals in linear frequency resolution. The mapping between linear frequencies in Hertz, and Mel-frequencies is given by:

$$f_{\text{Mel}}{=}2595 \log_{10}\left(\frac{f_{\text{Hz}}}{700}+1\right) \tag{2.5}$$

The power in each frequency band (filter) is:

$$X'[m] = 20 \log_{10}\left|\sum_k X[k]H[k,m]\right|, \quad m = 1,\ldots,M \tag{2.6}$$

where $M$ is the total number of bands, and $H[k,m]$ is the $m^{\text{th}}$ filter and $k$ is the frequency bin index. Typically triangular filters are used but other shapes can also be employed (e.g. rectangular, Hanning, Gaussian). Finally the logarithmic Mel-spectrum is decorrelated via Discrete Cosine Transform (DCT):

$$\phi[l] = \sum_{m=1}^{M} X'[m] \cos\left(\frac{\pi}{M}\left(m-\frac{1}{2}\right)\right) \tag{2.7}$$

for $l = 1,\ldots,L$, where $L \leq M$ is the the number of coefficients after the DCT. The resulting DCT coefficient, $\phi[l]$, are the Mel Cepstral Coefficients. MFCCs are associated with the concept of timbre, and are the features of choice in MIR audio-based similarity systems. Nevertheless, it is not uncommon to also estimate the differences MFCCs obtain in one or two consecutive frames: $\Delta$-MFCCs and the $\Delta^2$-MFCCs, or use other MFCC-related descriptor such as the Mel-Frequency Spectral Irregularities [West and Lamere, 2007], or the Spectral Contrast Features [Jiang et al., 2002], although some authors argue that these are not as robust as the MFCCs [Sigurdsson et al., 2006, West, 2008]. The MFCCs were the main type of features we used in our experiments; we tested the inclusion of different number of coefficients but from our experience, using twelve coefficients was proved sufficient for our purposes, and using more coefficients did not improve the performance of the methods we tested, nor did the inclusion of $\Delta$-MFCCs or $\Delta^2$-MFCCs (see results in Appendix B).

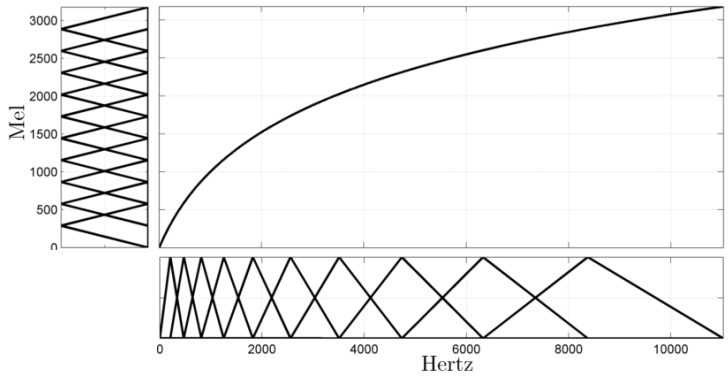



Figure 2.2: Schematic plot of non-uniform spaced triangular filters in the original power spectrum ($x$-axis) into Mel frequency bands. Using a non-uniform filter bank is equivalent to using a uniformly spaced filter bank in the Mel domain.

### Spectral Features

In addition to the MFCCs, it is common to include other descriptors extracted from the raw audio. There is a large number of features (in the hundreds) that can be chosen for this purpose. These are usually grouped according to some rough categorization, such as energy, temporal, spectral, harmonic, as in [Klapuri and Davy, 2006], but other feature taxonomies are also common (see for e.g. [Herrera et al., 2003, Manjunath et al., 2002, Peeters and Rodet, 2004]). It is beyond the scope of this thesis to give a overview of all the possible features that can be used in audio-based MIR classification scenarios. Next, we briefly describe just the ones we used in our research. These are the spectral centroid, the spectral flux, the spectral roll-off, and the zero crossing rate. These four features, in conjunction with thirteen Mel cepstral coefficients (including the zero order one) were, so to speak, the "elected recipe mixture" of features in three of our publications [Marques et al., 2010, 2011a,b], and are commonly used by many researchers [Aucouturier and Pachet, 2004, Ness et al., 2009, Pampalk, 2006b, Pohle et al., 2005, Seyerlehner, 2010, Tzanetakis and Cook, 2002]. This set of features, which we will refer from now on as the M17feat set[8], is based on spectral characteristics of the audio. In order to describe the features, we first introduce the energy-normalized magnitude spectrum:

$$\tilde{X}[k] = \frac{|X[k]|}{\sum_{j \in \mathcal{K}} |X[j]|} \tag{2.8}$$

where $X[k]$ denotes the discrete Fourier spectrum (see Equation 2.2), $k$ the frequency index, and $\mathcal{K} = \{0, \ldots, \lceil \frac{N}{2} \rceil\}$ the set of the non-negative frequency indexes (for a audio frame of $N$ samples long).

[8] We will make use of the M17feat set in several examples and experiments described further on in this thesis

- **Spectral centroid (SC)** is the center of mass of the (normalized) spectrum. This feature is closely related with the "brightness" of sound timbres, which is derived from an analogy with visual brightness. Bright sounds are associated with high frequency contents, and one way to measure the brightness is through the spectral centroid, which is given by the weighted mean of the frequency bins with their magnitudes as the weights:

$$\mathrm{SC} = \sum_{k \in \mathcal{K}} k \tilde{X}[k] \tag{2.9}$$

- **Spectral flux (SF)** is a measure of how quickly the power spectrum is changing. This can be defined as the Euclidean distance between the spectral magnitudes of consecutive frames:

$$\mathrm{SF}_i = \left( \sum_{k \in \mathcal{K}} \left( \tilde{X}_i[k] - \tilde{X}_{i-1}[k] \right)^2 \right)^{\frac{1}{2}} \tag{2.10}$$

  where $\tilde{X}_i[k]$ and $\tilde{X}_{i-1}[k]$ are the normalized magnitude spectra of the current and the previous frames, respectively.

- **Spectral roll-off (SR)** frequency is the frequency index $R$, below which a certain fraction $\lambda$ of the spectral energy resides:

$$\mathrm{SR} \quad \text{so that} \quad \sum_{k=0}^{\mathrm{SR}} |X[k]|^2 = \lambda \sum_{k \in \mathcal{K}} |X[k]|^2 \tag{2.11}$$

  where in our experiments, we set $\lambda = 0.85$ (another commonly chosen value is $\lambda = 0.95$).

- **Zero crossing rate (ZCR)** is the number of times a time-domain signal changes its sign. This feature is calculated in the time domain, and although it is not spectral feature per se, it is closely related to high-frequency contents of the signal. Pitched instruments (which result in quasi-periodic waveforms) will tend to have low values for the ZCR, while white noise or percussion instruments typically have high values. For a given frame of $N$ samples long of a temporal signal $x[n]$, the ZCR in that frame is computed as

$$\begin{aligned} \mathrm{ZCR} &= \frac{1}{2N} \sum_{n=1}^{N-1} \left| \mathrm{sign}\left(x[n]\right) - \mathrm{sign}\left(x[n-1]\right) \right| \\ \text{where} \quad \mathrm{sign}(x) &= \begin{cases} +1 \text{ if } x \geq 0 \\ -1 \text{ if } x < 0 \end{cases} \end{aligned} \tag{2.12}$$

### Pitch Class Profile Coefficients

The Pitch Class Profiles (PCPs) are features derived from a logarithmic frequency resolution inspired in Western-based representation of music: the equal tempered tonal scale. In this scale,

each octave is divided into twelve semitones (half steps) established by the tuning systems used on a piano, each semitone comprised of 100 cents. An octave is the simplest interval in music, where two notes one octave apart are essentially perceived as the same note; in terms of the fundamental frequency (pitch), one of the notes has half or double the frequency of the other. The relationship between the frequencies in Hertz and in cents are obtained via (using the musical note A4 as reference):

$$\begin{aligned} f_{\text{Cent}} &= 1200 \log_2 \left( f_{\text{Hz}} / \left( 440 \times 2^{-\frac{57}{12}} \right) \right) \\ f_{\text{Hz}} &= \left( 440 \times 22^{-\frac{57}{12}} \right) \times 2^{\frac{f_{\text{Cent}}}{1200}} \end{aligned} \tag{2.13}$$

Pitch Class Profiles represent the intensity of each of the twelve semitones. These are sometimes referred as chroma features, and show a high degree of robustness in timbre variations and are closely related to the harmonic and/or tonal contents of music. They have been used in a variety of MIR areas, such as chord recognition [Bello and Pickens, 2005, Fusjishima, 1999, Mauch and Dixon, 2010, Yoshioka et al., 2004], key detectors [Pauws, 2004], synchronization and alignment [Hu et al., 2003, Joder et al., 2010], cover song identification [Ellis and Poliner, 2007, Gómez and Herrera, 2006, Serrà et al., 2008], music structure analysis [Goto, 2003, Paulus et al., 2010], just to name a few. There are several ways of calculating the PCPs, either by combining spectral representations with binning strategies or using multi-rate filter banks (see for e.g., [Cabral et al., 2005, Gómez, 2006, Lerch, 2012, Müller and Ewert, 2011, Stein et al., 2009]). We used a similar strategy as in [Goto, 2003], and we computed the PCPs using a frequency range of six octaves, from the third to the eight (from 130 Hz to 8.7 kHz). We first converted the Hertz frequency to cents, and summed up the power spectrum in each 100-cent bins (in each semitone), resulting in $6 \times 12 = 72$ power measures[9]. These were then folded into 12 PCP coefficients (i.e. summing the power spectra in each of the semitone bins independently of the octave they were in). The experiments we conducted with the PCPs pertain to genre classification problems, but unlike the results reported in [Ellis, 2007], thet did not contribute to an increase in performance of our classification systems. We believe that there are two main reasons for this negative results. The first one is due to folding process of the signal spectrum used to obtain the PCPs. This folding, which at the same time confers the robustness of these features to timbre variations, also destroys the "color" of the sound - a property that has been empirically proven quite effective for genre classifications, by the many works found in literature using only timbre related descriptors. The second reason has to do with the use of the magnitude of the spectrum in the computation of the PCPs (instead of the log-magnitude).

[9] In our experiments, most of the audio signals were sampled at a frequency rate of 22050 Hz, and the frame size was set to 2048 samples ($\approx 93$ ms). This results in a frequency resolution of approximately 11 Hz per frequency bin in the STFT representation. In order to compute the chroma related descriptors, the frequency resolution was artificially increase via zero-padding. The frame size was increased to $2^{16}$ which results in a frequency resolution of $\approx 1.3$ Hz per frequency bin.

The amplitude of the Fourier coefficients can show large variations in comparison to its logarithm counterpart, which can be detrimental for classification purposes. Tests we conducted with the 72 coefficients obtained from the cent spectrum on a genre classification task show that when log-amplitude is not used, the performances drop, on average, ten percentage points (see Table B.1). Additionally, when the log-magnitude is used, the performances are approximately the same to the ones obtained with the traditional MFCCs, and similar results are obtained when projecting the 72 chroma-related coefficients into a 12-dimensional space (either via principal component analysis or discrete cosine transforms). Even though there is usually a clear distinction between the properties of timbral and chroma related features, since they are both obtained from the Fourier spectrum, and as long as the feature extraction process is not too destructive, fairly similar performances can be obtain with a wide range of spectral descriptors (see results described in the Appendix B).

### 2.2.4 On Low-Level Spectral-Based Descriptors

During our work, different types of feature arrangements were tested, in part to determine if particular sets, descriptor classes, or pre-processing schemes were more beneficial than others, in part to start from the same feature set used by other authors, to simplify comparisons between theirs and our methods. Many of the experiments on this subject were made in the initial stages of our research, where we measured the impact of different feature combinations on genre classifications methods. These focused mainly on MFCCs and chroma descriptors, and on various parametrization issues, such as the number of coefficients, frame sizes, etc. In the Appendix B.1 we report on several tests we conducted on different feature combinations, and on particular issues such as hop and frame size, and other signal processing related matters. Our observations only confirm that the MFCCs (between 10-20) are effective enough for audio similarity tasks, although complementing them with other descriptors (like the in the M17feat set) contributes to a slight performance increase. Our experience on spectral-based features, timbre and chroma related, indicate that fairly similar performances can be obtained with different spectral descriptors. Next, through a simple example, we show in loose terms, how different sets of spectral descriptors populate the feature space. The intent is to give a rough idea of the strong correlation present between different spectral feature representations, and intuitively explain the reason for obtaining comparable performances with these features. We then proceed to two other feature related examples: the first one is on how the spectral features are distributed among different data sets, and the second on how class-dependencies affect the feature distributions. Here we want to show a general perspective on how features from distinct data sets and features from different genres populate the feature space; this will set the stage for other examples, namely the one in Figure 3.5 of the next chapter, and the discussion on what genre-related information

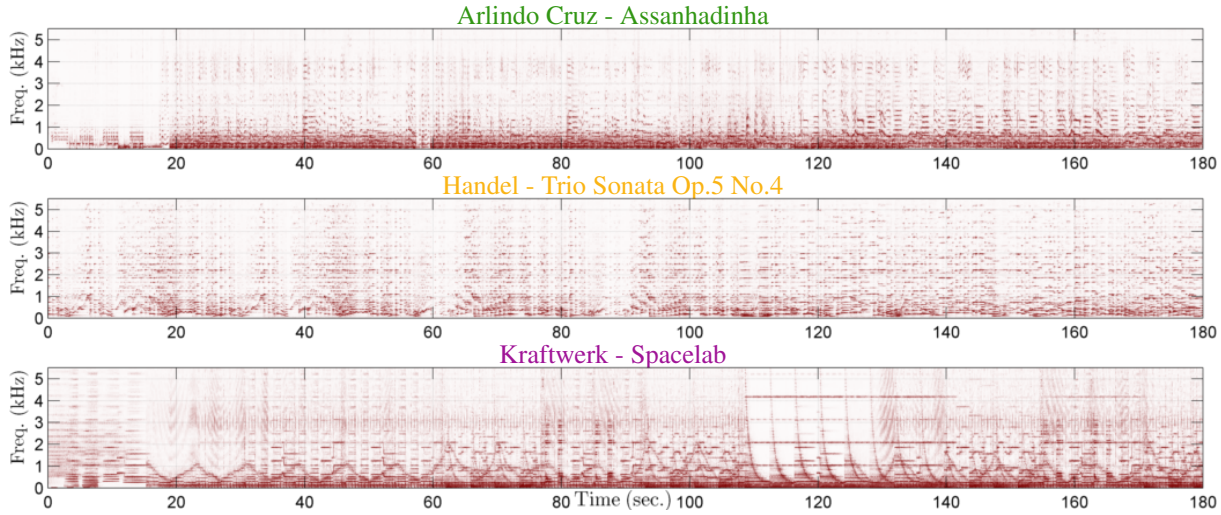



Figure 2.3: ,
high values corresponding do dark red colors.]Magnitude spectrograms of 3 minute excerpts of three songs (≈3900 frames). The magnitude values have been normalized between [0,1], high values corresponding do dark red colors. The maximum displayed frequency was set to ¼ of the sampling frequency for visualization purposes. The color on name of each song is the same as the corresponding feature points in the scatter plots of Figure 2.4.

can be inferred from low-level feature distributions, which we address in Chapter 4.

**Example: Spectral Representations of 3 Distinct Songs**

In this first example, we picked three songs with readily distinguishable musical sonorities. We want to analyze how perceptually contrasting songs occupy the feature space. The intention is to do a basic sanity check for classification methods that rely solely on the feature distributions, and show that they can deal with the simple case of data divided in three distinct "classes". The selected songs are:

- **Assanhadinha** by **Arlindo Cruz**: This song is a typical Pagode song, , which is a sub-genre of Samba, with a loud upbeat sonority, with lots of percussion instruments, people singing along, and accompanied by a banjo line (or Cavaquinho). This song belongs to the LMD data set.
- **Trio Sonatas, Op.5 No.4** by **George Fredric Handel**: This is a Baroque piece, with violins and cello, and with a fast tempo (Allegro), performed by The Brook Street Band. This song belongs to the ISMIR04 data set.
- **Spacelab** by **Kraftwerk**: This is a song from the first Kraftwerk album "The man-machine" from 1978. The sonority heavily based on synthesizers and drum machines. This is a song

from the CAL500 data set.

The data in this example consisted of 3 minutes audio excerpts from each song. The audio was mono, sampled at a rate of 22050Hz. We extracted several spectral descriptors from 93 milliseconds analysis windows of the audio, with $50\%$ overlap. The features are all computed on a frame by frame basis, and with the exception of the zero crossing rate, they are all based on the log-magnitude spectrum of the frames. The features we used are:

MELfeat: 50 descriptors corresponding to the log-square root power in each filter of the Mel filter bank (Equation 2.6[10]).

MFCCfeat: 13 MFCCs, including the $0^{\text{th}}$ order one, obtained via DCT transformation of the previous Mel descriptors.

M17feat: 17 coefficients: the zero crossing rate, spectral centroid, rolloff frequency, spectral flux, and 13 MFCCs, including MFCC0.

CENTfeat: 72 coefficients obtained by from the log-magnitude of the cent spectrum, corresponding to 12 semitone per octave, and 6 octave bands total (from 130Hz to 8.7kHz)

PCPfeat: 12 pitch class profile coefficients (or chroma)- the folded version of the previous coefficients[11].

The spectrograms of the three songs are shown in Figure 2.3, where differences in the temporal evolution of each song are promptly noticeable. To see how this spectral diversity is reflected in the features, we performed two linear, dimensional reduction transformations commonly used for data visualization: principal component analysis (PCA), and linear discriminant analysis (LDA). The first in an unsupervised method, while the second is supervised, and they are both described in Appendix A. PCA assumes that the most informative directions are the ones with the highest variance, and through an orthogonal linear transformation projects the original data into this new coordinate system. Typically the directions with the lowest variance are discarded, but the information in the data is preserved if all the principal components are kept. In LDA that is not the case. LDA is a generalization of the Fisher's discriminant analysis, where the transformation tries to maximize the distance between classes and at the same time minimize the variance within each class. This method project the original data into a $|\Gamma|-1$-dimensional subspace, where $|\Gamma|$ is the number of classes, which for this case is a 2-dimensional subspace. In Figure 2.4 are the 3 and 2 dimensional PCA and the 2-dimensional LDA projections of the spectral feature sets previously mentioned.

[10] Actually, this equation acts in the power domain, so to be exact, the coefficients we used are half this value.

[11] These coefficients are the logarithm values of the "regular" PCPs coefficients.

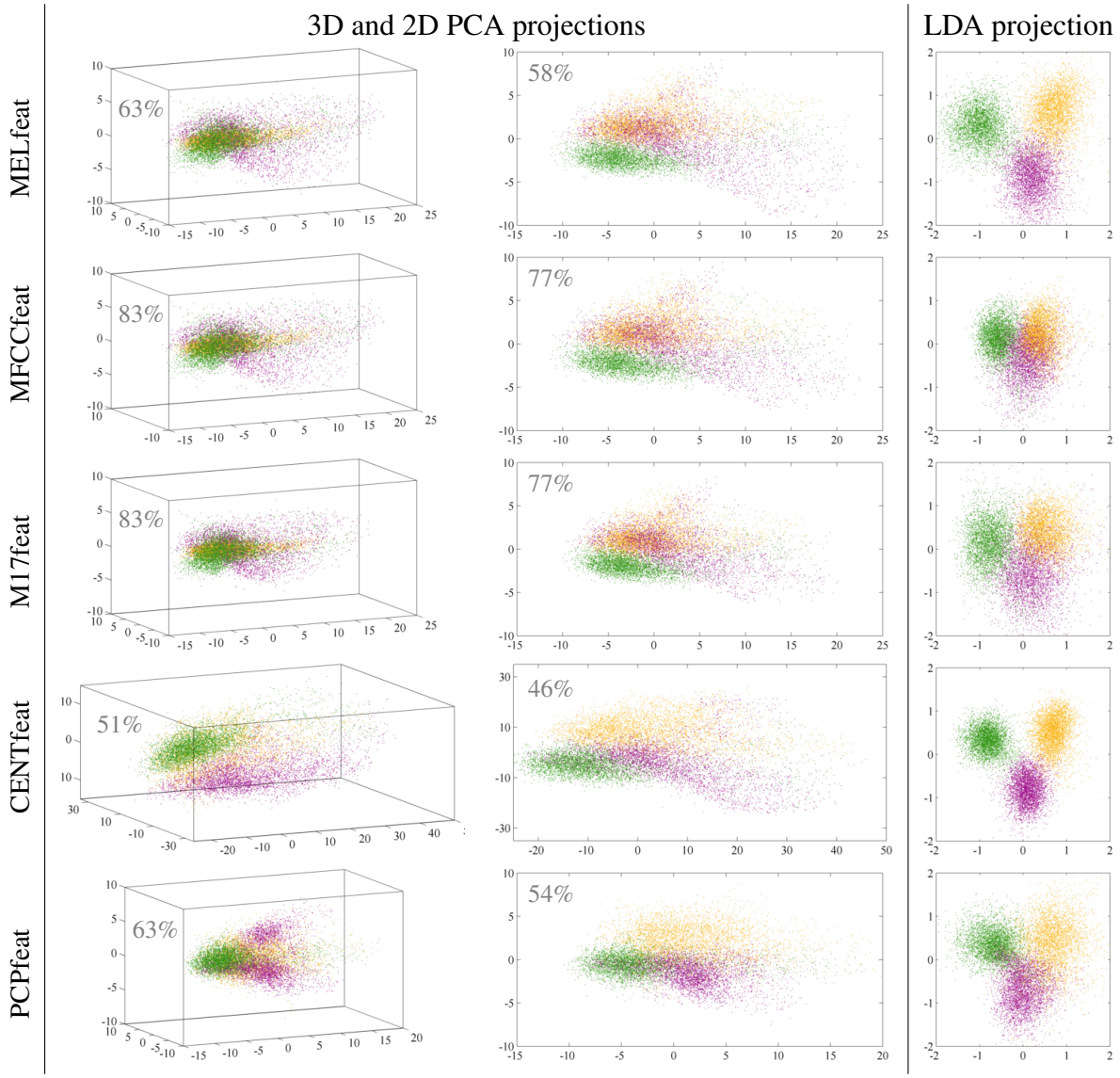



Figure 2.4: Scatter plots of different sets of spectral features obtained through PCA or LDA projections. Each color correspond to the features of one of the three songs: ■ Arlindo Cruz - Assanhadinha, ■ Handel - Trio Sonata Op.5 No.4, and ■ Kraftwerk - Spacelab. In the upper left corner of the PCA projections, are the total variance of the first three, and the first two principal components.

In PCA plots there is a strong resemblance between the projections obtained with three of the feature sets: the MELfeat, the MFCCfeat and the M17feat. This is not surprising, in particular for the first two projections. The MFFCfeat are derived from a discrete cosine transformation of the MELfeat, and since the DCT is an approximation of the PCA transformation, applying PCA to both of these features yields very similar results. The reason why the M17feat have similar distributions are also understandable since 13 of the 17 dimensions of the features in this

set are identical to the MFCCfeat. There is however, from our point of view, some differences between these timbral features and the remaining two, which are associated with chroma, the CENTfeat and the PCPfeat. This is particularly noticeable in the 3-dimensional scatter plots. The CENTfeat have the lower percentage of the total variance in the first principal components, which is expected since it is also the feature set with the highest cardinality. However, the PCPfeat, also have a low variance, equal to the MELfeat set, cardinality forty, and the same cardinality of the MFCCfeat set: twelve. This is a sign that there is a more evenly distributed dimension wise occupation of the feature space, which may also imply randomness. If that is the case, it is an explanation for the poor results obtained on genre classification with these features (performances were about 10% lower than with MFCCfeat - details in Table B.1). PCA plots show some differences compared to the remaining ones, but there are still some noticeable correlations between all the plots.

For the LDA case, all plots show a reasonably good separation between features of the three songs, with the CENTfeat and the MELfeat obtaining the best results. Note though, that the quality of the separation obtained by this two types of features is, to a large extent, due to the dimensionality of these features rather than any discriminative qualities they may possess. The CENTfeat are comprise of 72 coefficients and the MELfeat 40 coefficients; in these high dimensional spaces it is easier to obtain better separations of the three songs than it is in lower dimensions, i.e. in the 12 or 17 dimensional spaces of the remaining feature representations. Furthermore, the good separations obtained with LDA are due to oversimplified nature of the problem: we only considered 3 distinct "classes" (i.e. songs) when in a typical testing scenario, the data is comprised of several hundreds or thousand songs. Figure 2.6 shows the LDA results for the same 3 songs, but when the LDA projection was calculated based on a whole data set: the LDA transformation does not separate well the three songs in this situation.

**Example: Feature Distributions of Three Distinct Data Sets**

In this example, we analyze how the features are distributed among different data sets. Our intent is to show the features diversity present in most data sets, and the high feature superposition found between them. Figure 2.5 shows the scatter plots for 3 data sets: ISMIR04, LMD, and CAL500 (see Appendix C, for a descriptions of these data sets, and others used in this dissertation). The first two were designed for genre classification while the third was for autotagging. In terms of musical variety, the ISMIR04 data set has greatest diversity, while the LMD has the smallest, but all three data sets are quite different from each other. In the ISMIR04 set approximately half of the tracks are classical music songs, a genre which is not present in the other two sets. The LMD is uniquely composed of Latin music, and although there are ten different genres represented in this set, the majority of the songs are clearly identifiable as Latin American music

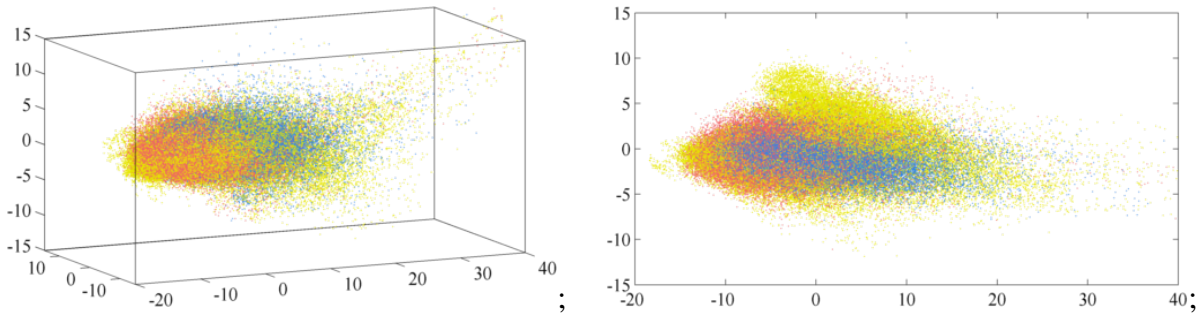



Figure 2.5: Superimposed 3D and 2D scatter plots of a subset of $5{\times}10^4$ feature vectors from three data sets: ■ ISMIR04, ■ LMD, and ■ CAL500. The original data is composed of 17-dimensional vectors - the M17feat set, and the plots were obtained through PCA projections.

songs. The CAL500 is mostly composed of pop music songs of the last four decades, although there are also other musical type songs such as Jazz, electronic, or Latin music ones. The figure shows the 2-dimensional and 3-dimensional scatter plots obtained through a PCA projection. The principal components were obtained based on the data of the ISMIR04 set only. The reason is to facilitate comparisons with the plots in Figure 2.6, which were obtained with the same PCA projections and where we differentiate the ISMIR04 data by genres. The figure shows that there is a high superposition between vectors of the different sets, although there are some regions that seem predominantly populated by vectors of one of the sets. This is noticeable in the 2D plot (top view from the 3D one) where the upper part of the plot is largely composed of points form the ISMIR04 set - the points with the yellow color. Looking at the Figure 2.6, we can see that these points belong to the JazzBlues and World genres. At first glance these results are a bit unexpected, at least if we group the data in terms the ISMIR04 classes. In the LMD set, which could be loosely classified as belonging to the World class, the features are not distributed in the manner they are in the ISMIR04 set. The CAL500 set, which in terms of genre is composed of many PopRock songs, the points also seem a bit out of place. Finally the features from the Classical genre are in regions of high superposition between all three data sets, despite the fact that this genre is not present in two of the three sets. The intent of this example is to give a rough idea of how low-level features are distributed, and the high superposition found between data set - in this case, clearly distinguishable ones. This example is also a sign of the difficulty of inferring high-level musical concepts such as genre based on the low-level descriptors. This issue is further explored in the next example.

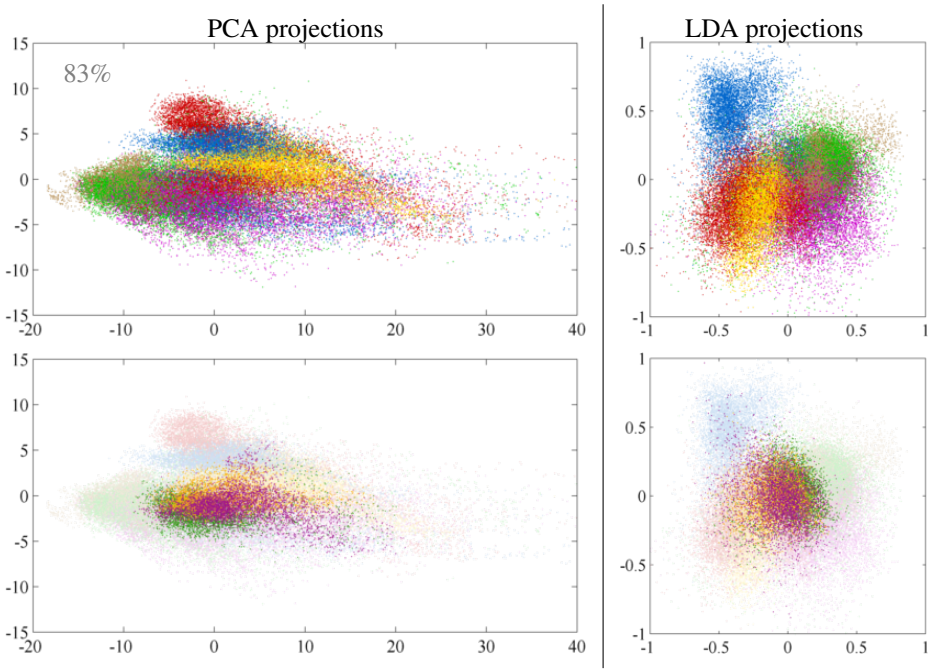



Figure 2.6: **Top plots** - 2D, LDA and PCA projections (83% corresponds to the variance of the first two principal components) of a subset of $6{\times}10^4$ feature vectors from the ISMIR04 data set - $10^4$ points per genre. The features used to represent the data are the M17feat. Colors correspond to genre: Classic, Electronic, JazzBlues, MetalPunk, PopRock, World.
**Bottom plots** - 3 songs used in the first example of this section, projected with the same transformation used in the top plots. The LDA projection obtained with the ISMIR04 data set does not achieve the separation shown in Figure 2.4, when only the data from the 3 songs is considered.

### Example: Feature Distributions of Distinct Genres

In this example we investigate how individual low-level features are distributed in terms of genre. The problem of inferring genre from these descriptor will be analyzed more carefully in Chapter 4. For this example we chose the ISMIR04 data set because the data is divided into six broad classes with readily discernible music characteristics: Classical, Electronic, JazzBlues, MetalPunk, RockPop, and World. This relatively small number of classes is suited for data visualization, and the loose genre taxonomy facilitates interpretation compared to other sets, for e.g. the LMD set where there are strong correlations between different genres. In Figure 2.6 are the scatter plots of PCA and LDA projections of this data set, where points from different genres are represented with different colors. We are interested in surveying how the features are distributed in terms of this six categories; we are not concerned in reflecting the a priori class distributions of this particular data set, hence, in Figure 2.6, there is an equal number

of points per class. Although there is a significant amount of superposition between points of different classes, the figure also shows that some regions of the feature space have a strong class-dependency. This is particularly noticeable in the scatter plots obtained from the LDA projections of the features. This substantiates the use of bag of frames methods (which are addressed in Section 2.3.3) and corroborates the fact that their performance is clearly above random. Nevertheless, we must point out that the purpose this example is just to give an a pictorial introduction to the genre classification problem, although in some aspects, it is an oversimplification of the task. For instance, the original PCA data lives in 17-dimensional space, and the LDA in a 5-dimensional space, therefore these representations do not show the "whole picture". Additionally, each song does not correspond to a single point in the feature space: and is composed by a sequence of feature vectors. This is shown in the bottom plots of the figures, where it is noticeable that the points corresponding to individual songs have variations that span over several class-specific regions (this fact is also depicted in Figure 3.5(c) of the next chapter).

### 2.2.5 Block-Level Features and Other Mid-Term Descriptors

On their own, the features just described are not sufficient to capture the dynamics of the music signals. This could be achieved by using techniques that directly take in account the signals temporal structure, like for e.g. the Hidden Markov Models, but reported performances of many standard time dependent methods are at best comparable to static ones, and no satisfactory ways of modeling the dynamics of music have yet been found. Some progress, though, has been made by taking an alternate approach. Instead of using complex time-dependent models, the temporal variations are encoded by dividing the audio signal into a series of windows a few seconds longs, and performing some temporal or statistical processing on the feature frames encompassed by those windows. In this fashion, a new set of features, also known as block-level features[12] can be calculated tailored to capture timbral, rhythmic, and harmonic and other aspects of the music signal that otherwise would be discarded by the frame vector representations. This way the same machine learning techniques can be applied to these new features, the progress being in the features themselves rather than in the classification or estimation algorithms. Recently there has been series of this type of descriptors proposed in literature [Pohle et al., 2009, Schindler and Rauber, 2012, Seyerlehner et al., 2010a, West, 2008], but most are based on the works of Tzanetakis [Tzanetakis and Cook, 2002] and Pampalk [Pampalk et al., 2002].

[12]The term was borrowed from Seyerlehner and we use it to refer to features that are obtained on a consecutive sequence of audio frames - i.e. a block of consecutive frames. Note also that we use the term more loosely than by the original author. For Seyerlehner some "explicit temporal processing has to be done" [Seyerlehner, 2010, p. 111], while we do not make such restriction.

Tzanetakis uses a *Timbral Texture Window* of one second, to compute the means and variances of spectral descriptors such as the centroid, roll-off, flux, MFCCs, and zero crossing rate, extracted from the short term audio frames present in the texture window (43 frames total). For classification, the mean of the texture window descriptors is used along two other long term audio characteristics based on the beat and pitch histograms; this results in a single feature vector representation of each song. The beat histogram describes how much periodicity is in the music signal at different tempo levels, and where each bin corresponds to a beat period. Several beats are extracted of the whole song, based on the peaks of the autocorrelation function of the signal's envelope, computed via Daubechies discrete wavelet transforms [Daubechies, 1988]. In order to account for the rhythmic contents of the music excerpts, the authors use several features derived from the beat histograms, like the relative amplitudes, rations and locations of the two highest peaks. The pitch histograms account for the tonal content present in the audio. An histogram is computed based on the energy contents of each semitone (i.e. each 100 cents). A one octave folded version of the histogram, corresponding to the Pitch Class Profiles previously described, is also calculated. The folded histogram contains the harmonic/melodic content, whereas the unfolded version also contains the pitch range of the piece. In a analogous manner to the rhythmic features, pitch descriptors are extracted from the folded and unfolded histograms. Since its appearance, this type of features have been made publicly available through the software framework MARSYAS (Music Analysis, Retrieval and Synthesis for Audio Signals) and are now common used in the MIR community (see for e.g [Flexer et al., 2005, Marques et al., 2011a, Meng et al., 2005, Ness et al., 2009, Peeters, 2007, Scaringella and Zoia, 2005]).

Another popular, rhythmic related set of block-level features are the *Fluctuation Patterns* proposed by Pampalk in [Pampalk et al., 2002, 2005] (for details see Section 2.2.4 of [Pampalk, 2006b]). The fluctuation patterns describe the periodic loudness fluctuations over time, or more precisely, they are a measure of the loudness amplitude modulation per frequency band. The fluctuation patterns are computed based on the several seconds long windows of the spectrogram of the audio excerpt (e.g 6 seconds in [Pampalk, 2001] and 3 seconds in [Pampalk, 2006b]), or other time-frequency representation, like the sonograms in [Lidy and Rauber, 2005]. For each segment, the frequencies are bundled into 20 Bark bands Zwicker [1961], and in each band the amplitude modulation of the loudness in the range of 0-10Hz are computed via FFT. The result is a matrix whose columns are the modulation frequencies (from 0-10Hz, with a total of 60 frequency steps), and rows are the Bark critical bands. Each music is described by the median fluctuations patterns from all the segments in the piece, represented in a $20 \times 60$-dimensional vector. Many descriptors inspired on this work have bee proposed in literature. For instance in [Pohle et al., 2009], the onset patterns are presented, which are basically the fluctuation patterns with some modifications. They analyze only onset parts of the signal, use a Cent scale,

increase the resolution of the Fourier transform, and represent the periodicities in a log scale instead of a linear one. A set of coefficients is then obtained via DCT. In [Seyerlehner, 2010] several block-level features are proposed, also inspired on the fluctuation patterns. These are based on the Cent spectrum, and are obtained by applying standard signal processing techniques to each block of the audio signal. The new block features are the spectral patterns (with delta and delta-variance versions) which characterize timbral contents of the blocks, the correlation patterns that model rhythmic and harmonic contents, and the spectral contrasts patterns for modeling the tonal contents. Fluctuation patterns have also become an household name in MIR community and many authors use them (see for e.g. [Moerchen et al., 2006, Schindler and Rauber, 2012, Seyerlehner et al., 2008, Turnbull et al., 2008b]). In all these works, the main strategy is to encode the temporal information by processing longer durations of the audio signal (seconds instead milliseconds) while the models are kept time-independent, and there are other similar approaches [Annesi et al., 2007, Bertin-Mahieux et al., 2008, Casey et al., 2008a, West, 2008], just to name a few. Note that, the process of extracting the block-level features, i.e. cutting the signal into a series of fixed-sized overlapping windows, only converts the sequence of short-term feature vectors that represent the audio excerpt, into a shorter sequence of descriptors. However, in order to use standard classification methods, which most works do, the descriptors sequence has to be converted into a single vector. The modus operandi is to compute over the whole song sequence, means, variances, and other statistics and summarize the results in a single vector. This strategy has the advantage that one can append to the single-vector representations other descriptors that are not extracted on a block-by-block basis, as the case of the beat histograms, which are calculated over the whole song.

Two final notes must be made about block-level features. First, converting a whole song into a single vector often results in very high dimensional representations, which are then used as inputs to standard machine learning systems. Nevertheless in high-dimensional spaces, point distances start to behave strangely and can compromise the use of these methods. In Section 6.2 we will address this issue. Second, we would like to point out that block-level features are the outcome of signal processing and machine learning methods applied to the frame-based features. This dilutes the line between what is feature extraction and what is classification, since "good" features can be handled by simple methods. Next, we review the most common classification techniques.

## 2.3 Music Classification Methods

In this section we review the problems of genre classification and autotagging and survey the most representative methods found in literature. These two problems are also studied exten-

sively in this thesis and therefore have received a special status compared to other classification tasks such as artist or mood classification. With the exception of autotagging, all others fall under the standard classification scenario, where each song or audio excerpt is associated with a single label, and the objective is to train a system on labeled data in order to predict the labels in unseen data. Many of the approaches found in literature for artist, mood and genre classification share the same design strategies and algorithms, although different sets of features may be better suited for each task. For instance, in mood classification rhythmic descriptors are the features of choice (albeit complemented with timbral and/or harmonic features), while in artist or genre classification many works, including ours, used exclusively timbral features. Although, there is a strong resemblance between mood and the other classification tasks, the problem itself has specificities that are beyond the scope of this thesis. For a review of mood classification (i.e. music emotion recognition) we refer to [Dhande and Tiple, 2013, Fu et al., 2011, Kim et al., 2010, Laurier, 2011, Sturm, 2013a]. Artist identification has received some attention in content-based MIR, but substantially less that its genre counterpart. Furthermore, the same features and classification techniques are often tested with both tasks (for e.g. in [Bergstra et al., 2006, Mandel and Ellis, 2005, Pampalk, 2005, Seyerlehner et al., 2010a]), and in our work we also opted for the same approach (see Section 3.4.3). Autotagging is also a classification problem, but distinct from the rest because it is a multi-label problem: each song is annotated with a set of labels that are not mutually exclusive like in standard classification. Autotagging is addressed next in Section 2.3.2, and in Chapter 5 we describe in depth the testing methodologies, the performance metrics, and explore aspects of this task related to generalization and evaluation issues.

All these tasks have been used to evaluate the performance of MIR similarity systems. These are related to just certain characteristics of the more general audio-based music similarity problem. Nevertheless audio-based music similarity has many limitations of its own. The objective of music similarity systems is to generate a score that describes the resemblance between songs. In other words, ideally they should approximate the relation: $\text{Song}_\text{A}$ *sounds like* $\text{Song}_\text{B}$, which in itself is not a well defined concept [Ellis et al., 2002]. Furthermore, pairwise song similarities grows quadratically with the number of songs in the data set and direct estimations via listening tests is only feasible for small data sets. In addition, listening tests should encompass the opinions of several subjects do to the noisiness of single user ratings [Amatriain et al., 2009], which further complicates the problem of building annotated collections with pairwise song similarity scores. To put it plainly, similarity is intrinsically subjective and this leads to ground truth ambiguity and raises serious difficulties when it comes to evaluating systems and comparing their performances [McKay and Fujinaga, 2006]. It is therefore understandable that the majority of researchers use classification tasks as a workaround for music similarity, in which automatic evaluation procedures are straightforward to implement.

It should be noted there is another task that is directly related to audio-based music classification [Fu et al., 2011]: instrument recognition. The purpose is to identify the instruments that are present in different intervals of the raw audio. Unlike genre, mood or artist classification, instrument identification is done at a segment level and therefore it is closer to sequence labeling rather that a standard classification problem. Furthermore each segment can be composed of several instruments playing simultaneously, which can be interpreted as a multi-label classification scenario. Instrument recognition is closely related to the area of Computational Auditory Scene Analysis (CASA) [Wang and Brown, 2006], where the aim is to audio break down the audio into individual constituents the same way humans do. For the case of music signals, these may be comprised of higher level music descriptors such as notes, chords, beats, melody, instruments, and so on. Instrument recognition presents many challenges, and the approaches used for this task differ substantially from the ones used to tackle genre, artist, mood classification and autotagging (see for e.g. [Hamel et al., 2009, Heittola et al., 2009, Herrera et al., 2003, Leveau et al., 2007, Martins et al., 2007]). Despite this fact, instrument recognition is an important aspect of music similarity [McKay and Fujinaga, 2005]. This and other music sub-components should be taken in consideration in order to obtain useful and meaningful music similarity measures.

### 2.3.1 Genre Classification

Genre classification has been present since the "early days" of the MIR community, and is one of the most studied areas in audio-based MIR. In [Aucouturier and Pampalk, 2008] genre classification has been called a "'flagship application", and in [Sturm, 2012b], an exhaustive survey on the subject, there are 467 referenced works on music genre classification including 13 doctoral dissertations [Ahrendt, 2006, Aucouturier, 2006, Gauss, 2009, Gouyon, 2005b, Mckay, 2010, Meng, 2006, Pampalk, 2006b, Pérez-Sancho, 2009, Pohle, 2010, Schnitzer, 2011, Seyerlehner, 2010, Tzanetakis, 2002, West, 2008]. Genre is probably the most popular way to categorize music, and the most widely used meta-data for browsing and searching music collections [Aucouturier and Pachet, 2003, Lee and Slaney, 2006, McKay and Fujinaga, 2006]. Many studies have been conducted on music genres, how they develop, on the role of cultural factors, and how humans categorize the diversity present in music [Aucouturier and Pachet, 2003, Brackett, 1995, Fabbri, 1999, Lippens et al., 2004]. Not only music genres are intrinsically related to sociological and cultural constructs, but the process itself in an adaptive malleable one where over time different genres are merged and others are split into sub-genres. In addition, there is the difficulty (if not impossibility) to systematize genre taxonomies [Sordo et al., 2008], which the absence creates an ambiguity and a high degree of subjectivity between label attributions. Furthermore, there are other limitations such as evaluation issues [Craft et al., 2007, Sturm, 2013c],

and practical questions such as mislabeled data [Sordo et al., 2008, Sturm, 2012c, 2013b], and data set creation. There are debates if genre can be inferred from low-level features, or if automatic classification systems are able to recognize genre at all [Wiggins, 2009]. Despite all of its weaknesses, genre remains one of the most studied areas in MIR [Aucouturier and Pampalk, 2008, Casey et al., 2008b, Fu et al., 2011, Sturm, 2012b], and continues to be an active research topic (see for e.g. [Agarwal et al., 2013, Costa et al., 2012, Hamel et al., 2013, Koerich, 2013, Salamon et al., 2012]). One of the reasons is because genre remains a efficient way to search and browse music collections, and even though there are confusions, redundancies and inconsistencies in attributing taxonomies, switching and navigating between different taxonomies is a natural process for most users [Aucouturier and Pachet, 2003, McKay and Fujinaga, 2006]. Note also that genre is based on more formal sociocultural agreements than, for instance, mood classification and general similarity measures which are strongly dependent on subjective factors and preferences.

The first works on genre classification rely on short-term frame-based features, which almost invariably include the all-purpose MFCCs, and other spectral based features commonly associated with timbre characteristics of the audio [Aucouturier and Pachet, 2002, Herre et al., 2001, Logan and Salomon, 2001, Soltau et al., 1998, Tzanetakis et al., 2001]. These works (and many others including our own [Langlois and Marques, 2009a]) are associated with the concept of timbre similarity, since most rely on features related to timbre characteristics of the audio [Aucouturier and Pachet, 2004]. The strategy to consider timbral features was in part inspired by psychological study by David Perrott and Robert Gjerdinger presented in 1999 at the annual meeting of the Society for Music Perception and Cognition [13]. In their study, commonly known in the MIR community by the study on "Scanning the Dial", they report on the human ability to categorize musical genre, where participants exceeded by far random classification, even on very short audio excerpts ( 250 ms - classifying ten genres with about 50% accuracy). Two other curious studies document the capability of carps (fish) to discriminate between the genres blues and classical [Chase, 2001] and the capability of pigeons to distinguish between the composers Bach and Stravinsky [Porter and Neuringer, 1984], and both the fish and birds seem to be very good at it. This contradicts the notion that genre is mainly a cultural issue and that the perception of music is not a specialized but rather generalistic. Also the fact that humans only need a quarter of a second to recognize genre suggests that music does not need to be broken down into higher-level representations, such as instrumentation, melody, orchestration, and such, in order for genre to be recognized, and that only a small localized timbral texture of the music signal is sufficient. Nevertheless, many counter examples can be found. For example,

[13] This work is extensively cited by the early works on genre classification and music similarity but was only published in 2008 in the Journal of New Music Research [Gjerdingen and Perrott, 2008]. In the same journal issue, Aucouturier and Pampalk also tell an interesting story about this paper [Aucouturier and Pampalk, 2008].

classical pieces such as Mozart piano sonatas or Bach's Golderg Variations interpreted by Glenn Gould on the piano have the same timbral characteristics as many of the works made by jazz pianists Bill Evans and Abdullah Ibrahim, and they clearly do not belong to the same genre. Genre and timbre are definitely correlated as shown in [Gjerdingen and Perrott, 2008] and the many works on genre classification using exclusively timbral features are substantially better than random guesses, which also corroborates this fact. Nowadays however, the general opinion in the MIR community is that this information alone is not enough, thus the efforts focused on descriptors that encompass other meaningful musical facets such as the ones described in Section 2.2.5.

### 2.3.2 Music Autotagging

Music autotagging refers to the task of automatically classifying music audio excerpts with respect to a number of high-level concepts (the "tags") from potentially very diverse facets such as emotions, musical instruments, genre, usage, etc. That is, based on human-annotated songs, autotagging systems are trained in a supervised fashion in order to predict the labels[14] of new songs. The systems are tailored to the fact that the task is more difficult than genre classification in that the number of classes is usually much higher (genres correspond in fact to one among many facets), and models must account for the possibility that multiple labels usually apply to a given excerpt. This is commonly known as multi-label classification problem and is usually solved taking one of two approaches. The first is to convert the multi-label problem into a set of binary classifications problems - one for each label. The instances with the labeled class are taken as positive examples while the other instances are considered negative examples. The second approach is to convert standard classifications techniques[15] to deal with multiple labels such as k-nearest neighbors [Zhang and Zhou, 2005], neural networks [Zhang and Zhou, 2006], and support vector machines [Elisseeff and Weston, 2001, Godbole and Sarawagi, 2004]. For a review of multi-label learning techniques we refer to [Madjarov et al., 2012, Tsoumakas et al., 2010]. In the next section, we describe some of the popular methods used in the context of music autotagging. An important aspect of music autotagging is how to evaluate system performances. There are several issues related to classification techniques and testing methodologies that make the evaluation of autotagging systems a much more complex process that the case of standard single-label classification. These are addressed in depth in Section 5.1.

[14] In this thesis the terms "tags", "labels" are used interchangeably, and refer to textual meta-data used to characterize songs or audio excerpts.

[15] Two of these techniques, namely k-nearest neighbors and support vector machines are used in our work and are described in Section 3.3. For details of other techniques such as neural networks or classification and regression trees we refer to [Bishop, 2006, Duda et al., 2012, Mitchell, 1997].

Tags are textual annotations that form a semantic description that can be used for example to search or recommend music. Methods that use this textual information are context-based methods which were briefly described in the beginning of this chapter. Autotagging, however, is a purely content-based method since the objective is to use only the audio information to predict the tags that are associated with a song. In a sense this is a step towards bringing together context and content-based music similarity techniques. Autotaggers that are able to predict well tags from the audio signal, can be used as a first processing stage by context-based methods to search and recommend similar songs. Nevertheless, as of today, autotagger have not yet attained a satisfactory performance to be able to aid context-based approaches, but this will probably change in the next few years. From a historical perspective in MIR research, autotagging can also be considered as evolution of the genre classification [Aucouturier and Pampalk, 2008] and other single label classification tasks, where now genres artists, mood and so fourth are just some of the tags among a large set of possible tags that can be used to characterize music contents. We briefly review the main ways to collect tags because depending on the approach, different types of tags are assigned to the music, and these may not be related to the audio signal at all. This can hinder the performance of content-based methods. Another detrimental factor is that some collection processes are more prone to misspelled tags, duplications, malicious labels, and other types of errors. There are four majors ways to collect tags [Bertin-Mahieux et al., 2010, Sordo, 2011, Turnbull et al., 2008a]:

- **Human Surveys:** This method is the most straightforward one. Humans listen to songs and tag them. One exemplary case is the Music Genome Project, where songs are annotated by musicologists using over 450 attributes to describe a song. The Music Genome Project is a trademark owned by Pandora Media Inc., which has the well known system Pandora[16], an internet radio that exploits the music genome annotations to create playlists. However the process of annotating songs is very costly in terms of human resources, time and money (in [Tingle et al., 2010, Turnbull et al., 2008a] the authors report that each song takes about 20mn to 30mn to annotate and that about 15000 new songs are annotated each month). The result of this process are high-quality tags closely related to musical content and they make sure that there is a high level of reviewer agreement for the same songs, which also adds a degree of objectivity for the tags. Nevertheless, since the Music Genome Project is used for commercial applications, and since they have spent enormous efforts in building it, it is unlikely that they will make it public (at least in its entirety), even for research purposes.

  The Computer Audition Laboratory at the University of California San Diego (UCSD) has made available a small data set comprised of 502 songs, called the CAL500[17], which

[16] www.pandora.com

[17] See details in Appendix C.5.

was annotated by payed undergraduate students [Turnbull et al., 2008a,b]. This data set is extensively used in research, including in our own work. Some limitations of this data set are the small size of the music corpus, and the subjectivity associated with some of the tags (e.g. usages).

The main advantage of survey methods is that the vocabulary (the set of tags) can be restricted to a well defined set of keywords, that transmit acoustically objective properties of the music. It is also important that the reviewer can recognize these music related characteristics. Therefore, survey methods are expensive since they can have costs (time, money, etc) in both training reviewers and in terms of time spent in the process of song annotation. This of course hinders scalability: this approach can not cope with the enormous amounts of new music that are produced every year, and only a small portion can ever be annotated.

- **Games:** The MIR research community has developed a few online tagging games to collect tags. This approach has the benefit to overcome the monetary costs associated with the survey methods, specially when annotations are given by groups of experts, as in the case of the Music Genome Project. Furthermore, it is relatively easy to impose some restrictions on the vocabulary which results in relatively clean tags like in the case of the ListenGame [Turnbull et al., 2007]. However there are other examples of music tagging games that do not impose a structured vocabulary, and where users are allowed to enter "free text": Tagatune [Law et al., 2007] and MajorMiner [Mandel and Elis, 2007]. However there is an incentive for users to provide appropriate tags for songs or song excerpts. In Tagatune in order for tags to be accepted two users must agree on them, and in MajorMiner the entered text is compared against previously collected tags. Therefore data acquired in this fashion is usually considered a lot cleaner than data mined from web documents or social tags. There are also some disadvantages. As Turnbull et. al. point out [Turnbull et al., 2008a], when players have a pre-defined set of tags to chose from, they tend to favor simple tags with generic content over more descriptive ones: "grundge" over "distorted electric guitar" (sic in the article). This contributes to the problem of sparsity in the annotated sets, with the simple more generic tags assigned to a large number of songs, while other tags are rarely chosen. Of course this could be solved with a large enough number of players, but it is not easy to create a successful game which the main purpose is to collect musical related tags based on an analysis of the audio.

  In our work, we use a data set derived from the Tagatune game, the Magnatagatune (MTT). The tags in the MTT set do not obey to restricted vocabulary, and have problems such as misspelling, duplications and others (details in Appendix C). We processed it and used a cleaner version of it, that we named the MAGTAG5k .

- **Web Documents:** The web is a rich source of information about music, but web contents are highly heterogeneous, unstructured and noisy. When mining the web, some of the retrieved wed-pages are irrelevant, and most of the content on relevant web-pages is also useless [Levy and Sandler, 2007]. Therefore the challenge lies in how to extract pertinent music information from the web. A number of authors have used this approach. For instance, Celma et. al [Celma et al., 2006a] mined MP3 web-blogs, Whitman and Ellis [Whitman and Ellis, 2004] mined music web-sites for music and artist related information, and in [Knees et al., 2008] the authors queried search engines with music related questions (e.g. artist name, album name, music review, etc) and collected relevant top hits. In all the cases, the information mined from the web is noisy, and standard language processing methods such as stop words and stemming techniques [18] are used to clean the documents. Furthermore, Term Frequency-Inverse Document Frequency (TF-IDF) representations, or some other form of document scoring technique (like relevance scoring proposed in [Knees et al., 2008]) are used to measure the general importance of the retrieved information. The tags extracted in this fashion, and due the unstructured, free-form nature way they are created, have the same problems than social tags, which are discussed next.

- **Social Tags:** The most common method for collecting tags is through large online music platforms that enable users to annotate the songs they listen to. These are commonly referred to as social tags [Lamere, 2008], and have no restrictions in terms of what words or phrases are used to characterize songs (among researchers, one paradigmatic example of a platform used to collect social tags is the Last.fm[19] - for instance, some works that use or describe data from Last.fm are [Bertin-Mahieux et al., 2008, Eck et al., 2007, Green et al., 2009, Lamere, 2008, Mandel et al., 2011, Sordo et al., 2010, Turnbull et al., 2007]). Unlike surveys or games, social tags are an abundant source of information, but the collection process is noisy, unstructured, and made by non-music experts. Therefore, social tags along with tags mined from web documents are considered more problematic than those obtained by the two other methods (surveys and games), and can pose some difficulties for those who want to exploit them. Misspellings and synonyms are common. The same tags can also convey different meanings (polysemy): for e.g. the tag "love" can mean that it is a romantic song or a favorite song of the tagger. This example also reveals another problem: many tags are used to convey personal tastes or experiences. Tags like "love", "seen live", or purely nonsensical ones ("random") have no relation with the audio and can not be predicted with autotagging systems. There is also the issue

[18] Stop words are words considered irrelevant for the query, and stemming is the process of reducing the words to their root, (see for e.g. [Porter, 1980]).

[19] www.last.fm

of malicious tagger behaviors, like artists tagging their own material in an attempt to make it more popular, or competitors tagging others material for the opposite reasons, or simply because of personal dislike (e.g. tagging pop artist Paris Hilton with "brutal death metal" or "your ears will bleed" [Lamere, 2008]). Moreover, new or unpopular items are rarely tagged (the cold start and long tail problems mentioned in Section 2.1.2), and music tagging behavior may be biased by age or location [Lamere, 2008] and may not be representative of the music tastes of the general population. Despite all of these limitations, social tags can bring a rich a complex view of music since they are made up of contributions of several thousands (or millions) users. So it is evident that social tags can be a valuable tool for MIR researchers, even though it is not completely understood how best to use them.

A few final notes about tags are in order. Ideally music should be annotated by musical experts that use a well defined fixed vocabulary (taxonomy), and that for each song confirm if all the tags are applicable or not. This is not the case for most of the publicly available data sets. The ones that are available can have several problems, specially when tags are collected in a unstructured and free-form nature. Two characteristic common to the majority of data used to train autotagging systems can also hinder their performances. These are the weak labeling and the sparsity of tags in most data sets. Weak labeling means that songs that are not annotated with a given tag do not necessarily mean that the tag should not be present. The sparsity is a consequence of very few tags applied in a large number of audio items, while most of the tags are rarely used. In Chapter 5 we will show that this tag imbalance makes evaluation measures fragile in a sense that small alterations in the tag sets can yield drastically different results.

### 2.3.3 Classification Algorithms

In this section we review the state of the art techniques for single and multi-label audio-based classification with special emphasis on genre classification and autotagging algorithms. Single-label classification encompasses other tasks besides genre, such as mood or artist classification, and the approaches here described can also be used in these settings. Furthermore, and although autotagging is a multi-label problem, many of the autotagging approaches are built upon previous work in genre and artists classification – which is not surprising since multi-label problems can be converted into several single-label binary problems, although it is also common to use classification methods adapted to handle multi-label scenarios. When necessary, we will mention which authors use which setting. It is also important to point out that classification tasks are often used as a proxy for music similarity. While classification involves groups of songs divided in classes, similarity measures involve computing distances between pairs of songs. As we shall see next, some classification models can be easily converted to yield similarity

measures between songs, but not all the models can be adapted in this fashion.

We start by analyzing the methods commonly known in MIR as the Bag-of-Frames (BoF) methods, a term borrowed from the "bag of words" models in text retrieval. The reason for the name comes from the fact that BoF methods disregard the audio signal temporal information. Audio-based descriptors are obtained in a frame by frame basis, and so the audio signal is converted into a sequence of feature vectors. BoF methods ignore the order of the vectors and build generative or discriminative models based on their long term distributions.

BoF generative approaches take an intermediate step to classification. They first model the features probability of each class before proceeding to classification – and in the BoF case, assume that the low level descriptors that represent each song are independent and identically distributed (i.i.d). Many of the early works opted for this generative approach, for example in [Logan and Salomon, 2001] a k-means algorithm was used to model spectral similarity between songs (and genres), and in [Aucouturier and Pachet, 2002] a similar strategy for genre classifications was done using Gaussian mixture models (GMMs) trained with the classical Expectation Maximization (EM) algorithm [Dempster et al., 1977] (see Section 3.1 for details of the two methods). In [Logan and Salomon, 2001] to measure distance between models the authors use the earth movers distance [Rubner et al., 2000], a technique used to compare two cluster representations. Clusters are analogous to "piles of earth" and the amount of "earth" (probability mass) necessary to move in order to convert one set of clusters into another measures the (dis)similarity between the two models. The GMMs are an estimate of the feature distributions, and an information theoretic way of measuring the differences between distributions is via a symmetrized version of the Kullback-Leibler (KL) divergence (Equation 3.23). In practice some approximations are necessary for the GMM case, and in [Aucouturier and Pachet, 2002] Monte Carlo sampling is used for that purpose, although other strategies are also available [Vasconcelos, 2001]. GMMs have been extensively used in MIR for audio-based classification and similarity estimation: for e.g. in [Aucouturier and Pachet, 2004, Berenzweig et al., 2004, Burred and Lerch, 2003, Flexer et al., 2005, Jensen et al., 2009, Jiang et al., 2002, Turnbull et al., 2008b, Tzanetakis and Cook, 2002]. With probabilistic models, one can also determine class memberships via maximum likelihood or maximum a posteriori estimation. One model is build for each class and new songs are attributed to the class with the highest likelihood (or maximum a posteriori probability). For classification, the models are built using data from songs from the same class, but it is also straightforward to estimate them based on the feature vectors from individual songs. Therefore, this type of approach can be used for more general music similarity measures (i.e. similarity between songs). For comparing two songs, one can either measure similarity via KL-divergence, or via maximum likelihood – i.e. compute the likelihood of the features from $\text{Song}_\text{A}$ using the model from $\text{Song}_\text{B}$. Note that this is possible, because each song is composed of several feature vectors, but a limitation is that the audio has

to be long enough in order to have enough data for a reliable estimation process.

BoF discriminate approaches do not need to estimate class conditional distributions and learn a direct map between inputs and labels. Nevertheless they need to deal with single vectors, not sequence of features which is how songs are represented. The most straightforward path is to compute means, variances and other statistics over the low-level feature-sequence and summarize the results in a single vector. The process of collapsing a sequence enables the use of standard classification techniques but leaves out the signals temporal information. In MIR research, the most common classifiers are k-nearest neighbors (k-NN) [Costa et al., 2004, Pampalk et al., 2005, Panda and Paiva, 2012, Pohle et al., 2009, Seyerlehner et al., 2008, Tzanetakis and Cook, 2002] and support vector machines (SVM) [Barrington et al., 2008, Costa et al., 2004, Mandel and Ellis, 2005, Ness et al., 2009, Panda and Paiva, 2012, Scaringella and Zoia, 2005] (note that some authors test both classifiers, including ourselves in [Marques et al., 2011a,b]). Most of these works, use bock-level features which carry temporal information that otherwise would not be available in a frame by frame basis. From a conservative point of view, they are not BoF models. However, these higher-level descriptors also form a sequence of feature-vectors, and the temporal information is loss in single-vector representations.

Many works use other standard machine-learning approaches: artificial neural networks (ANN) [Matityaho and Furst, 1995, McKay], classification and regression trees (CART) along with linear discriminant analysis [West and Cox, 2004], in [Huang et al., 2012] they extract via a random forest algorithm (a set of binary decision trees) audio vocabulary that is then processed with text retrieval methods (TF-IDF[20]), and [Ahrendt and Meng, 2005] use a multi-class logistic regression. Begstra et. al. [Bergstra et al., 2006] use AdaBoost (meta-algorithm that combines weak-learners to produce a strong classifier [Freund and Schapire, 1999]), in [Bertin-Mahieux et al., 2008] a stochastic version of the former called FilterBoost, and in [Foucard et al., 2011] boosting is also used with CART trees as weak learners. These are just a few examples of the classic audio-based approaches, but there are many others. Among the methods found in literature, we find one particularly interesting, because of its simplicity and, at the same time, the competitive performances it is able to achieve. It was proposed by Mandel and Ellis [Mandel and Ellis, 2005] and song or class models are estimated with a single multivariate Gaussian distribution. Their algorithm came in first in the artist identification task and third[21] in the genre classification task, in the MIREX 2005 competition. Since then it has become a standard audio similarity algorithm and has been used and adapted by many authors (for e.g. in [Flexer et al., 2010, Pampalk, 2006a, Pohle et al., 2009, Schnitzer et al., 2011, Seyerlehner et al., 2010a]) -

[20]TF-IDF stands for Term Frequency Inverse Document Frequency. It is a numerical feature used in text mining that measures the frequency of a word (term) weighted by how much information that word provides - if is common or rare across the document collection.

[21]The top two algorithms were from the same author [Bergstra et al., 2005].

note that three of these algorithms also won first place in Audio Music Similarity and Retrieval tasks: [Pampalk, 2006a] in MIREX 2006, [Pohle et al., 2009] in MIREX 2009, and [Seyerlehner et al., 2010a] in MIREX 2010. One of the main advantages of this generative approach is that model comparisons based Kullback-Leibler divergence have a close-form solution (see Section A.3.4), thus making the method scalable due to its computational simplicity. The single-Gaussian models are usually applied in the context of music similarity (i.e. distance estimation between song pairs). For classification, even thought the same strategy could also be used, another approach is often chosen. The songs are still modeled with single multivariate Gaussian densities, but individual songs are represented by a single vector composed of the Gaussian parameters (means and the covariance matrix entries). This makes possible the use of standard classification techniques, and in this respect, another curiosity stands out: classifications based on 1-NN achieve top or competitive performances [Mandel and Ellis, 2005, Pohle et al., 2009]. Part of the reason why some of these very simple methods achieve good performances has to do with the way the music signals are represented. For example in [Pohle et al., 2009], the authors report improvements in performance when onset patterns and coefficients (block-level features proposed by the authors), instead of fluctuation patterns alone, and tests on the ISMIR04 set showed accuracies above 90% which is among the top results (see performances in Table 3.4 where we compare our approach to some of the representative methods found in literature). Nevertheless, the performance of these simple methods is also comparable to more elaborate ones, even when the same feature representations are used, which is a bit counter intuitive.

### 2.3.4 Practical Considerations: Artist Filter

The evaluation of content-based algorithms in tasks such as genre recognition, and label inference, is typically done in a supervised manner, where part of the data is used for training and the other for testing. In these conditions, Pampalk [Pampalk et al., 2005] observed that over-optimistic results are obtained when songs from the same artist are in both training and test sets. Naturally, songs from the same artist often sound similar to each other. This is also reflected in spectral feature representations and therefore it is relatively easy for algorithms to identify these songs. This is known as the *artist effect*, and to counteract it one should use an artist filter: i.e. songs from the same artist should not be present in both training a test sets. Experiments reported in [Pampalk, 2006b, p. 66] showed that several algorithms optimized with no artist filter can have high accuracy differences (up to 50% in some tests) compared to artist filtered optimizations. Nowadays, artists filter is a common practice when evaluating similarity measures, and in our experiments we used artist filtered versions of two publicly available data sets: the Latin Music Database and the Magnatagatune data set (see Appendix C for details).

# Chapter 3

# VQMMs: Vector Quantization Markov Models

The focus of this chapter is on a method we developed for audio-based music similarity tasks: the Vector Quantization Markov Models (VQMMs). In VQMMs a single codebook is used to quantize all the feature vectors independently of the class or label associations, and prior to building any models. Once the quantization process is complete, the classification is done via a first order Markov model of the transitions of the quantized descriptors. The underlying strategy is simple: it consists of representing music with a finite-set of symbols and modeling their time-evolution.

This design structure of the VQMMs explicitly divides the training phase in two distinct stages: in the first stage the features are quantized and converted into a sequence of discrete symbols, and in the second stage a model of the symbols transitions is build. The quantization phase is an unsupervised training process while the second stage is a class dependent model estimation; the training of both processes can be done separately. This architecture confers an high degree of flexibility to the system since different quantization and time-modeling techniques can be applied in each stage and different combinations of these techniques can be tested. In this chapter we present the functioning details of these two stages, describe the different adaptations to the VQMMs necessary for dealing with conceptually different tasks, such as genre or artist classification, autotagging, and playlist generation, and report on the results obtained on these tasks using for each of the VQMMs stages several distinct algorithms, and combinations thereof. The VQMMs proved to be an effective enough way to tackle several similarity related problems, and showed performances on par with other state of the art algorithms. The tests were conducted on several publicly available datasets to facilitate comparisons. For this end, we also implemented a few standard classification algorithms commonly used by MIR community to have full control of all the parametrization and execution steps: the hidden Markov models, his-

togram and mixture of Gaussians density estimation techniques, and support vector machines. The tests with these alternate approaches resulted, on most occasions, in lower performance levels than the VQMMs, and hence, contributed to strengthen the validity of our method.

Having said that, we would like to add that during our research journey we noticed a few unexpected behaviors in several of the tasks we addressed; this led us to investigate a few aspects on inferring mid to high level musical concepts from multi-dimensional feature distributions, and on more problem-dependent issues such as data set biases, evaluation metrics, and generalization capabilities in genre and autotagging scenarios. Our observations indicate that many of these issues can have significant impact on the robustness of a wide range of audio similarity methods (including ours), and in a sense, work against the positive results obtained with the VQMMs. We are aware that our method is quite simple, specially compared to time-modeling approaches found in literature (see, for e.g. [Bogdanov et al., 2011, Coviello et al., 2011, Mandel et al., 2011]). Even so, its performance levels are on par with more elaborate techniques, which raises the question of why is this so. This also brings to mind the work of Mandel and Ellis [Mandel and Ellis, 2005], where class densities were approximated by a single Gaussian function, and despite the simplicity, the models achieved very competitive performances and are today common usage. We believe these and other counter intuitive behaviors are symptoms that something in not quite right in audio-based music similarity. This observation does not come as a surprise to the MIR community, the glass ceiling limit has been present for a decade, and other works have analyzed many aspects of this problem such as evaluation settings and methodologies [Craft et al., 2007, Sturm, 2012b], distance metrics [Jensen et al., 2007], hub formation [Flexer et al., 2012, Mandel et al., 2011], and other challenges that have hampered efforts to build systems capable of bridging the semantic gap between low-level audio features and high level musical concepts. In analyzing these matters, the VQMMs were a very useful tool. The flexibility of VQMMs derives from the separation of the first stage, the discrete representation of the feature space, from the transition model estimation of the second stage. In the first stage, we can control several aspects of the codebook generation, and tailor the codebooks to favor distinct characterizations and partitions of the feature space. For example, the quantization detail can be controlled varying the number of codewords used, or the codewords selection process can be altered to emphasize discriminative capabilities or label dependencies. This will help us examine, for instance, how features from different genres or tags populate the space and how this affects classification results. The second stage models the symbols transitions, and in this stage we can test how distinct partition affect the models and weigh the benefits of using this method compared to using the first stage alone, which can be easily converted to a classical bag of frames classifier.

In the next two chapters we will explore several issues that we believe need to be looked at more carefully when addressing two specific audio similarity related tasks: genre classification

and autotagging. The objective of this chapter is to give an overview of our method and how it performs on standard tasks. We describe the algorithms we used in the two stages of the VQMMs, and summarize the results obtained on publicly available data sets. We present the theoretical background and formulations that we will use throughout this thesis, and give in general terms a review of the operation and performance of our system. In that sense, some of the experiments we conducted will only be mentioned briefly here, and a more careful analysis will be left for other chapters.

This chapter is organized as follows. Section 3.1 describes the codebook generation procedure. In Section 3.2 we present the Markov based model for music classification and similarity estimation. Other methods for classification and similarity estimation, that were implemented and tested in this thesis are analyzed in Section 3.3. In Section 3.4 we perform several experiments under controlled environments to study specific issues, such as clustering methodologies, quantization detail, second order time dependencies and other questions that affect the VQMMs. Then we report the results obtained on two tasks, genre classification and autotagging and compare our performance with other methods found in literature. We continue with tests on similarity related tasks, namely artist identification and playlist generation, and describe the studies we did with HMMs and with other time-dependent models as a substitute for the Markov models, in the second stage of our approach. The chapter finalizes with some considerations on what the experiments presented so far can tell us about genre and autotagging.

## 3.1 Stage 1: Codebook Generation Procedures

The first stage of our approach consists in finding a more compact representation of the data by selecting a set of prototype vectors, or codewords, that represent well enough the feature space. This is achieved via standard clustering or density estimation techniques. A single codebook is used to quantize all the feature vectors, independently of their association with a given class or label. The process of codebook creation and subsequent feature quantization is done in a completely unsupervised manner. This is accomplished in two steps, the first selects a subsample of the data and the second builds the codebook based on that subset. This strategy alleviates the computational load associated with the tasks we undertook, and similar schemes have been adopted by other authors [Hoffman et al., 2009, Li and Sleep, 2005, Seyerlehner et al., 2008].

In the first step, the features are sub-sampled, ideally with enough points to have a thorough representation of the data space and maintain the computational load manageable. In many experiments we opted for selecting $k_1$ vectors from each music piece, which empirically, was found to be a simple and adequate scheme for our purposes. Furthermore, the tests we

conducted on this matter indicate that as long as we select a fairly representative subset of the feature vectors, the value $k_1$ or the method used to select the frames have a minor impact on our algorithm performance. For now, and unless otherwise specified, we will assume the sampling is done by selecting $k_1$ frames from each track.

In the second step, from the sampled sub-set are extracted $k_2$ feature vectors that comprise the final codebook. The number of symbols in the codebook, $k_2$, has an important role since it controls how well the feature space is characterized, and also how well our transition models are estimated. Intuitively, too few symbols are detrimental to our system since they lead to poor representations of the feature space, and high $k_2$ values can also result in weak models due to the lack on sufficient information to estimate the transition probabilities. In other words, the total number of transitions is $k_2^2$, and high $k_2$ values need large amounts of training data which may not be available. We will see in Chapter 5 that there are situations that contradict our expectations.

Next we describe the various clustering techniques that we used in either the first or the second stages of the codebook creation process. These are reasonably well known clustering strategies, and the reader may want to skip ahead if already familiar with them. The methods are random sampling, k-Means clustering, and mixture of Gaussian models. These have different levels of complexity and generalization capabilities, and we included the random sampling to have a base point of comparison with the other methods. However, before proceeding with the description of the clustering methods, there are a couple of points that need to be addressed about the codebook creation process. First we must stress that clustering is a very rich and active research topic [Filippone et al., 2008, Jain, 2010]. The goal of data clustering is to discover the *natural* grouping of a set of instances, which is one of the fundamental ways of learning. It is intrinsically related to density estimation since its purpose is to find the underlying data structure, but it is also linked to classification, feature extraction, and compression. It is not our intent to make original contributions in this vast area but rather use the available tools to construct a finite representation of the audio features, and to study how these representations affect the performance of ours and other techniques. This brings us to the second point which pertains to the codebook generation process and how it can be modified to favor different characterizations of the feature space. This can be done in several ways: a) by varying the level of information used in the codeword selection process, starting with a random choice and proceeding with more informative representations, b) by altering the level of detail of the quantization process (i.e. the total number of codewords) c) by emphasizing label dependencies either building the codebooks based on feature subsets with particular label associations or by using a selection process that favors codewords with high discriminative capacities. The different codebook building processes were useful for studying issues such as the influence of diverse partitions and different levels of quantization detail of the feature space on the performance of

genre and autotagging methods.

**Random Sampling:** This procedure is a trivial way of selecting a sub-set of frames or the codewords comprising the final codebook. The selection is done according to an uniform distribution. We used this method as a reference point for comparing and analyzing the benefits of other codebook generation techniques that use different amounts of supervised information. The bulk of the tests with codebooks obtained with uniform random sampling will be described in Chapter 4.

**k-Means Clustering:** k-Means is one of the most well-known unsupervised clustering algorithms which divides a set of $N$ vectors $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ into $K$ partitions (or clusters), with $K$ fixed a priori. The process minimizes the following objective function:

$$\mathcal{E} = \sum_{i=1}^{K} \sum_{\mathbf{x}_j \in \mathcal{S}_i} \mathrm{dist}\left(\mathbf{x}_j, \mathbf{c}_i\right) \tag{3.1}$$

where $\mathcal{S}_i$ is the $i^{\mathrm{th}}$ partition and $\mathbf{c}_i$ its corresponding mean, or centroid, and where $\mathrm{dist}(\mathbf{p}, \mathbf{q})$ is metric, or a distance measure between two elements, $\mathbf{p}$ and $\mathbf{q}$, of $\mathcal{X}$. In our tests we used the Euclidean and the cosine distance. The Euclidean distance between two vectors $\mathbf{p}$ and $\mathbf{q}$ is:

$$\mathrm{dist}_{\ell_2}\left(\mathbf{p}, \mathbf{q}\right) = \|\mathbf{p} - \mathbf{q}\| = \|\mathbf{q} - \mathbf{p}\| = \sqrt{(\mathbf{p} - \mathbf{q})^\top (\mathbf{p} - \mathbf{q})} \tag{3.2}$$

The cosine distance is:

$$\mathrm{dist}_{\cos}(\mathbf{p}, \mathbf{q}) = 1 - \frac{\mathbf{p}^\top \mathbf{q}}{\|\mathbf{p}\|\|\mathbf{q}\|} = 1 - \cos(\theta) \tag{3.3}$$

$\mathrm{dist}_{\cos}(\mathbf{p}, \mathbf{q})$ is a bounded function within $[0, 2]$, and $\theta$ is the angle formed by the two vectors. For orthogonal vectors, the distance value is $1$ and for parallel ones is either $\{0, 2\}$, depending on their direction. This formula is derived from the Euclidean dot product between two vectors: $\mathbf{p}^\top \mathbf{q} = \|\mathbf{p}\|\|\mathbf{q}\| \cos(\theta)$. Note that the cosine distance does not depend on the vector norms, therefore the distances between any two (non-zero) vectors $\mathbf{p}$ and $\mathbf{q}$, or their normalized counterparts, $\mathbf{p}/\|\mathbf{p}\|$ and $\mathbf{q}/\|\mathbf{q}\|$ yield identical measures.

The minimization of the objective function $\mathcal{E}$ is done by alternating between an assignment step, where each vector is assigned to the cluster with nearest centroid, and an update step, where the new partition means are calculated. This process is repeated iteratively until the centroids no longer change.

In our experiments, the k-Means algorithm proved to be an effective enough clustering technique due to scalability and computational reasons, and although simple, it was often our first choice compared to the other more elaborate methods here described.

**Mixture of Gaussian Models:** An alternative way of selecting the most representative frames in each song is to choose the ones with the highest likelihood. This entails the estimation of the distribution of the feature vectors $\mathbf{x}$. One standard way is to use Gaussian Mixture Models (GMMs) as an approximation to the real distribution:

$$\begin{aligned} p(\mathbf{x}) &= \sum_{i=1}^{K} \omega_i \, \mathcal{N}(\mathbf{x} - \mathbf{c}_i, \mathbf{\Lambda}_i) \\ \text{and} \quad \mathcal{N}(\mathbf{x} - \mathbf{c}_i, \mathbf{\Lambda}_i) &= \frac{1}{\sqrt{2\pi|\mathbf{\Lambda}_i|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{c}_i)^\top \mathbf{\Lambda}_i^{-1}(\mathbf{x} - \mathbf{c}_i)\right) \end{aligned} \tag{3.4}$$

where $p(\mathbf{x})$ is the probability density resulting from the sum of $K$ Gaussian densities, with mean $\mathbf{c}_i$, covariance matrix $\mathbf{\Lambda}_i$, and a priori probability of $\omega_i$ (with $\sum_i \omega_i = 1$). The parameters of the distribution $p(\mathbf{x})$ are estimated via the Expectation Maximization (EM) algorithm [Dempster et al., 1977]. This representation of the features distribution is also intrinsically related to clustering . Each individual Gaussian has assigned to it a partition of the feature space, where its density function has higher values than the rest. This probabilistic approach to clustering is more flexible than the previous k-Means algorithm. In fact, k-Means can be viewed in the light of GMMs as the particular case of equiprobable Gaussian densities with identical, isotropic covariances matrices, and with the centroids corresponding to the individual mean vectors of each Gaussian. Nevertheless, the EM algorithm is sensible to parameters like the number of Gaussians, the dimension and the number of data points, and can result in ill-conditioned solutions. This was verified in numerous experiments in which we used GMMs as a frame selection procedure, where we managed to train only MoGs with a reduced number of kernels that, in many situations, was too small for our objectives.

In the context of the VQMMs the MoGs models were used as a frame selection procedure, but also note that MoGs can be directly used in classification. This approach is one of the most common procedures in MIR, where class or label dependent distributions are estimated with a mixtures of Gaussians, and classification or label estimation is done via maximum likelihood (see, for e.g. [Aucouturier, 2006, Ellis, 2007, Flexer et al., 2010, Pampalk, 2006b, Turnbull et al., 2008b]).

### Further Considerations

Clustering is a rich area of research and there are many other clustering techniques we could also have used to generate the codebook. Amongst the most popular are, for example, the Isodata [Ball and Hall, 1965], spectral clustering [von Luxburg, 2007], Self Organizing Maps (SOMs) [Kohonen, 1982], competitive networks [McClelland and Rumelhart, 1986], non-negative matrix factorization [Paatero and Tapper, 1994], and many others. These methods have different levels of complexity, and some are more fit than others to extract complex patterns from the

data. However in regard to the first stage of our approach, our experience indicates that in audio similarity related tasks such as genre classification and autotagging, it is not critical to have a "good" characterization of the feature space in terms of the models used in its representation; the overall performance of the VQMMs is more affected by the level of quantization detail rather than the efficiency/complexity of the clustering methods used (these matters will be discussed in Section 3.4.1). The fact that different clustering algorithms do not alter the VQMMs performances significantly, is due in part, to the non-stationary nature of music signals. For example, the features of an individual song can cover a wide set of areas of the data space and jump erratically from region to region (see for e.g. Figure 3.5); the second stage of the VQMMs models the transitions of the quantized feature in the data space, and in that sense, it is fairly robust to more or less informed representation of the features, as long as the transitions are preserved.

## 3.2 Stage 2: Markov-Based Models

In this section, we first describe a Markov-based language model technique which is used in the second stage of our two-stage approach. Then, we describe the different model adaptations necessary to take into account when dealing with distinct tasks, such as genre classification, autotagging, or playlist generation. We also present other time-modeling procedures used as alternatives for the bi-gram based ones. These were implemented in order to quantify the benefit of taking into account temporal correlations besides the second order ones. These include the use of uni-gram in conjunction with bi-grams, tri-grams, and variable length n-grams.

### 3.2.1 Markov Models

The set of $k_2$ vectors obtained during the previous step is used to form a codebook that allow us to transform a track into a sequence of symbols. For each frame $\mathbf{x}$ a symbol $s$ corresponding to the nearest centroid $\mathbf{c}_i$ is assigned:

$$s = \underset{i=1\ldots k_2}{\operatorname{argmin}} \operatorname{dist}(\mathbf{x}, \mathbf{c}_i) \tag{3.5}$$

For the distance function, $\operatorname{dist}(\cdot)$, we tested the Euclidean and the cosine distances (Equations 3.2 and 3.3), although in our published experiments we only report the results obtained with the Euclidean distance since the cosine metric yielded sub-optimal performances. Once tracks are transformed into sequences of symbols, a language modeling approach is used to build classifiers. One of the benefits of our method is that once the models are computed, there is no need to have access to the audio files and features since only the sequences of symbols are used.

A Markov Model is built for each category by computing the transition probabilities (bi-grams) for each set of sequences. The result is a $k_2 \times k_2$ probability transition matrix, $\mathbf{T}_\upsilon$ for each model $\upsilon \in \Upsilon$, in a set containing the probabilities[1] $P(s_j|s_i)$ of a symbols $s_j$ being preceded by $s_i$:

$$\mathbf{T}_\upsilon = \begin{bmatrix} P(s_1|s_1) & P(s_2|s_1) & \dots & P(s_{k_2}|s_1) \\ P(s_1|s_2) & & \dots & P(s_{k_2}|s_2) \\ \vdots & & \ddots & \\ P(s_1|s_{k_2}) & P(s_2|s_{k_2}) & \dots & P(s_{k_2}|s_{k_2}) \end{bmatrix}$$

Each line $i$ in this matrix contains the transition probabilities of the symbol $s_i$ to all other $k_2$ symbols in the codebook, and therefore must sum, to one: $\sum_{\forall k} P(s_k|s_i)$=1. These are estimated using frequency counts, which are afterwards converted to probability estimates by normalizing each line $i$ by the total number of $s_i$ occurrences. To prevent unwanted situations where there are unused symbols[2], we initialize the transitions matrices diagonal elements with ones, before computing the frequency counts. In other words, the transitions $P(s_i|s_i)\ \forall i$ start with a count of one.

The matrices $\mathbf{T}_\upsilon$ can still contain zero-frequency transitions, and in such cases they can not be used directly for likelihood-based estimations. Many solutions to this problem have been studied by the Natural Language Processing community. Collectively known as "smoothing", and these include approaches such as the Expected Likelihood Estimator and the Good-Turing estimator [Manning and Schutze, 2002]. These particular techniques were not suited for our case because the size of our vocabulary is much smaller than that commonly used in Natural Language Processing. We opted for two different strategies, both of them consisting on on adding a small probability mass value to the transition counts. In the first one, a fixed constant, $\epsilon$, was chosen empirically and added to the whole transition matrix after the row normalization. We are aware that this results in total transition probability masses that are greater than one, but this does not affect the classification process since the likelihood of the data is biased for all the models by the same small offset. In the second one, we group the transitions by their first symbol $s_i$, and add a small and fixed percentage of the total number of $s_i$ occurrences, before performing the normalization - i.e. before computing the $P(s_k|s_i)\ \ \forall k$. In terms of the transition matrix, this is done on a line by line basis, and each line is added its own constant. Depending on the task at hand and on the size of the training set, more or less smoothing may be required. In both

[1] At this stage we are purposely omitting the class membership in $P(s_j|s_i)$, not to clutter the the formulation, and since we are describing a general model without going into specifics - this will be done next. However, for mathematical rigor, where we mention $P(s_j|s_i)$, it should be $P(s_j|s_i, \upsilon)$ instead. We will use the more complete notation when describing the implementation of this approach in different application environments.

[2] Note that during the codebook creation, which is done in an unsupervised fashion, all the centroids must have a non-zero number of allocated points and all the symbols are used; however since the transition matrices are estimated with sub-sets of all the training data, there can be cases where certain symbols are left unused.

solutions, i.e. either adding a fixed constant to the whole matrix after the transition probabilities normalizations, or a different constant in each line prior to the normalization process, this can be done with a simple adjustment. We performed some tests in situations where there was a reduced amount of data to train the models, such as in similarity between songs or experiments on small music collections, and here we used more smoothing. However, in most of our tests conducted on publicly available datasets, fixing the percentage to $5\%$ of the total of each symbol occurrences or setting $\epsilon = 10^{-10}$ proved to be sufficient for our purposes. Additionally both of this smoothing procedures yielded very similar performances, and ultimately it was irrelevant which one was adopted. Therefore, and unless otherwise mentioned, we will assume that a fixed $\epsilon = 10^{-10}$ was used in the tests here reported.

Once we have the transition matrices, we can estimate the likelihood of a genre or some other label for each song, and also obtain a similarity score between two pieces. There are, nonetheless, some model adjustments and subtleties that need to be taken in consideration, depending on what type of problem we are dealing with. Next we describe three types of testing scenarios commonly encountered in MIR similarity tasks, and the different implementation routes that need to be taken in each one.

### 3.2.2 Single-Label Classification:

This is the classic scenario in classification where there is a finite number of mutually exclusive classes. This is the case for example, in genre or mood classification and in artist identification tasks. In these settings, the test are conducted on a track by track basis, and the likelihoods of the track given each class are estimated. For a class $\gamma \in \Gamma$, where $\Gamma$ is the set of all mutually exclusive classes, and for the sequence $\mathcal{S} = \{s_1, s_2, ...s_N\}$, the likelihood of the model $\gamma$ is:

$$P(\mathcal{S}|\gamma) = P(s_1|\gamma) \prod_{n=2}^{N} P(s_n|s_{n-1}, \gamma) \tag{3.6}$$

or, in its logarithm form,

$$\begin{aligned} \mathcal{L}_\gamma(\mathcal{S}) &= \ln(P(\mathcal{S}|\gamma)) \\ &= \ln(P(s_1|\gamma)) + \sum_{n=2}^{N} \ln(P(s_n|s_{n-1}, \gamma)) \end{aligned} \tag{3.7}$$

The winning model $\hat{\gamma}_\mathcal{L}$ is the one that maximizes the likelihood:

$$\hat{\gamma}_\mathcal{L} = \underset{\forall\ \gamma \in \Gamma}{\text{argmax}}\ \mathcal{L}_\gamma(\mathcal{S}) \tag{3.8}$$

This formula can be adapted to incorporate the models' priors, and output the probability of each model given the sequence. This is done through the popular Bayes rule, which converts

the likelihood function into an a posteriori probability:

$$\begin{aligned} P(\gamma|\mathcal{S}) &= \frac{P(\mathcal{S}|\gamma)P(\gamma)}{P(\mathcal{S})} \\ &= \frac{P(\gamma)P(s_1|\gamma)\prod_{n=2}^{N} P(s_n|s_{n-1},\gamma)}{\sum_{\forall\gamma\in\Gamma} P(\gamma)P(\mathcal{S}|\gamma)} \\ P(\gamma|\mathcal{S}) &\propto P(\gamma)P(s_1|\gamma)\prod_{n=2}^{N} P(s_n|s_{n-1},\gamma) \end{aligned} \tag{3.9}$$

The denominator $P(\mathcal{S})$ is the likelihood of the data for all models[3], which is constant and can be discarded. The likelihood and the a posteriori models are equivalent only for equiprobable models. For classification, the winning model is the one with the highest probability:

$$\hat{\gamma}_{\text{MAP}} = \underset{\forall\ \gamma\in\Gamma}{\text{argmax}} \ln\left(P(\gamma|\mathcal{S})\right) \tag{3.10}$$

where $\sum_{\forall\gamma_i\in\Gamma} P(\gamma_i|\mathcal{S}) = 1$. This functional takes into account the prior mode probabilities, however, in practice more meaningful results might be obtained with the likelihood functional because some models can be over-represented, and this biases the model priors.

### 3.2.3 Autotagging:

Autotagging is a multi-label classification problem with a finite number of overlapping classes. That is, each class is not mutually exclusive like in the previous case. The common approach to solve it is either convert this into several binary classification problems, which we will discuss later, or adapt the score functions to directly perform multi-label classification. One possible adaptation is to do some sort of ranking like often performed in autotagging problems (see for example, [Bertin-Mahieux et al., 2008, Turnbull et al., 2008b] and many others). Here, we distinguish between two sub-tasks: annotation and retrieval. In annotation the tags are sorted according to their relevance in each test song, and only a fixed number of the top tags are chosen; note, however, that ranking procedure, optimal performance usually can not be achieved since the number of chosen tags may not correspond to the actual number tags in the song. In annotation, establishing which top tags can be done via direct application of Equation 3.7, which we rewrite here for clarity and referencing purposes:

$$\begin{aligned} \mathcal{L}_\theta(\mathcal{S}) &= \ln(P(\mathcal{S}|\theta)) \\ &= \ln(P(s_1|\theta)) + \sum_{n=2}^{N} \ln(P(s_n|s_{n-1},\theta)) \end{aligned} \tag{3.11}$$

[3] In Bayesian inference settings, $P(\mathcal{S})$ is also known as the evidence, and among other things, can be used to measure how well the prior is adjusted to the observations

where $\theta \in \Theta$, is one of the tags that constitutes the set, $\Theta$, of all possible tags. The annotation process boils down to selecting the tags with the top scores, $\mathcal{L}_{\theta_i}(\mathcal{S}) \ \ \forall \theta_i \in \Theta$.

In retrieval, and applying the same ranking solution, the task is selecting the top songs with a given tag. In this case we cannot directly use the previous equation because it is necessary to account for the symbol sequence length (otherwise shorter songs would come in first). This is done in the next model for the normalized likelihood of the song's symbol sequence $\mathcal{S}$ for the tag $\theta$:

$$\begin{aligned} \mathcal{L}_{\text{n}\theta}(\mathcal{S}) &= \ln\left(\left(P(s_1|\theta)\prod_{n=2}^{N} P(s_n|s_{n-1},\theta)\right)^{\frac{1}{N}}\right) \\ &= \frac{1}{N}\left(\ln\left(P(s_1|\theta)\right) + \sum_{n=2}^{N}\ln\left(P(s_n|s_{n-1},\theta)\right)\right) \end{aligned} \tag{3.12}$$

This normalization is done by averaging the log likelihood of the sequence by the number of symbols. This enables score comparisons between song of different lengths. During testing, the retrieval is done by selecting the songs with top scores, $\mathcal{L}_{\text{n}\theta}(\mathcal{S}_i) \ \ \forall \mathcal{S}_i$.

In binary classification scenarios, two likelihood models are built for each tag $\theta$ using Equation 3.11: one, $\mathcal{L}_\theta$, with the positive examples and the other, $\mathcal{L}_{\bar{\theta}}$, with negative examples - songs that did not have the tag $\theta$. With this two models, one can directly assign a tag if the positive score is higher than the negative one: $\mathcal{L}_\theta / \mathcal{L}_{\bar{\theta}} > 1$. This process can be viewed as assigning a binary tag vector of length $|\Theta|$ to each unlabeled song, where a $1$ means the corresponding tag is present and a $0$ otherwise. This avoids the evaluation biases created by annotating each song with a fixed number of tags, and was our preferred approach in autotagging problems. We can easily convert this likelihood functional into an a posteriori probability score like in Equation 3.9, however the likelihood method may be more appropriate since unbalanced positive and negative model probabilities can be particularly salient in autotagging. There are other implementation strategies that can be used in this situation such as constructing a unique background model and test all the positive models against it. This is a common strategy inspired from automatic speech recognition scenarios. This and other ways to tackle for model estimation in autotagging are addressed in Section 3.4. Also note that, in retrieval environments, where some sort of ranking is required, we can use the ratio of scores $\mathcal{L}_\theta / \mathcal{L}_{\bar{\theta}}$ between binary models and choose the top ones - and since this is a ratio between likelihoods, no normalization is necessary.

### 3.2.4 Distance Between Music Pieces:

The models described so far are trained with integer sequences extracted from sets of songs. In the limit, a model can be trained with a single song. We used this approach for comparing two songs, where we trained two models, $\theta_1$ and $\theta_2$, one with each song sequence $\mathcal{S}_1$ and $\mathcal{S}_2$, and

used the following divergence:

$$\mathcal{D}\left(\mathcal{S}_1, \mathcal{S}_2\right) = \mathcal{L}_{n\theta_1}\left(\mathcal{S}_1\right) + \mathcal{L}_{n\theta_2}\left(\mathcal{S}_2\right) - \mathcal{L}_{n\theta_1}\left(\mathcal{S}_2\right) - \mathcal{L}_{n\theta_2}\left(\mathcal{S}_1\right) \tag{3.13}$$

where $\mathcal{L}_{n\theta}\left(\mathcal{S}\right)$ is given in Equation 3.12. Although symmetric, this divergence is not a proper distance metric since $\mathcal{D}(\mathcal{S}_1, \mathcal{S}_2) = 0 \Rightarrow \mathcal{S}_1 = \mathcal{S}_2$ is not verified. To evaluate this functional we conducted some tests on small collection sets. We automatically generated playlists based on the divergence of Equation 3.13 between a seed song, and the rest of the tracks, which were then ordered according to their relevance. The reason to choose this evaluation strategy is because there are no publicly available "ground truth" data sets for pairwise song similarity[4]. The playlist generation tests we conducted with our method showed some interesting results - albeit in somewhat controlled scenarios. These along with implementation issues related to similarity estimation between two songs will be addressed in more detail when reporting the results on this task in Section 3.4.3.

### 3.2.5 Different Temporal Correlation Orders:

Our classification method is based on models of bi-gram probabilities whereas most approaches (i.e. the bag of frames techniques) rely on the classification of frame-based feature vectors or on estimates of statistical moments of those features computed on wider temporal windows. In order to quantify the benefit of taking into account transition probabilities other approaches were developed, but these were discarded because of substandard performance, and recurrent implementation and convergence problems that rendered some techniques unstable. We will report later on in this chapter some results in classification tasks pertaining to these methods that were documented in our publications or in our personal technical reports. The focus of this analysis is to understand if other orders of temporal correlation, besides the second are beneficial. These include tri-grams and n-grams but also include the "zero order" model based on the symbol frequencies of the codebook.

**Uni-grams and bi-grams:** We tested a linear combination of uni-grams and bi-grams. The model is given by:

$$\mathcal{L}_{1|2\gamma}(\mathcal{S}) = \alpha\mathcal{L}_{1\gamma}(\mathcal{S}) + (1-\alpha)\mathcal{L}_{\gamma}(\mathcal{S}) \tag{3.14}$$

where $\alpha \in [0, 1]$, and $\mathcal{L}_{1\gamma}(\mathcal{S})$ is the score of a sequence $\mathcal{S}$, independently of the symbols order. This is a simplification of the likelihood score, $\mathcal{L}_{\gamma}(\mathcal{S})$, in Equation 3.10. For the model $\gamma \in \Gamma$,

[4]The two main reasons why there is no such data sets are: 1) Pairwise song scores are too numerous (e.g. a 1000-song data set would require about half a million of such pairwise scores). 2) Pairwise scores are hard to obtain since they are derived, one way or another, from some sort of human-based similarity evaluation.

where $\Gamma$ is the set of all possible class models, the likelihood is:

$$\mathcal{L}_{1\gamma}(\mathcal{S}) = \ln\left(P(\mathcal{S}|\gamma)\right) = \sum_{i=1}^{N} \ln\left(P(s_i|\gamma)\right) \tag{3.15}$$

This representation gives us a rough estimate of how the feature space is populated, since it is based on frequency counts of vector quantized intervals of this space. With this discretization, probability mass estimates can be easily obtained for each class or model, and information-based theoretical measures can be devised. This will be covered in more detail in the following subsection, where we analyze the different aspects of this histogram representation.

**Tri-grams:** We also investigated methods based on tri-grams transition counts. In this case, the Markov model is a generalization of Equation 3.7, which results in a transition tensor or "cube matrix" with symbols probabilities conditioned to the previous two observed symbols. For a model $\gamma$ belonging to the set, $\Gamma$ of all models, the log-likelihood score of the sequence $\mathcal{S}$ is:

$$\begin{aligned}\mathcal{L}_{3\gamma}(\mathcal{S}) &= \ln(P(\mathcal{S}|\gamma)) \\ &= \ln(P(s_1|\gamma)) + \ln(P(s_2|s_1,\gamma)) + \sum_{n=3}^{N} \ln(P(s_n|s_{n-1}, s_{n-2}, \gamma))\end{aligned} \tag{3.16}$$

However, adding a extra order of temporal dependence increases by an order of $k_2$ the total number of transitions, i.e comparing to bi-gram counts, instead of $k_2^2$ there are now $k_2^3$ possible transitions. This has several negative implications which can seriously hinder the use of the tri-gram models. On one hand, the increase in total number of possible transitions means that for the same amount of training data, the probabilities are estimated with less data than their bi-gram counterparts, which results in extremely sparse transition matrices. On the other hand, and from a purely practical point of view, the processing load and memory requirements are greatly intensified and it can easily reach levels that render inoperable tri-gram based implementations. Even moderately small values of $k_2$, for e.g. $k_2 = 100$, result in transition tensors with one million entries, and in many tasks, a relatively high number of these tensors has to be estimated. For instance, in the CAL500 data set, which is used for autotagging, there are $174$ labels. In order to test this approach on this data set, $174$ transition tensors -one per tag- would have to be estimated, or if we consider positive and negative tag models, twice that amount. This, along with the fact that the tri-gram models did not show superior performances to the bi-grams counterparts (see Section 3.4.4), led us to abandon this type of models.

**Variable length n-grams:** Finally, we also tested methods based on variable length n-grams, obtained with an adaptation of the Lempel-Ziv compressing algorithm [Ziv and Lempel, 1977].

This approach was inspired by Li and Sleep [Li and Sleep, 2005], where a feature vector is extracted from each sequence and similarity is calculated by the inner product between two of such vectors. We opted by a moderately different approach. We built a global dictionary by parsing all the training data with the LZ78 algorithm [Ziv and Lempel, 1978]. Note, that this global dictionary is composed of variable length codewords consisting of consecutive discrete symbols. Also, not all the codewords resulting from the parsing process were included in the dictionary: only the ones with more that a predefined number of hits, along with all individual symbols (see Section 3.4.3 for details). After building the dictionary we used the frequency of occurrence in each category to estimate the probability of the codewords given the class. For classification, we first count the number of times each codeword appears in the test sequence. This is done via an adaptation of Lempel-Ziv encoding schemes, where we use the global dictionary to initialize the encoder, and do not allow it to grow[5]. Then the likelihood of the test sequence given each class model is calculated, the winning class being the one with the highest score. This is done via Equation 3.15, where we used the Lempel-Ziv codewords instead of the individual symbols $s_i$. The underlying assumption in this method is that the order of the codewords is unimportant. However, there is important difference compared to the bag of frames classifiers: the short-time dependencies of the audio signal are implicitly modeled by the multi-gram codewords.

The results obtained with these different time dependent models are summarized in Section 3.4.3. The performance of these models was inferior compared to the VQMM, and possible causes are also addressed along with the results.

## 3.3 Alternative Tested Approaches

In this section we describe three alternative methods we tested. The first is a well known technique for modeling time-varying processes: the hidden Markov models (HMMs). The other two are standard classification algorithms: the k-nearest neighbors (k-NNs) and the support vector machines (SVMs). HMMs, k-NNs or SVM are fairly common approaches used in MIR similarity tasks (see, for e.g., [Aucouturier, 2006, Pampalk, 2006b, Seyerlehner, 2010]), but we chose to implement them to have a full control of all the implementation steps, and thus be able to make comparisons with the VQMMs more sound. Furthermore, these alternative implementations rely on a quantized version of the features, i.e. the output of the first stage, in the context of our two stage approach. This fact also facilitates analogies between the VQMMs and these systems, which is often not the case for most works found in literature since they depend on continuous representations of the feature space. Also note that the HMMs can use directly the

[5]This is an adaptation of the Lempel-Ziv-Welch algorithm [Welch, 1984], rather than the LZ78.

quantized sequences, but the other two methods cannot because they work with single vectors as inputs. For this end, the quantized sequences were converted into a vector format using a histogram based strategy, which we will also detail in this section.

### 3.3.1 Hidden Markov Models

A technique commonly used to model time-varying processes is the hidden Markov models (HMMs). HMMs have been around since the late 1960s [Baum and Petrie, 1966], but only became popular in the late 1980s, partly due to its success in speech applications. The HMMs are an extension of the observable Markov models where the observations are a probabilistic function of the states, which are hidden. In other words, in HMMs there is a doubly embedded stochastic process, the first one is hidden, and can only be observed through another stochastic process that produces the observations. This confers a higher degree of flexibility than its observable counterparts, and are often one of the first choices in temporal pattern recognition applications such as automatic speech recognition [Jelinek, 1997, Rabiner and Juang, 1993], handwriting [Hu et al., 2000, Makhoul et al., 1994], and gesture recognition [Nam and Wohn, 1996], natural language modeling [Manning and Schutze, 2002], just to name a few.

A HMM is characterized by the number of hidden states, the initial state distribution, the transition probabilities between hidden states, and the observation (or emission) probabilities in each state. Considering that $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ is a sequence of $N$ observations, there is also another sequence of unobserved (latent) variables, $\mathcal{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N\}$, that describe the states in the HMM. Each vector $\mathbf{z}$ is a $L$-dimensional binary vector with only one element equal to 1 and all the other equal to 0, with $L$ being the total number of states in the HMM. The probability distributions of the variables $\mathbf{z}_n$ (with $n = 1, \ldots, N$) obey the Markov property, which assumes that the current state is only dependent on the previous one:

$$p(\mathbf{z}_n|\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{n-1}) = p(\mathbf{z}_n|\mathbf{z}_{n-1})$$

This state transition probabilities are usually represented by a square matrix $\mathbf{A}$ of $L \times L$, where each element $a_{ij} \equiv p(z_{n,j} = 1|z_{n-1,i} = 1)$, with $0 \leq a_{ij} \leq 1$ and with $\sum_j a_{ij} = 1$. We can write this conditional distribution explicitly in the form:

$$p(\mathbf{z}_n|\mathbf{z}_{n-1}, \mathbf{A}) = \prod_{j=1}^{L} \prod_{i=1}^{L} a_{ij}^{z_{n-1,i} z_{n,j}} \tag{3.17}$$

The variable $\mathbf{z}_1$ does not have any previous state and it is described by the marginal distribution $p(\mathbf{z}_1)$, which can be represented by the probabilities $\pi_l \equiv p(z_{1,l} = 1)$:

$$p(\mathbf{z}_1|\boldsymbol{\pi}) = \prod_{l=1}^{L} \pi_l^{z_{1,l}} \quad \text{where} \quad \boldsymbol{\pi}^\top = [\pi_1, \ldots, \pi_L] \quad \text{and} \quad \sum_{l}^{L} \pi_l = 1 \tag{3.18}$$

The specification of the HMM model is complete by defining the emission probabilities, which are the conditional probability distributions of the observed variables

$$p(\mathbf{x}_n|\mathbf{z}_n, \phi) = \prod_{l=1}^{L} p(\mathbf{x}_n|\phi)^{z_{n,l}} \tag{3.19}$$

where $\phi$ are the parameters that govern the emission distributions, which can assume a wide range of parametric forms, such as discrete mass functions, Gaussians, mixtures of Gaussians, and many others. The joint distribution over both the observed and the latent variables is then given by

$$p(\mathcal{X}, \mathcal{Z}|\boldsymbol{\pi}, \mathbf{A}, \phi) = p(\mathbf{z}_1|\boldsymbol{\pi}) \left( \prod_{n=2}^{N} p(\mathbf{z}_n|\mathbf{z}_{n-1}, \mathbf{A}) \right) \prod_{m=1}^{N} p(\mathbf{x}_m|\mathbf{z}_m, \phi) \tag{3.20}$$

The objective in HMMs training is to find, based on a given output sequence(s), the best set of state transitions and output probabilities. This can be achieved via maximum likelihood, where the likelihood function is obtained by marginalizing the previous equation over the latent variables

$$p(\mathcal{X}|\boldsymbol{\pi}, \mathbf{A}, \phi) = \sum_{\mathcal{Z}} p(\mathcal{X}, \mathcal{Z}|\boldsymbol{\pi}, \mathbf{A}, \phi) \tag{3.21}$$

Direct optimization of the likelihood leads to complex expressions with no closed form solutions, and becomes intractable even for time processes with small durations. However, a local maximum likelihood can be derived efficiently using a form of the EM algorithm, also known as the forward-backward or the Baum-Welch algorithm, which iteratively updates the model parameters to better explain the observed sequences (further information on HMMs can be found, for e.g., in [Bishop, 2006, Rabiner, 1989]).

In the context of our two-stage approach, the HMMs training is done with discrete symbol sequences, resulting from the quantification process of the music pieces in the training set. We tested two different architectures commonly adopted in the context of HMMs: the left-right model (Figure 3.1) and the fully connected model (Figure 3.2). The experiments pertain to the genre classification task, where one HMM was trained for each genre and classification was based on the likelihood scores of each model. The results are presented in Section 3.4.4. The performances are inferior to the ones obtained with the VQMMs. Similar negative results were reported by other researchers [Aucouturier, 2006, Flexer et al., 2005, Scaringella and Zoia, 2005, Soltau et al., 1998], where HMMs did not fare as well as other methods, namely time-independent ones.

### 3.3.2 Histogram-Based Classification Systems

We tested another feature representation strategy based on histograms of the symbol occurrences. Here, each song represented by a $k_2$-dimensional vector consisting of the normalized

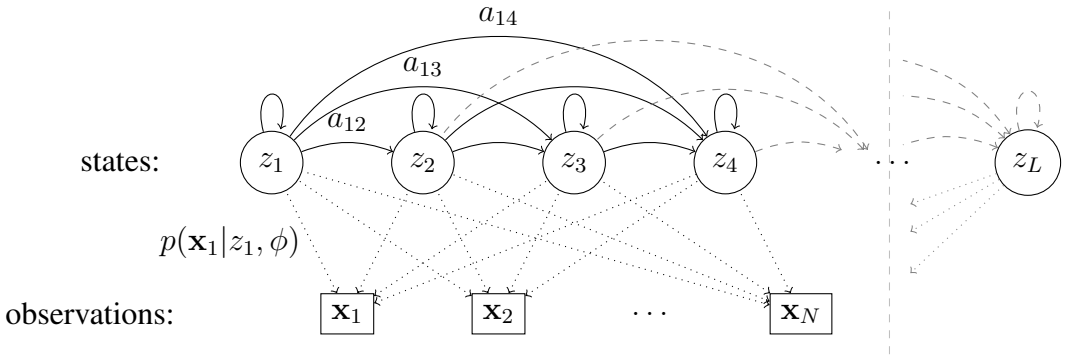



Figure 3.1: An example of the state transition diagram of an HMM with left-right connected states with three delays. States are represented with circles, while the observations are represented with squares. $a_{ij}$ is the probability to transition from state $z_i$ to state $z_j$. $p(\mathbf{x}_n|z_j, \phi)$ is the probability to emit the observation $\mathbf{x}_n$ in state $z_j$. In this type of configuration, once the state has been vacated, it cannot be re-entered.

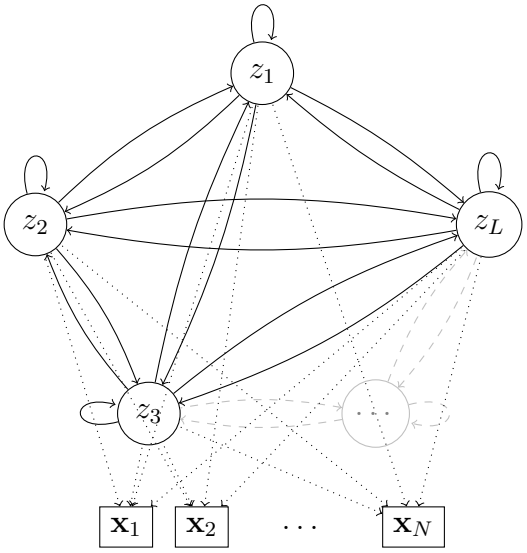



Figure 3.2: The state transition diagram of an HMM with fully connected $L$ states, and the corresponding sequence of $N$ symbols (observations) generated by this model. States are represented with circles and observations with squares. In this type of configuration there is no restrictions to the state transitions, and any given state can transfer to any of the other states.

symbol occurrences - that is, each dimension is the number of occurrences of the corresponding symbol divided by the total number of symbols in the song. With this alternative data characterization we tested two classification algorithms, commonly used in MIR: k-nearest neighbors and support vector machines, which we will describe shortly. Note that in order to use these two algorithms as many other standard techniques in MIR scenarios, each audio excerpt, typically represented by a sequence of feature vectors, has to somehow be converted into a single vector. In our case this was done by the histogram representation, and the classification algorithms treat the histograms as points in a $k_2$-dimensional space [6]. With this representation we can also obtain an approximation of the features distribution for a given set of songs, or for the whole data set. This is due to the fact that the vector quantization stage of our approach divides the feature space into a set of $k_2$ regions[7], one for each centroid, with the region consisting of all points closer to that centroid, and the histogram counts give us an estimate of how densely each one is populated. This is important to ascertain, for instance, if in genre classification the low-level feature capture somehow the (musical) specificities of each genre, which is a common underlying assumption in many methods founds in literature. Our findings contradict this assumption, but this will be one of the issues analyzed in the next chapter. For now, we will address just the mathematical framework necessary to contextualize the experiments reported in this chapter.

**k- Nearest Neighbors:**

The k-NN algorithm is one of the simplest of all machine learning algorithms. It is a non-parametric technique, where a given point is classified by a majority vote of its neighbors. This is known as instance-based or lazy learning: no discriminative functional needs to be derived (as in Neural Networks, Linear Discriminants, SVMs, and many others) and the classification process is done concurrently with the testing phase. That is, for a given test vector $\mathbf{x}$, one finds the closest $k$ vectors in the whole training set, and assigns to $\mathbf{x}$ the most populous class amongst its $k$ neighbors. For the distance metric, we used in our experiments the Euclidean distance and the cosine distance (Equations 3.2 and 3.3).

We also used two other functionals inspired from a probabilistic perspective. Both distances are based on the Kullback-Leibler divergence [Kullback and Leibler, 1951] between two (discrete) densities, $\mathbf{p}$ and $\mathbf{q}$:

$$\mathrm{KL}\left(\mathbf{p}\|\mathbf{q}\right) = \sum_{i=1}^{k_2} \ln\left(\frac{\mathbf{p}_i}{\mathbf{q}_i}\right)\mathbf{p}_i \tag{3.22}$$

[6] Note, however that with the histogram representations, the points lie at most in $k_2-1$ dimensional space. This is due to the fact that any histogram vector $\mathbf{p}$, must fulfill $\sum_{i=1}^{k_2} p_i = 1$. This restriction takes away one degree of freedom.

[7] These regions are commonly known as the Voronoi cells, after the work of Georgy Voronoi [Voronoi, 1908].

where in our case, $\mathbf{p}$ and $\mathbf{q}$ are obtained by the normalized song histograms, and $\mathbf{p}_i$ is the $i^{\text{th}}$ bin of the histogram $\mathbf{p}$. In order to use this divergence, the distributions $\mathbf{p}$ and $\mathbf{q}$ must have non-zero entries; however this can happen if one or more symbols from the codebook are not used in the representation of a given music piece. To overcome this limitation we add one hit to all histogram bins before performing the normalization. Additionally this divergence is not symmetric since $\mathrm{KL}(\mathbf{p}\|\mathbf{q}) \neq \mathrm{KL}(\mathbf{q}\|\mathbf{p})$, so we used two adaptations:

$$\mathrm{dist}_{\mathrm{KL}}(\mathbf{p},\mathbf{q}) = \tfrac{1}{2}\mathrm{KL}(\mathbf{p}\|\mathbf{q}) + \tfrac{1}{2}\mathrm{KL}(\mathbf{q}\|\mathbf{p}) \tag{3.23}$$

$$\mathrm{dist}_{\mathrm{JS}}(\mathbf{p},\mathbf{q}) = \tfrac{1}{2}\mathrm{KL}(\mathbf{p}\|\mathbf{m}) + \tfrac{1}{2}\mathrm{KL}(\mathbf{q}\|\mathbf{m}) \qquad \text{where } \mathbf{m} = \tfrac{1}{2}(\mathbf{p}+\mathbf{q}) \tag{3.24}$$

the first distance is a symmetric version of the Kullback-Leibler divergence, and the second one is the Jensen-Shannon divergence. Both of these divergences are not strictly a distance, since they do not fulfill the triangular inequality (although the squared root of $\mathrm{dist}_{\mathrm{JS}}(\mathbf{p},\mathbf{q})$ does [Endres and Schindelin, 2003]).

**Support Vector Machines:**

The Support Vector Machines (SVMs) is one of the most widely used algorithms in machine learning. First devised by Vladimir Vapnik, the standard version (soft margin) later presented in the seminal paper [Cortes and Vapnik, 1995], the SVM is a supervised learning algorithm for binary classification which implicitly maps the training points into a high (or infinite) dimensional space, and constructs an hyperplane that separates the examples as cleanly as possible while maximizing the margin between the nearest, well classified points of the two classes - the support vectors. Given a set of training instances-label pairs $(\mathbf{x}_i, y_i)$, for $i = 1,\ldots,N$, with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1,+1\}$, it can be shown that for the linear case, the SVMs find the solution for the following optimization problem:

$$\begin{aligned} &\min_{\mathbf{w},b,\xi} \tfrac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i \\ &\text{subject to:} \quad y_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right) \geq 1-\xi_i, \quad \text{with } \xi_i \geq 0\ \forall i \end{aligned} \tag{3.25}$$

where $\xi_i$ are non-negative slack variables[8] which measure the degree of misclassification of the data, and $C$ is a parameter chosen by the user which controls the penalty for the errors. The term $\frac{1}{2}\|\mathbf{w}\|^2$ maximizes the margin, while $\sum \xi_i$ is a upper bound for the training error, since for an error to occur, the corresponding $\xi_i$ must exceed unity. Figure 3.3 is a representation of the hyperplanes, $\mathbf{w}$, $\xi$, and the support vectors for a 2D example. To create nonlinear classifiers one

[8] In an optimization problem, a slack variable is a variable added to an inequality constraint to transform it into an equality.

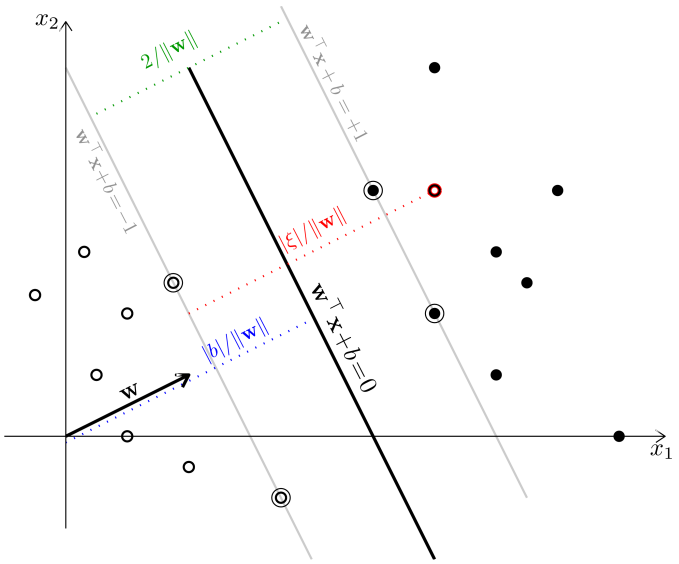



Figure 3.3: Linear separating hyperplanes for non-separable tow-class case (dots and circles). The decision boundary corresponds to the black line, while the margins are the parallel gray lines in both sides of the decision boundary. The points on top of the gray lines are the support vectors - the ones circled.

can preprocess the data by a nonlinear function $\Phi$: use $\Phi(\mathbf{x}_i)$ instead of $\mathbf{x}_i$. This way the data is mapped into a richer feature space where one can construct a linear separating hyperplane, which is equivalent to a nonlinear decision boundary in the original feature space. Note that the training algorithm only depends on the data through the dot products in the transformed, high-dimensional feature space, i.e. on functions of the form $\Phi(\mathbf{x}_i)^\top\Phi(\mathbf{x}_j)$. To keep the computational load manageable, the mappings, $\Phi$, used in SVMs schemes are chosen so that the dot product may be computed in terms of a kernel function, $K$, so that $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^\top\Phi(\mathbf{x}_j)$. This way, one only needs to use the kernel function in the training process without ever needing to explicitly know what $\Phi$ is[9]. Among the most commonly used kernels are the linear, the polynomial, the Gaussian or radial basis function (RBF) and the sigmoidal kernels. In our experiments, namely in [Marques et al., 2010, 2011b], we used the RBF kernel, while in [Marques et al., 2011a], a linear kernel was used. This was the system proposed in [Ness et al., 2009], which we used as a benchmark in order to better compare our experiments with previous literature and to facilitate the reproducibility of our experiments. SVMs have been originally designed to solve a two class problem but there are many strategies to allow SVMs to work with more than two classes - the most common being to convert a multi-class problem into several two-class problems. Further information on SVMs can be found in [Burges, 1998, Smola and

[9]This is also known as the "kernel trick". Also note that not every function is a valid kernel function. For that to happen, $K$ must satisfy the Mercer's condition [Mercer, 1909]

Schölkopf, 2004, Vapnik, 1995].

## 3.4 VQMMs Operational Experiments

The objective of this section is to present the functioning details of our method, how it fares in different testing environments, and how it compares to other techniques.

We start this section with a description of how different parametrization, design methodologies, and other functional requirements affect the way the VQMMs operate. We divide this analysis in two parts, one for each stage of the VQMMs. In the first part, we investigate how different codebook generation algorithms and different codebook sizes influence our method's performance. In the second, we analyze the symbol transition estimation process and how different codebook sizes and signal lengths affect it. We also conduct tests to measure the benefits of modeling the symbol transitions probabilities compared to just using the information present in the quantized features without accounting for the temporal dependencies.

We then continue with an overview of the VQMMs performances obtained on several music similarity related tasks. We first report the results obtained on the genre and autotagging tasks, and then describe the experiments we did on artist identification and playlist generation. Genre and autotagging play an important role in our research, and they are analyzed in detail in the next two chapters. In that sense, the results presented here are just a summary of the tests were carried out on several publicly available data sets used for these two tasks. We compare these with others found in literature, briefly address some data set or evaluation related problems but do not explore these matters any deeper, since they are addressed latter on. We then describe the experiments we did on artist identification and playlist generation. Artist identification task is similar to genre classification since they both are single label classification problems, and typically there is a strong correlation between artist and genre. However, genre is by far the most popular task, and in our research it was the predominant choice for evaluating the VQMMs in single label testing scenarios; therefore, the results artist identification are somewhat limited compared to its genre counterparts. The same can be said for the playlist generation task. The reason for not conducting an extensive batch of tests on this tasks has to do with the subjectivity of the evaluation process, and the inherent difficulty in constructing testing scenarios that can be used for comparison with other authors.

We then describe the experiments we did on alternative methods for the second stage of the VQMMs. These consist of using uni-grams in conjunction with bi-grams, using tri-grams, variable length n-grams, and hidden Markov models. We then finalize this chapter with a brief summary and some commentaries on the results presented so far.

### 3.4.1 Algorithmic Design and Model Tuning

The objective in this section is to analyze distinct parametrization and methodologies, and how they affect the performance of the VQMMs. To that end, and not to clutter this section with unnecessary results, we restrict ourselves to a specific set of features, the M17fea feature set (introduced in Section 2.2.4), and our tests were performed on publicly available sets, with a leaning for two specific data sets: the ISMIR04 and the CAL500 (see Appendix C). The reason to choose the M17feat set is because they are a common mix of features used by us and many other researchers. The reason to favor these particular sets is because they are two of the most popular ones among the MIR community, and are associated with two specific tasks that we analyze in detail and constitute the core of the next two chapters: genre classification and autotagging. In this way, the results and experiments here described also will set the stage for a posterior study of these tasks.

Finally, we would like to add that the issues covered in this section do not comprise the bulk of the studies and experiments we conducted on the VQMMs mode of operation. Other issues that potentially could benefit our algorithm, such as the choices of low-level audio features, window and hop sizes, distance metrics and others were also analyzed, and the results are presented in Appendix B. These were left for the appendix, because they do not significantly contribute to the understanding of how the VQMMs operate and how different parametrizations and methodologies affect them.

#### Codebook Generation Methodologies and Parametrization

The codebook creation process is divided into two steps. In the first step, a sub set of audio features is chosen, and with this sub set, the final codebook comprised of $k_2$ elements is built. We investigated various ways of obtaining this sub set. We tested several algorithms for feature selection, namely, Gaussian mixture models, k-Means and random selection. The approach consisted in using these techniques to pick $k_1$ feature vectors in each song, and with that construct the sub set on which the final codebook is based. The results are shown in Table 3.1 where performance measures (accuracies and F-scores[10]) were obtained with VQMMs with codebooks of fixed size $k_2 = 200$ trained with k-Means, and with different values for $k_1$ and different feature selection schemes. The results indicate that the VQMMs' performances are almost un-affected by the value of $k_1$ and by the selection procedures. This is somewhat expected when the selection process yields a fair representation of the feature distributions, and in the experiments we undertook, we observed that even random sampling selection procedures

[10]The evaluation metrics used for autotagging, such as the per-tag F-scores used here, are presented in detail in Section 5.1.

| | | ISMIR04 | CAL500$_{\text{TOP97}}$ |
|---|---|---|---|
| | Meth. | Acc.+(std. dev.) | F-scores+(std. dev.) |
| $k_1 = 25$ | GMMs | 80.91±0.42 | 31.90±0.90 |
| | k-Means | 81.21±1.12 | 33.45±0.73 |
| | random | 81.48±0.49 | 33.01±0.52 |
| $k_1 = 50$ | GMMs | 81.17±1.84 | 33.87±0.91 |
| | k-Means | 81.10±0.36 | 33.50±0.14 |
| | random | 81.10±0.61 | 33.29±0.43 |
| $k_1 = 100$ | GMMs | 81.10±0.64 | 33.06±0.74 |
| | k-Means | 81.10±0.34 | 33.14±0.29 |
| | random | 81.43±0.78 | 33.09±0.36 |

Table 3.1: VQMMs average accuracies for genre classification using the ISMIR04 set and average per-tag F-scores obtained with a binary tag attribution (see Section 5.1) for autotagging using the CAL500 set (only the 97 most frequent tags were considered). The VQMMs used codebooks with a fixed size of $k_2 = 200$; all the codebooks were generated via k-Means clustering, based on a sub set of feature vectors obtained from a selection of $k_1$ vectors per music piece. Different selection approaches were used: random, k-Means, and GMMs (with ten mixtures). Both set were split in two parts, one for training and the other for testing, and the tests were repeated 5 times.

proved, on most occasions, to be sufficient for our purposes.

The second step of the codebook creation process consists in choosing $k_2$ codewords to represent the feature space. The value of $k_2$ and the methods used to obtain the final number of codewords are important in the context of the VQMMs since they control how well the features are characterized. We tested different selection algorithms and different levels of quantization detail by altering the values of $k_2$. Note that in this last case, the value of $k_2$ also has a direct impact on the estimation of the symbol transition matrix and therefore the results need to be read carefully. In other words, the VQMMs performance levels are influenced by the detail of the feature space discrete characterizations in two ways. In one hand, a high level of detail may help capture certain regions of the feature space that are correlated to some specific musical facet, such as, for example, belonging to a certain instrument, genre, etc. On the other hand, a high $k_2$ values also imply that more transition coefficients need to be derived from the same fixed number of training examples, resulting in noisier estimations. This two functioning behaviors work against each other in a sense that the benefits gained by using thorough feature space representations are counteracted by the resulting poor estimations of the transition probabilities.

The tests pertaining to the different methods used for building the final codebook are summarized in Table 3.2. The codebook size was fixed to $k_2 = 200$ and the frame sub set was

obtained by a random selection of $k_1 = 50$ feature vectors per song. The results indicate that there is little benefit in using more complex methods (that potentially yield better representations) compared to simple ones such as k-Means or even random sampling, either in genre classification or in autotagging tasks. This outcome is somewhat unexpected and is one of the counter intuitive behaviors that we mentioned earlier. There are a few factors that may help explain why more informed data spaces representations fare as well as random ones. In the next chapter we will report on several systematic tests we conducted to study if high level musical concepts such as genre can be inferred from low-level feature distributions. We believe this is an important issue to address because many methods rely on probabilistic or discriminative mappings of the feature distributions. Furthermore, the "good" or "bad" codebook representations also reflect how well the feature space is characterized, and controlling the choice of clustering methods and quantization detail can help clarify how far can genre inference methods go based on audio feature distributions.

The tests with different levels of the quantization detail are reported in Table 3.3 (the table also shows results obtained with k-NN classifiers but this will be addressed in the next section). The results refer to experiments we made on genre classification and autotagging problems using three data sets, the first two - the ISMIR04 and the LMD - for genre, and CAL500 for autotagging. We tested several codebooks of different sizes, obtained via k-Means. For the genre classification, the VQMMs obtain better results with relatively large codebooks sizes ($k_2 \geq 200$), while for autotagging, the performance of the VQMMs is higher for relatively small values of $k_2$, and decreases for large codebooks. This contrasting behavior is in part explained by the conceptual and evaluation differences between the two tasks. Genre is a single label problem, while in autotagging there are several labels (tags) that are not mutually exclusive. The performances are based on average per tag F-scores which can only transmit a general measure over the whole 97 tags involved in this particular data set. Nevertheless, this does fully explain why small codebook sizes are better suited for autotagging tasks. From our experiments, the results indicate that there is a wide variability between labels in terms of ideal codebook sizes, and at the same time more informed feature space representations seem to do just slightly better than random ones. In order to analyze these observations, we must look at several aspects of this complex problem, and in Chapter 5 we address this and other autotagging related issues, while in the next chapter we concentrate on genre.

### Markov Models

In the second stage of the VQMMs the symbols transition probabilities are estimated. This approach can not model all the intricacies present in music signals, nevertheless this simple representation does capture some time-dependent signal characteristics. In Table 3.3 are the

| | ISMIR04 | CAL500$_{\text{TOP97}}$ |
|---|---|---|
| Meth. | Acc.±std.dev | F-scores±std.dev |
| k-Means | 81.10±0.36 | 33.50±0.14 |
| random | 82.41±0.67 | 32.45±0.98 |

Table 3.2: Average accuracies and per-tag F-scores obtained with VQMMs with a codebook of size $k_2 = 200$ generated by random selection or by k-Means clustering. The codebooks were built with random selection of $k_1 = 50$ points per song. Both sets were split in two sub-sets (one for training and the other for testing), and the performances were obtained by averaging the results over 5 tests runs.

| ISMIR04 (accuracies) | | | | | | |
|---|---|---|---|---|---|---|
| Methods | $k_2=25$ | $k_2=50$ | $k_2=100$ | $k_2=200$ | $k_2=400$ | $k_2=800$ |
| k-NN$_{\ell_2}$ | 70.5 | 72.7 | 74.3 | 76.3 | 77.5 | 75.7 |
| VQMMs | 74.2 | 79.1 | 80.9 | 82.3 | 81.9 | 82.9 |
| LMD (accuracies) | | | | | | |
| Methods | $k_2=25$ | $k_2=50$ | $k_2=100$ | $k_2=200$ | $k_2=400$ | $k_2=800$ |
| k-NN$_{\ell_2}$ | 46.4 | 49.9 | 54.9 | 57.0 | 57.4 | 55.8 |
| VQMMs | 60.9 | 66.0 | 70.0 | 70.8 | 68.9 | 68.3 |
| CAL500$_{\text{TOP97}}$ (F-scores) | | | | | | |
| Methods | $k_2=25$ | $k_2=50$ | $k_2=100$ | $k_2=200$ | $k_2=400$ | $k_2=800$ |
| k-NN$_{\ell_2}$ | 28.5 | 28.7 | 29.9 | 30.1 | 30.0 | 29.4 |
| VQMMs | 44.2 | 43.2 | 41.1 | 32.8 | 23.1 | 20.0 |

Table 3.3: Performance scores (in percentage values) on three data sets, for the k-NN (with k=5) and VQMMs classifiers trained with the same discrete sequences. The sequences were derived from quantizations of the feature space with a different number of total codewords $k_2$ (i.e. level of quantization detail). The codebooks were generated via k-Means algorithm with a sub set of features obtained with a random selection of $k_1 = 50$ samples per song. For the genre classification task, codebooks with $k_2 \geq 100$ show approximately constant accuracies for a wide range of values, while for autotagging the smaller codebook sizes show better results. In all three cases, the VQMMs performances are consistently superior to their k-NN counterparts.

results of the VQMMs compared to k-NNs classifiers. One specific characteristic of these tests is that both methods used the same discrete representations of the data. The VQMMs were trained with symbols sequences, while the k-NNs were trained using symbol histograms of the same sequences. The k-NN classifiers can be seen as time-independent version of the VQMMs since they are based solely on the output of the first stage, and so are an intuitive base


Figure 3.4: VQMMs genre dependent, symbol transition matrices for codebooks of different sizes, trained with the ISMIR04 data set. The codebook were created using the k-Means algorithm on a sub set of features obtained from a random selection of $k_1 = 50$ vectors per song. Darker colors represent higher values of the transition probabilities, as indicated by the color bars at the right hand side of each line; the codebook sizes are on the left hand side. For visualization purposes, the symbols in the transition matrices were grouped by genre (i.e. the first symbols have highest conditional class distribution $p(\Gamma|s)$ for the genre Classic, then for Electronic and so fourth), and within each genre ordered by decreasing mutual information (this sorting scheme was used in the experiments of Section 4.2.2 where the procedure is explained in detail). The value of $k_2$ has a direct impact on the transitions matrices: small values result in more densely packed matrices while high value in sparse ones. From the figure it is noticeable that different genres have different transitions patterns: Electronic, PopRock and World have more densely packed matrices compared to other genres. Also note the strong correlations between patterns of the same genre matrices for varying codebook sizes. For visualization purposes, all the transition matrices have been represented with images of equal sizes (note that in order to preserve the true sizes, the last matrices - with $k_2 = 800$ - should be $256$ times bigger that the ones with $k_2 = 50$).

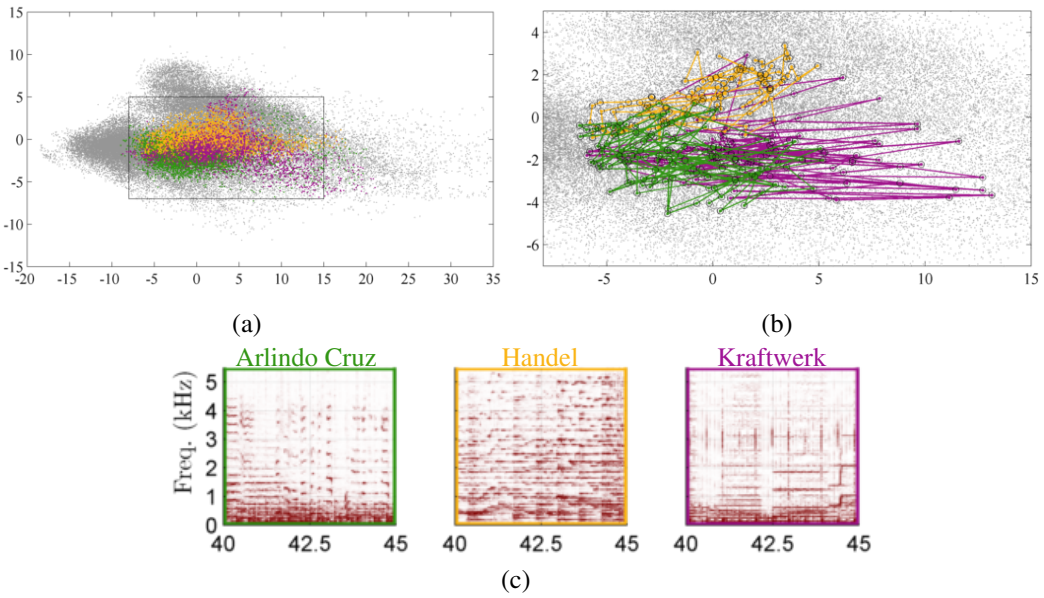



Figure 3.5: The objective of this figure is to illustrate, through a simple example, the temporal variations present in different music signals. For that, we come back to the three songs the previously used in Section 2.2.4 (see Figure 2.3 for the spectrogram representations). Figure 3.5(a) shows the feature vectors of the 3 songs projected in the sub space of the first 2 principal components of the ISMIR04 data set, whose points are also shown in gray. The song excerpts are 3 minutes long, and the corresponding features (about 4 thousand) span a considerable part of the regions occupied by all the songs in the ISMIR04 set. Figure 3.5(b) shows as 5 second excerpt of the three songs, with the feature vectors connected according to their time evolution; the region depicted corresponds to the rectangle in Figure 3.5(a) ; the spectrograms of the 5 second audio clips are in Figure 3.5(c). The time evolution of the three signals occurs in distinct ways. Consecutive features in the Handel signal are not far from each other while the Kravftwerk features jump wildly between consecutive instances; the Arlindo Cruz feature evolution are more erratic than the Handel ones but less than the Kraftwerk features (note that this was also observed for the remaining parts of the signals, but due to the signal lengths, the transitions visualization would have become obscured). The three signals can be classified as belonging to the World, Classical, and Electronic genres, using the ISMIR04 label terminology. We chose these songs because they reflect, in loose terms, the transition behavior typically found in songs belonging to these genres in the ISMIR04 data set. This observation comes from the empirical knowledge we acquired through experimentation, and its intent is to show that different sonorities present in music signals are also reflected at a feature transition level. This is also noticeable in Figure 3.4, where the Electronic and World class have more evenly filled transition matrices than other genres counterparts such as Classical.

of comparison to use. For the ISMIR04 test set, the results show that VQMMs consistently obtain higher accuracy scores, and similar results were obtained on the LMD data set. This is an indication that some class dependent information is present in the symbols transitions. This can also be observed in Figure 3.4 where are depicted the genre transitions matrices obtained with the ISMIR04 training set. To facilitate comparisons, the symbols were grouped by class and ordered by decreasing mutual information between symbols and genres, so the first lines of the transition matrices are from symbols associated with the genre Classical, then with the genre Electronic, and so fourth. This ordering scheme was derived from the informational theoretical analysis that we did on codebook representations (see Section 4.2). The figure also shows how the probabilities populate the transition matrices for different codebook sizes, and it is visible that for smaller codebooks the matrices are more densely packed, while large codebooks result in sparse transition matrices. It is also noticeable the different transition patterns in each genre, although for $k_2 = 50$ they start to get blurred. Genres such as Electronic RockPop and World have more evenly filled transitions which is a sign that tracks from these genre occupy large regions of the feature space, while the other genres, Classic, JazzBlues and MetalPunk seem to be more confined to restricted areas, judging by the gaps in the transition matrices. Figure 3.5 is an example that songs with different sonorities can have very distinct characteristics at a feature transition level (see explanation in the figure legend). In order to have a better understanding of the genre transition patterns, we need to look more carefully at the codewords, how they populate the feature space, and how the signals dynamics are reflected in that quantized space. A genre dependent analysis of the feature space is done in the next chapter, and then it will become clearer how these genre patterns are created. Nevertheless, these examples show that there is some information present in the feature transitions, and in our approach, we capture it through a Markov model. The model extracts only second order information, but this is a simple and effective way of incorporating some of the signal dynamics. In Section 3.4.4 we report on tests we conducted using other temporal models.

### 3.4.2 Genre Classification and Autotagging

The objective of this section is to resume the VQMMs performances on genre classification and autotagging tasks, and position them amongst other methods found in literature.

The accuracies with the VQMMs for the genre classification task are presented in Table 3.4, along with other published results. The tests were conducted on three publicly available data sets, commonly used by the MIR community to evaluate genre classification tasks: the ISMIR04, the GTZAN, and the LMD data sets. Note that this is by not means an exhaustive listing

of all the works [11] - the ones presented in the table were selected because they are some of most representative methods and/or because they showed top performances on this tasks. The first two lines of the table are the results obtained with our method. In regard to the ISMIR04 set, the VQMMs performances are above or on par with most of the methods. The two exceptions are the scores obtained in [Panagakis et al., 2009] and by [Pohle et al., 2009], which were clearly above all others. Panagakis et. al. use a novel technique based on non-negative tensor factorization and sparse representation of auditory features (note that a related technique is used by [Holzapfel and Stylianou, 2007], where GMM models are trained on non-negative matrix factorization based features [Paatero and Tapper, 1994]). However, it seems that the results obtained by Panagakis et. al. arise from a flaw in the experiments, at least according to [Sturm, 2013c][12]. Sturm obtained accuracies around 80% with the same method on the GTZAN compared to 92% reported in [Panagakis et al., 2009]. This is also an indication that the performances reported on the ISMIR04 may also be over-optimistic. The second method that clearly out-performed ours is the one by [Pohle et al., 2009]. This is one of the cases previously mentioned in Section 2.3.3, in which exceedingly simple classification methods achieve competitive performances (in this particular case, the authors used a 1-NN classifier). The strength of the approach is in the features they used - block-level features known as onset patterns - and not at an algorithmic level. Compared to the rest of the methods the VQMMs achieved good performances.

However, this was not the case for the GTZAN set. In general, VQMMs accuracies are poor compared to other reported values. Note that this particular set, although is one of the most reference genre sets in literature, it also has some problems that affect classification performances. As documented in [Sturm, 2012c, 2013b], the problems include duplications (7.2% excerpts come from the same recordings), mislabeling (10.6% of the excerpts), and some tracks are degraded by noticeable distortions. Additionally, in what concerns the VQMMs, the GTZAN has a particular characteristic, not found on the other music sets that significantly affects its performance: each song excerpt is only thirty seconds long. The tests we conducted on song duration and the impact it can have on the VQMMs mode of operation are reported in Appendix B, where the experiment consisted in measuring the VQMMs performances in the LMD set, as the songs durations were reduced to a few tens of seconds. We observed that the performance levels of the VQMMs start to drop for song durations below approximately one minute. In order for our method to work properly, the songs have to be long enough for a reasonable estimation of the transition probabilities, which we empirically found to be around a couple thousand samples[13].

[11] A survey on genre classification by Bob Strum [Sturm, 2012b] referenced a total of 435 papers, with the GTZAN data set appearing in 23% and the ISMIR04 in 17% of them.

[12] From a personal communication between Bob Sturm and Yannis Panagakis.

[13] This confirms the results reported in the Appendix B; with the low-level features settings used, two thousand samples corresponds approximately to a minute and a half of audio.

The third test set used in our genre classification experiments is the LMD. This set has the particularity of all the genres being Latin music genres with a strong similarity between themselves in regard to instrumentation, rhythmic and harmonic contents, which makes genre discrimination a challenging task with this set. Table 3.4 shows two works, [Costa et al., 2012, Lidy et al., 2010], that obtained noticeably higher performances compared to the VQMMs. Note that the tests pertain to both the original set [Silla et al., 2008] (marked with "(FULL)"), and an artist filtered sub-set of 900 songs - which increases the difficulty of the classification task. We find these two works interesting because both use novel sets of features, which work well for LMD case but do not show significant performances gains for the ISMIR04 set, compared to ours and other methods. This may mean that the proposed audio descriptors are more tailored to capture the specificities of non-Western music (see discussion on the subject in [Lidy et al., 2010]). In [Lidy et al., 2010] a wide range of audio features is used, many biased towards rhythmic contents such as Inter-Onset Interval Histogram Coefficients [Gouyon, 2005b], rhythm patterns and histograms, and one pertaining to timbral contents namely the Statistical Spectrum Descriptors computed on 24 Bark-scale bands. The authors remarked that "rhythm and timbre play a major role in discriminating Latin music genres", and the best results were obtained when these features were combined in a hybrid manner. In [Costa et al., 2012] a different type of features is extracted from spectrogram images. These are based on Local Binary Patterns (image processing technique used for texture classification [Ojala et al., 2002]) combined with a zoning strategy (sub-divisions in frequency bands) where different classifiers are used in each band. Once again the results showed high performances for the LMD set, but not for the ISMIR04. The VQMMs performances are substantially lower than the two described methods: more that 10% points compare to [Costa et al., 2012] (71.76% vs 82.33%), and about 5% less than [Lidy et al., 2010]. Note, however, that these are fairly recent and top of the line results; for the MIREX 2009 contest [14], which used the original LMD, in all of the 33 participants, the best accuracy was 74.66% [Cao and Li, 2009], significantly below the 83.32% accuracy we obtained.

In autotagging tasks, it is common for authors to use different testing methodologies which can make comparisons difficult. For instance in autotagging it is commonly divided into two sub-tasks: annotation and retrieval (see Section 5.1 for an overview of these sub-tasks and for a description of the performance metrics commonly used in autotagging). Furthermore in annotation, some authors base their tests on a ranking scheme while others on a binary tag attribution scheme - ranking, however is the overwhelming choice. Table 3.5 summarizes the performances obtained with the VQMMs on the CAL500 set. We tried to replicate most of the conditions reported by other authors, in ranking evaluations with both annotation and retrieval tests. We stress that some of the referenced results are not directly comparable between themselves due

[14] http://www.music-ir.org/mirex/wiki/2009:Audio_Genre_Classification_(Latin_Set)_Results

| | Data Sets | | | | | |
|---|---|---|---|---|---|---|
| | ISMIR04 | | GTZAN | | LMD | |
| VQMMs | 83.4 | ½ | 65.7 | 3-F | 71.76 | 3-F |
| | 81.6 | 10-F | 66.1 | 10-F | 83.32 (FULL) | 10-F |
| [Costa et al., 2012] | 80.65 | ½ | | | 82.33 | 3-F |
| [Flexer et al., 2005] | 78.19 | ½ | | | | |
| [Holzapfel and Stylianou, 2007] | 74.1 | 5-F | 72.9 | 5-F | | |
| [Lidy et al., 2007] | 81.4 | ½ | 76.8 | 10-F | | |
| [Lidy et al., 2010] | 80.37 | 10-F | | | 88.06 (FULL) | 10-F |
| [Mayer et al., 2010] | 81.3 | 10-F | 72.6 | 10-F | | |
| [Panagakis et al., 2009] | 94.38 | ½ | 92.4 | 10-F | | |
| [Pampalk et al., 2005] | 84 | ½ | | | | |
| [Pohle et al., 2009] | 90.4 | 10-F | | | | |
| [Tzanetakis and Cook, 2002] | | | 61.0 | 10-F | | |

Table 3.4: **Genre Classification Task:** VQMM accuracies compared to other results reported in literature, on three publicly available data sets: ISMIR04, LMD, GTZAN. The details of these sets are in the Appendix C. Note that LMD refers to an artist filtered version sub set of the original Latin Music Database, (marked (FULL)). The "k-F" means a k-fold cross validation methodology with the results averaged over all the folds. For the ISMIR04 data set, the "½" means that half of the data was used for training and the other for testing (the same division as in the original contest). Comments are in the text.

to different test settings. In particular, the number of tags used in the experiments ($3^{\text{rd}}$ column) can have a significant impact on performances, and comparisons should only be made between algorithms using the same tag sub-sets. Also note that in annotation, even trivial classification schemes can obtain relatively high precision or recall scores, but only truly valid classifiers are able to obtain high values in both metrics - thus the F-score is a better indicator since it is the harmonic mean between precision and recall. Table 3.5 reports performances of some of the most popular autotagging algorithms along with other methods that have recently emerged. [Turnbull et al., 2008b] is probably the most referenced paper in this area. The authors use a Gaussian mixture model for each tag, also common in audio-based genre classification. Two other popular works are [Bertin-Mahieux et al., 2008, Hoffman et al., 2009]. [Bertin-Mahieux et al., 2008] proposes an online version of AdaBoost with rejection sampling to be able to deal with large data sets. In this work decision stumps (a decision tree with two leaves) are used as weak learners. In Hoffman et al. [2009], a probabilistic model called Codeword Bernoulli Average (CBA) is used. CBA is based on a vector quantized codebook representation of the

| CAL500 data set – RANKING (TOP 10) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | Annotation | | | Retrieval | | |
| | | | Pr. | Re. | F-s. | MAP | AROC | Eval. |
| | VQMMs | FULL | 35.8 | 13.5 | 19.5 | 36.6 | 68.0 | 10-F |
| | | TOP97 | 44.3 | 19.5 | 27.1 | 45.0 | 68.9 | 10-F |
| [Bertin-Mahieux et al., 2008] | FilterBoost | FULL | 28.1 | 13.1 | 17.9 | 30.5 | 67.8 | 10-F |
| [Coviello et al., 2011] | HEM-DTM | TOP78 | 47 | 25 | 30 | 48 | 69 | 5-F |
| in [Ellis et al., 2013] | HEM-DTM | TOP97 | 44.6 | 21.7 | 26.4 | 44.6 | 70.8 | 5-F |
| [Ellis et al., 2013] | BoS | TOP97 | 43.3 | 26.3 | 26.7 | 48.9 | 74.4 | 5-F |
| [Hoffman et al., 2009] | CBA | FULL | 28.6 | 16.2 | 20.7 | 39.0 | 75.9 | 10-F |
| in [Miotto et al., 2010] | CBA | TOP 97 | 36.1 | 21.2 | 26.7 | 42.5 | 69.1 | 5-F |
| [Miotto et al., 2010] | D-M | TOP 97 | 44.1 | 23.2 | 30.3 | 44.3 | 69.7 | 5-F |
| [Turnbull et al., 2008b] | GMM | FULL | 26.5 | 15.8 | 19.8 | 39.0 | 71.0 | 10-F |
| in [Miotto et al., 2010] | GMM | TOP 97 | 40.5 | 20.2 | 26.9 | 43.3 | 69.8 | 5-F |

Table 3.5: **Autotagging Task**: Results are grouped by algorithm. Note that some authors have tested the algorithms of other authors. Miotto et. al, report results on GMMs with the code provided by Turnbull et. al, and on CBA with code from [Hoffman et al., 2009]. [Ellis et al., 2013] uses the algorithm in [Coviello et al., 2011]. The 2$^{nd}$ column refers to the methods in the referenced publications of the 1$^{st}$ column (see text for a brief description of each method).

songs, and defines a collection of binary random variables that determine whether or not a tag is present in a song. The CBA model parameters are estimated with the standard EM algorithm.

Some of the more recent methods have focused on developing approaches that can take into account the temporal dynamics present in music. In [Coviello et al., 2011] this is done with dynamic texture mixtures (DTM), a model used in [Barrington et al., 2010] for audio segmentation. Dynamic texture models treats a sequence of feature vectors as a sample from a linear dynamical system (a generative model similar to HMMs), and in [Barrington et al., 2010] different segments of the audio are modeled with different dynamic textures. In [Coviello et al., 2011] this idea is adapted for autotagging, where first song-level DTMs are build, and then a hierarchical expectation maximization algorithm is used to select the DTMs that better represent a given tag (their algorithm is referred to as HEM-DTM). In [Ellis et al., 2013], the authors address some limitations of the HEM-DTM algorithm, like the choice of time scales and the number of parameters necessary for DTM estimation, and propose a new method called the Bag of Systems (BoS) representation (note that all the authors in [Coviello et al., 2011] appear in [Ellis et al., 2013]). This approach uses a codebook representation of the audio, with generative models (DTM and GMM) instead of codewords. The codebook is trained in an unsupervised fashion, and several learning strategies are used to build codeword-models –

based on audio fragments (a single model is used for each segment), based on individual songs (small number of model for each song), or based on the entire audio collection (model selection is done via EM or hierarchical-EM algorithms). Each song is characterize by an histogram of the codeword-models. Once the songs are represented with histograms, tag attributions are done via codeword Bernoulli averages or with logistic regression.

Overall, the VQMM performances were satisfactory, and other tests also showed competitive performances when tags were differentiated by categories such as emotional content, instrumentation, and other general facets as originally proposed in [Turnbull et al., 2008b] (see Appendix B.2). We also conducted tests on another autotagging data set, the MTT and an artist filtered version of this set, the MAGTAG5k. Results with this sets are presented in Chapter 5, along with the ones for the CAL500 (see also Appendix B).

### 3.4.3 Artist Identification and Playlist Generation

#### Artist Identification Task

One of our objective with this task is to assess the performance of our method when models are based on songs from the same artist. This task is closely related to genre classification, but in this case, a model is build for each artist. We evaluated our method using two data sets: JAZZ17 and ARTIST20 (see Appendix C), and the results were reported in [Langlois and Marques, 2009a]. These were conducted in the early part of our research process that led to this thesis, and in this sense, it is important to contextualize them in time. Then we did not have a clear notion of the role of the $k_1$ and of the $k_2$ parameters in our VQMM algorithm, as we have now. Thus, we tried two different values for $k_1$, 100 and 200, which were unnecessary (even a smaller value for $k_1$ would suffice), and tested also the same values for $k_2$, which were more or less appropriate for the genre classification with the ISMIR04 data set, but in the light of what we know now, other values may have been better.

JAZZ17: This is a Jazz music data set and is based on authors' collection. It contains 543 tracks from 17 artists. With the JAZZ17 data set our interest was to see if our system is able to distinguish songs that belong to a single genre. Because of the reduced number of albums per artist, 50% of each artist's songs were randomly selected for training while the other half was used for test. For two sets of parameter values ($k_1$ and $k_2$) the training and test was repeated ten times. Table 3.6 shows the average accuracy and the standard deviation observed on the test sets. Table 3.7 contains a confusion matrix obtained with one of the test runs. As can be seen in the confusion matrix, the majority of misclassifications occur between songs with strong vocals. For instance, 3 songs from Diana Krall are classified as Norah Jones's, 5 songs from Sarah

Artist Identification with JAZZ17 data set

| $k_1$ | $k_2$ | mean Acc. | std. dev. |
|---|---|---|---|
| 100 | 100 | 73.49% | 1.75 |
| 200 | 200 | 74.25% | 2.25 |

Table 3.6: Average accuracy and corresponding standard deviation obtained with ten test runs on an artist identification task with the JAZZ17 data set.

Vaughan as Anita O'Day's, 4 songs from Ella Fizgerald are recognized as Sarah Vaughan's, and 3 songs from Anita O'Day as Ella Fitzgerald's songs. Nevertheless, other misclassifications do not have such an obvious explanation. We may conjecture that some errors are due to similar orchestration arrangements, like the case of 3 of Frank Sinatra's songs being attributed to Ella Fitzgerald. Others may be due to other specificities, such as the unusual high timbre voice of Chet Baker, and thus the confusion between Chet Baker's songs and other women interpreters (or vice-versa) such as Sarah Vaughan, Anita O'Day or Nina Simone. Nevertheless, there are some cases that elude plausible explanations - this was also noticed in a playlist generation task that we conducted with this data set. The presence of these somewhat counter intuitive errors has also been reported by many authors, namely in playlist generation experiments [Aucouturier and Pachet, 2004, 2007, Berenzweig, 2007, Pampalk et al., 2005]. The authors noticed that the unwanted "intruder" songs have also the characteristic of appearing as nearest neighbors of a large number of other songs, and are referred as "hubs". Hubs are not exclusive to audio music similarity and have been observed in many areas such as image processing [Hicklin et al., 2005], text mining [Radovanović et al., 2010], and biometrics [Yager and Dunstone, 2010], but it is not until recently that the MIR community is discovering the full implications of the hub phenomenon [Flexer et al., 2010, 2012, Karydis et al., 2010, Radovanović et al., 2010, Schnitzer et al., 2011, 2012]. In fact, hubs may be one of the main causes for the errors here mentioned, but also the culprit of many pathological behaviors found in MIR. We report our research on this subject in Section 6.2.

ARTIST20: This data set contains 1412 tracks from 20 artists, and each artist is represented by 6 albums. The results obtained with the ARTIST20 dataset are shown in Table 3.8 . We used two different setups:

1. 50% of an artist's songs are randomly selected and used for training while the other half is used for testing.

2. For each artist an album is randomly selected for test and the other five albums are used for training - this was the strategy suggested in [Ellis, 2007].

| | AD | AT | BH | CB | CJ | DE | DK | EF | FS | JC | LY | MD | NJ | NS | OP | SV | TM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| AD | **15** | 0 | 0 | **1** | 0 | 0 | 0 | **3** | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| AT | 0 | **18** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 |
| BH | 0 | 0 | **17** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| CB | 0 | 0 | 0 | **20** | **1** | 0 | 0 | 0 | **2** | 0 | 0 | **1** | 0 | **1** | 0 | **2** | 0 |
| CJ | 0 | 0 | 0 | **1** | **2** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| DE | **1** | 0 | 0 | 0 | 0 | **5** | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 |
| DK | 0 | 0 | 0 | 0 | 0 | 0 | **14** | 0 | 0 | 0 | 0 | 0 | **3** | 0 | 0 | 0 | 0 |
| EF | **2** | 0 | 0 | 0 | 0 | 0 | 0 | **9** | 0 | 0 | **2** | 0 | 0 | 0 | 0 | **4** | 0 |
| FS | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **3** | **20** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| JC | 0 | 0 | 0 | **1** | 0 | 0 | 0 | **1** | 0 | **2** | 0 | **2** | 0 | 0 | 0 | 0 | 0 |
| LY | **1** | 0 | **1** | **4** | 0 | 0 | 0 | **1** | 0 | 0 | **11** | 0 | 0 | 0 | **2** | **1** | 0 |
| MD | 0 | 0 | 0 | **1** | **1** | 0 | 0 | 0 | **1** | 0 | 0 | **14** | 0 | 0 | 0 | **2** | 0 |
| NJ | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | **16** | 0 | 0 | 0 | 0 |
| NS | 0 | 0 | 0 | **1** | 0 | 0 | 0 | **1** | 0 | 0 | **1** | 0 | 0 | **9** | 0 | **1** | 0 |
| OP | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | **1** | **11** | **1** | 0 |
| SV | **5** | 0 | 0 | **1** | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | **1** | **9** | 0 |
| TM | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **9** |

Artists names abbreviations:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **AD** | Anita O'Day | **CJ** | Clifford Jordan | **FS** | Frank Sinatra | **NJ** | Norah Jones | **TM** | Telonious Monk |
| **AT** | Art Tatum | **DE** | Duke Ellington | **JC** | John Coltrane | **NS** | Nina Simone | | |
| **BH** | Billie Holliday | **DK** | Diana Krall | **LY** | Lester Young | **OP** | Oscar Peterson | | |
| **CB** | Chet Baker | **EF** | Ella Fitzgerald | **MD** | Miles Davis | **SV** | Sarah Vaughan | | |

Table 3.7: Confusion matrix obtained with the JAZZ17, in one of the ten test runs we conducted with this data set. Incidentally, these are the results obtained with the first test run and are not the best results.

As we can see, choosing the training and testing sets randomly leads to significantly better results than keeping one album for test. This is due to the "album effect" [Kim et al., 2006, Pampalk, 2006b].

These results show that despite the name of the task, it is clear that, at least in our case, the problem solved is not the Artist Identification problem. Indeed, our method aims at classifying songs using models based on timbre. Different albums of the same artist may have very different styles, use different kinds of instruments, sound effects and recording conditions. If a sample of each artist's style is found in the training set it is more likely that the classifier will recognize a song with similar timbre. If every songs of an album are in the test set, then the accuracy will depend on how close are the mixtures of timbre of this album from those of the training set. This is confirmed by the standard deviation observed with both approaches. When trying to avoid the "album effect" we observe a large variation of performance due to the variation of the datasets. In one of our tests we reached an accuracy of 62.3% but this was due to a favorable combination of albums in the training and test sets.

Notwithstanding these observations the results are interesting. In particular with the JAZZ17 data set, we can see that the timbre-based classification is quite accurate even with music pieces that belong to the same genre.

Artist Identification with ARTIST20 data set

| | | Random Selection | | Album Filtered | |
|---|---|---|---|---|---|
| $k_1$ | $k_2$ | mean Acc. | std. dev. | mean Acc. | std. dev. |
| 100 | 100 | 57.40% | 0.74 | 45.28% | 7.27 |
| 200 | 200 | 59.14% | 1.49 | 48.98% | 7.96 |
| Dan Ellis [Ellis, 2007] - MFCCs | | | | 54% | |
| Dan Ellis [Ellis, 2007] - MFCCs + Chroma | | | | 57% | |

Table 3.8: Average accuracy and corresponding standard deviation obtained with ten test runs on an artist identification task with the ARTIST20 data set. In the "Random Selection" case, 50% of each artist songs were randomly selected for training, while the other half was chosen for testing. In the "Album Filtered" results, for each artist a randomly selected album was used for test, while the other albums were used for training. This was the strategy used in [Ellis, 2007], and we include in the last two lines of this table the results reported therein.

### Playlist Generation Task

In [Langlois and Marques, 2009a] we reported the results obtained with our method in an automatic playlist generation problem, which unlike genre classification, autotagging, artist identification, or mood estimation, is a task directly related to music similarity estimation - albeit with the inherent subjectivity in the evaluation process. Table 3.9 presents a more complete version of these results. The task consisted in generating playlists with the top 20 most similar songs, from four well-known Jazz songs as seeds, based on the JAZZ17 data set. We built a transition model with every seed song and picked the songs that had the smallest divergence given by Equation 3.13. The objective of these experiments is to gain some insight on how our method performs on a similarity estimation task, namely to see if the choices brought up by the VQMMs make some sense, at least in a controlled environment. The reader should bear in mind that these are not systematic tests, as some reported in this and other chapters, such as the majority of the experiments in genre classification or autotagging; here we were less strict with our testing methodology. We did not perform album or artist filtering in the data set, and our analysis of the results, which is presented next, has unavoidably some part of subjectivity.

- **Thelonious Monk, Smoke Gets in Your Eyes:** This seed song is characterized by a salient piano accompanied by a saxophone. The most similar songs are from the same artist and same album. Then we find songs from Anita O'Day which can be described as voice accompanied with piano. Following are two pieces from the Jazz pianist Oscar Peterson. The two songs from Sarah Vaughan are less expected in this list. Finally Art Tatum has obviously his place in this list.

- **Norah Jones, Come Away With Me:** This playlist is composed of songs where vocals are the dominant timbre. It is interesting to note that with one exception, the artists that appear in this list are all women. The timbre of Chet Baker's voice is rather high and sometimes may be confused with a women's voice. However, John Coltrane's "Village Blues" appears as an intruder in this list.

- **Sarah Vaughan, Lover Man:** This playlist is dominated by songs where vocals are the dominant timbre. As with the previous playlist, Chet Baker high timbre also explains the artist presence in this list.

- **John Coltrane, Blue Train:** The song "Blue Train" by John Coltrane is characterized by Saxophone solos and trumpet. Excluding the songs from the same album, the songs found in the playlist are performed by Miles Davis, Dizzy Gillespie whose trumpets are assimilated with saxophone and Ella Fitzgerald and Frank Sinatra who are accompanied by a strong set of copper instruments.

### 3.4.4 VQMMs Alternative Second Stage Methods

We report in this section the results of other methods we tested for the second stage of our algorithm. Like the VQMMs, these methods also have as inputs discrete symbols obtained from the quantization of the low-level features, but the temporal dynamics are not modeled (at least exclusively) with bi-gram transition probabilities. The objective of these experiments was to determine if temporal correlations, besides the second order, or if the use of HMMs to model the symbols dynamics was advantageous. The experiments we conducted consisted in modeling the discrete symbol sequence with uni-gram in conjunction with bi-gram transition probabilities, with tri-grams transition probabilities, with n-grams obtained using Lempel-Ziv parsing, and with hidden Markov models. Next, we concentrate on reporting the tests and the results obtained with each modeling technique on a genre classification task with the ISMIR04 data set; the functioning details of the methods were previously described in Sections 3.2.1 and 3.3 and therefore we will only mention them superficially. We would like to point out that these experiments pertain to the early part of our research, and due to poor performances or due to implementation drawbacks where abandoned. These issues will also be addressed in the next paragraphs.

Table 3.9: Playlists generated from four different seed songs (songs "0" - first song on the list), in the JAZZ17 data set.

| | PLAYLIST 1 | | PLAYLIST 2 | | PLAYLIST 3 | | PLAYLIST 4 | |
|---|---|---|---|---|---|---|---|---|
| | Artist | Song | Artist | Song | Artist | Song | Artist | Song |
| 0 | **TM** [a] | Smoke Gets in Your Eyes | **SV** | Lover Man | **NJ** | Come Away with Me | **JC** | Blue Train |
| 1 | **TM** | Off Minor | **SV** | Prelude to a Kiss | **NJ** | Come Away with Me [b] | **JC** | Moment's Notice |
| 2 | **TM** | Evidence | **SV** | I'm Glad there Is You in This | **DK** | Cry Me a River | **JC** | Lazy Bird |
| 3 | **TM** | Eronel | **SV** | I'll Never Smile Again | **NJ** | Feelin' the Same Way | **JC** | Locomotion |
| 4 | **TM** | Reflections | **SV** | It Might as Well Be Spring | **DK** | Guess I'll Hang My Tears Out To Dry | **EF** | It Ain't Necessarily So |
| 5 | **TM** | Well You Needn't | **AD** | Boogie Blues | **JC** | Village Blues | **EF** | I Got Plenty o' Nuttin' |
| 6 | **TM** | We See | **NJ** | Come Away With Me | **DK** | Every Time We Say Goodbye | **FS** | I've Got You Under My Skin |
| 7 | **TM** | Round About Midnight | **SV** | Jim | **DK** | The Night we Called it a Day | **MD** | So What |
| 8 | **AD** | Lonesome Road | **AD** | Teo for Two | **NJ** | Don't Know Why | **MD** | Freddie Freeloader |
| 9 | **AD** | Rosetta | **NJ** | Come Away With Me | **DK** | I Remember You | **EF** | A Woman is a Sometime Thing |
| 10 | **OP** | Cotton Tail | **CB** | My Funny Valentine | **DK** | Walk On By | **SV** | Jim |
| 11 | **AD** | Them there eyes | **SV** | Embraceable You | **DK** | I've Grown Accustomed To Your Face | **FS** | Pennies From Heaven |
| 12 | **OP** | Take the A Train | **SV** | September Song | **SV** | Prelude to a Kiss | **DG** | November Afternoon |
| 13 | **SV** | The Nearness of You | **SV** | Over the Rainbow | **DK** | Too Marvelous For Words | **MD** | Bess oh Where's my Bess |
| 14 | **AD** | Once there Lived a Fool | **SV** | April in Paris | **DK** | The Boy from Ipanema | **FS** | The Way You Look Tonight |
| 15 | **OP** | In a Mellow Tone | **SV** | Shulie a Bop | **NJ** | Lonestar | **EF** | There's a Boat Dat's Leavin' Soon for NY |
| 16 | **SV** | East of the Sun | **DK** | Guess I'll Hang My Tears | **CB** | My Funny Valentine | **EF** | Dream A Little Dream of Me |
| 17 | **TM** | Hackensack | **EF** | Im Just Lucky So And So | **DK** | The Look of Love | **JC** | I'm Old Fashioned |
| 18 | **AD** | Ain't Misbehavin | **CB** | Time After Time | **NJ** | Lonestar [b] | **EF** | Basin' Street Blues |
| 19 | **AT** | Yesterdays | **EF** | How High The Moon | **DK** | Este Seu Olhar | **MD** | All Blues |

[a] Artists' Names Abbreviations

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **AD** | Anita O'Day | **DG** | Dizzy Gillespie | **FS** | Frank Sinatra | **NJ** | Norah Jones | **TM** | Thelonious Monk |
| **AT** | Art Tatum | **DK** | Diana Krall | **JC** | John Coltrane | **OP** | Oscar Peterson | | |
| **CB** | Chet Baker | **EF** | Ella Fitzgerald | **MD** | Miles Davis | **SV** | Sarah Vaughan | | |

[b] different version

**Using uni-grams and bi-grams:** In this approach, the classification of a sequence depends on a linear combination of uni-grams and bi-grams:

$$\mathcal{L}_{1\oplus\gamma}(\mathcal{S}) = \alpha\mathcal{L}_{1\gamma}(\mathcal{S}) + (1-\alpha)\mathcal{L}_{\gamma}(\mathcal{S}) \quad \text{(Equation 3.14)}$$

where $\mathcal{L}_{\gamma}(\mathcal{S}) = \ln(P(s_1|\gamma)) + \sum_{n=2}^{N} \ln(P(s_n|s_{n-1},\gamma))$ (Equation 3.7)

and $\mathcal{L}_{1\gamma}(\mathcal{S}) = \sum_{n=1}^{N} \ln(P(s_n|\gamma))$ (Equation 3.15)

where $\alpha \in [0,1]$. The results for a codebook size of $k_2 = 200$ are shown in Table 3.10.

When $\alpha = 1$, only uni-grams are taken into account whereas $\alpha = 0$ reverts to the case where only bi-grams are considered. As we can see in this table, the introduction of uni-grams in the classification process does not seem to be beneficial. This is confirmed by the analysis of the feature distributions in terms of genre and their discriminative capacities, which will be addressed in detail in the next chapter. Nevertheless, the examples already given in this chapter also indicate that uni-gram probabilities are not ideal for genre inference due to the high levels of feature superposition.

| $\alpha$ | 1 | 0.5 | 0.0 |
|---|---|---|---|
| accuracy | 71.88% | 77.64% | 81.89% |

Table 3.10: Genre classification results with ISMIR04 test set using a weighted sum of uni-grams and bi-gram based models.

**Tri-grams:** The most obvious evolution from the Markov models is to use time dependencies higher than the second order. In the tri-gram model, the current symbol is dependent on the two previous ones:

$$\begin{aligned}\mathcal{L}_{3\gamma}(\mathcal{S}) &= \ln(P(\mathcal{S}|\gamma)) \\ &= \ln(P(s_1|\gamma)) + \ln(P(s_2|s_1,\gamma)) + \sum_{n=3}^{N} \ln(P(s_n|s_{n-1},s_{n-2},\gamma))\end{aligned} \quad \text{(Equation 3.16)}$$

This has an serious drawback: as we include higher order dependencies, the more transition probabilities we have to estimate with the same amount of data. Additionally, from a practical perspective, several implementation and operational problems also arise in this situation. The high number of transitions requires - $k_2^3$ for each model - requires the use of programming strategies such as sparse matrix allocation, writing on disk the transition matrices, and other programming tricks to circumvent the memory demands associated with this methods, which in turn, significantly decreases the algorithm functioning speed. In Table 3.11 are the performances obtained with the tri-gram models. We tested different codebook sizes, but the results showed that there is no significant performance difference between the bi-gram and tri-gram

| | $k_2=25$ | $k_2=50$ | $k_2=100$ | $k_2=200$ | $k_2=300$ |
|---|---|---|---|---|---|
| accuracy (%) | 78.6±1.3 | 79.8±1.6 | 81.1±1.5 | 81.4±0.8 | 81.5±1.0 |

Table 3.11: Genre Classification with ISMIR04 test set using tri-grams for various codebook sizes. The codebooks were generated via k-Means clustering, with a data set made from a random selection of $k_1=50$ vectors per song. Results were obtained via 5-fold cross validation, and scores were averaged across the 5 folds.

transition models. Furthermore, the use of tri-gram models becomes prohibitive when a large number of models has to be estimated (e.g. in Autotagging scenarios) or when the number of symbols is moderately high (e.g for $k_2 \geq 500$). This lead us to abandon the tri-gram model in favor the second order Markov transition model.

### LZW Parsing

Another way to model a time-varying process is to discover emerging multi-grams patterns in the audio feature sequences. For that end, we used an adaption of the Lempel-Ziv compressing algorithm (based on the work of [Li and Sleep, 2005]) to create a global dictionary composed of variable length n-grams, by parsing all the training sequences independently of the class membership. The Lempel-Ziv based method builds a dictionary based on symbol occurrences, and the dictionary grows as new combinations of symbols appear in the training songs; once all the training songs are parsed, the dictionary is frozen. We performed some tests based on a maximum likelihood approach, where we used the class dependent codeword probabilities to assign the test songs a likelihood score. The conditional probability of a codeword, $z_i$, in the dictionary is given by:

$$P(z_i|\gamma) = \frac{\text{n. occurrences of } z_i \in \gamma}{\text{total n. codeword occurrences in } \gamma} \tag{3.26}$$

where $z_i$ is the codeword $i$, $\gamma$ is the class ($\gamma \in \Gamma$). The probabilities $P(z_i|\gamma)$ were estimated on codeword frequencies in the training set. For classification, we counted the number of times the codewords appeared in the test sequence. During testing, the common dictionary is used to parse the songs, but unlike Lempel-Ziv compression schemes, the dictionary is not allowed to grow. The classification is performed by maximum likelihood estimation.

$$\begin{aligned} \hat{\gamma} &= \underset{\forall\gamma\in\Gamma}{\operatorname{argmax}} \left(\mathcal{L}(\mathcal{Z}|\gamma)\right) \\ \mathcal{L}(\mathcal{Z}|\gamma) &\propto \log(p(\gamma)) + \sum_{i=1}^{N} \log\left(p(z_i|\gamma)\right) \end{aligned} \tag{3.27}$$

where $\mathcal{Z} = \{z_1, \ldots, z_N\}$ is this the set of codewords extracted from parsing the test song. In this metric, the underlying assumption is that the order of codeword occurrences is unimportant. This is equivalent to the naïve Bayes or the "bag of frames" classifier with the difference that the short-time dependencies of the audio signal are implicitly modeled by the multi-gram codewords.

The results with this approach are summarized in Table 3.12, but these were somewhat disappointing and showed consistently worst performances that the VQMMs counterparts. We conducted several experiments with n-grams in order to determine the effectiveness of this method, but operational and implementations issues along with substandard performance levels lead us to abandon this line of research. One of the problems associated with this method is the size of the n-gram dictionaries resulting from the Lempel-Ziv codeword selection schemes. If no pre-processing is done, the number of codewords can easily reach several thousands. This happens when a moderately low number of $k_2$ symbols are used to represent the (quantized) feature space; for $k_2$ values above a few hundreds, the number of n-grams rapidly grows to a few tens of thousands (see Table 3.12). In this situation, the estimations of the codeword conditional probabilities become unreliable, and consequently the use of the likelihood-based functional of Equation 3.4.4 is also questionable. In order to reduce the number of codewords, we opted to only include in the n-gram dictionary codewords with a frequency count above a predefined number of hits. This simple strategy allowed us to have a reasonable amount of control over the final size of the dictionary. Nevertheless, this method fares poorly with small dictionaries. One of the factors that can help explain the inferior performances has to do with the method itself. The Lempel-Ziv based codeword selection favors codewords (sequences of a few consecutive symbols) that occur frequently and, in the long run, this converges to a minimum entropy code. Nevertheless, in order for this to happen, the algorithm has to be provided with long enough sequences with salient re-occurring patterns. Based on our results, we believe that our Lempel-Ziv based approach was not efficient enough to extract possible re-occurring patterns present in the discrete symbol sequences derived from the music signals. The difficulty also comes from the fact that, for a given training sequence, the Lempel-Ziv selection scheme can output different dictionaries if different time instances are chosen for the starting point of the algorithm in the sequences. This is due to the transitory behavior in this type of codification procedures, which tends to fade away after the algorithm has run for a sufficient long time. From empirical analysis, we noticed that the n-gram dictionaries can differ significantly from each other, even when dictionary generation process is run with the same parameters and on the same training sequences - the only change being the time offsets. This lack of consistency in the n-gram selection also affects the performance of the method, which we found not to be robust enough for our purposes.

| | | $k_2=50$ | | $k_2=100$ | | $k_2=200$ | | $k_2=400$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | \|Dic.\| | Acc.(%) | \|Dic.\| | Acc.(%) | \|Dic.\| | Acc.(%) | \|Dic.\| | Acc.(%) |
| cdw freq. $\geq$ | 2 | 7080±102 | 67.9±1.0 | 9255±170 | 71.2±1.0 | 11545±142 | 70.2±2.1 | 15012±339 | 68.4±2.3 |
| | 3 | 2342±81 | 64.5±5.0 | 3027±96 | 64.4±4.1 | 3671±139 | 63.8±2.1 | 4222±165 | 56.4±3.1 |
| | 4 | 823±66 | 46.3±5.7 | 1141±56 | 45.2±11.5 | 1503±37 | 42.8±4.9 | 1867±74 | 38.0±9.2 |
| | 5 | 359±15 | 27.8±8.9 | 504±14 | 33.3±11.6 | 735±14 | 32.8±11.9 | 989±18 | 39.3±11.3 |

Table 3.12: Percentage of correctly classified songs on the ISMIR04 test set, for n-gram based classifiers. The dictionaries were obtained by selecting the n-gram codewords with a frequency count above a predefined number of hits (left column). The resulting dictionary lengths and corresponding accuracies are given for discrete sequences obtained with different levels of quantization detail ($k_2$ values). The tests were repeated 5 times, and the values are the mean of the five tests runs, along with plus or minus one standard deviation.

### Hidden Markov Models

A commonly used technique to model time dependencies is the hidden Markov models. We tested the HMMs on the genre classification task with the ISMIR04 data set. The discrete sequences used to train the VQMMs were also used in the HMM's training. We used different models to ascertain the impact of the number of states and the structure of the model on the performance of the classifier. We tested two architectures for the HMMs: the left-right models (Figure 3.1) and fully connected models (Figure 3.2). The models were trained [15] using the Baum-Welch algorithm on sequences belonging to the same genre. For classification, we use the Viterbi algorithm to decode a given sequence (music piece) and calculate the probabilities with the HMM's trained for different genres. The music is then assigned to the genre with the highest probability. We used left-right models with 1, 2 and 3 delays, and a fully connected model. We also tested these models with 10 and 20 hidden states. The results, shown in Table 3.13, indicate that neither the state number nor the type of structure have a significant impact on the performance.

## 3.5 Summary

In this chapter we presented the VQMMs, a method for audio-based similarity-related applications, described its functioning details, summarize the results we obtained with our approach and compare them to other approaches found in literature. We started the theoretical details of clustering techniques that we used to represent the feature space with a discrete set of symbols,

[15] We used the HMM Toolbox by Kevin Murphy [Murphy].
See http://www.cs.ubc.ca/ murphyk/Software/HMM/hmm.html for details.

| HMMs | LR-1 | LR-2 | LR-3 | FC |
|---|---|---|---|---|
| 10 states | 68.3% | 69.3% | 68.7% | 69.1% |
| 20 states | 69.1% | 69.8% | 69.5% | 69.5% |

Table 3.13: Percentage of correctly classified songs on the ISMIR04 test set, for various HMM structures with 10 and 20 hidden states. In the left column is the type of model used: LR-$n$ means a left-right model where each state can transfer to itself and $n$ states right to it, and FC means the fully connected model.

and then introduced Markov process used by the VQMMs to model the second order time dependencies present in the quantized signals (i.e. songs). We also described other methods we implemented and tested, to establish a base of comparison with our method, like the HMMs, SVMs, or the k-NNs classifiers, and studied alternative ways of incorporating other time dependencies besides the second order, in the second stage of the VQMMs. However, the experiments we conducted with other time-dependent models for the second stage of our approach did not prove to be beneficial.

To measure the VQMMs performances in audio-based classification tasks, we undertook several tests on several publicly available sets so we could make direct comparisons to results reported in literature. The core of experiments focused on genre classification and autotagging problems. In terms of performance, the VQMMs were not as good as some recent methods, but were able to obtain higher or comparable results to the bulk of the methods used in these task. One exception was the performances the VQMMs obtained on the GTZAN data set, which were clearly below average. The reason for this behavior has to do with the length of the audio excerpts in the set, which were too short for the VQMMs to be able to function properly - this is one of the principal limitations of our approach. We also obtained sub-optimal performances on an artist identification task, but in the context of this dissertation, this is a secondary problem and we did not further explore this issue. Instead we concentrated our efforts on genre and autotagging, namely on some aspects of this tasks that we find pertinent, and in our perspective have only been superficially addressed in literature. These issues are presented in the next two chapters.

# Chapter 4

# Audio-Based Genre Classification

Our objective in this chapter is to explore a particular aspect of the genre classification problem: the influence of diverse partitions of the data space on the performance of a genre classifier. We want to study how the feature space is populated, and analyze if there is sufficient information to infer genre from the feature distributions. In other words, is there a typical set of low-level features values for a " e.g. Jazz or Classical" frame? We believe this is an important topic to address because many music similarity and genre classification systems, and in particular the bag of frames approaches, rely on probabilistic or discriminative mappings of the feature space. In other words, starting from a data representation space defined by local signal features, it is implicitly assumed that specific regions of this representation space do *globally* capture musical specificities and, in the context of genre classification, can globally represent genre. Therefore, building good genre models traditionally focuses on determining in an automatic fashion which these specific representative regions are.

Throughout our experiments, the data space is fixed and is defined by the same set of features derived from short-term frames of the audio signals. In our codebook-based methods we use a single codebook to quantize all the feature vectors in the data sets, independently of the classes, and prior to building any genre models. In this way, we separate the partitioning of data representation space from the process of creating classification models, which are trained only with discrete symbols, derived from the quantized low-level features vectors of the audio. In our experiments, we tailored the codebook generation process to favor certain characteristics of the data. We explored the inclusion of more or less information about the provenance of the frames used to create the codebook, either by selecting the most representative frames via k-Means or via density models of their distributions, or by random sampling the data instances (see Section 4.1.2). We also explore building codebooks restricting the training frames to a particular genre. We performed systematic tests with the different codebook generation approaches, and monitor the impact on the classification rate. Moreover, we conduct a information theoretic

analysis of the results accompanied by a visual examination of different feature representations.

Our findings indicate there is no apparent benefit in seeking a thorough representation of genres in the low-level features space. They also suggest that there is a high superposition of genres, and that there are very few regions that are directly correlated to a single class. In other words, BoF classification strategies that implicitly assume that specific regions of the feature space globally capture musical-related genre information can only go so far due to inherent performance limitations. The results show that a randomized and unsupervised data representation permits to build genre models that are as good as those built from thoroughly supervised data representations. At a feature selection level, more informed codebook generation strategies such as GMMs or k-Means fair as well as uniform random sampling. Considering class-dependent distributions, we observe that there is no significant benefit in seeking the most thorough representation of each class in the data space defined by local signal features. This is confirmed by the experiments in Section 4.2, where the codebooks are analyzed from an information theoretic perspective. We show that the majority of frames have a large class overlap, while only a few exhibit some level of class dependency. Nevertheless, it is also shown that selecting the codebook elements based on their class-discriminative capacities does not lead to a performance increment. We finalize this chapter with a discussion on why sub-optimal representations of the feature space have similar performance in terms of genre recognition as more informed ones.

## 4.1 Experiments in Genre Classification

Part of the experiments described in this chapter were presented in two of our publications [Marques et al., 2010, 2011b]. Here we give a more complete version of the results, and present some additional experiments that we hope will contribute to a better understanding of the challenges and limitations in genre classification. Next, we describe our experimental setup, the classification algorithms tested, and the evaluation metrics used. We then report the experiments we conducted on frame selection and genre classification.

### 4.1.1 Experimental Setup

**Classification Methods:** The methods we tested were previously described in Chapter 3. These are based on a codebook approach, which implies that each music piece (both for training and testing) is first converted into a sequence of symbols, obtained via vector quantization of the audio features. For classification, we used our own model, the VQMM, which relies on the symbol sequence, or used classical methods, namely SVMs and k-NN with several distance metrics which were also described in Section 3.3.2. Note that the former classifiers rely on

histograms of the symbol frequency of each song. For convenience, next we briefly list the functionals used:

VQMM: We used the likelihood functional to calculate the score $\mathcal{L}_\gamma(\mathcal{S})$, of the sequence $\mathcal{S}$ for the class $\gamma$:

$$\mathcal{L}_\gamma(\mathcal{S}) = \ln(P(s_1|\gamma)) + \sum_{n=2}^{N} \ln(P(s_n|s_{n-1},\gamma)) \qquad \text{(Eq. 3.7)}$$

The winning class is the one with the highest score.

k-NN: For nearest neighbor classifier we used $k=5$ (the number of neighbors), and the following distances (between vectors $\mathbf{p}$ and $\mathbf{q}$):

- Euclidean norm: $\text{dist}_{\ell_2}(\mathbf{p},\mathbf{q}) = \|\mathbf{p}-\mathbf{q}\|$ (Eq. 3.2)
- Cosine: $\text{dist}_{\cos}(\mathbf{p},\mathbf{q}) = 1 - \frac{\mathbf{p}^\top \mathbf{q}}{\|\mathbf{p}\|\|\mathbf{q}\|}$ (Eq. 3.3)
- Kullback-Leibler: $\text{dist}_{\text{KL}}(\mathbf{p},\mathbf{q}) = \frac{1}{2}\text{KL}(\mathbf{p}\|\mathbf{q}) + \frac{1}{2}\text{KL}(\mathbf{q}\|\mathbf{p})$ (Eq. 3.23)
- Jensen-Shannon: $\text{dist}_{\text{JS}}(\mathbf{p},\mathbf{q}) = \frac{1}{2}\text{KL}\left(\mathbf{p}\|\frac{\mathbf{p}+\mathbf{q}}{2}\right) + \frac{1}{2}\text{KL}\left(\mathbf{q}\|\frac{\mathbf{p}+\mathbf{q}}{2}\right)$ (Eq. 3.24)

SVM: We used LibSVM environment [Chang and Lin, 2011] in conjunction with the Weka software package [EL-Manzalawy and Honavar, 2005, Hall et al., 2009], to implement the SVM classifier. We used a radial-basis function kernel with a length-scale $\epsilon=1/k_2$ and a regularization cost of $C=2000$ (these parameters were obtained empirically).

**Evaluation Methodologies:** We conducted our experiments on two different data sets: the ISMIR04 and the LMD data sets (see Appendix C). We report the accuracy obtained over test sets only, both for the ISMIR04 and LMD data sets. For the evaluation on the ISMIR04 data set, we kept the original training-testing division proposed in the ISMIR 2004 genre classification contest. The evaluation on the LMD data set, first follows a three-fold cross validation procedure: two folds are used for training and one for testing, with all the permutations of the folds. In each experiment, the performance measures were averaged over a 10 test runs (this results in an over all of 30 runs LMD data set).

**Audio Features:** The features tested are the M17feat feature set. These features are described in Section 2.2.4, and were also used in the feature distribution related examples given in Chapter 3.

**Codebook:** In our experiments, results were obtained with various codebook generation techniques. Several values for $k_2$ were tested (see, for e.g. Table 3.3), but we decide to fix $k_2 = 200$ in most of our tests, not to clutter our analysis with unnecessary numbers. Empirically, it found to be sufficient to reach state of the art performances, and higher $k_2$ values did not

significantly alter the results. Therefore, and unless specified otherwise, $k_2$ parameter is set to 200.

### 4.1.2 Experiment 1: Frame Selection

The frame selection schemes used for the codebook generation are usually based on information pertaining to the feature densities, obtained via Gaussian mixture models or via the k-Means algorithm. Several codebook generation schemes were implemented, for selection the $k_1$ feature vectors per track and for selecting the final $k_2$ elements of the codebook

$k_1$-SELECTION
1. Build GMM model for each song (using three kernels and full covariance matrices). The resulting density probability function is used to select the $k_1$ most representative frames from each track.
2. Select frames randomly according to a uniform distribution. In this case the resulting set of frames will follow the original distribution of data but will include less representative frames.

$k_2$-SELECTION
1. Select the $k_2$ centroids obtained with the k-Means algorithm.
2. Choose randomly $k_2$ samples among this set, according to a uniform distribution.

Three combinations of these algorithms are used in the following sections and are referred as GMM + k-Means (GK), Random + k-Means (RK) and Random + Random (RR). The purpose of the various combinations of algorithms (GK, RK and RR) is to experiment codebook generation techniques that use less and less information from the data set. While the GK technique is an attempt to base the codebook on the most representative frames, RR is a technique where, although the resulting codebook follows the distribution of the data, it is likely to include less likely representative regions of the space.

We systematically used different frame selection procedures for the codebook generation, and monitor the influence on the classifiers performance. We tested several informed representations of the data-space (via Gaussian mixture models or the k-Means algorithm), versus random sampling. Table 4.1 shows the average accuracy obtained on the ISMIR04 and LMD test sets. One can see that, for each classification algorithm (SVM, k-NN and VQMM) if we consider the standard deviation around the average results, there is almost no difference between the various codebook generation techniques.

**Data-Independent Codebooks:** In light of the previous results, we decided to test a more extreme method for codebook generation: to chose codebook elements on a regular grid. Of

| | ISMIR04 | | | LMD | | |
|---|---|---|---|---|---|---|
| Codebook | GK | RK | RR | GK | RK | RR |
| k-NN$_{\ell_2}$ (5) | 75.53 ±1.01 | 75.72 ±0.81 | 74.72 ±1.24 | 55.98 ±2.33 | 58.00 ±3.14 | 58.03 ±2.92 |
| SVM | 73.20 ±0.56 | 74.98 ±0.53 | 74.81 ±1.29 | 59.60 ±1.79 | 62.41 ±1.14 | 62.63 ±0.87 |
| VQMM | 81.81 ±0.38 | 82.13 ±0.57 | 83.03 ±0.64 | 69.93 ±1.40 | 71.61 ±1.09 | 71.58 ±1.41 |

Table 4.1: Overall accuracies obtained with different classifiers (lines), and different frame selection techniques (columns), for the ISMIR04 data set (left), for the LMD data set (right).

course this is not feasible since our data lie in a 17-dimensional space and in order to collect only two points on each axis we would end with a codebook of $2^{17}$ elements. A possible solution is to generate the codebook elements according to a low-discrepancy sequence such as the Sobol sequence. A Sobol sequence generator produces a series of points $\mathbf{x}_i$ in a $d$-dimensional hypercube $\mathbb{I}^d = [0,1]^d$ such that for any integrable function $f$ the series converges as fast as possible and is such that:

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n} f(\mathbf{x}_i) = \int_{\mathbb{I}^d} f(\mathbf{x}) d\mathbf{x} \tag{4.1}$$

The objective is to map the space with as few points as possible while minimizing the holes. An interesting property is that if points of a Sobol sequence are projected on a sub-space, they are not superimposed and the holes are minimized. Following this approach, we construct a codebook by generating $k_2$ points of a Sobol sequence in a $\mathbb{I}^{17}$ hyper-cube and scale these points to fit our data space. The only information used from the dataset is the minimum and maximum values of the feature vectors. These codebooks are much less efficient in terms of memory because the mapping involves large portion of the space where there is few or no data. In this context, very large codebooks are necessary to obtain results comparable to those obtained with other methods. Several codebook sizes were experimented. For each codebook, the evaluation is based on the accuracy obtained with the VQMM classifiers. Figure 4.1 summarizes the results for codebook sizes ranging from 200 to 40000. One can see that the accuracy increases with codebook size until 20000 symbols and then levels off around 81% in the case of the ISMIR04 dataset and around 65% for the LMD dataset.

### 4.1.3 Experiment 2: Genre-Dependencies

Experiments of the previous section showed that there is apparently no clear advantage to model the statistics of the short-term feature vectors and that at this level an unsupervised approach can be considered. Indeed the as we described it, the codebook approach does not require labeled data i.e. it is not assumed that the feature vectors belong to a particular class.

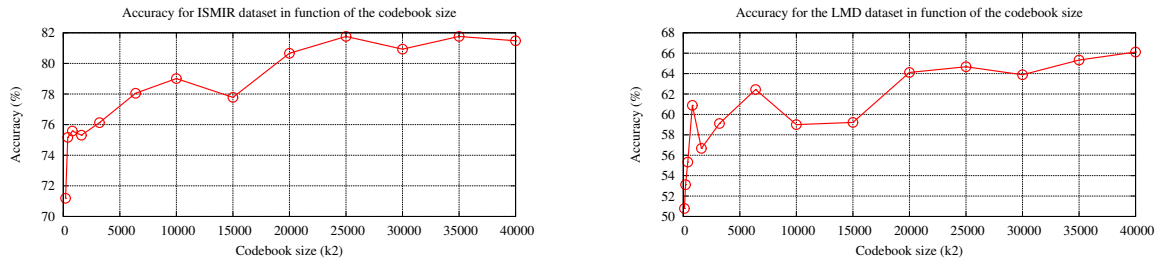



Figure 4.1: Accuracy curves on the ISMIR04 test set (left) and on the LMD test set (right) obtained with codebooks of different sizes ($x$-axis), based on Sobol sequences. For the LMD dataset, the results are averaged over three folds.

In this section we explore opposite point of view assuming that short-time frame vectors do belong to a class. Several codebooks are build using feature vectors from a single class. The assumption is that if we base the representation on elements of a single class and compare the accuracy obtained we should be able to evaluate the benefit of having a class-dependent representation at this level.

In this way, the universe of all possible training frames is significantly reduced, compared to the standard procedure, where all the available training data is used. In our experiments, codebooks built with frames from only one genre are referred as "only-X", while codebooks generated using frames from all classes are referred as "all-genres".

Table 4.2 shows the accuracy obtained on the test set when the codebook is based solely on one class and the classifier is based on SVM, k-NN, or VQMM models. All codebooks are build using the Random k-Means procedure. One can see that for example, using only frames from the Rock-Pop category for the construction of the codebook leads to a decrease of performance of $\approx 1\%$ when comparing to the codebook that is based on all genres (VQMM classifier). When looking at the worse case, the difference with the reference codebook is $\approx 3.6\%$ in the case of the codebooks based on Classical or Metal-Punk music. It means that with a representation based solely on Metal-Punk music the classification rate over all genres decreases by only 3.6%. This decrease in accuracy is rather small if we consider the perceived difference between the musical genres. The same experiment was performed with the LMD data set. The results shown in Table 4.3 share the same characteristics: in the majority of cases there is a small difference in accuracy ($\approx 1\%$) compared with the results obtained with the reference codebook. A notable exception is Tango. This result can be explained by the fact that the tango music of the LMD database is composed mainly by old recordings (from 1917-1935) with very poor sound quality. The set of feature vectors extracted from this genre have a limited frequency range and do not account for the variety found in other genres.

| ISMIR04 — only one genre | | | | | | |
|---|---|---|---|---|---|---|
| | VQMMs | SVM | k-NN$_{\ell_2}$ | k-NN$_{\cos}$ | k-NN$_{\mathrm{KL}}$ | k-NN$_{\mathrm{JS}}$ |
| all genres | 83.03±0.64 | 74.81±1.29 | 76.28±0.82 | 73.72±0.80 | 77.20±0.64 | 76.47±0.62 |
| only-Classical | 79.36±0.72 | 71.40±1.15 | 69.60±0.98 | 70.04±0.68 | 74.05±1.16 | 74.02±0.60 |
| only-Electronic | 82.63±0.73 | 73.54±1.25 | 73.62±0.68 | 70.14±0.75 | 74.25±0.47 | 74.23±0.73 |
| only-JazzBlues | 80.70±0.53 | 73.29±0.68 | 71.22±1.08 | 70.49±1.13 | 75.69±0.93 | 75.64±0.95 |
| only-MetalPunk | 79.45±0.85 | 71.22±1.21 | 69.23±1.02 | 67.15±1.02 | 73.95±0.97 | 72.84±0.83 |
| only-RockPop | 81.88±0.27 | 72.91±0.97 | 71.26±0.50 | 69.31±0.79 | 75.01±0.86 | 75.08±0.98 |
| only-Word | 81.26±0.75 | 73.17±1.50 | 72.22±1.53 | 71.17±0.85 | 76.78±0.62 | 75.91±0.71 |

Table 4.2: Results for the ISMIR04 data set. Results obtained with codebooks generated with data from a single genre, with Random k-Means selection method. In each line are the accuracies (mean value obtained in ten runs) and the corresponding standard deviation. For comparison, the first line contains the results obtained with codebooks computed with all the genres.

| LMD — only one genre | | | | | | |
|---|---|---|---|---|---|---|
| | VQMMs | SVM | k-NN$_{\ell_2}$ | k-NN$_{\cos}$ | k-NN$_{\mathrm{KL}}$ | k-NN$_{\mathrm{JS}}$ |
| all genres | 71.58 ±1.41 | 62.63 ±0.87 | 58.40 ±3.29 | 60.49 ±2.30 | 61.36 ±2.55 | 61.29 ±2.43 |
| only-Axé | 71.84 ±1.79 | 61.14 ±1.04 | 57.64 ±2.91 | 59.36 ±1.70 | 60.67 ±2.65 | 60.38 ±2.12 |
| only-Bachata | 70.68 ±1.39 | 60.74 ±1.10 | 56.96 ±2.07 | 57.16 ±2.32 | 59.58 ±1.65 | 59.98 ±1.57 |
| only-Bolero | 68.92 ±1.67 | 58.03 ±0.96 | 54.22 ±2.71 | 55.09 ±2.50 | 57.22 ±2.45 | 57.24 ±2.27 |
| only-Forró | 71.61 ±1.33 | 62.14 ±1.06 | 58.24 ±2.38 | 60.62 ±1.47 | 60.85 ±3.12 | 60.98 ±2.90 |
| only-Gaúcha | 71.99 ±1.29 | 61.52 ±0.89 | 57.93 ±2.67 | 60.36 ±2.34 | 60.47 ±2.41 | 60.78 ±2.11 |
| only-Merengue | 70.72 ±1.51 | 60.74 ±0.95 | 56.29 ±2.53 | 56.96 ±2.23 | 59.00 ±1.98 | 58.69 ±2.39 |
| only-Pagode | 71.89 ±1.88 | 61.71 ±1.28 | 56.64 ±2.65 | 57.11 ±2.48 | 58.87 ±1.48 | 59.38 ±1.25 |
| only-Salsa | 71.02 ±1.39 | 61.14 ±0.78 | 56.65 ±2.47 | 57.49 ±1.83 | 59.20 ±1.74 | 59.62 ±1.63 |
| only-Sertaneja | 71.58 ±1.54 | 61.50 ±1.20 | 57.60 ±2.83 | 59.13 ±1.46 | 60.56 ±2.49 | 61.13 ±2.22 |
| only-Tango | 53.81 ±4.40 | 44.21 ±2.76 | 44.78 ±2.56 | 44.73 ±2.59 | 50.82 ±1.67 | 50.60 ±1.92 |

Table 4.3: Results for the LMD data set. Results obtained with codebooks generated with data from a single genre, with Random k-Means selection method. In each line are the accuracies (mean value obtained in thirty runs) and the corresponding standard deviation. For comparison, the first line contains the results obtained with codebooks computed with all the genres.

## 4.2 Information-Theoretic Analysis

Results from the previous sections show that performance of the classifiers is not significantly altered by the frame selection methods, and it is possible to achieve good levels of performance without knowing the feature distribution via Sobol sequences. Even when the training instances in the codebook generation process are restricted to only one class, performance levels are not greatly affected. This raises questions on how the short-term feature vectors are distributed in the data representation space. Is there a strong class overlap? Are there genre-specific regions of the feature space? The objective of this section is to shed a light on this issues, through the analysis of the symbols distributions from a information-theoretic point of view (for a more detailed description of the information theoretic measures used here see Appendix A.3).

### 4.2.1 Statistical Distributions of Feature Vectors

The entropy of the codebook gives us a measure of how the symbols are distributed among the clusters. The (normalized) symbol entropy is:

$$\mathcal{H}(\mathcal{S}) = \frac{-1}{\ln k_2} \sum_{k=1}^{k_2} P(s_k) \ln(P(s_k)) \tag{4.2}$$

where $P(s_k)$ is the a priori probability of symbol $s_k$ (cluster $s_k$, with $k = 1, \ldots, k_2$). The normalizing constant $\log k_2$, limits the entropy, $\mathcal{H}(S)$, to values in the interval $[0, 1]$. High entropy values are associated with codebooks that have even symbol distribution, while low entropy values correspond to codebooks with a few predominant clusters. The same reasoning can be applied to the conditional symbol entropy for a given genre. The (normalized) symbol entropies for a particular genre is:

$$\mathcal{H}(\mathcal{S}|\gamma_i) = \frac{-1}{\ln k_2} \sum_{k=1}^{k_2} P(s_k|\gamma_i) \ln(P(s_k|\gamma_i)) \tag{4.3}$$

where $P(s_k|\gamma_i)$ is the conditional probability of symbol $k$, given the genre $\gamma_i \in \Gamma$. In this case, high conditional entropy values are associated to a wide coverage of the data space by the feature vectors belonging to genre $\gamma_i$. On the other hand, low values mean that the short-term instances of the genre are quantized by a restricted set of clusters, and therefore are confined to specific regions of the data space. This is confirmed by the results in Figure 4.2, where the symbol distributions are shown for the different classes in the ISMIR04 data set, for codebooks created with frames from only one genre. Each line pertain to a single codebook, and in it are represented the symbols distribution by genre (columns). For comparison, in the last line are the genre distributions for a codebook created using all genres. The conditional entropy of

the symbols is a indicator of the frame diversity of the genre. For instance, for the codebook based on Electronic music, the class distributions have a more uniform behavior than the class distributions for the codebook base on MetalPunk, which tells us that the frames of this genre are indeed less diversified. This is also noticeable inspecting the symbol distribution of the Classical genre when the codebook was created based on MetalPunk. The fact that the distribution is very sparse is an indicator that there is little superposition between frames of the two genres; and the reverse situation also confirms it (MetalPunk feature distributions from a codebook based on Classical frames). The Figure 4.2 shows that the plots in the diagonal have the highest entropy in each line. This is natural since the codebooks were generated with frames belonging to the same genre, and therefore, the symbols for that specific genre are more evenly distributed among the codebook clusters. The figure also shows that there is a significant number of clusters with a strong genre superposition. Similar behaviors were observed inspecting the codebooks for the LMD data set.

We conducted further tests to study the discriminative capacity of the feature vectors and to analyze to what extent they characterize musical genres. An intuitive measure of the clusters discriminative capacity is the mutual information, $\mathcal{I}(\Gamma; s_k)$, shared between a given symbol $s_k$ and the genres:

$$\begin{aligned} \mathcal{I}(\Gamma; s_k) &= \mathcal{H}(\Gamma) - \mathcal{H}(\Gamma|s_k) \\ &= \mathcal{H}(\Gamma) + \sum_{i=1}^{|\Gamma|} P(\gamma_i|s_k) \ln(P(\gamma_i|s_k)) \end{aligned} \tag{4.4}$$

where $|\Gamma|$ represents the total number of genres. $\mathcal{H}(\Gamma)$ is the entropy of all genres, independently of the symbols, and is a fixed value greater or equal to the conditional entropy $\mathcal{H}(\Gamma|s_k)$. A high level of shared mutual information implies low values for the conditional entropy $\mathcal{H}(\Gamma|s_k)$. Low entropy values mean that the genre distribution for $s_k$ is dominated by one of the genres. On the other hand, symbols with high entropy values have a weak discriminative capacity, since, in these clusters the genres tend to be approximately equiprobable.

Figure 4.3 shows the conditional genre distribution given the symbols $p(\Gamma|s_k)$, for the ISMIR04 and LMD data sets, for a codebooks created with frames from all classes. The mutual information of the symbols is given by the bottom plots in each figure. We grouped the symbols by genre (i.e. by the most probable class), and within each genre, the symbols were ordered by decreasing mutual information. High mutual information values correspond to dark colors, while low values are light colored. It is noticeable the genre transitions in these plots; in the ISMIR04 plots reflect the class imbalance in this set, while in the LMD ones the number of symbols is more or less evenly distributed for each class. The figure shows that a majority of symbols exhibit a strong class overlap, and therefore have a low discriminative capacity. However a minority of symbols show high class-dependencies. This is particularly noticeable for

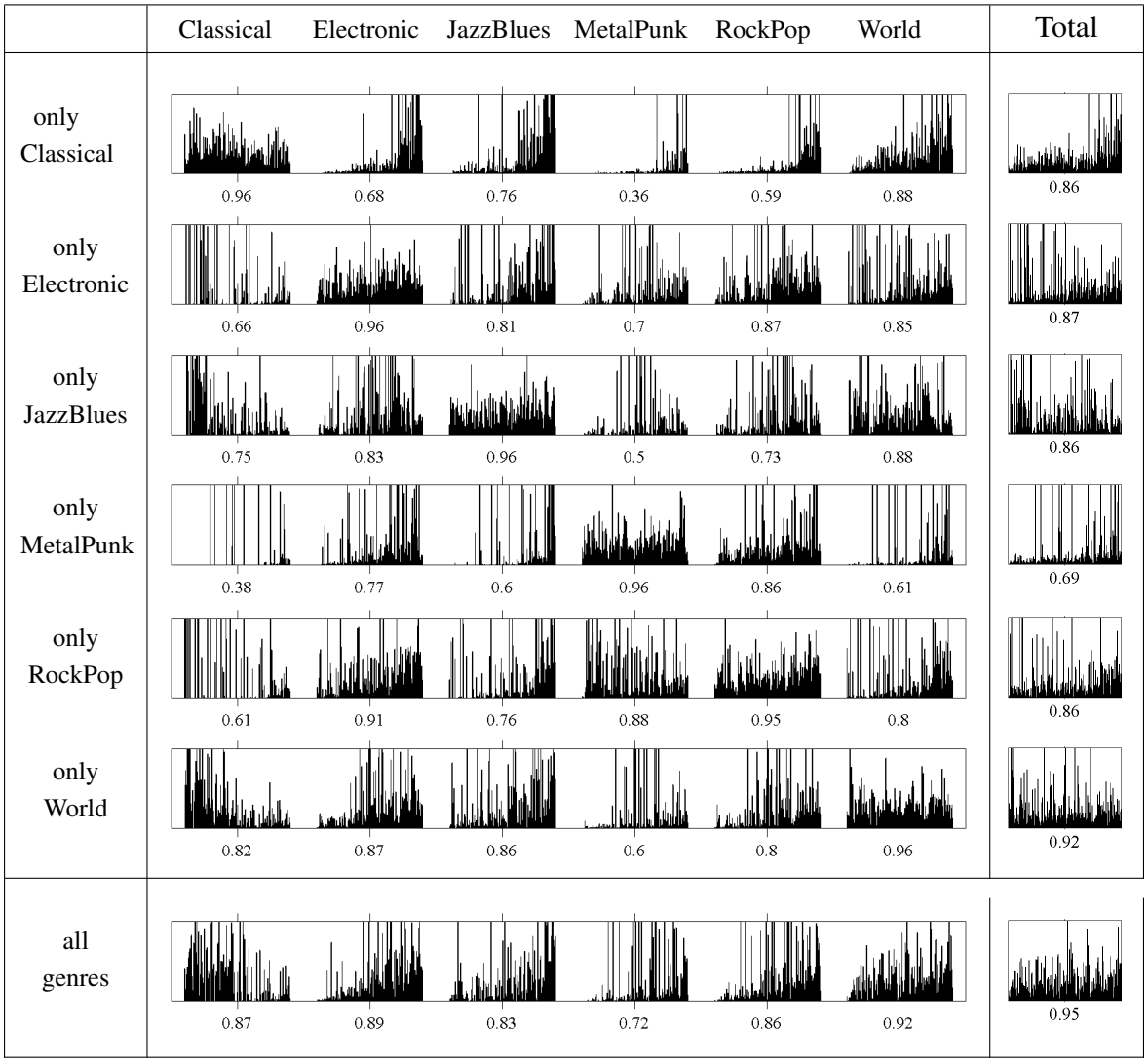



Figure 4.2: Symbol distributions by genre for the ISMIR04 data set, for several codebooks with $k_2 = 200$, created using frames from only one genre. Each line corresponds to a specific codebook, while the columns correspond to the different genres distributions for that codebook. In the last column are the global symbol distributions (for all genres). For comparison, in the last line are the genre distributions for a codebook created using all genres. For visualization purposes, y-axis on all the plots is limited to $0.02$. The numbers beneath each plot are the conditional symbol entropies (Equation 4.3). In each line, the symbols were ordered in decreasing mutual information (Equation 4.4). The ordering is the same for all the plots in given row.

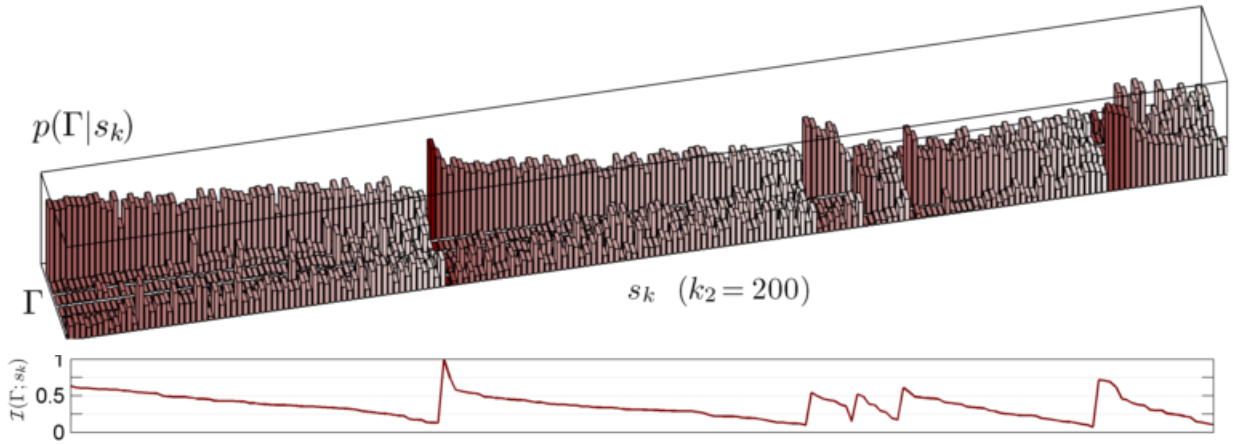
the class Tango - the rightmost group of symbols in the LMD bar plot. Empirically, we observed that this is due to an artifact from poor recording conditions rather than a meaningful musical characteristic[1]. This means that specific region(s) of the feature space belong only to frames from the class Tango.

So far, the experiments we conducted indicate that there is, in fact, some feature/genre dependencies, but they also show that meticulous partition schemes of the feature space are as good as random or less representative ones. Furthermore, it remains unclear what contribution, if any, have the more discriminative symbols and their overall percentage in the performance of classification algorithms. These questions are addressed in the next experiment.

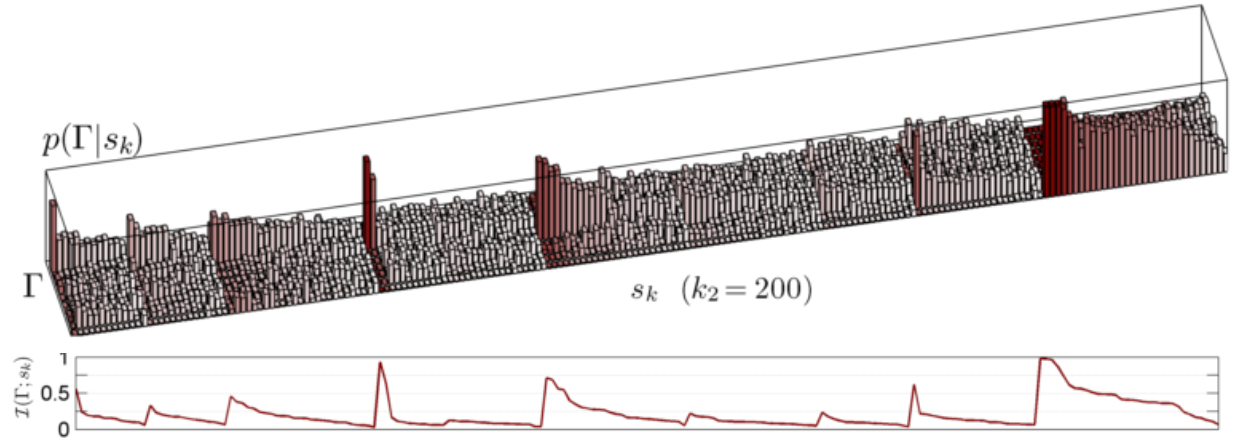
### 4.2.2 Experiment 3: Discriminative Codebooks

The previous experiments showed that there is at least some minor dependencies between short-time frames and genres, since a small subset of symbols have predominant classes (although the vast majority appears in every class). In this section, we tested the influence of genre information present in the symbols (or the lack of) on the overall classifier performances. Our objective is to ascertain if codebooks comprised of clusters with high discriminative capacity yield better classification accuracies than codebooks where the genres are evenly distributed among all clusters. To build such codebooks, we opted to first create a codebook with $k_2 = 400$, using frames from all genres, and selected a subset of $200$ codewords to generate a new codebook. The selection process was based on the mutual information between the symbols (codewords) and genres (Equation 4.4). We selected three codebooks: one with the 200 most discriminative symbols (with the highest mutual information), another with the 200 less discriminative symbols, and a final one with the 200 in the middle. We are aware that quantizing the feature vectors with the new codebooks will change the symbol distribution, and, therefore, will change its mutual information contents. Nevertheless, Figure 4.4 shows that codebooks created from a subset of codewords within a given discriminative range, will also have a high number of symbols within that range. In Table 4.4 are the accuracies obtained on the ISMIR04 data set, with the proposed codebook-generation method. The results show that there is no significant variations in the classifiers performances for the different codebooks. This indicates, that in codebook representations, each symbol discriminative capacity is not essential for genre classification.

[1]Note that the sonority from old recordings is itself a characteristic. Although it could (should) be used to improve classification, this attribute has very little relation to the genre Tango. It is likely, for instance, that Charleston or other recordings from the same period exhibit similar spectral signatures.




(a)




(b)

Figure 4.3: Conditional class distribution, $p(\Gamma|s_k)$, for the ISMIR04 (Figure 4.3(a)) and the LMD (Figure 4.3(b)) data sets for a codebook, with $k_2 = 200$, trained with all genres. The symbols (normalized) mutual information, $\mathcal{I}(\Gamma; s_k)$ (Equation 4.4) are shown in the bottom plots. High mutual information values are dark colored while low ones are light colored. For both sets, the symbols were first grouped by genre, and within each genre, the symbols were ordered by decreasing mutual information. For the ISMIR04 set, the bar plot shows the probabilities of the symbols for the following genres: Classical, Electronic, JazzBlues, MetalPunk, RockPop, and World. For the LMD set the genre ordering is the following: Axé, Bachata, Bolero, Forró, Gaúcha, Merengue, Pagode, Salsa, Sertaneja, and Tango. The peaks in the mutual information plots indicate the genre transitions. For both sets, the majority of the symbols has low mutual information, and only a small minority has values above 0.75. Furthermore, for the LMD set, there are three genres, Gaúcha, Pagode, and Salsa, that have value below 0.25.

| ISMIR04 — discriminative codebooks | | |
|---|---|---|
| | VQMM | SVM |
| all symbols | 83.03 ±0.64 | 74.81 ±1.29 |
| Top 200 | 82.54 ±0.75 | 74.55 ±1.09 |
| Middle 200 | 83.05 ±0.67 | 74.13 ±1.08 |
| Bottom 200 | 82.00 ±0.70 | 74.02 ±0.75 |

Table 4.4: Results for the ISMIR04 data set, obtained with discriminative codebooks (different discritizations of the feature space). The codebooks generated with a subset of 200 codewords (i.e. symbols) from larger codebooks with $k_2 = 400$. Top 200 corresponds to codebooks that were created with the 200 most discriminative codewords and bottom with the 200 least discriminative ones. In each line are the mean accuracies and the corresponding standard deviation, obtained on ten test runs. For comparison, the first line are the results for all genres presented in Table 4.2.

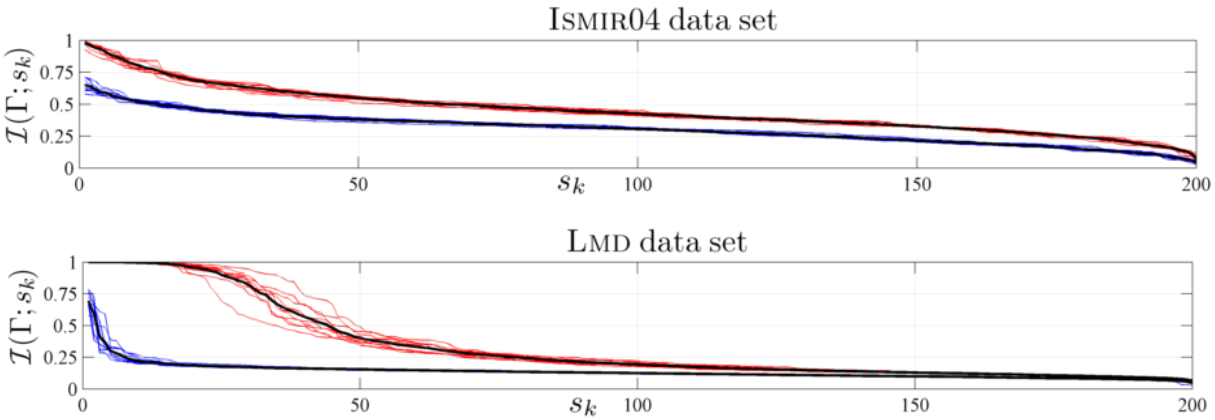



Figure 4.4: Mutual information curves for the most and the least discriminative codebooks. Symbols with high mutual inforamtion values are ploted first ($x$-axis). In red are the curves for the codebooks created with the most discriminative codewords, and in blue the codebooks with the less discriminative. The plots comprise ten tests runs on two data sets (ISMIR04 top and LMD bottom). Curves in black represent the mean values.

## 4.3 Discussion

In this chapter we analyzed the problem of inferring music genre based on spectral features. Our findings indicate there is no apparent benefit in seeking a thorough representation of genres in the low-level features space. This also suggest that there is a high superposition of genres in the feature space, and that there are very few regions that are directly correlated to a single class. In other words, BoF classification strategies that implicitly assume that specific regions of the feature space globally capture musical-related genre information have inherent performance limitations. In ours experiments, we studied diverse partitions of the feature space by varying the amount of information regarding the provenance of audio frames. The results show that a randomized and unsupervised data representation permits to build genre models that are as good as those built from thoroughly supervised data representations. At a feature selection level, more informed codebook generation strategies such as GMMs or k-Means fair as well as uniform random sampling. As far as feature/genre dependencies, selecting frames from a single genre to generate the codebook does not significantly deteriorate the performance of the tested classification approaches. We also analyzed if sellecting codewords with a high discriminative power (i.e. high class dependancy) was benefitial, but the symbols' discriminative capacity did not have a noticeable impact on classification performances.

An obvious question arises from our observations: why are BoF classification methods effective, if there is litte genre information in low-level descriptors? Since BoF methods rely on probabilistic or discriminative representations of the feature space, this is a sign, that on average, there are regions more densely populated by instances of a particular class. Note that we do not challenge this fact, our experiments only suggest that there exist a high superposition of low-level features. Global statistics seem to be sufficient for the performances obtained by BoF methods. In Figure 2.6 we already seen scatter plots of point distributions by genre for the ISMIR04 data set, and the figure clearly shows class dependent regions. However, this information is a bit measleading since we ploted a equal number of points per class; this does not reflect the class a priori distributions which in this particular case are very unbalanced - having done so, points from the genre Classical would cover a much wider area. Furthermore, why do clusters representations of more discriminative regions do not seem to help the classification process? Part of the reason lies in the low-level feature characterization of the audio signal. As seen in Figure 2.6 (bottom plots), songs are not restricted to a single point, they are composed of sequences of feature vectors that occupy wide regions of the feature space. Therefore, the points in the feature sequences may wander into more descriminative regions, but overall, this may not be enough for classifiers to take advantage of this fact, specially if the discriminative areas are as scarce as our experiments suggest.

# Chapter 5

# Music Autotagging

Autotagging was briefly introduced in Section 2.3.2, and in Table 3.5 we reported overall performances of the VQMMs along with other authors results. In this chapter we examine the problem of autotagging more thoroughly. In Section 5.1, we analyze the metrics used for evaluation of autotagging systems. These are based on generalization of the performance measures of binary classification problem and are used in many other research areas of Information Retrieval. We review the annotation and the retrieval problems commonly used in the context of autotagging, and the different set of evalution measures that are applied to each task. Then we describe three sets of experiments, where we demonstrate the relevance of a number of music autotagging issues that we believe are, to the best of our knowledge, only addressed superficially in current literature. The setup for our experiments is presented in Section 5.2. The first experiments (Section 5.2.2) addresses issues related to the notion of "fragility" in autotagging evaluation methodologies. We show that small alterations to the set of tags can alter significantly systems performances and how tag distribuitions effect them. In the second experiment (Section 5.2.3), we address the issue of generality of autotagging models. In the third experiment (Section 5.2.4), we address limitations of exploiting tag correlations in a second processing stage. We finally propose a discussion on these issues and directions for future work in Section 5.3.

## 5.1 Evaluation Measures

Autotagging is a multi-label problem where each song is characterized by a set of tags. This problem is inherently different from traditional classification, such as genre or artist classifications, where each song is associated with a single label belonging to a set of disjoint classes. In multi-label scenarios the labels are not mutually exclusive, and several labels must be assigned

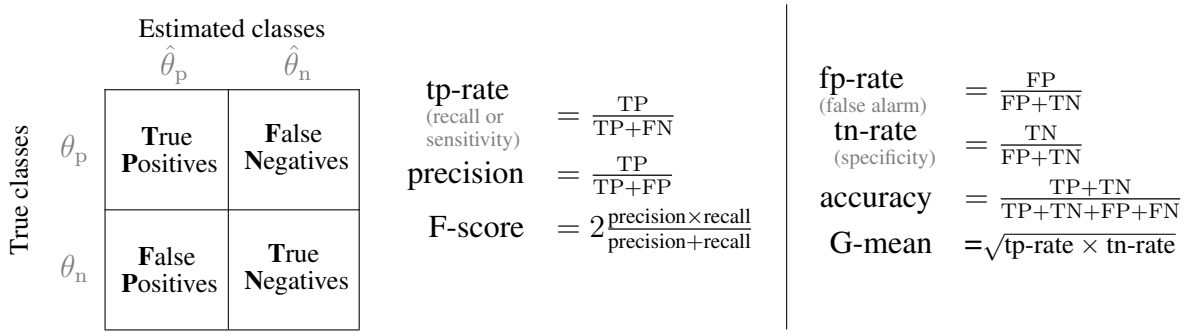


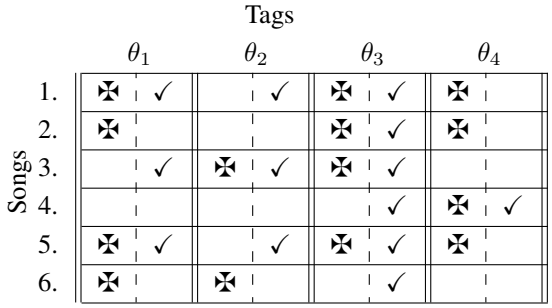

Figure 5.1: Confusion matrix for a two class problem and common performance metrics derived from it. $\theta_{\mathrm{p}}$ corresponds to the class with positive examples and $\theta_{\mathrm{n}}$ with negative ones, and $\hat{\theta}_{\mathrm{p}}$ and $\hat{\theta}_{\mathrm{n}}$ are their hypothesized counterparts. For clarity, counts such as TP (True Positives), and TN are in upper case and rates such as true and false positive rate (tp-rate and fp-rate) are in lower case. Note that often other nomenclature is used (names in gray color beneath tp-rate and fp-rate)

to each instance. This corresponds to one of the two commonly used testing scenarios in autotagging: the *annotation* problem. For a given song, the objective is to predict a list of tags that best characterizes it. The other commonly used testing scenario is the *retrieval* problem, where for a given tag (or set of tags) the objective is to retrieve the top songs that are relevant in terms of the query tag(s). In retrieval, the songs are ordered according to their relevance, while in annotation no ranking is required. Although the two tasks are obviously related, the different testing methodologies require different evaluation measures. Next we describe the most common measures used in autotagging[1], starting with the ones for annotation, and then the ones used in a retrieval context. Both are derived from generalizations of the metrics of the binary classification problem. Additionally, we address some issues specific to autotagging that need to be taken in consideration when computing these metrics. Some aspects, such as the way score averages are obtained can alter significantly the evaluation results, and the models architecture can condition the evaluation procedures used and impose a limit on the maximum achievable performances. These aspects are analyzed in greater detail in the final part of this section.

### 5.1.1 Annotation

Annotation, and multi-label classification in general, can be tackled as a whole, i.e. devising methods that map directly each instance into a set of labels, or the problem can be converted into simpler binary classification problems, one for each label. The evaluation, however, is usually

[1]Note, that these metric are also adopted in a variety of other research fields such as Information Retrieval, Data Mining, and Machine Learning, (see [Fawcett, 2006] for an overview).




| per tag, per-song, and global precision and recall scores | |
|---|---|
| $\text{precision}_t$ | $= \frac{1}{4}\left(\frac{2}{3} + \frac{1}{3} + \frac{4}{6} + 1\right) = \frac{2}{3}$ |
| $\text{precision}_s$ | $= \frac{1}{6}\left(\frac{2}{3} + 1 + \frac{2}{3} + \frac{1}{2} + \frac{2}{3} + 0\right) = \frac{7}{12}$ |
| $\text{precision}_g$ | $= \frac{8}{13}$ |
| $\text{recall}_t$ | $= \frac{1}{4}\left(\frac{2}{4} + \frac{1}{2} + \frac{4}{4} + \frac{1}{4}\right) = \frac{9}{16}$ |
| $\text{recall}_s$ | $= \frac{1}{6}\left(\frac{2}{3} + \frac{1}{3} + \frac{2}{2} + 1 + \frac{2}{3} + 0\right) = \frac{4}{9}$ |
| $\text{recall}_g$ | $= \frac{8}{14} = \frac{4}{7}$ |

Figure 5.2: Synthetic autotagging example with six songs represented by the lines in the table, where each song can be annotated with a set of four possible tags $\Theta = \{\theta_1, \theta_2, \theta_3, \theta_4\}$ (the table columns). Each column is divided in two by a dash-line, in the left are the ground truth annotations (the symbol "✠" indicates the presence of the tag), and in the right side of each column are the classification results (the "✓" symbols are the positive predictions). For example the tag $\theta_3$ was assigned to all the songs, but songs number 4 and 6 do not have the tag. The precision and recall evaluation metrics are calculated in three different ways: averaging the results on a per-tag and per-song basis, and globally. Note that the three ways of estimation the same evaluation metrics results distinct values.

based on metrics derived from the binary classification problem. Note, that in many information retrieval problems, and particularly in autotagging, accuracy alone is not sufficient to give a clear idea of the performance. Due the the sparsity of the annotations found in most data sets, accuracies can be over inflated by conservative classifiers that assign the "negative" class more often (i.e. the absence of the tag). Other measures are more suited for this type of problem with skewed class distributions; all of them are derived from the confusion matrix (i.e. the contingency table) of the binary classification problem. Figure 5.1 shows the confusion matrix and the equations for several common metrics that can be calculated from it. Given two classes, $\Theta = \{\theta_\text{p}, \theta_\text{n}\}$, and the classifier predictions, $\hat{\Theta} = \{\hat{\theta}_\text{p}, \hat{\theta}_\text{n}\}$, there are four possible outcomes represented in the confusion matrix: true positive (TP), false negative (FN), false positive (FP), and true negative (TN). In the diagonal of this matrix are the correct decision while the other elements, FP and FN, represent the classification errors. In autotagging, and for each song, the positive class $\theta_\text{p}$ indicates the presence of the tag $\theta$, while the $\hat{\theta}_\text{p}$ and $\hat{\theta}_\text{n}$ are the positive and negative classifier predictions, respectively. The most common reported measures are the *precision*, *recall*, and the *F-score*, although some authors also use the G-mean [Seyerlehner, 2010]. The precision is the number of correctly classified positive songs divided by the total number of positive predictions. The recall is the number of correctly classified songs over the total number of (true) positive songs. The F-score is the harmonic mean of the precision and the recall. It is common either to report evaluations based on F-scores alone, or the F-scores complemented by the precision and recall values, but it is not wise to present isolated precision or recall measures. The reason is because precision values can be artificially inflated

with "conservative" tag attribution schemes: classifiers that assign a positive class only with strong evidence (and make few false positive errors) have typically high precision scores (and low recall ones). Recall scores, on the other hand, can be inflated choosing "liberal" classifiers: for the case of the trivial positive classifier (i.e. assigns the tag to all the songs), the recall is equal to one - the maximum value possible. While trivial models can achieve relatively good scores in one of these two measures, only a truly valid model can obtain simultaneously high values in precision and recall. This is not the only issue that needs to be taken into consideration when using these evaluation measures in the context of autotagging. In testing conditions, the measures in Figure 5.1 are usually not reported in terms of each tag individually due to the high number of tags present in a typical data set. Hence, some averaging is normally done either in a per-tag, per-song or global basis:

Per-Tag: Scores are computed based on the evaluation measures for each tag individually, which are then averaged over the whole set of tags. The values calculated in this fashion are usually low compared to the two other averaging methods. The reason is that classification methods typically obtain low scores in a non-negligible portion of the tags, which brings down the overall scores. Nevertheless, results collected this way are considered more reliable then the per-song and global measures in a sense that they are not influenced by the strong imbalance in tag distributions, commonly encountered in most autotagging sets. For example, the top ten most frequent tags in the MAGTAG5k data set account for over 40% of all annotations, and classifiers that predict well these tags start off with high per-song and global scores, independently of how well the rest of the tags are estimated (this and other issues related to the tag distributions and their impact on autotagger performances will be addressed in Section 5.2.2). To avoid confusion, metric computed in this fashion will be accompanied, from now on, with the underscore letter "$t$" to specify the per-tag based averages (e.g. F-score$_t$).

Per-Song: Scores, in this case, are computed for each song individually, and are then averaged over the whole set of songs. That is, for each song the evaluation scores are calculated based on the tags attributed to the song. Computing performances on a per-song basis, instead of a per-tag basis, may intuitively seem a more appropriate metric[2]. However, per-song metrics are usually biased by the most frequent tags (for e.g. the tag "song_recorded" is present in 88% of the songs in the CAL500 data set): systems that are good in predicting these tags obtain high per-song score

[2]Scores on a per-song basis are a measure of how well each song is annotated. This may seem as important as estimating how well tags are predicted, but per-song metrics are biased by the most frequent tags and produce over optimistic results.

even if their performance is sub-optimal in other tag predictions. Note that a similar effect also happens with global metrics. To denote score computed in a per-song basis, we will use the underscore letter "$s$" (e.g. F-score$_s$).

Global: Global scores are calculated based on the results of the whole data set, independently of the songs or tags. For example, true positive counts are based on the number of times all the tags were correctly predicted, false positive counts are the number of times any given tag was incorrectly assigned, and so on. Metrics computed in this fashion also suffer from similar biases as the per-song metrics, which can be over inflated by predicting well the most frequent tags. The scores computed on a global basis will be denoted with the letter "$g$" (e.g. F-score$_g$).

Figure 5.2, through a synthetic example, illustrates the three ways of computing the recall and precision scores and shows that different values are obtained depending on the type of averaging used. The metrics reported in Figure 5.1, along with the averaging processes just described, are a commonly used evaluation measure in the context of annotation.

### 5.1.2 Retrieval

In retrieval it is more useful to rely on Receiver Operating Characteristics (ROC) graphs rather than the metrics previously described for annotation (see [Fawcett, 2006] for an overview on the subject). ROC graphs are 2-dimensional plots that depict the tradeoff between true hits and false alarms (tp-rate as a function of the fp-rate). A binary classifier produces a single point in the ROC plot. However, retrieval is based on a selection of the top songs from a ranked list that reflect, in some manner, the relevance of a given tag: i.e. the top songs are assigned the tag while the bottom ones do not. Different values for the number of chosen songs correspond to different points in the ROC graph. In practice, a plot is made for all the possible values of the top chosen songs, starting with a single song and finishing with the whole set of songs. Note that the ranking scheme used in autotagging retrieval tasks is also conditioned to the type of classifiers used. Some classifiers such as classification and regression trees or SVMs produce a single class decision for each instance. These classifiers produce a single point in the ROC curve, and are not applicable in retrieval context where some sort of ranking is needed. For classifiers that produce a continuous output, different thresholds can be applied to predict class memberships and to construct the ROC curve. In this situation, varying the threshold from $+\infty$ to $-\infty$ (from conservative to liberal classifiers) is equivalent to the ranking process found in retrieval. In Figure 5.3(a) is an example of the ROC curve obtained using different thresholds on the points of Figure 5.3(b): i.e. for a given threshold value, points above it are classified as positive and below it as negative. In Figure 5.3(a) are also represented some of the relevant

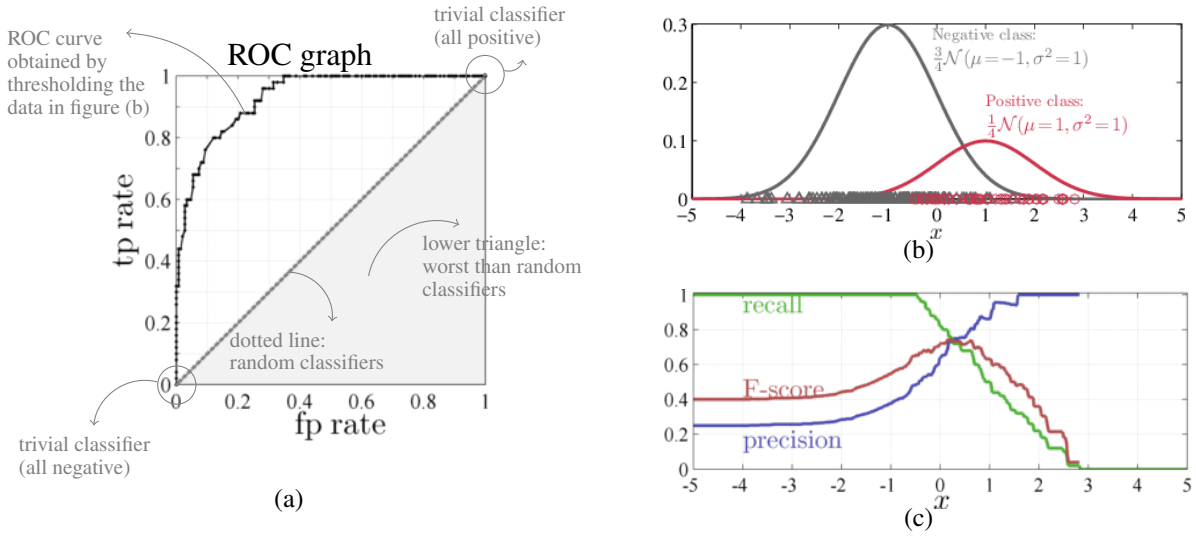



Figure 5.3: (a) ROC graph and properties associated with certain points and areas of this space (see text for details). The figure also shows the ROC curve (in black) obtained by thresholding the values of the points in (b). (b) Synthetic example, for one-dimensional data divided in two classes: the points belonging to the positive class are represented by red circles, and the ones for the negative class are the gray triangles. This figure also shows the class-dependent distributions (unit variance Gaussians) weighted by the class priors (the negative prior is three times larger than the positive one). The classifier considered in this example is a simple threshold, where points below it are assigned the negative class and points above it the positive class. The ROC curve and the curves in Figure (c) are produced by varying the threshold from $-\infty$ to $+\infty$. This procedure is also useful for estimating the threshold that yields the best tradeoff between precision and recall.

properties of the ROC graphs. A perfect classifier corresponds to the point (0,1) in the ROC graph (the upper left corner). Informally, a classifier is better than another if it is closer to this upper left corner. The point (0,0) is the case of assigning the negative class to all the instances, while the point (1,1) is the case of issuing all positive classifications. The diagonal dotted line in the graph represents the strategy of randomly guessing the classes. Classifiers corresponding to points below this diagonal fare worst than random guesses, but in this situation, negating the classifiers output produces a point in the upper left triangle. The ROC curve is also an useful tool when deciding which threshold is better suited for a given classifier and for deciding the tradeoff of true hits and false alarms. Nevertheless, 2-dimensional depictions of classifiers performances are not straightforward to compare, and it is often easier to convert the curves to a single value that represents the expected efficiencies. One common approach is to calculate the *Area Under the ROC Curve* (AUC) [Bradley, 1997]. This area is a portion of the unit square, therefore it is

| | Annotation | | | | | | | Retrieval | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $P_t$ | $R_t$ | $F_t$ | $P_g$ | $R_g$ | $F_g$ | G-mean | Mean AUC | Mean AP | P-N |
| VQMMs | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | ✓ | ✓ | |
| [Marques et al., 2011a] | | | ✓ | | | ✓ | | | | |
| [Bertin-Mahieux et al., 2008] | ✓ | ✓ | ✓ | | | | | ✓ | ✓ | ✓ |
| [Coviello et al., 2011] | ✓ | ✓ | ✓ | | | | | ✓ | ✓ | |
| [Hoffman et al., 2009] | ✓ | ✓ | ✓ | | | | | ✓ | ✓ | |
| [Miotto et al., 2010] | ✓ | ✓ | ✓ | | | | | ✓ | ✓ | ✓ |
| [Ness et al., 2009] | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | | ✓ | | |
| [Seyerlehner et al., 2010b] | | | ✓ | | | ✓ | ✓ | | | |
| [Turnbull et al., 2008b] | ✓ | ✓ | ✓ | | | | | ✓ | ✓ | |
| [Yang et al., 2012] | | ✓ | | | ✓ | ✓ | | | | |
| [Zhao et al., 2010] | ✓ | ✓ | ✓ | | | | | | ✓ | |

Table 5.1: Evaluation metrics reported on published works that address the task of autotagging. We included the performance measures used in this thesis (the VQMM line), noting that in Section 5.2, other forms of evaluation are also used. This is also true for some of the works referred in this table (see details in text).

comprised between 0 and 1 (although classifiers with AUC values of ½ and below are useless). In autotagging retrieval problems the AUC is one of the most used evaluation metric, but since each individual tag produces one ROC curve, the common approach is to average the AUC values for all the tags.

Another typical evaluation metric is the *Mean Average Precision* (MeanAP). Intuitively, precision is the most suited metric for evaluating retrieval problems, since it reflects the percentage of correctly assigned positive predictions. However, as previously mentioned, this criterion privileges conservative classification schemes. This can be circumvented by computing the *Average Precision* (AP), which takes into consideration the order in which the songs are presented. The average precision is calculated moving down the list of ranked songs, and every time a true positive prediction is made the precision is calculated; the AP is the mean of all those precisions, and the MeanAP is the AP averaged over all the tags. A related evaluation measure is the top-N precision, which is inspired on search engines that typically limit the number of returned query results. This is a simple modification of plain precision, which is the fraction of true positive predictions in the top-N songs of the ranking (normally the value of N is between 10 and 50).

### 5.1.3 Autotagging Models and Evaluation Procedures

Table 5.1 is a non-exhaustive list of works that use the evaluation metrics just described. These are not the only ones used in autotagging problems, for example in [Yang et al., 2012] the Spearman's rank correlation coefficient [Yule and Kendal, 1968] is used as an evaluation criterion, and in [Ness et al., 2009] tag accuracies are also reported; notwithstanding, the performance metrics addressed in this section are by far the most popular choice for autotagging evaluation scenarios, and are also the ones adopted in our tests. However, the evaluation choices presented in Table 5.1 do not differentiate between distinct testing methodologies, particularly the ones used in the annotation part of the autotagging problem, and these can influence the maximum performance achieved by the different techniques. There are two typical setups in annotation: one is to choose for each song a predefined number of labels with the highest scores, the other is to assign the tags based on a binary attribution scheme. Assigning to each song a fixed number of tags puts a limit in the performance that can be achieved by autotagging systems. If the number of labels in a song is less than the predefined number of assigned tags, then some incorrect attributions may be chosen just because they are part of the top list. In the reverse situation, songs with more tags than the established number of attributions, meaningful labels may be left out. This testing setup is used in [Bertin-Mahieux et al., 2008, Miotto et al., 2010, Turnbull et al., 2008b], where precision, recall and F-scores are computed annotating each song in the CAL500 with 10 labels, while the average number of tags per song in this set is approximately 26, which limits the maximum achievable performance. This upper bound is reported in Tables 3.5 and B.4 along with ours and other authors results. When tags are assigned by binary attribution schemes (e.g. building positive and negative models for each tag) there is no impediment in achieving maximum performance.

These two distinct types of evaluations are tied to the design of the models used for testing. In the previously cited works, a single model is used for each tag, and the adopted strategy to annotate unlabeled songs, is to rank tags by their scores, and to choose the top portion. In binary attribution, typically two models are used to predict each tag, a positive and a negative one, and the label is assigned when the positive model has a higher score than the negative one, and therefore, different songs are annotated with a different number of tags. Note that although the models used in autotagging can constrain the evaluation procedures, this does not translate into a strict imposition. As previously mentioned, in retrieval one cannot use classifiers that output a discrete label since the results need to be ordered by their relevance; however, on most occasions it is fairly simple to "look inside" the classifier and convert the output to a continuous value. For example, in decision trees, the leaf nodes can output the class percentages instead of the most frequent class, and in SVMs one can see how far away a given point is from the boundary, instead of just transmitting on which side of the boundary the

point is. In annotation a similar adaptation can be made for single models to produce binary tag attributions, for e.g., by thresholding the output of the classifiers. The Figure 5.3(c) is an example of this type of procedure, where the optimal threshold corresponds to the point of intersection of the two Gaussian functions (i.e. the maximum a posteriori estimate). In testing conditions, the threshold can be established, e.g. by measuring performances obtained on a validation sub-set of the training data.

In this section we presented a general introduction on the performance measures commonly used in information retrieval problems. In what concerns autotagging, we also addressed relevant details specific to this task that may not be evident to someone outside the MIR community. In summary, autotagging can be divided into two main problems, annotation and retrieval, and distinct performance metrics are used in each situation. Additionally different averaging methods can yield very different performance values (e.g., per-tag and global results in Table 3.5). The three averaging techniques, per-tag, per-song and global, can be applied either in annotation and retrieval problems, but in retrieval, the overwhelming majority of the results presented in literature are only based on per-tag averages, while for annotation it is common to use several averaging schemes. In annotation tasks, there is also the question of how the systems are evaluated: when a fixed number of tags is chosen, this puts a limit on the maximum achievable performances. These many aspects of autotagging are fairly well known to the MIR community, but make the evaluation more complex compared to genre or other single label classification problems. Furthermore, we will see next, that small alterations to the set of tags to be learned, e.g. considering just a subset of the most frequent tags, can significantly alter the performance, and conceal sub-optimal results, a fact that is often ignored in literature.

## 5.2 Experiments in Autotagging

In this section we analyze three aspects of the autotagging problem that, in our perspective, have only been superficially addressed in the literature (the results here presented were also published in [Marques et al., 2011a]). First we show that the autotagging task must be evaluated more carefully than it is usually done, and that small alterations to the set of tags to be learned can dramatically alter performance results and sometimes hide weaknesses. Second, through a series of tests on different data sets, we show that the generalization capacities of autotagging systems are poor and fail to learn similar tags from different origins. Third, we address the limitation of exploiting natural tag correlations, the rationale behind 2-stage system architectures [Aucouturier, 2009, Lin, 2008, Miotto et al., 2010, Ness et al., 2009, Pachet and Roy, 2009, Seyerlehner et al., 2010b], where in the second processing stage tag co-occurrences are modeled in order to "correct" the predictions done in the first stage.

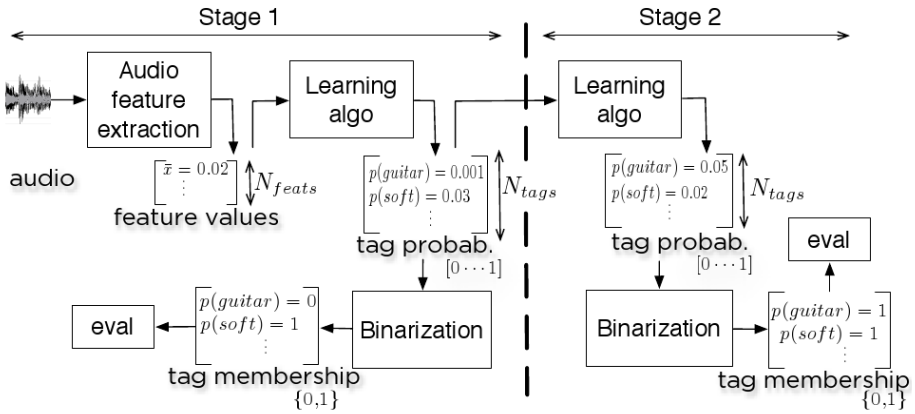



Figure 5.4: Generic 2-stage music autotagging framework (training of learning algorithms not represented; audio feature extraction can be statistics or time series).

Next, we introduce the setup used in our experiments, starting with the autotagging classification systems that we tested, the evaluation procedures and data sets used in our tests. We then proceed to the experiments we conducted on each of the three issues just mentioned.

### 5.2.1 Experimental Setup

**Tested Autotagging Systems:** We tested two types of models, VQMMs and SVMs, which were described in Chapter 3, and were used in the experiments on genre classification. However, in this tests the SVM architecture and input features used are quite different from the ones used in the previous chapter. We used two SVM systems, the first one (referred to as $\text{SVM}_\text{B}$) serves as a benchmark system. This is the method proposed in [Ness et al., 2009], which is available under the GNU Public License Version 2, in MARSYAS[3]. The second, referred as $\text{SVM}_2$ is an adaptation of the first one.

$\text{SVM}_\text{B}$: In this system, frame features are collapsed in a two steps process (texture windowing and computation of global mean and standard deviation) into a 64-dimensional feature vector for the whole audio excerpt [Ness et al., 2009]. This system implements an architecture with two stages of processing, illustrated in Figure 5.4. A multi-class SVM classifier is used in both stages. We report below on the performance of using just the first stage of processing alone, or the whole system. Note that performances of this benchmark system have also been reported in the 2010

[3]All the authors of [Marques et al., 2011a] are grateful to Ness and Tzanetakis for kindly providing and commenting the code used in experiments reported in the publication, and in this chapter.

MIREX evaluation[4].

$SVM_2$: This system is a 2-stage system similar to the $SVM_B$, with the difference that it externalizes the learning algorithm and directly uses the libSVM software package (http://www.csie.ntu.edu.tw/~cjlin/libsvm). Thus, we have more flexibility to setup the parameters for the learning algorithm than using the system in [Ness et al., 2009], and we chose to perform the data normalization via the libSVM package and not the MARSYAS code.

VQMM: The models are trained with the discrete version of the features. The features used in our experiments are the M17feat set described in Section 2.2.4. Two models were build for each tag, $\theta$, a positive one, $\mathcal{L}_\theta$, and a negative one, $\mathcal{L}_{\bar{\theta}}$. We used the likelihood functional to calculate the score $\mathcal{L}_\theta(\mathcal{S})$, of the sequence $\mathcal{S}$ for the tag $\theta$:

$$\mathcal{L}_\theta(\mathcal{S}) = \ln\left(P(s_1|\theta)\right) + \sum_{n=2}^{N} \ln\left(P(s_n|s_{n-1},\theta)\right) \qquad \text{(Eq. 3.11)}$$

and the negative score, $\mathcal{L}_{\bar{\theta}}(\mathcal{S})$ was calculated in a similar fashion. The tag was assigned to the sequence if $\mathcal{L}_\theta(\mathcal{S}) \geq \mathcal{L}_{\bar{\theta}}(\mathcal{S})$. The tests with the VQMMs were made with this positive/negative model pair, and with a fixed codebook size for all the tags. Empirically, we found that better results were obtained by using a codebook size of $k_2 = 50$ for the CAL500 data and $k_2 = 200$ for the MAGTAG5k set (in [Marques et al., 2011a], $k_2 = 200$ was used in all experiments).

The $SVM_B$ was chosen in order to have a fair point of comparison with the other tested methods: it represents a contribution towards solving the autotagging problem that rates among the best in the MIREX evaluation (2010)[5]. Furthermore, SVM models (and the 2-stage architectures) are a common choice of classifiers in autotagging problems, and collapsing the frame features into a single vector is the standard approach in this and other MIR tasks. Table 5.2 presents a comparison of the performance achieved with these three systems. In summary, we claim that the presented approaches are on par with the state-of-the-art as described in recent literature. In what concerns our approach, the VQMMs, their performance is also reported in Table 3.5 along with a more extensive list of published results, and in Tables B.4 and B.5 for annotation and retrieval tasks via ranking (as in [Bertin-Mahieux et al., 2008, Turnbull et al., 2008b] - with the results of this two works included for comparison).

[4]We strictly followed indications from the MARSYAS readme file for MIREX 2010 submission and used version 4178 of MARSYAS. Note that we checked that similar results were obtained with the latest MARSYAS version at the time of writing (i.e. 4418).

[5]see details on http://www.music-ir.org/mirex/wiki/2010:MIREX2010_Results

| | CAL500 | MAGTAG5K | MTT |
|---|---|---|---|
| $SVM_B$ | 0.452\|0.245 | 0.312\|0.083 | 0.347\|0.136 |
| $SVM_2$ | 0.464\|0.269 | 0.423\|0.176 | 0.347\|0.136 |
| VQMMs | 0.429\|0.299 | 0.411\|0.171 | 0.318\|0.158 |

Table 5.2: F-score$_g$ | F-score$_t$ for $SVM_B$, $SVM_2$, and VQMMs on the CAL500, MAGTAG5K, and the MTT data sets. For the CAL500 and the MTT sets, results were obtain via 2-fold cross validation, while for the MAGTAG5K it was with 3-fold cross validation. Note that the MTT is an unprocessed version of the MAGTAG5K data set, with no artist filter.

**Data Sets and Evaluation Methodologies:** For our experiments on the subject of autotagging, we considered mainly two data sets, the CAL500, the MAGTAG5K, although we also performed some tests on the Magnatagatune (MTT), and on two genre classification data sets previously utilized in this thesis: the ISMIR04 and the LMD (the general description of these data sets is presented in Appendix C). The CAL500 is one of the most referenced data sets in the context autotagging, and the MAGTAG5K is a cleaned and artist filtered version of the MTT, another very popular data set used for this task. The reason for using the MAGTAG5K instead of the full version is because the MTT reveals significant number of problems with annotations, such as misspelling (e.g. "rokc" instead of "rock"), impossible combination (e.g. "rock" and "no-rock" tags in the same song), nonsense values and so fourth, although only few papers consider these potential problems when reporting on autotagging experiments (these are discussed in greater detail in the appendix).

Another important issue inherent to autotagging systems is which evaluation methodology we should use to measure their performance. The most common methodology consists in splitting the data into training and test sets, using the first set to train an autotagging model and the second one to evaluate it. To ensure that the results of the evaluation are not sensitive for the particular training-test partitioning, we can use the $n$-fold cross validation methodology [Mitchell, 1997]. In the $n$-fold cross validation, the data are randomly partitioned into $n$ folds. For each fold, we use $n-1$ of those folds of data for training and the remaining for testing. Then, we can summarize the $n$ results using simple functions, as for example, mean and standard deviation [Mitchell, 1997]. However, it seldom takes into account artist filtering in the definition of the training and test datasets, a method whose importance has been demonstrated in music similarity research [Flexer, 2007] (over-optimistic results can be achieved when the same artists are present in both sets). Taking this additional factor into account, the evaluation methodology should agree with a number of constraints related to the statistics of the data, i.e. the number of folds should not be higher than the number of artists per tag, nor than the number of excerpts per tag. For instance, constraints from CAL500 favors a 2-fold cross-validation or holdout val-

| CAL500 | | | | MAGTAG5K | | | |
|---|---|---|---|---|---|---|---|
| Top Ten Tags | | Bottom Ten Tags | | Top Ten Tags | | Bottom Ten Tags | |
| Frq. | Tag | Frq. | Tag | Frq. | Tag | Frq. | Tag |
| 444 | Song.Recorded | 6 | Instrument.HarmonicaSolo | 1693 | guitars | 14 | woodwind |
| 339 | Instrument.MaleLeadVocals | 6 | Instrument.AcousticGuitarSolo | 1590 | classical | 13 | drone |
| 326 | Song.Texture.Electric | 6 | Vocals.Monotone | 1214 | slow | 13 | monks |
| 319 | NOT.Emotion.AngryAgressive | 6 | Vocals.Duet | 1160 | singing | 13 | soprano |
| 296 | NOT.Emotion.BizarreWeird | 6 | Instrument.Organ | 1072 | strings | 11 | clarinet |
| 287 | Song.Quality | 6 | Genre.CountryBlues | 927 | drums | 11 | scary |
| 278 | Song.Texture.Acoustic | 6 | Genre.Bebop | 797 | no.singing | 10 | no.beats |
| 275 | Instrument.DrumSet | 5 | Genre.BestSoul | 778 | electro | 9 | birds |
| 246 | NOT.Song.VeryDanceable | 5 | Usage.WithFamily | 758 | man.singing | 7 | reggae |
| 231 | Song.HighEnergy | 5 | Genre.Swing | 745 | fast | 5 | water |
| 23.26% of all attributions | | 0.44% of all attributions | | 41.98% of all attr. | | 0.41% of all attr. | |

Table 5.3: Top and bottom tag frequencies for the CAL500 and the MAGTAG5k data sets. The CAL500 consists of 502 songs annotated with a vocabulary of 174 tags, while the MAGTAG5k is comprised of 5259 songs annotated with a total of 137 tags. This table shows that both data sets have a strong class imbalance relative to the most and least frequent tags. Underneath each column is the percentage of the total number of tag attributions that corresponds to the top/bottom tag subsets.

idation (instead of 10-fold cross-validation [Bertin-Mahieux et al., 2008, Hoffman et al., 2009, Turnbull et al., 2008b]). We report results with the 2-fold cross validation (with a 50% split). In MAGTAG5k, some tags have few instances, from few artists (e.g. tag "water" has 16 songs from 6 artists). Thus, we chose to set the maximum number of folds to 3 (ensuring at least 2 different artists per tag per fold) and report on results with 3-fold cross-validation. We can clearly see in Table 5.2 that very different results are obtained when considering data and methodology issues discussed here and when not. To facilitate reproducible research, the whole MAGTAG5k data is available[6].

### 5.2.2 Experiment 1: Tag Distributions and Evaluation Scores

The purpose of this experiment is to show that autotagging evaluation measures can be substantially affected by the tag distributions and by the choice of subsets of tags used in testing situations. One common testing methodology is to remove the least frequent tags and estimate performances on a tag subset comprised of the most frequent tags. The motivation behind this approach has to do with the fact that the least frequent tags are associated with a very small

[6]Please follow this link: http://tl.di.fc.ul.pt/t/magtag5k.zip.

number of songs, making it hard to train models for these tags, specially if cross-validation training/testing is used. Disregarding the least frequent tags is done, e.g., in [Barrington et al., 2008, Coviello et al., 2011, Ellis et al., 2013, Foucard et al., 2011, Miotto et al., 2010, Ness et al., 2009, Yang et al., 2012], where results are compared with other published works on the same data sets, but more often than not, no attention is payed to the subset of tags chosen to estimate the systems performances. We show that this type of comparisons are flawed because removing the least frequent tags can significantly increase system performances, in contrast to using the full set of tags. Additionally, we also alert to the fact that over optimistic evaluation scores can be obtained with data with a high number of annotations per song, like the CAL500 set, or by excluding from the score averages tags that were not chosen during the classification process.

Music datasets typically have a strong imbalance in tag distributions, and results on a per-tag or global basis can differ significantly. For example the top 10 most frequent tags comprise $23\%$ of all annotations in CAL500 and more than $40\%$ in the MAGTAG5k, while the 10 least frequent tags correspond to less than $0.5\%$ of the annotations in both datasets (see Table 5.3). This imbalance drives global scores artificially high. The reason is simple: since the most common tags account for a large percentage of all annotations, classifiers that predict these tags well start off with high global scores. Figure 5.5 shows the F-scores on CAL500 and MAGTAG5k, when the most frequent tags (left) or the least frequent tags (right) are removed from the dataset. The purpose is to illustrate that removing the most and the least common tags can have a large impact on evaluation scores, independently of the classifier used. The figure shows that two conceptually distinct autotagging models, the $\text{SVM}_2$ and VQMMs, have a similar reaction to the removal process; this also applies to the $\text{SVM}_\text{B}$ (left out not to clutter the figure). Once again for clarity, we also excluded the precision and recall values, since they showed a similar behavior to tag removal as the F-score values (and the same can be said for per song averages). Results confirm the dependence of global scores on the most common tags (also noted in [Bertin-Mahieux et al., 2008, Miotto et al., 2010, Seyerlehner et al., 2010b, Turnbull et al., 2008b]): the left plot shows a sharp decrease in $\text{F-scores}_g$ when the top tags are removed ($\text{F-scores}_t$ also decrease, albeit relatively less). This indicates that the most frequent tags are on average better classified and have a substantial effect on the global performances. This is also seen in Figure 5.12, where the most common tags - the ones represented by larger circles - have high scores, and the least frequent tags low scores. The right plots of Figure 5.5 show how the two systems are affected by the removal of the least frequent tags. For the global scores, the behavior of the two systems is distinct: while, for the $\text{SVM}_2$ classifier, the least frequent tags have a small effect on the global scores, for the VQMMs the effect is more noticeable, particularly for the MAGTAG5k set. However, in relation to the per-tag scores, the removal of the least frequent tags has a dramatic impact on both systems. This also indicates that the

least frequent tags are, on average, worst classified and greatly affect the per-tag metrics. As previously said, removing the least frequent tags is a common practice in many works, but as shown with this experiment, direct comparisons between results obtained on the same data sets should not be made unless the same tag sub-set is also used, a fact that is most often ignored. For instance, in [Miotto et al., 2010] the evaluation on the CAL500 set is obtained by excluding the 77 least frequent tags. Inspecting the right plots, we can see that excluding around 80 least frequent tags translates in an increase above ten percentage points in the F-scores$_t$ for both systems and for both data sets.

The use of different tag sub sets invalidates comparisons between autotagging systems, but there are other issues that influence evaluation metrics. One is how thoroughly the songs are annotated. For example, the CAL500 set has a high number of tags per song (an average of 26 tags per song): a trivial classifier (i.e. always predicting all tags "on") has a precision of $\approx 15\%$ (with $100\%$ recall). This starting point yields a F-score$_t$ of $26\%$, which is misleadingly high. Note however, that autotagging systems that obtains F-scores$_p$ comparable to those of trivial classifiers does not necessarily mean that they are worthless. For trivial classifiers the F-scores$_g$ and the F-scores$_t$ have the same value, which for the F-scores$_g$ is much lower than most reports in literature, hence a good indicator of the system's sub-optimal performance. Another issue is how to account for the cases when a system does not recommend one or more tags. The consequence are undefined per-tag precision and F-scores values. In [Turnbull et al., 2008b] it is suggested to use for precision the empirical prior of the word, which is similar to using a "random" model to estimate the tag, while in [Zhao et al., 2010], the tags are excluded from the evaluation. From our experience, excluding from the evaluation tags that were not recommended can add non-negligible gains to performances, particularly in cross-validation testing scenarios with a high number of folds. This was observed in some preliminary tests we conducted in autotagging and that are reported in Table B.3 (Section B.2 of Appendix B), where we obtained a 5% point gain over replacing with zero the undefined scores - which experimentally, yielded practically the same results as the approach in [Turnbull et al., 2008b].

These non-obvious complexities that arise in autotagging scenarios can hinder comparisons between different methods and can also conceal sub-optimal performances. It is therefore important to report both per-tag and global scores, and ideally, also document how the individual tag performances are related to the a priori tag frequencies in the datasets used. Additionally, the fact that most frequent tags are, on average, better classified than least frequent ones also raises the question of what are autotagging systems really learning: are the models able to extract relevant musical concepts from the data or are they just learning the a priori tag probabilities. If indeed the systems are just learning the tags a priori distributions, then one would expect them to have a poor generalization capacity. This is the theme of the next set of experiments.

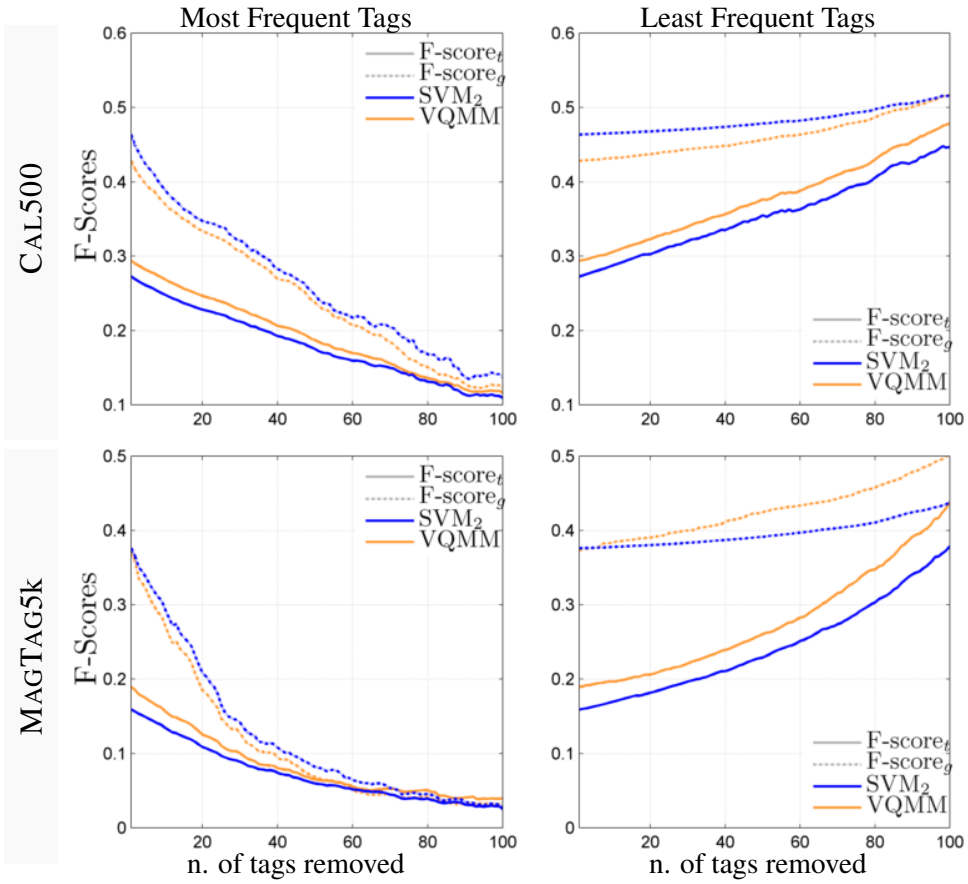



Figure 5.5: F-score$_g$ and F-score$_t$ on CAL500 for SVM$_2$ and VQMM autotaggers, as the most frequent (left) or the least frequent tags (right) are removed. The per-song scores were not included in these graphs because they are almost identical to the global scores, and therefore are redundant. The left plots show a sharp decrease in global scores when the most frequent tags are removed. Most frequent tags, which account for a significant percentage of all annotations in most data sets, can drive global and per-song averages artificially high. Therefore, per-tag averages are often preferred when reporting systems performances. However, an aspect widely ignored is the fact that removing the least frequent tags can also add a significant boost to the per-tag metrics, as seen in the right plot.

### 5.2.3 Experiment 2: Generalization Issues in Autotagging Systems

The objective of this experiment is to study autotagging models' ability to generalize. For this purpose, we selected songs annotated with 35 tags common to both MAGTAG5K and CAL500 (see Table 5.4), and evaluated how the systems perform when trained and tested with different sets. We also used in our tests two genre classification sets, the ISMIR04 and LMD, to analyze the models behavior in terms of tag annotation frequency. Our findings indicated that autotag-

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| (0.3\|2.5) | acoustic.guitar | (0.3\|1.0) | sax | (8.3\|2.4) | electro | (6.8\|5.8) | rock | (1.1\|11.8) | acoustic |
| (1.7\|6.9) | bass | (11.4\|0.8) | strings | (1.2\|1.3) | folk | (0.3\|2.0) | soft.rock | (4.4\|1.4) | ambient |
| (9.9\|1.9) | drums | (6.2\|4.2) | synth | (0.5\|0.5) | funk | (0.3\|0.9) | world | (0.2\|5.7) | happy |
| (1.0\|5.3) | electric.guitar | (0.3\|0.9) | trumpet | (0.2\|0.9) | hip.hop | (0.4\|0.3) | duet | (0.5\|4.6) | mellow |
| (0.5\|0.8) | horns | (6.2\|0.4) | violins | (1.6\|1.4) | jazz | (6.8\|3.8) | female.singing | (0.3\|2.5) | sad |
| (0.6\|0.3) | organ | (0.6\|1.0) | blues | (4.1\|3.1) | pop | (8.1\|14.4) | man.singing | (4.7\|4.4) | soft |
| (6.4\|3.6) | piano | (2.0\|1.4) | country | (0.4\|1.0) | punk | (0.3\|0.4) | talking | (2.4\|0.9) | weird |

Table 5.4: Shared tags (35 total) between the MAGTAG5k and the CAL500 data sets. The tags are divided into 5 main facets: instruments, genre, vocals, acoustic qualities, and emotion. In parenthesis are the tag percentages of the total number of annotations - left for the MAGTAG5k, and right for the CAL500. Songs without any tags were excluded from the MAGTAG5k data set, resulting in a total of 4548 songs (instead of the original 5259 songs). The number of songs in the CAL500 remained the same (502) since all the songs had at least one tag. For convenience, the tag names of the MAGTAG5k set were used since they are more concise.

ging techniques show very limited generalization capabilities. They also are far from the ideal of learning relevant musical concepts from the audio, independently of the data used in training (as long as it represents well enough the tag), and from being able to automatically tag sounds of different origins.

Figure 5.6 shows the F-scores for each of the 35 tags obtained by the three tested systems: $\text{SVM}_\text{B}$, $\text{SVM}_2$ and the VQMMs. The left plots show the F-scores obtained with CAL500 test set for the models either trained with the MAGTAG5k ($x$-axis) or with the CAL500 data ($y$-axis). The right plots show the results obtained with MAGTAG5k test set, for the systems trained with the CAL500 ($x$-axis) or with the MAGTAG5k ($y$-axis). When the training and testing was performed on the same data sets, we used a 2-fold cross validation for the CAL500 and a 3-fold for the MAGTAG5k. The results shown in the figures ($y$-axis values) are based on the averages of the test folds. When the testing set was different from the training one, the whole data set was used to estimate the F-scores ($x$-axis values). In the plots, each tag is represented by two circles, their size proportional to the tags percentage of total annotations in the data sets: blue for CAL500, orange for MAGTAG5k (the smaller circle is always drawn on top of the larger one). We realize that in Figure 5.6, the individual tags are hard to identify, specially when there is a high number of tags bundled together. The purpose of the plots is not to show the individual tag scores but rather transmit a general idea of how the overall performances are affected by training the models with data from different origins. On these plots a model that performs equally well when trained with either data set would be on the diagonal. Circles above the diagonal correspond to models that achieve a better performance when trained and tested on the same data, while circles below the diagonal are for models that perform better when trained

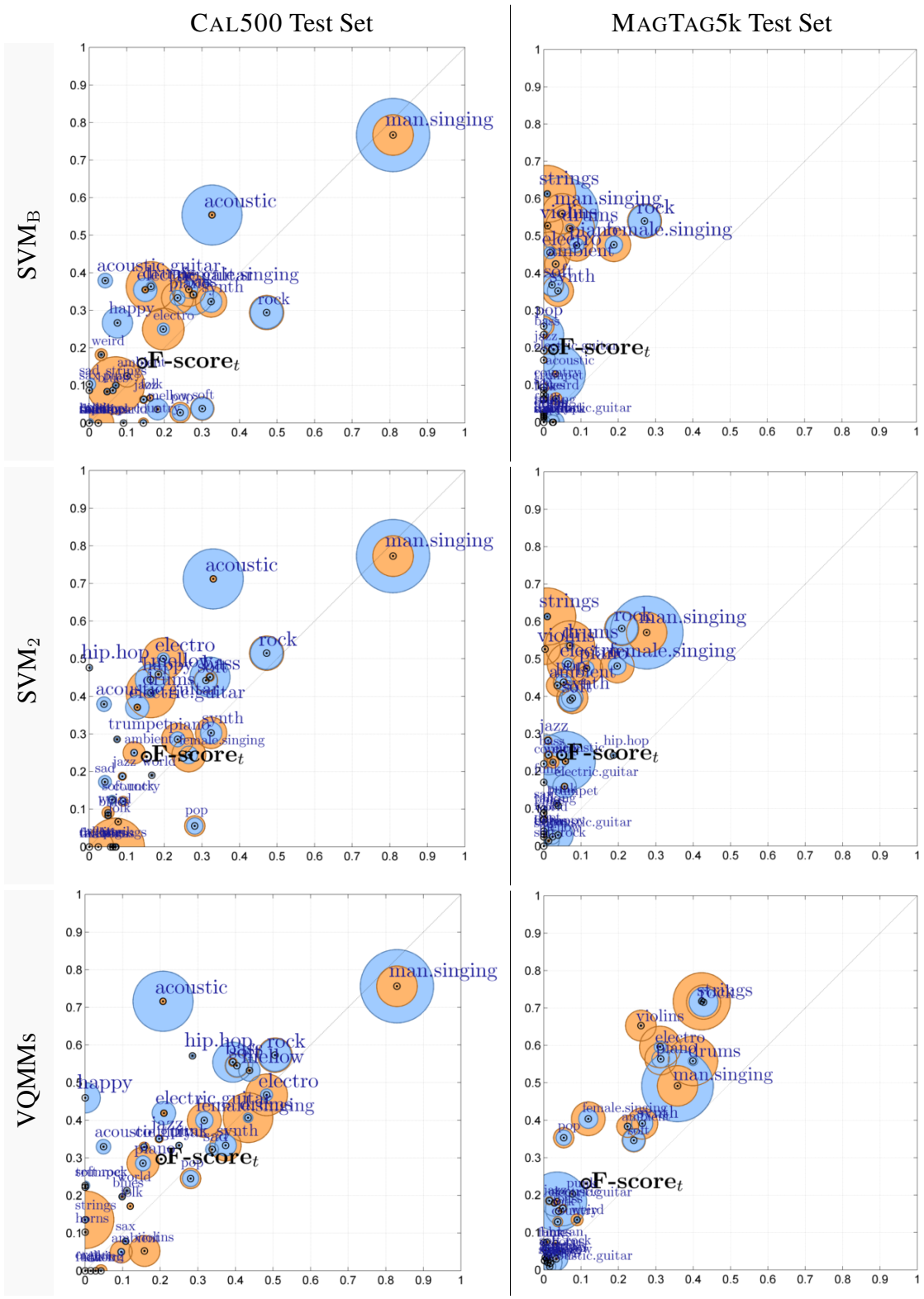



Figure 5.6: Generalization tests on a subset of 35 tags common to the CAL500 and the MAGTAG5k data sets (see Table 5.4). Three autotagging systems were tested: the $SVM_B$ (1st line), the $SVM_2$ (2nd line), and the VQMMs (3rd line). In the 1st column are the F-scores obtained by the 3 systems on the CAL500 *test* set, either trained with MAGTAG5k ($x$-axis) or CAL500 ($y$-axis). In the 2nd column are the F-scores for the MAGTAG5k *test* set, either trained with CAL500 ($x$-axis) or MAGTAG5k ($y$-axis) - see text for more details and comments. The size of the circles are proportional to the tag frequencies. Each tag has two circles, the blue color corresponding to the CAL500 tag frequencies and the orange to the MAGTAG5k - the smaller circle was drawn on top of the larger one for every tag. In black are the average per tag F-scores on the whole set of tags.

and tested on different data.

Looking at the plots for the CAL500 test set (left plots), there are a non-negligible number of tags close to the diagonal line, and a few lie in the lower triangular part of the plots. This means that these tags obtain similar scores trained with either sets and some even perform better when the models are trained with MAGTAG5k set. However very few tags exhibit simultaneously a good performance and generalization capacity. One exception is the tag *man.singing* that achieves high scores with all the three systems, and even performs slightly better when trained with the MAGTAG5k set (above 80% points). For models trained with MAGTAG5k, the tag *rock* had the second highest score on all three algorithms, but the values are significantly less than those obtained for *man.singing* tag (between 47% and 50%). Furthermore, there are only 11 tags for the $\text{SVM}_\text{B}$ that achieve an F-score above 10% on both trained versions of the model, 15 for the $\text{SVM}_2$, and 20 for the VQMMs. These numbers drop down to 3, 6, and 10 tags, for the $\text{SVM}_\text{B}$, $\text{SVM}_2$, and VQMMs respectively, when the F-score threshold is raised to 30% points.

The results pertaining to the MAGTAG5k test set (right plots) are quite different from the ones for the CAL500. For the SVM-based systems most tags are grouped along the $y$-axis, meaning that they obtain a relatively good score when trained and tested on the MAGTAG5k set but perform poorly when trained with CAL500. The VQMMs show a better generalization for a wider set of tags than the SVM counterparts, but still well bellow the results obtained with the CAL500 test set. For example, if we exclude tags that have an F-score bellow 10% points (on both trained versions of each system), then only 2 tags remain for the $\text{SVM}_\text{B}$ models (*female.singing* and *rock*), 5 tags with the $\text{SVM}_2$ (the previous two plus *man.singing*, *piano* and *hip.hop*), and 12 with the VQMMs. If the F-score threshold is raised to 30%, then only 6 tags remain with the VQMM models (*drums*, *piano*, *strings*, *electro*, *rock*, and *man.singing*), and none are left with the two SVM systems.

Some general comments can be made about the plots of Figure 5.6. One is that the majority of tags are above the diagonal, which means that better results are obtained when models are trained and tested on the same set. Another is that tag frequencies and model performances are correlated when models are trained and tested on the same data set, but not when different data sets are used. This is seen more clearly in Figure 5.7 which shows the same results as Figure 5.6, but each plot is separated in two, one for each training set. Additionally, there is also a noticeable difference in scores between test sets, which is not surprising since the two sets have very distinct characteristics [7]. For the CAL500, all three systems have a non-negligible portion of the tags positioned along (or close to) the diagonal of the plots. For the MAGTAG5k,

[7]The number of songs in the CAL500 is about ten time less that the MAGTAG5k, and for the subset of 35 tags, the CAL500 has, on average, 4.7 annotations per song while the MAGTAG5k has 2.0.

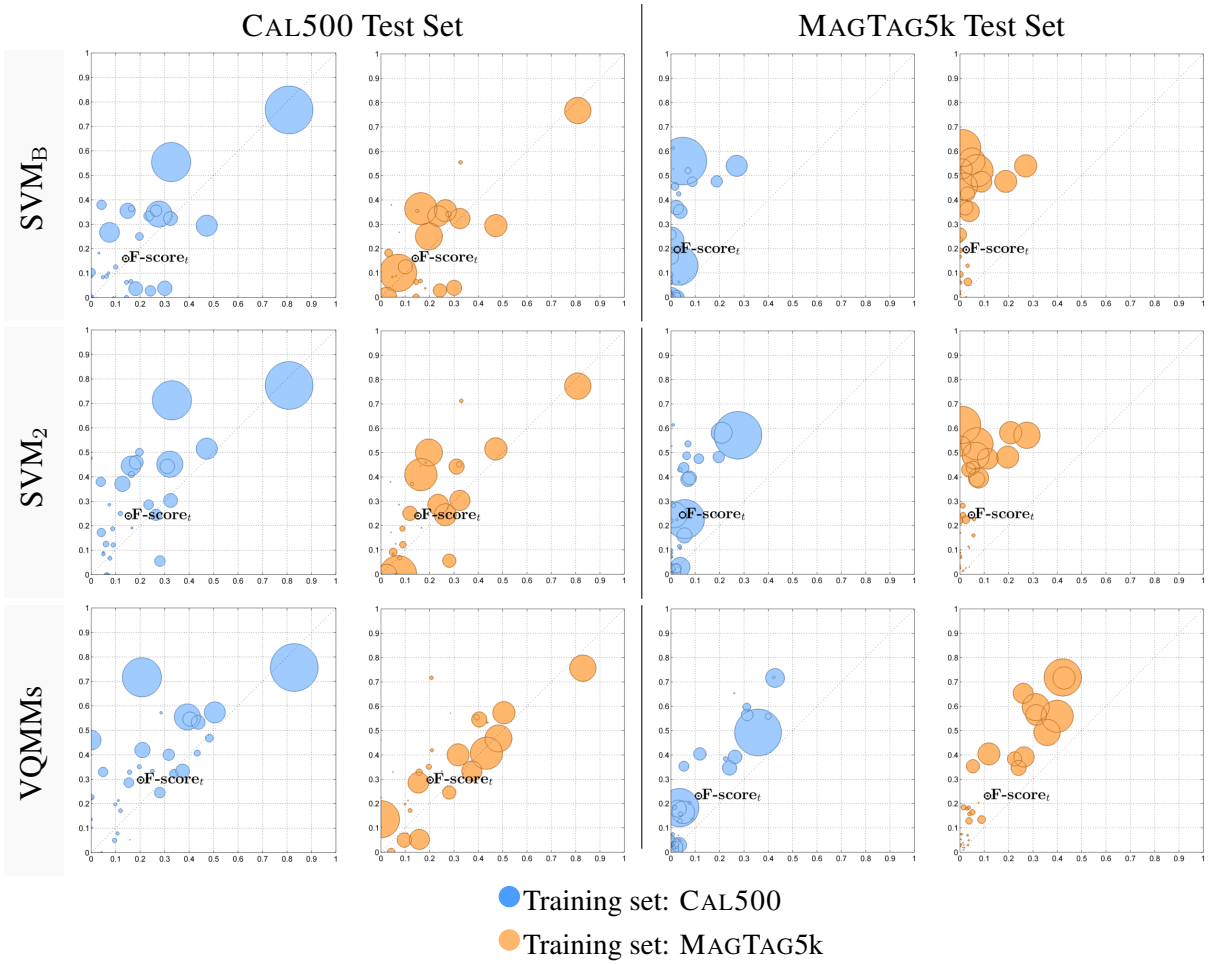



Figure 5.7: The same results on generalization test presented in Figure 5.6, with tag frequencies separated by data set: blue for tag models trained with the CAL500 set and orange for models trained with the MAGTAG5k set (i.e. each plot of Figure 5.6 is a superposition of two of the plots show here - tag names are omitted for clarity). When models are trained and tested on the same data set (1st and 4th columns), higher frequency tags are better classified than lower frequency ones, but when training and testing is performed on different data sets (2nd and 3th columns), no apparent correlation is noticeable between tag frequencies and F-scores.

the majority of tags are along the $y$-axis for the SVM models; for the VQMMs there is a high number of tags with low classification scores, and the remaining ones are positioned well above the diagonal line compared to the left side plot. Overall, the VQMMs show slightly better performances than the SVM counterparts, but the fact that higher scores are obtained when the same set is used for training and testing, or the fact that tag frequencies are correlated to performances, are indications that all the models have poor generalization capabilities. This is further corroborated by the next set of experiments.

Models were also tested on the LMD and the ISMIR04 genre classification data sets. These

two data sets were not created for autotagging tasks (no ground truth is available) so our analysis is based on tag assignment frequency. We processed the music pieces from these data sets with $SVM_2$ and VQMM models trained with CAL500 and MAGTAG5k. Figure 5.8 shows the proportion of songs from a given test set to which each tag was assigned by the $SVM_2$ models[8], while Figure 5.9 shows the results obtained with the VQMM models. Each color/shade corresponds to a training set and each row to a test set. For the $SVM_2$ models, we can see that when testing with CAL500 (first row) and training with MAGTAG5k (orange) nine tags are assigned to all songs. When testing with MAGTAG5k (second row), models trained with CAL500 (blue) recognize very few tags. When testing on LMD and ISMIR04 we observe a strange phenomenon: the proportion of music per tag is almost the same for both datasets and for all tags. This is also noticeable in Figure 5.10 where we show annotation frequencies on the ISMIR04 and the LMD sets for the $SVM_2$ and VQMM models trained with the MAGTAG5k set. The first row confirms what was seen on Figure 5.8: with $SVM_2$ the proportion of songs per tag is almost the same independently of the test set. Note that the LMD and ISMIR04 datasets have very distinctive characteristics that should be reflected in terms of classification into tags. When VQMMs are used, different anomalies are observed. When testing with the CAL500, and for VQMMs trained with either set, the tag annotation is done in a liberal fashion (i.e. tag assignments are excessively high). For the ISMIR04 and the LMD test sets, very few tags are recognized and these tags are over-represented. Moreover the same tags seem to be over-represented in both datasets (i.e. the tags *drums*, *synth*, *eletro*). When comparing the two rows of the plot, we can see that the two autotagging techniques have a very low level of agreement, for both test sets.

These experiments show that models obtained with autotagging techniques at the level of the state-of-the-art show very limited ability to generalize to new datasets and that the level of performance observed on a single finite dataset is somewhat misleading. Current autotagging techniques are still far from the long-term goal that is to allow automatic tagging of sounds independently of their origin.

### 5.2.4 Experiment 3: Exploiting Tag Correlations

Music tags are often correlated, and this is illustrated in Figure 5.11 where the tag correlation coefficients for the CAL500 and the MAGTAG5k data sets are shown. This is the rationale behind implementing a 2-stage architecture, where a second stage of processing, modeling tag co-occurrence relationships, can "correct" the imperfect tag predictions of the first stage (see illustration in Figure 5.4). Two rationales support this approach: (1) take advantage of correlation between labels, (2) use higher level features for classification. However, both rationales make

[8]The results obtained with the $SVM_2$ are similar to the ones obtained with $SVM_B$ (not included here).

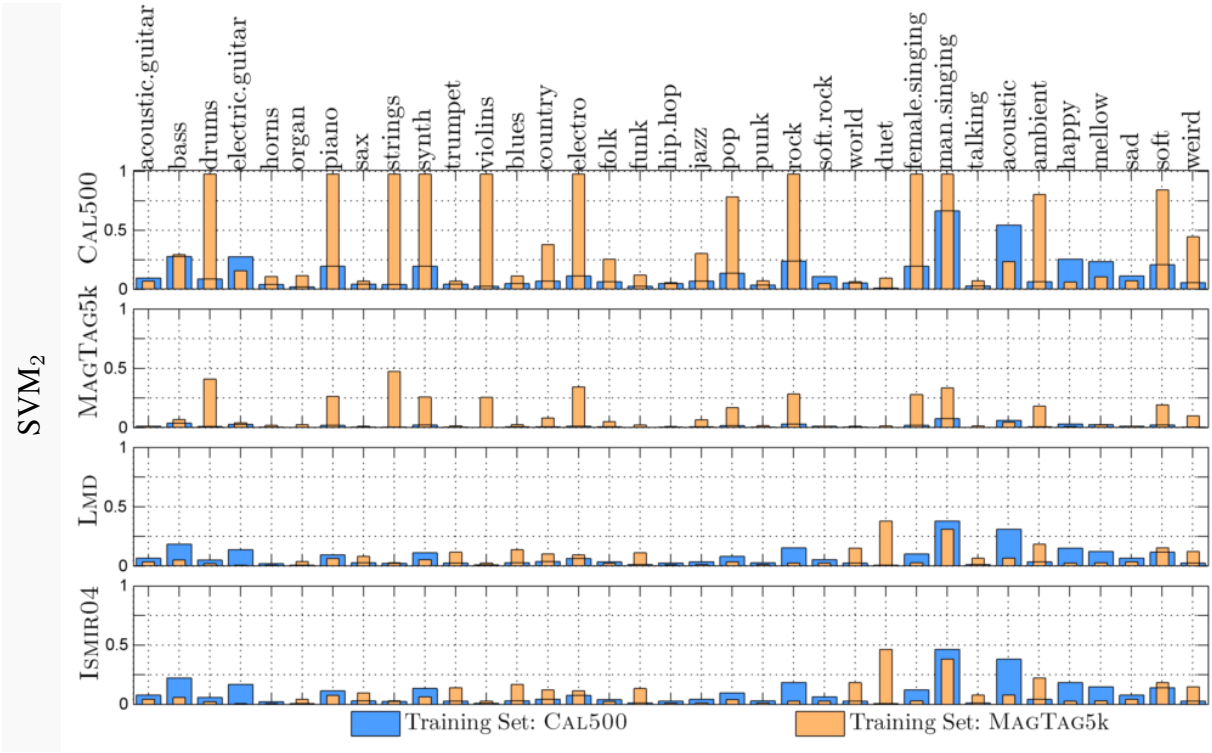



Figure 5.8: Proportion of music pieces for which each tag was assigned in the corresponding test set (rows). Models are trained using $SVM_B$ with two training sets CAL500 (blue) and MAGTAG5k (orange).

most sense when the first stage works satisfactorily. The influence of performance of first stage on performance of second stage is not always considered explicitly. A number of authors report on performance improvements with this procedure over the one-stage approach [Aucouturier, 2009, Miotto et al., 2010, Ness et al., 2009, Pachet and Roy, 2009, Seyerlehner et al., 2010b].

In Table 5.5, we compare $SVM_2$ against the Benchmark, considering either stage 1 only or both stages. The first column reports results on MAGTAG5k while the second reports results with the data and evaluation methodology from [Ness et al., 2009]: 2-fold over the whole MTT data, without artist filtering. Looking at results for the $SVM_B$, we can see that although results of the first stage (first row, second column) are very similar to those published in [Ness et al., 2009] (see Table 1 in that paper), the second stage in fact impairs results from the first stage only, i.e. the opposite phenomenon than [Ness et al., 2009]. Similar improvements for the second stage as those published can only be found when considering unadapted evaluation methodologies (e.g. no artist filter) and noisy and redundant data, as illustrated in the second column.

Results also show that the second stage of $SVM_2$ does appear to bring a small improvement on the first stage. However, we can gain more insights on the actual effect of the second stage by looking at Figure 5.12 which illustrates the difference in tag's individual F-scores between

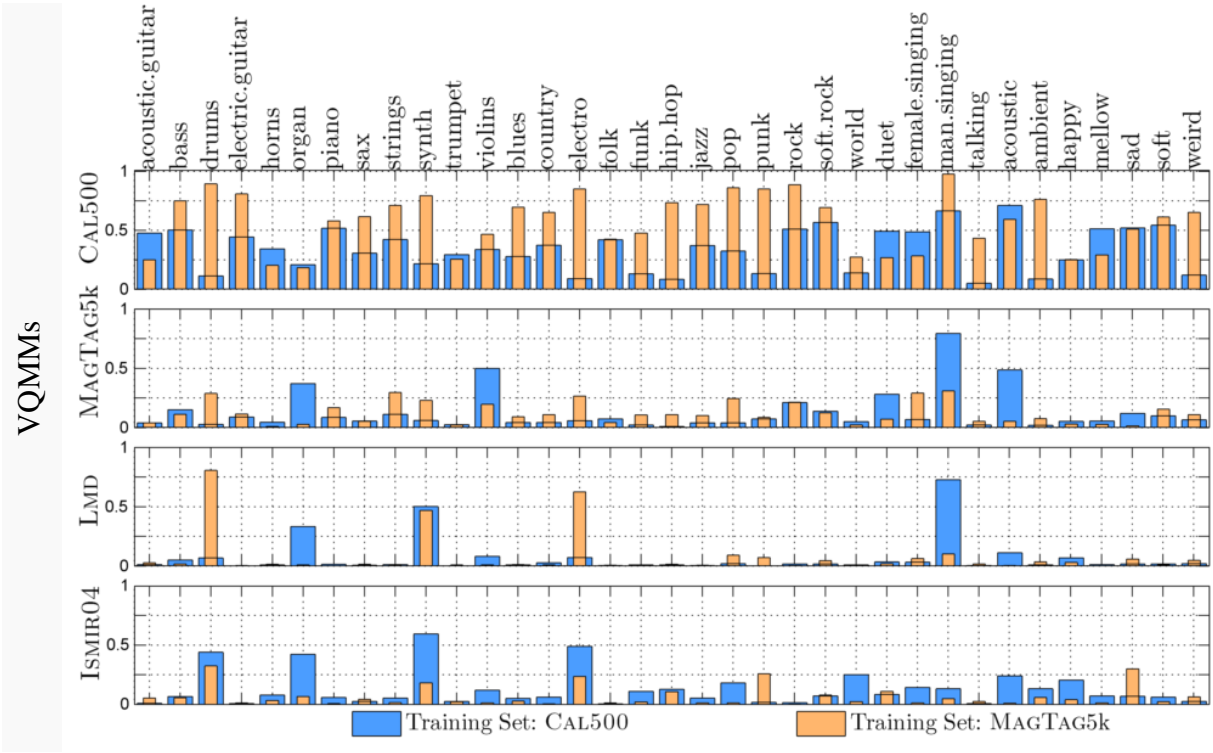



Figure 5.9: Proportion of music pieces for which each tag was assigned in the corresponding test set (rows). Models are trained using VQMMs with two training sets CAL500 (blue) and MAGTAG5k (orange).

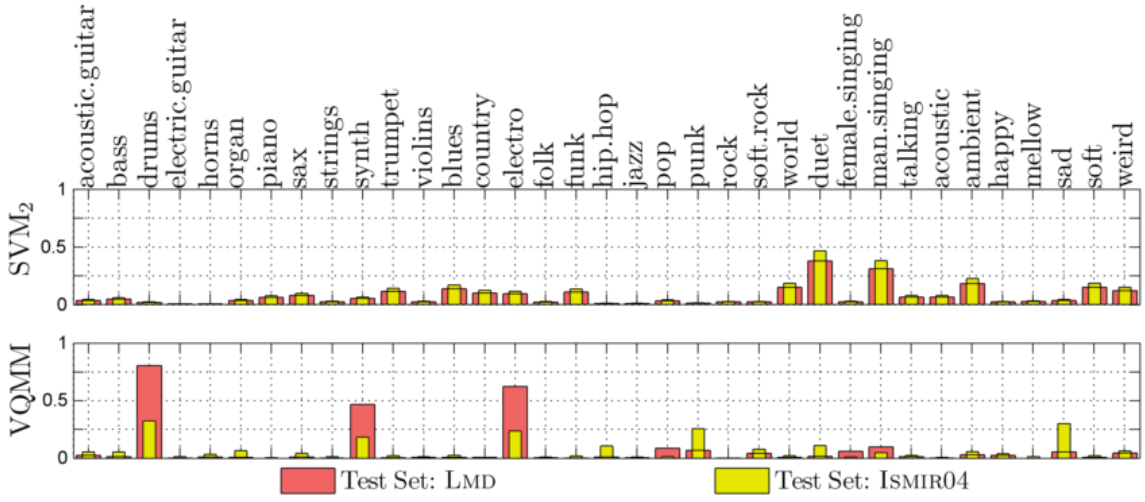



Figure 5.10: Proportion of music pieces for which each tag was assigned for two kinds of models (rows) and two test sets (colors). On tags for $SVM_2$ (top) and VQMMs (bottom) trained with the MAGTAG5k data set and tested with the LMD and with the ISMIR04.

using only one stage of processing vs using both stages. For a given data point (i.e. a particular tag) to lie above the diagonal means that the second stage improves results, while below the

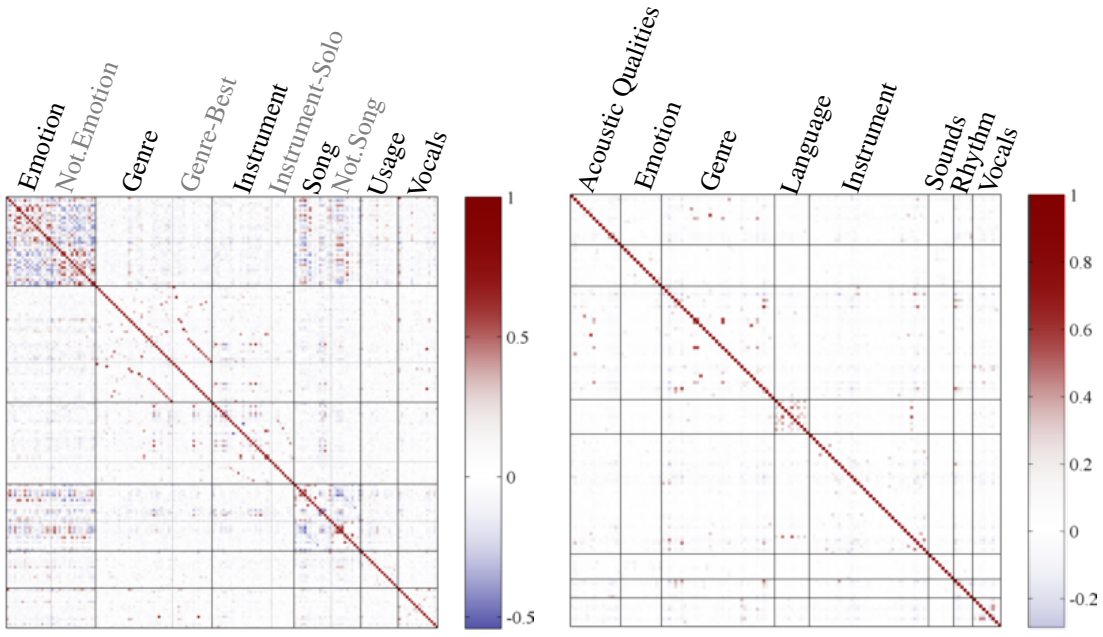



Figure 5.11: Correlation coefficient matrices for the CAL500 and the MAGTAG5k sets. The CAL500 data set has a much higher number of per-song annotation than the MAGTAG5k set, having the corresponding matrix larger areas of red and blue (positive and negative correlations).

| | MAGTAG5k | MTT |
|---|---|---|
| $SVM_B$ stage 1 | 0.409\|0.164 | 0.342\|0.126 |
| $SVM_B$ both stages | 0.312\|0.083 | 0.347\|0.136 |
| $SVM_2$ stage 1 | 0.411\|0.165 | 0.341\|0.127 |
| $SVM_2$ both stages | 0.423\|0.176 | 0.347\|0.136 |

Table 5.5: Comparison of F-score$_g$|F-score$_t$ for different configurations of the MTT dataset: MAGTAG5k, and 2-fold cross-validation over unprocessed MTT data set (no artist filter).

diagonal means impairing results from stage 1. For the $SVM_B$ (left plot), the decrease in overall performance can be seen on almost all tags individually. For $SVM_2$ (middle plot), if average results are better with both stages, we can see that not all tags are affected in the same way by the second stage: some improve (these are above the diagonal) while others do not. In our opinion, this distribution around both sides of the diagonal indicates that no clear pattern of improvement can be identified with the 2-stage procedure.

A possible reason for the inability of the system to take advantage of existing tag correlations may be because it is trained on data that only represents *estimations* of these correlations (and relatively bad ones, as indicated by the performance of stage 1). Hence we modified $SVM_2$

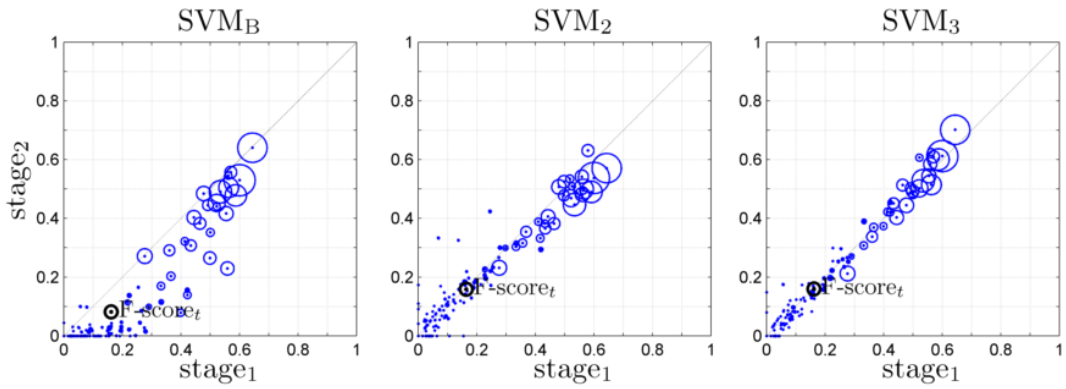



Figure 5.12: Performance of stage 1 vs both stages, MAGTAG5k. Individual tag F-scores are represented by circle centers. $x$-axis are the stage 1 F-scores, and $y$-axis both stages. Radii are proportional to corresponding tag frequencies.

with binary SVMs in stage 2, trained with *true* tag annotations instead of probability estimations from stage 1. We refer to this system as SVM$_3$. Overall, SVM$_3$ reaches F-score$_g$ and F-score$_t$ of 0.411 and 0.162, therefore slightly below the performance of SVM$_2$ and comparable to using only stage 1 (see table 5.5). However, when looking at the case of individual tags, i.e. rightmost plot of Figure 5.12, we can spot an interesting pattern: improvements with stage 2 seem higher for tags with better performance in stage 1. In other words, this seems to indicate that a minimum performance in stage 1 should be expected for a given tag (i.e. for its probability estimation) to be useful in a second stage. Although proving this claim will require more data, we wish to argue here that this pattern appears as a logical and desirable property for an autotagging system, and it indicates clear directions for future work: e.g. improving stage 1; tailoring stage 2 classifier to a selection of particular tags (e.g. the most reliable, the most "influential" [Aucouturier, 2009]) instead of processing all tags the same way.

## 5.3 Discussion

The experiments described here have shown that two diverse techniques, on par with the state-of-the art in music autotagging, fail to achieve their goal in several aspects. It was shown that autotagging tasks must be evaluated more carefully than what is usually done, that changing the set of tags or altering the evaluation measure (per tag vs global F-score) may dramatically alter the results, sometimes hiding weaknesses. It was also shown that these techniques fail the generalization test. Finally it was shown that the performance achieved with these techniques is not sufficient to be able to take advantage of the correlations between tags. Research in music genre classification and music similarity has seen recent progresses but its adaptation to autotagging shows severe drawbacks. What are the causes of these negative results?

We believe that there are several aspects of this complex problem that can hinder performances of autotagging systems. It is our opinion that more emphasis to some key differences between autotagging and genre classification should be given in autotagging research, in particular with reference to data recollection and annotation [Tingle et al., 2010]. Some tags correspond music facets that are more subjective than music genre tags (which are themselves subject to debate). They also can have multiple meanings, as in the case of Instrument tags: a song tagged "piano" can mean e.g. that piano is salient all over the song, or that there is a piano accompanying (but it may be relatively quiet), or that some parts have piano (but may have a short temporal span). In autotagging the procedure used to obtain ground truth differs from one data set to another, which results in a lack of consistency. Public data sets are limited in quantity and in many cases present errors or incompleteness. Also, where data sets for genre classification are usually limited to 10-20 genres, it is common to deal with hundreds of tags. This is not a problem per-se but in these conditions it is much more difficult to achieve good results for every tag and to follow good practices (artist filtering, $S$-fold cross validation). It is hard to build models based on extremely unbalanced data but it is even harder if the ground truth lacks consistency.

This results and previous observations lead us to propose some directions regarding future work in music autotagging: Different processing could be applied depending on categories of tags: (1) 2-stage architectures may be beneficial for some tags (e.g. tags with reasonable performance might help build models for other tags) but not for others (discussion in [Aucouturier, 2009] is also insightful on this matter). (2) Tag models could be differentiated according to temporal characteristics: models for tags that correspond to a short time span should be based on local features whereas tags that correspond to whole songs should use global features.

To favor reproducible research, the works described in the literature are based on closed data sets i.e. the training set and the test set are finite and fully described. If we could relax these constraints, we could, for example, build a model for a *saxophone* tag using *any* data. The public data set would then be used for evaluation only. We believe that the training data set is part of the model design and should not be constrained.

# Chapter 6

# Conclusions and Future Research Directions

## 6.1 Conclusions

In this dissertation, we presented a content-based approach for automatic music classification, label inference and music similarity estimation. The method is based on discrete representations of the audio features, which are obtained via vector quantization. This is done in an unsupervised fashion, meaning that at this stage there are no class labels associated with the data. The outcome of this process is a characterization of the music signals with sequences of discrete symbols. The classification models are based on a first order Markov process trained with the symbols transitions. Our algorithm contrast with the standard BoF approaches in one particular aspect: it explicitly models some temporal dynamics of the music signal – although we acknowledge that this is done at a very basic level. The fact that this simple time-dependent approach can capture some of the music signal dynamics, can be seen by the different patterns that emerge in the class-dependent transition matrices (e.g. in Figure 3.4). Our method was evaluated on multiple music collections, and to access its quality, we compared it to the state of the art. We paid special attention to genre and autotagging tasks. In these problems, and based on results obtained on publicly available data sets, our algorithm achieved performances comparable to many of the approaches found in literature.

However, part of the contributions in this work are not directly related to the approach we proposed. They arise from the studies we conducted on genre classification and autotagging, and our findings also cast some doubts on the effectiveness of the VQMMs and other algorithms, and their capacity of inferring high level musical concepts from the audio in a meaningful and useful way.

In the context of genre classification, we analyzed the influence of diverse partitions of the feature space on the performances of several classification systems. Our experiments indicated that there is no apparent benefit in seeking thorough representation of the features space.

In the context of autotagging we showed that performance measures can be influenced by the choice of tags used for evaluation, and that scores can be inflated when the least frequent tags are omitted. Furthermore, the systems we tested showed poor generalization capacities, and were not able to take advantage of the correlation between tags.

## 6.2 Future Work

We have mentioned in this dissertation that some noticeable advances in content-based classification and similarity estimation were made at a feature representation level (i.e. using block-level features) instead of improvements at an algorithmic level. Throughout our experiments, we predominantly used low-level timbre related features. One natural course of action for future experiments is to test the VQMMs with block-level features as inputs. This is possible since these type of features also produce a sequence of vectors, which is essential for our method.

Another modification to our approach has to do with codebook sizes used in the vector quantization process. Our tests showed that the codebook size can influence the VQMMs performances (see Table 3.3). Furthermore, we also noted this behavior at a tag level, where certain tags are better estimated with small codebooks and others with large ones. Based on this observations, we devised a strategy that is equivalent to automatically adapting the codebook size for each tag. The strategy is presented next (note that the strategy is still "on paper" and we need to implement it and test it, to find out if it is beneficial).

1. The audio descriptors are quantized with a large codebook (e.g. $k_2 = 1000$). This yields a symbol set with same cardinality as the size of the codebook.
2. The number of symbols (i.e. centroids) are reduced by another clustering step, this time using the centroids as new data points.
3. This process is repeated several times to produce symbol sets of decreasing cardinality. Keep track of which symbols were merged during the each clustering step. This way, the music signals are represented by several symbol sequences of different cardinality, but note that the quantization of the audio features is performed only once.
4. To find out which symbol set is best suited for a given tag, a validation approach is used in the training phase of the Markov models: part of the training data is used only to measure the models performance (e.g. via F-scores$_t$)

5. Keep the symbol-set/model that has the highest validation scores and use it for testing.

We conclude this thesis with another research direction we plan to pursuit: the study of hubs. Our work on this subject is in a more advanced stage than the ones related to the VQMMs, and the results obtained so far are present next.

## Hubs and Other Effects of High-Dimensional Spaces

The formation of hubs is a phenomenon that occurs in high dimensional spaces, where certain data points keep appearing as nearest neighbors of large number of other points. This is a consequence of the property of *distance concentration*: one of the many aspects of the "curse of dimensionality", where pairwise point distances become approximately equal in high dimensions, rendering nearest neighbor searches meaningless. Hubs have an obvious negative impact in retrieval or recommendation systems: the hubs are often the unwanted first choices, while other objects remain "orphans" and are never selected. This undesirable effect has been observed in many application areas, such as image [Hicklin et al., 2005], text mining [Radovanović et al., 2010], and biometrics [Yager and Dunstone, 2010]. In MIR, hubs have been reported early on [Aucouturier and Pachet, 2004, 2007, Berenzweig, 2007, Pampalk, 2006b], but its only recently, that the MIR community is discovering the full implications that hubs have on audio-based similarity methods [Flexer et al., 2010, 2012, Karydis et al., 2010, Radovanović et al., 2010, Schnitzer et al., 2011, 2012]. Hubs are not an artifact but rather a property of high dimensional spaces, and the latter contributions indicate that they are one of the culprits for the "glass ceiling" performance limit in similarity systems. Kardis *et.al.* [Karydis et al., 2010] reported that hubs were responsible for up to $90\%$ of errors in a genre classification experiment, in [Flexer et al., 2012], hub-songs were judged by human evaluators to be less perceptually meaningful that non-hub ones, and in [Aucouturier and Pachet, 2007, Schnitzer et al., 2011], using a probabilistic distance metric that mitigates the hub problem, the performance gain in genre classification was on average $5\%$ points on several music databases. This works have shown that, at an algorithmic level, hubs are detrimental to a wide range of similarity methods. This has been observed in a wide range of distinct models [Aucouturier and Pachet, 2007, Flexer et al., 2012], and for different features parametrizations, such as fluctuations patterns, onset patterns, and others [Pampalk et al., 2005, Pohle et al., 2009, Seyerlehner et al., 2010a, West et al., 2006], where the authors report higher immunity hub levels compared to standard spectral-based features.

In this section, we address the problem of hubs and other pathologies associated with high dimensional spaces. Our objective is to measure the impact of high dimensional data on classification methods, in particular audio based ones. We focused our attention on ideal classification

scenarios, where points from the same class are grouped together in a tight cluster, and clusters from different classes positioned far away from each other[1]. In this situations, the classification process is trivialized, and simple methods suffice, such as centroid distances or k-nearest neighbors. We want to understand if, in high dimensions, simple classification problems can be solved by this trivial methods or if the pathologies associated with distances metrics render the classification processes meaningless. In a mathematical context, extensive studies have been made on the perplexing behavior of distance measurements in high dimensions (see, for instance [Aggarwal, 2001, Aggarwal et al., 2001, Ahn et al., 2007, Ertoz et al., 2002, Hall et al., 2005, Ververidis and Kotropoulos, 2009] and references within), and in a MIR based perspective, the subject has also been investigated [Aucouturier and Pachet, 2007, Flexer et al., 2010, Karydis et al., 2010, Radovanović et al., 2010, Schnitzer et al., 2011]. Our research confirms the results presented in these works, namely the evolution of point distance distributions in terms of dimensionality for academic cases (e.g. isotropic Gaussian data), and in terms of hubness measures for real data (e.g. MFCCs). In some of our tests, however, we observed a few unexpected results that, to the best of our knowledge, have not been reported in literature and may contribute to understand the odd behaviors of point distances in high dimensions. The unforeseen results involve tests with synthetically generated data with a mixture of Gaussians densities. In term of data dimensionality, the evolution of the distance distributions with this sets have two distinct modes: they either rapidly converge to the case of zero mean, unit variance white Gaussian noise, or exhibit multi-modal distributions as the data dimensionality increases.

Our studies on this subject are still on-going, and so far, we have only focused on the behavior of independent and identically distributed data in high dimensional spaces. Our analysis consists of two sets of experiments. The first one deals with synthetically generated data and the second experiment concerns real audio features, namely the all-purpose MFCCs. We analyze the two experiments in terms of hubness indicators and point distributions. We discuss the obtained results and propose directions for future studies on these issues. Next we briefly present the hubness indicators used in our tests.

### Hubness and Measures

The measures are used for evaluation purposes in our experiments used to measure "hubness" were inspired from previous works on the subject [Aucouturier and Pachet, 2007, Flexer et al., 2012, Karydis et al., 2010, Schnitzer et al., 2011]. The measures are based on the concept of $k$-occurrences, $N_k(\mathbf{x})$, which is defined next:

[1]This principle behind the Fisher criterion, which tries to maximize the inter-class scatter, and minimize the intra-class variances.

- ***k*-occurrences** $N_k$
  For a finite set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, of $N$ vectors $\mathbf{x} \in \mathbb{R}^d$, the number of $k$-occurrences $N_k(\mathbf{x})$, is the number of times the vector $\mathbf{x}$ appears in the $k$ nearest neighbor list of other vectors in the set. In our experiments, we used $k = 5$.

- **Number of orphans** $N_k = 0$
  The number of orphans is the number of objects with $N_k = 0$: objects that are never selected as nearest neighbors of other objects.

- **Hub counts** $H_m$
  Number of vectors with $k$-occurrences greater than $m$: $N_k \geq m$. In our experiments we measured hub counts with $m = \{10, 15, 25\}$.

**Experiment 1: Synthetic Data**

**Synthetic Data Sets:** In our tests with synthetic data, we used three readily distinguishable distributions: isotropic Gaussian data, and a mixture of three and four Gaussian models (from now on referred as Mo3G and Mo4G). We used the first distribution, isotropic Gaussian, as a reference point in our analysis, and because many works dealing with the subject of hubs and high dimensional spaces report on empirical and theoretical results with this distribution. The other two synthetic sets, the mixture of Gaussians, were tailored for classification, since they fit the profile of ideal scenarios, where each class is represented by a well defined cluster. Furthermore, the true class-densities are known, and optimal probabilistic models are easily implemented and tested since they have a simple closed form solution. The main discriminative elements in the mixture of Gaussians sets lie in the first two dimensions: the means of the individual Gaussians. The 2-dimensional scatter plots are represented in the first column of Figure 6.1, and for all of the three cases are zero mean and unit variance. For the isotropic Gaussian distribution, the data was generated in the same fashion for all dimensions, but for the Mo3G and Mo4G sets, we used two different ways to generated data values for $d \geq 3$:

Case 1: The data values are drawn from a zero mean, unit variance, Gaussian noise (2$^{\text{nd}}$ column of Figure 6.1). With this configuration, we want to study classification in the context of adding irrelevant information to extra data dimensions.

Case 2: The data values are drawn from a zero mean Gaussians with variances dependent on the cluster the data belongs to - 3$^{\text{rd}}$ column of Figure 6.1. Note that for the Mo4G set, the extra dimensions are zero mean iid, Gaussian noise with standard deviation $\sigma = ¼$, and therefore this situation is similar to the first case, since no relevant information is added. For the Mo3G set, the extra dimensions are zero mean, Gaussian noise with

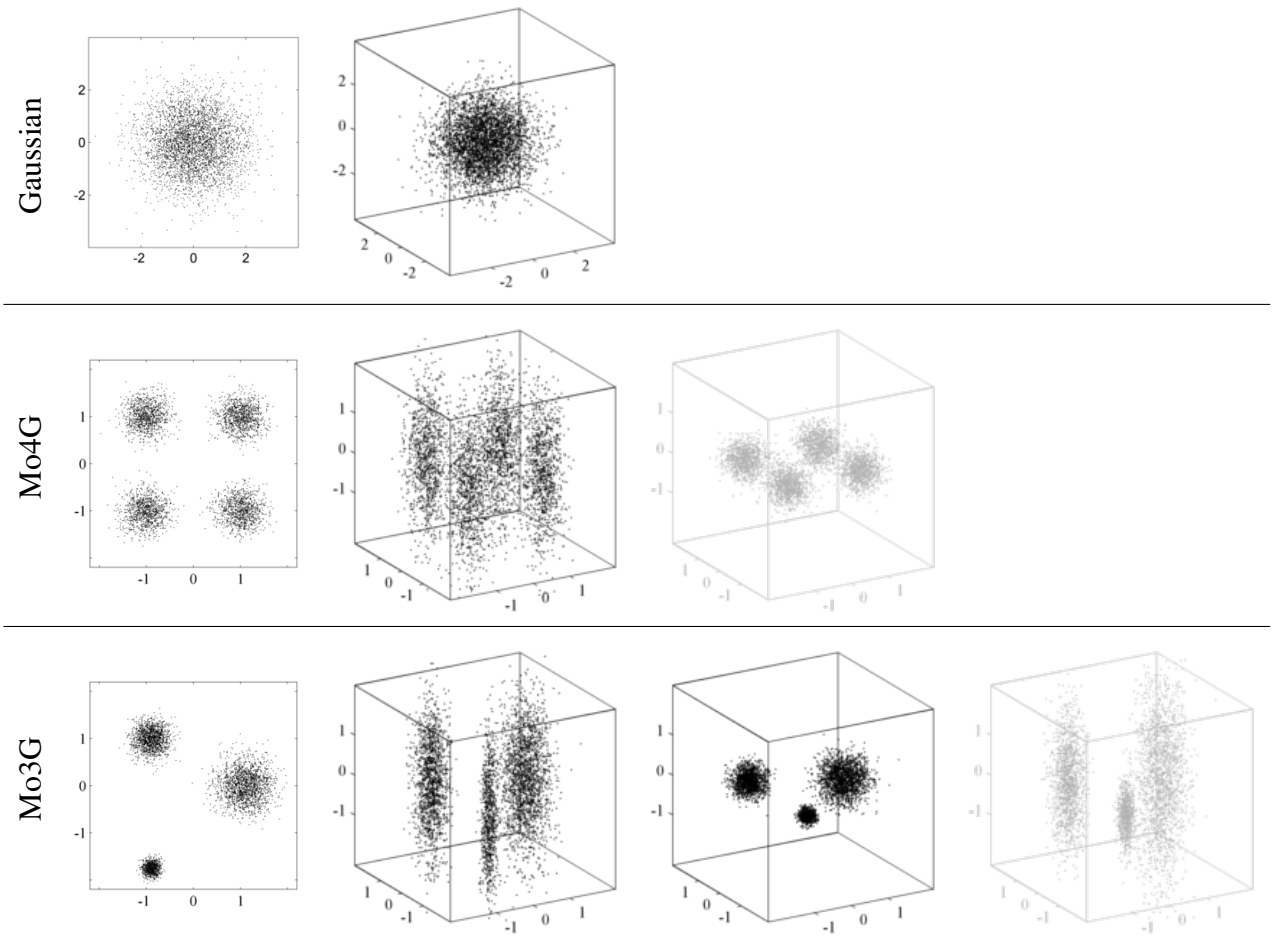



Figure 6.1: Three synthetic data sets (lines) used in our experiments. The first set is composed (for all dimensions) of zero mean, unit variance, white Gaussian noise. The other two sets are a mixture of four Gaussian (Mo4G) and of three (Mo3G) Gaussians. In the first column are the 2-dimensional scatter plots of a set of 4000 points. For the Mo4G the clusters have the same prior probability, $p = ¼$, and the same covariance matrix $\Sigma = \sigma^2\mathbf{I}$ with $\sigma = ¼$ (where $\mathbf{I}$ is the identity matrix), and for the Mo3G, the prior probabilities are $p_{1,2,3} = \{0.45, 0.35, 0.2\}$ and with covariance matrices $\Sigma_i = \sigma_i^2\mathbf{I}$, with $\sigma_{1,2,3} = \{0.3, 0.2, 0.1\}$. For the Gaussian mixture models, and for data dimensions $d \geq 3$, the data was generated in different ways (see text for a more detailed explanation). **Case 1** ($2^{\text{nd}}$ column): the extra dimensions (third and above) are zero mean, unit variance Gaussian noise. **Case 2** ($3^{\text{rd}}$ column): the extra dimensions are zero mean Gaussians, with the same variances of the individual clusters. Note that for the Mo4G set , the difference between the two distributions boils down to a scale factor for $d \geq 3$. Thus there is no change in the information contents in the data - to differentiate this situation we used a gray color for the scatter plots. For testing purposes, we also included a scaled version of the Mo3G set (Case 2), with unit variance in every dimension - lower right plot. Once again there is no change in information contents.

three different variances: the same variances as the individual Gaussians in 2D. This requires, in the data generation process, to know which Gaussian each point belongs to, and add the noise with the according variance. Therefore, and unlike the Mo4G, for the Mo3G case there is some extra information present in the data for $d \geq 3$, though the main discriminative elements lie in the first two data dimensions. We also include a scale version of the Mo3G with unit variance in every dimension. This was done in order to test the effects of data scaling. Note that this normalization is simply a scaling constant, since the individual data variances remain constant for $d \geq 3$, and the first two data dimensions are unaffected since they already have unit variance.

**Experiments:** In high dimensions, points tend to distribute themselves along the surface of a hypersphere centered at the data mean. The property of distance concentration can be generalized to any other reference point, and by induction, the distance between pairs of points becomes approximately equal in high dimensions. However, in low dimensions, pairwise distances can have very characteristic distributions. How do these distributions evolve as the dimensionality increases? Can we get a sense above what dimension the data become distributed in a hypersphere? Next, we analyze how pairwise distances, $\ell_2$ and cosine, vector norms (distances to the origin) evolve in relation to the data dimensions, for three described distributions: isotropic Gaussian, the Mo3G and Mo4G. We start with the Case 1 data, where the extra dimensions are simply Gaussian noise. Figure 6.2 shows the distributions of the vector norms ($2^{\text{nd}}$ column), and pairwise point distances ($3^{\text{rd}}$ column), for data dimensions of $d = \{2, 5, 10, 30, 50, 100\}$. The plots for the norm distributions were based on a set of $10^5$ iid samples, while for the pairwise ones, we used a smaller subset of $4000$ samples[2] - the ones in the scatter plots. For Gaussian random vectors, the distribution of distances is theoretically known ($\chi^2$ or non-central $\chi^2$-distributions), and the first line in Figure 6.2 shows the evolution of these distributions for increasing dimensionality values. The results are as expected: at lower dimensions, the shape of the $\chi^2$-distribution is more prominent than in higher dimensions, where they converge to a normal distribution[3]. For the other two distributions, their shape in low-dimensions are quite different from the Gaussian data, in particular for the 2D case. For the Mo4G set the Euclidean norm has a sharply peaked distribution, because all points are approximately equidistant from the origin. For the Mo3G, the points from the more compact Gaussian are further away from the mean than the remaining points, showing the peak on the right hand side of the 2D distribution. As for pairwise distances, the shape of the distributions for the mixture of Gaussians reflects the fact that only a fraction of the points - the ones belonging to the same Gaussian - are close

[2]Note, nevertheless, that $4000$ samples result in approximately 8 million different pairwise measures.

[3]The $\chi^2$-distribution is the sum of $d$ independent random variables, and due to the central limit theorem, it converges to a normal distribution for sufficiently large values of $d$.

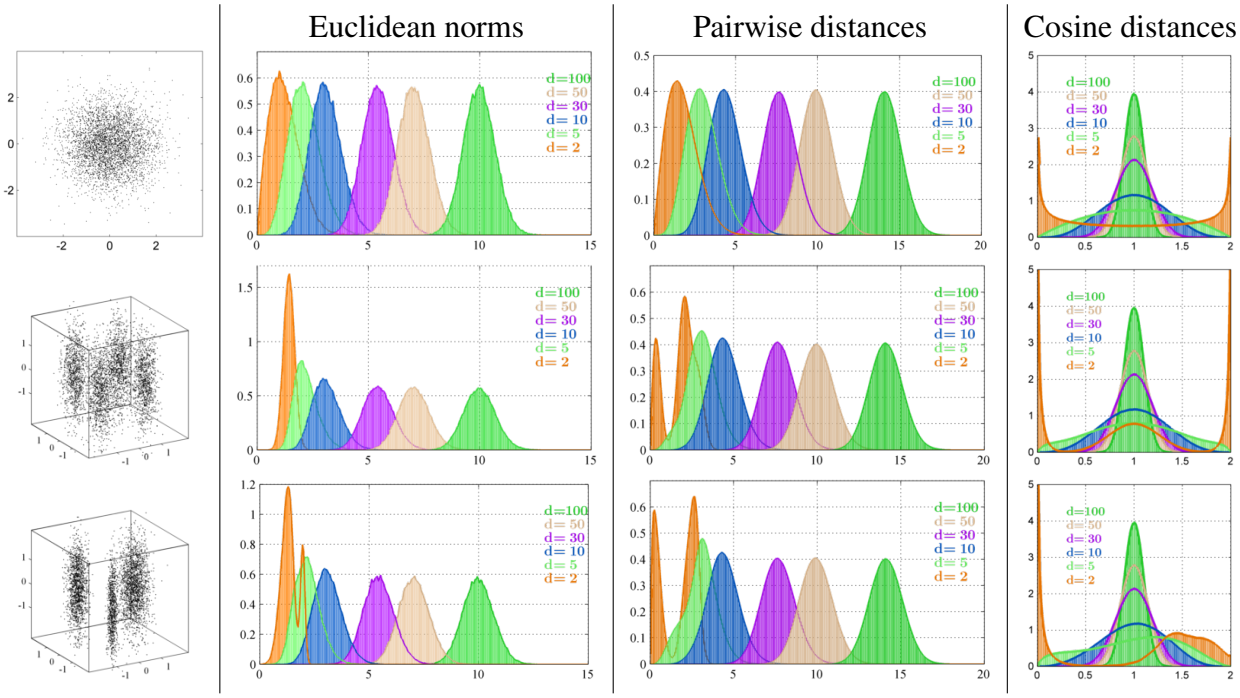
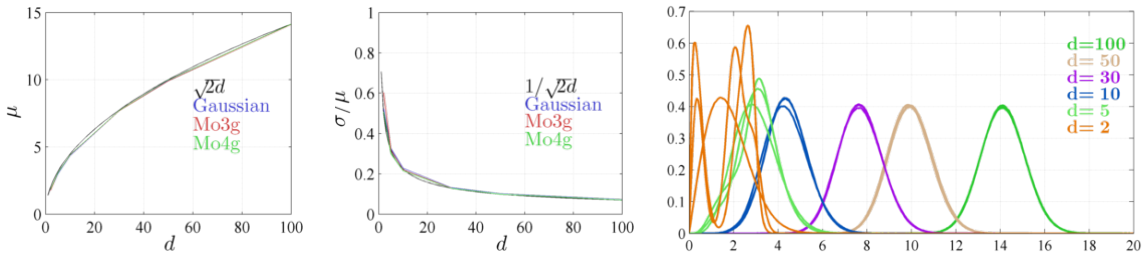



Figure 6.2: **Synthetic Data, Case 1.** Evolution of the distance distributions for different number of dimensions. 1st column: scatter plot of the data for 2-dimensional case. 2nd column: Euclidean norms . 3rd column: pairwise distances. 4th column: cosine distances.

to each other while the rest is far apart, thus the bi-modal shape for the 2D cases. The cosine distances also reflect the multi-modal shapes of the mixture distributions. The points belonging to the same Gaussian are responsible for the peaks around $0$ in both distributions. For the Mo4G case, and for a given cluster, the points in two other clusters are approximately orthogonal - thus the bump around $1$ - and the points remaining cluster are at a $180^{\text{o}}$ degree angle from the first ones - thus the other peak around $2$. For the Mo3G case, the angles between points in different clusters vary considerably, due to the specific cluster arrangement and individual cluster variances - thus the wide bump between $[1, 2]$. The evolution of the distance distributions as the data dimensionality increases show what was expected[4]: the average distance to the data mean grows according to $\sqrt{d}$, and the mean of pairwise distances according to $\sqrt{2d}$, since a significant number of pairwise values are around twice the average distance to the data mean. In Figure 6.3, the plots a) and b) illustrate the property of concentration of distances, as the data dimensionality increases, in the three previous examples. The first one represents the evolution of the pairwise distance means and the second the evolution of the ratio between the standard deviations and the means. The empirical results are practically indistinguishable from

[4]For Gaussian data, and for $\ell_2$ distance norms, it has been proven that, as the number of dimensions $d$ increases, the average distance to the data mean grows according $\sqrt{d}$, and the variance around the mean remains constant – this is also valid for other iid distributions and many $\ell_p$ norms [Aggarwal et al., 2001].




a) b) c)

Figure 6.3: **Synthetic Data, Case 1.** For the three examples in Figure 6.2, a) evolution the pairwise distance means, b) distance concentrations - ratio between the distribution's standard deviation and mean, c) superimposed pairwise distance distributions.

the theoretical ones, and similar behavior was also observed for the vector norms (distances to the origin). The less foreseen outcome is how fast different distributions converge to somewhat similar shapes in higher dimensions. In part c), the results for the pairwise distributions for the three examples are superimposed (the $3^{\text{rd}}$ column of Figure 6.2); for dimensions $d \geq 10$ the three distributions are almost identical. There are many works on the behavior and geometric representations of high dimensional an fixed sample size data sets (see for instance, [Aggarwal, 2001, Aggarwal et al., 2001, Ahn et al., 2007, Hall et al., 2005]). The studies in [Hall et al., 2005] showed that, as the dimensionality increases, each data vector becomes approximately located on the vertices of a regular simplex. This essentially means that the geometrical data structure becomes deterministic and the variables become almost independent : the randomness of the data lie in random rotations of this simplex [Ahn et al., 2007]. The cosine distances shown in Figure 6.2 also corroborate this finding since most of the data vectors become orthogonal - thus the sharp peak around 1 in the distributions. This simple example illustrates the problems that one may encounter when dealing with "high" dimensional data, in particularly, that even for moderate data dimensionality values ( i.e. $d \geq 10$), distances distributions become the same for all sets. As the data dimensions increase, the distances become concentrated around a given mean with a Gaussian distribution; this is an indication that point distances become diluted, and for the cases of the Mo3G and Mo4G sets, it is also harder to automatically estimate the individual Gaussians. This is evidenced in Figure 6.4, where the pairwise distances matrices are shown for the two sets and for data with different dimensions. For clarity, the data points have been ordered sequentially according to their Gaussian memberships. For $d$=2, the dark squares along the matrices diagonals reflect the obvious, the points within each Gaussian are closer to themselves than the ones from other Gaussians, but for $d \geq 10$ the patterns start to fade away.

Let us now look at the evolution of the distance distributions for the Case 2 where for dimensions $d \geq 3$ the data is generated from zero mean Gaussians with variances depending on the clusters they belong. Figure 6.5 shows this evolution for several data dimensions. The

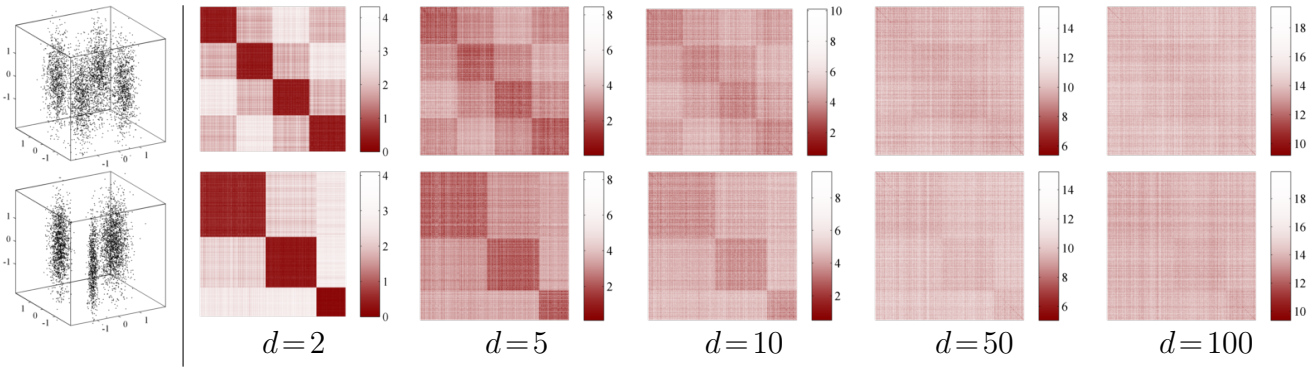



Figure 6.4: **Synthetic Data, Case 1.** Distance matrices for the Mo4G and Mo3G synthetic data (2D plots in the first column, 4000 points for each set) for several data dimension values (columns 2 to 6). The data has unit variance in all the dimensions, but only the first two dimensions carry relevant information, while the others (for $d \geq 3$) are zero mean, unit variance, white Gaussian noise (the same sets used in Figure 6.2).

last two columns of Figure 6.5 represent the pairwise distance matrices for data dimensions $d = \{2, 2000\}$, and for a set of $4000$ points. Note that the data sparsity is extremely high for $d = 2000$, since the number of data points is only twice the number of dimensions. The first line refers to the Mo3G, the other two are the Mo3G and the Mo4G; we made a distinction between the first and the last two sets, because for the last two, we already reported results on the same type of distributions (with different data scales). Before addressing the effects of scaling, let us analyze the behavior of the distance distributions for the Mo3G set. As the dimensions grow, the distance distribution becomes multi-modal with six clearly distinguishable levels: the distances from points within each Gaussian, and the points distances from a given Gaussian to the other two - thus the six color shades in the distance matrix for $d = 2000$. This matrix also shows that the points belonging to the Gaussian with the largest variance are further away from themselves than they are from the points of the other two Gaussians. This is also true for the points belonging to the Gaussian with the second largest variance; these points are closer to the ones from the more compact distribution than they are to themselves. From inspecting the distance matrices, we found that for $d = 100$ the points from the wider Gaussian are equidistant from all others, and for higher dimensions the patterns present in Figure 6.5 start to emerge.

The behavior of the distributions in the last two lines (scale versions of the Mo3G and Mo4G sets) is expected. For the Mo3G set (second line) the normalization changes the location and variances of the modes found in the first line, but the patterns found in the shape of the distribution and in the distance matrix still remain salient in high dimensions. A similar observation can be made about the Mo4G (third line), which resembles the previous example (see Figure 6.2, second line), although the mean distances now grow approximately four times

slower[5].

In our perspective, these experiments show some interesting aspects on the behavior of points distances in high dimensions. In the Mo3G and Mo4G sets, the pattern present in the distances matrices in low dimensions are a reflection of the multi-modality nature of the data. For Case 1, and for $d \geq 3$, the data is unit variance, white Gaussian noise, and the dimensions do not have to be very high ($d \geq 10$) for point distances to concentrate around a given mean, and for the distance matrices to become blurred (see Figure 6.4). The discriminative information in the data is only present in the first two dimensions, and adding noise to the extra dimensions seems to dilute this information. As a consequence, the well defined clusters in the 2-dimensional data become harder to identify in higher dimensions. For Case 2 data, the Mo3G distribution has some information present in dimensions $d \geq 3$, and the behavior of point distances contrasts with Case 1. The points from the two clusters with greater variances are further away from themselves than they are to the points in the cluster with the smallest variance. This unexpected behavior also makes it hard to estimate clusters in high dimensions, and can have an obvious negative impact on classification if we consider each cluster as a class.

**Experiment 2: Real Data**

**Audio Features:** The audio descriptors used in our experiments were the all purpose MFCCs. In order to study the effects of high dimensions, we made some modifications on the way theses coefficients are usually calculated. We used a 100 Mel filter bank to obtain 100 MFCC coefficients. Note that in this case, the MFCCs are just a decorrelated version of the log-power of the signal in each Mel band, and there is no loss of information in the Discrete Cosine transformation. The audio was obtained from excerpts from the ISMIR04 data set.

**Experiments:** In this section we repeat the tests we did on synthetic data with real audio features. In our experiments we used MFCCs, obtained from randomly sampling the features in ISMIR04 data set. We conducted our tests with two feature configurations: either we used the raw values of the MFCCs or we normalized the data so each dimension has zero mean and unit variance. The MFCCs features are decorrelated (via DCT), and typically the first few tens of coefficients contain relevant information, while the others are predominantly noise. When the data dimensions are normalized, the distribution of higher MFCC coefficients is close to a zero mean, unit variance, Gaussian. This is comparable, up to a certain degree, to the tests on the Case 1 synthetic data where for $d \geq 3$, the added values were white noise. For this case we observed that tight clusters present for $d = 2$ vanish in higher dimensions, and that

[5]The mean distances are actually a bit higher than $\frac{\sqrt{2d}}{4}$, since the first two dimensions have unit variance

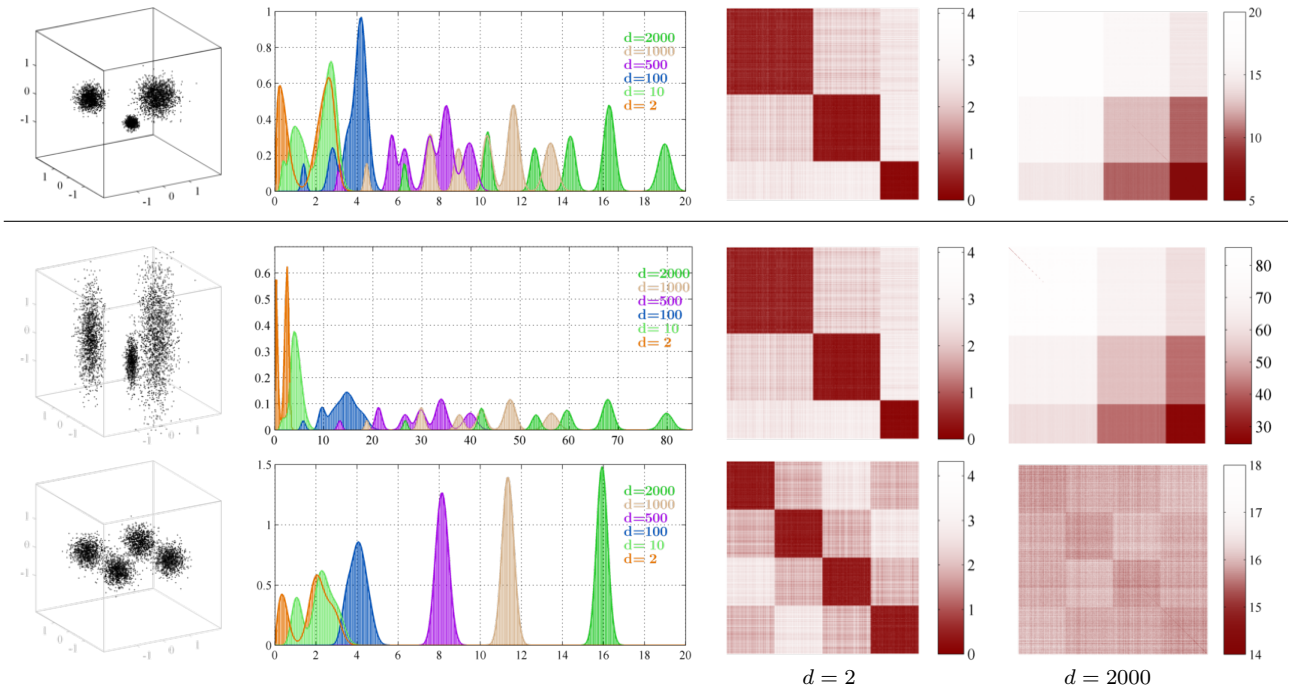



Figure 6.5: **Synthetic Data, Case 2.** First Column: "3-dimensional" scatter plot of the Mo3G and Mo4G sets. Second Column: pairwise distances. Columns 3 and 4: pairwise distance matrices for data dimensions $d = 2$ and $d = 2000$. In the first line are the results of the Mo3G (Case 2), and in the last two lines, the unit variance, scaled version of the Mo3G and Mo4G data (see text for details).

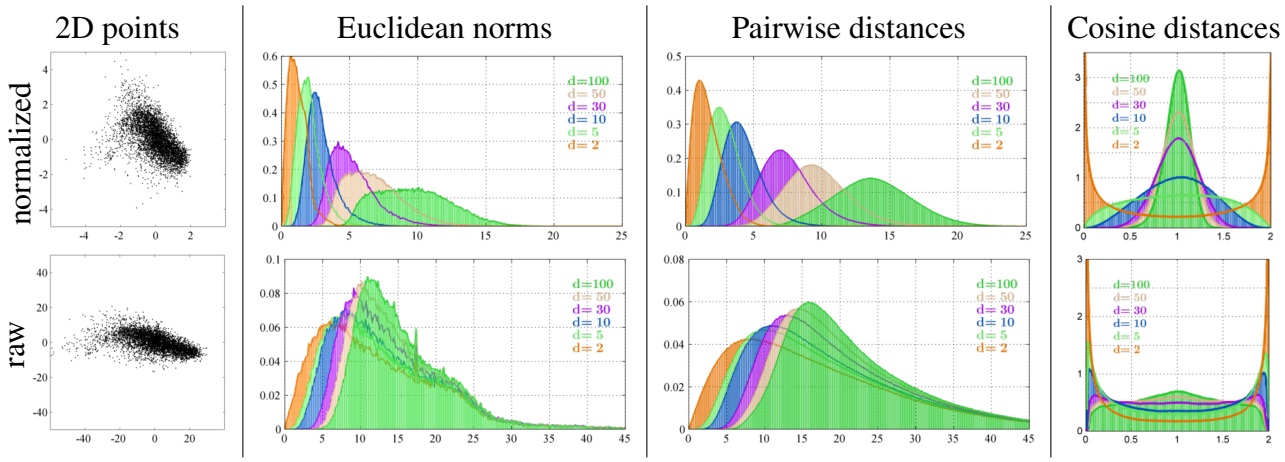



Figure 6.6: **Real Data, MFCCs.** Evolution of the distance distributions (relative to the mean (e.i. origin) - 2nd column, and pairwise distances- 3rd column), for different number of dimensions. The first column corresponds to the scatter plots of the distributions for the 2-dimensional case. The first line is relative to normalized MFCC data (e.i. zero mean, unit variance for all dimensions), and the second to the unprocessed coefficients.

white noise enhances the concentration of points distances, compared to Case 2 data. This effect is also felt, but to a lesser extent, with MFCC data. Figure 6.6 shows the evolution of the distance distributions for normalized MFCCs (1st line) and raw MFCCs (2nd line). For the normalized MFCCs, as the dimensions increase the pairwise distances tend to Gaussian shaped distributions, and the cosine distances show that a significant portions of the features vectors are orthogonal to each other (thus the large peak around 1). For the raw MFCCs, the results are very different from the ones found with normalized features. The evolution of the pairwise distances is slow and the shape of the distributions is far from Gaussian. The two sharp peaks found in the cosine distributions for $d = 2$ (at 0 and 2, meaning that most of the feature vectors are parallel to each other) are visible up to $d = 10$, and for high dimensions shape is approximately uniform - the peak found in the previous case only starts to appear for $d = 100$. Also note that for the normalized data, although the concentration of distances is felt much faster than in raw MFCCs, it is still slower than synthetic data (Case 1 - Figure 6.2).

### Hubness Indicators

So far we have addressed the impact of data dimensionality on point distance distributions. In this section we look at the previous experiments in terms of hubness indicators, and analyze how this measures evolve as the data dimensionality grows. We use the same synthetic and real sets[6], and except for Figure 6.7, all other results are based on pairwise distances of random

[6]Like in the previous experiments all the feature sets, synthetic and real, are composed of $5 \times 10^4$ iid samples.

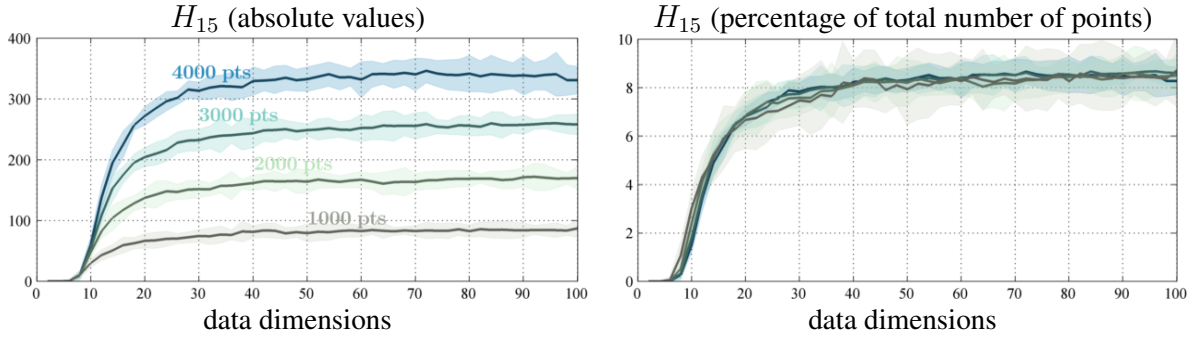



Figure 6.7: Euclidean distance, hub count statistics, $H_{15}$ for different data dimensions in terms of number of points (left plot) in terms of percentage of total number of points in the set (right plot). The data is generated according to the Mo3G (Case 2) distribution, and hub counts were estimated with four sets of different size. The curves are the mean values of ten test runs, and the shaded regions are plus and minus one standard deviation. The right plot shows, that as data dimensionality increases, the number of points, $\mathbf{x}$, with $N_k \geq 15$ comprise approximately 8% of the total data.

sub sets of $4 \times 10^3$ points. In order to obtain statistically sound results, this sampling process was repeated ten times in all the experiments, and hubness indicators were also re-calculated. We made a small modification to the way we estimated the hub counts $H_m$ (number of vectors with $N_k \geq m$), and the number of orphans (number of vectors with $N_k = 0$): instead of using absolute values (as in [Flexer et al., 2010, Karydis et al., 2010, Radovanović et al., 2010]), we list our results in terms of sample percentages of the whole data set (e.i. these measures are normalized by the total number of points). This way the hub and orphan counts become (more or less) independent of the number of points in the set. This is exemplified in Figure 6.7, where the left plot shows the absolute $H_{15}$ values for the Mo3G distribution, calculated with data sub sets with a different number of samples, and in the right plot are the same results normalized by the number of samples in the sets - which to us, are more intuitive.

The evolution of the hub counts $H_m$ and the number of orphans, $N_k = 0$ (in percentage of the total number of points) are represented in the top two plots of Figure 6.8, for the Euclidean distances. The figure shows the hubness indicators for four of the sets previously used in our tests: Gaussian (green), Mo3G - Case 2 with unit variance (blue), MFCCs raw (red) and normalized MFCCs (orange). We chose to use the Mo3G set with unit variance due to its characteristic behavior in terms of the evolution of hub and orphan counts. For the Mo3G data, there is a sudden increase in hub and orphan counts not observed in the other sets. The reason for this increase is due to the fact that the points in the two clusters with the largest variances become orphans as the dimensionality increases, while the ones in the tightest cluster become hubs - this is seen in Figure 6.9 (left top-three scatter plots). The evolution of the pairwise dis-

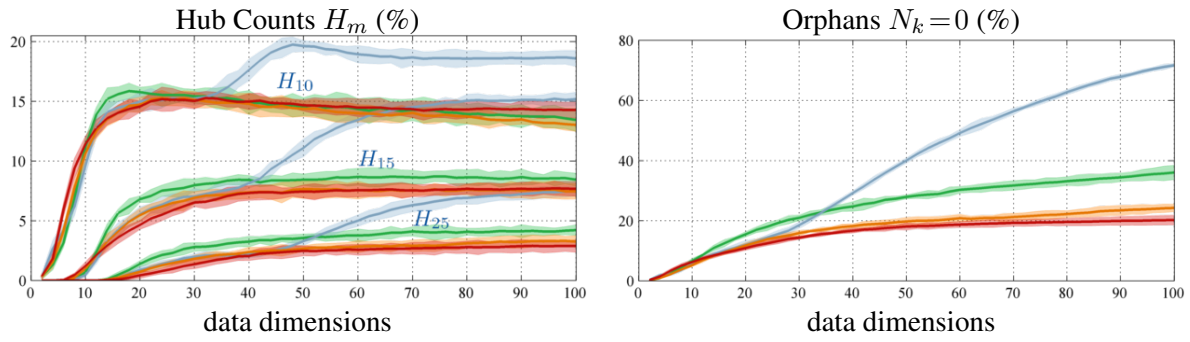



Figure 6.8: Euclidean distance, hubness measures - hub counts, number of orphans, skew of the k-occurrences distribution, and distance concentrations - for several data dimension values ($x$-axis). The different colors correspond to different data sets: Gaussian set, Mo3G set (Case 2 - normalized variance), MFCCs (raw), MFCCs (normalized). The curves are the mean values of ten runs, and the shaded areas correspond to plus or minus one standard deviations.

tances, depicted in Figure 6.5, already showed that in high dimensions the points from the two largest clusters are further away from themselves than they are to the ones in the tightest cluster, and therefore it is not surprising that these points become orphans, while the others become hubs. By inspection, we also observed a similar phenomenon with the non-normalized Mo3G distances, but the process happened at a much slower pace than the normalized Mo3G data. Moreover, looking at the hubness indicators, such as the $H_{15}$ counts of Figure 6.7, they do not exhibit the sudden increases present in Figure 6.8. This is a bit surprising since the data used in both tests is the same, apart from a scaling factor. This is also true for the raw and normalized MFCC features (although in this case, each data dimension was scaled with a different value). The normalization process also affects hub and orphan locations in point distributions of real data as seen in the right scatter plots of Figure 6.9. Note though, that hub and orphan counts in both sets are not very different (red and orange curves of Figure 6.8). The MFCCs curves also show that real data seems more immune to hubs and orphans than synthetic data: the orphan count is significantly less than the Gaussian case, and hubs with a high number of neighbors are less frequent.

**Discussion**

So far, our studies on hubs, orphans and the effects of high dimensionality on point distances have been limited to i.i.d. data. This is the initial part of a more general analysis that we intend to make on how high dimensional spaces affect classification techniques, particularly the ones tailored for audio-based classification. With that objective in mind, we tested synthetic data that was generated to fit the profile of ideal (trivial) classification conditions: the Mo3G and the

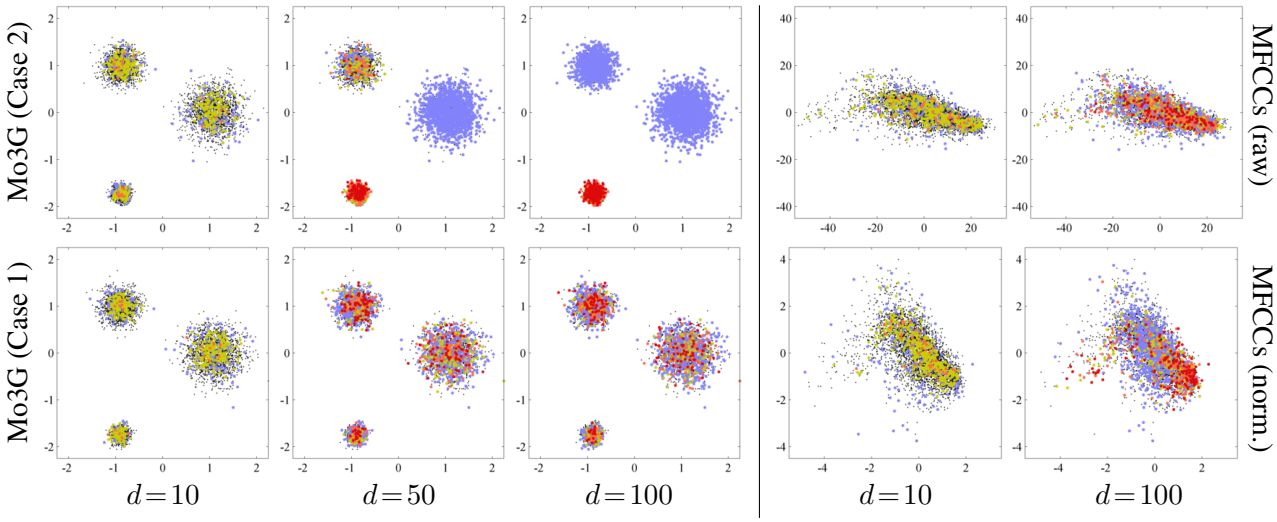



Figure 6.9: Scatter plots of hubs and orphans: • $H_{25}$, • $H_{15}$, • $H_{10}$, • $N_k = 0$ (black points are neither hubs or orphans). The left three plots are for the Mo3G sets: first line corresponds to the Case 2 with unit variance, and the second to the Case 1 set. The two right plots are for real data, in first line are the raw MFCCs, while in the second line are the normalized MFCCs.

Mo4G sets, where points are grouped in easily discernible and well separated clusters (at least in low dimensions). We tested two ways of growing the data dimensionality. The first was to simply add white Gaussian noise to data dimensions greater than 2. In this situation, distance distributions concentrate quite fast (for $d \geq 10$ as seen in Figure 6.4) and the patterns present in pairwise distance matrices for low data dimensions become diluted. The second way of increasing the data dimensionality, and this is only applicable to the Mo3G set, was to add Gaussian noise with different variances, depending on the cluster memberships. With this setting, points belonging the two clusters with largest variance are further apart from themselves than to points belonging to the cluster with the smallest variance. For both cases, the behavior of point distances in high dimensions raises doubts about the effectiveness of classifications methods, but further tests are need. For the synthetically generated data the probabilities density functions are known and therefor optimal maximum a posteriori classifiers can be built. We plan to test this ideal classifiers and compare the results with models trained in a supervised fashion. For real audio data, we believe that we can learn about the hub problem by first analyzing simple and controlled cases, instead of jumping directly to genre classification or music similarity evaluations as in [Flexer et al., 2012, Karydis et al., 2010], where the complexity of the tasks make results harder to interpret. For example, constructing a classification experiment with the three distinguishable songs presented in Section 2.2.4, or with a few songs with contrasting genres (e.g. Classical vs Metal), in our perspective, are good starting points to study the hub problem. Two other issues that we plan to analyze are the effects of data scaling and how descriptors extracted from blocks of consecutive frames react in terms of hubs and orphans compared to

low-level frame-based features. The experiments we undertook showed that scaling the data so that each dimension has unit variance has repercussions in terms hub and orphan distributions. We believe that this particular aspect is relevant because this type of data normalizations are quite common in MIR research (for e.g. in [Sturm, 2013c]) and its effects should be better understood.

# Appendix A

# Some Mathematics

## A.1 Metrics

A metric is function that returns a real number, and is a generalization of how we intuitively understand distances in 3-dimensional spaces - the Euclidean metric. For $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathcal{X}$, the metric $\text{dist} : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ must satisfy the following conditions: 1) Non-negativity: $\text{dist}(\mathbf{x}, \mathbf{y}) \geq 0$. 2) Identity: $\text{dist}(\mathbf{x}, \mathbf{y}) = 0$ if and only if $\mathbf{x} = \mathbf{y}$. 3) Symmetry: $\text{dist}(\mathbf{x}, \mathbf{y}) = \text{dist}(\mathbf{y}, \mathbf{x})$. 4) Triangular inequality: $\text{dist}(\mathbf{x}, \mathbf{z}) \leq \text{dist}(\mathbf{x}, \mathbf{y}) + \text{dist}(\mathbf{y}, \mathbf{z})$. There numerous metrics that have been proposed in literature (see Deza and Deza [2009] for an extensive listing), and we will come back to this subject later on. Next, we describe two common metrics that we used in our clustering algorithms: the Euclidean and the cosine distances.

1. The Euclidean distance between two $d$-dimensional vectors $\mathbf{p}$ and $\mathbf{q}$ is the length of the line segment that connects them:

$$\text{dist}_{\ell_2}(\mathbf{p}, \mathbf{q}) = \|\mathbf{p} - \mathbf{q}\| = \|\mathbf{q} - \mathbf{p}\| = \sqrt{(\mathbf{p} - \mathbf{q})^\top (\mathbf{p} - \mathbf{q})}$$

2. The cosine distance between two vectors $\mathbf{p}$ and $\mathbf{q}$ is:

$$\text{dist}_{\cos}(\mathbf{p}, \mathbf{q}) = 1 - \frac{\mathbf{p}^\top \mathbf{q}}{\|\mathbf{p}\| \|\mathbf{q}\|} = 1 - \cos(\theta)$$

   $\text{dist}_{\cos}(\mathbf{p}, \mathbf{q})$ is a bounded function within $[0, 2]$, and $\theta$ is the angle formed by the two vector. For orthogonal vectors, the distance value is $1$ and for parallel ones is either $\{0, 2\}$, depending on their direction. This formula is derived from the Euclidean dot product between two vectors: $\mathbf{p}^\top \mathbf{q} = \|\mathbf{p}\| \|\mathbf{q}\| \cos(\theta)$. Note that the cosine distance does not depend on the vector norms, therefore the distances between any two (non-zero) vectors $\mathbf{p}$ and $\mathbf{q}$, or their normalized counterparts, $\mathbf{p}/\|\mathbf{p}\|$ and $\mathbf{q}/\|\mathbf{q}\|$ yield identical measures.

## A.2 Linear Transformations

### A.2.1 Principal Component Analysis

Principal component analysis (PCA) is an orthogonal linear transformation that projects the data in the directions of maximum variance - the principal components. The underlying assumption is that the directions with the most variability carry the most information. The principal components form a new coordinate system, so that the projected data is linearly uncorrelated (e.i. has a diagonal covariance matrix). The first principal component has the highest data variance, the second component the second highest variance, and so on. PCA is a unsupervised non-parametric technique, and can be either obtained thru eigendecomposition of the data covariance matrix, or thru singular value decomposition (SVD) of the data matrix. Formalizing, let $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$ be a $d \times N$ matrix, containing the data: a total of $N$, $d$-dimensional vectors. For simplicity, we will assume that the data has zero mean. The empirical sampled covariance matrix $\hat{\mathbf{C}}_{\mathbf{x}}$ is:

$$\hat{\mathbf{C}}_{\mathbf{x}} = \frac{1}{N-1}\mathbf{X}\mathbf{X}^\top = \mathbf{V}\Delta\mathbf{V}^\top$$

where $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_d]$ is a $d \times d$ matrix containing the eigenvectors, $\mathbf{v}_i$ of $\hat{\mathbf{C}}_{\mathbf{x}}$, and $\Delta$ is a diagonal matrix with its eigenvalues. Note that symmetric matrices have orthogonal eigenvectors: $\mathbf{V}\mathbf{V}^\top = \mathbf{V}^\top\mathbf{V} = \mathbf{V}\mathbf{V}^{-1} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. This matrix $\mathbf{V}$, i.e. the principal components, can also be obtain by doing a singular value decomposition. SVD is a factorization technique that converts any real arbitrary, $p \times q$ matrix $\mathbf{M}$ into a product of three matrices:

$$\mathbf{M} = \mathbf{U}\Sigma\mathbf{V}^\top$$

where $\mathbf{U}$ is an orthogonal $p \times p$ matrix, $\mathbf{V}$ is an orthogonal $q \times q$ matrix, and $\Sigma$ is a rectangular diagonal matrix with the non-zero singular values of $\mathbf{M}$. SVD and eigendecomposition are closely related:

- The columns of $\mathbf{U}$ are the eigenvectors of $\mathbf{M}\mathbf{M}^\top$.
- The columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{M}^\top\mathbf{M}$.
- The diagonal elements of $\Sigma$ are the squared roots of the eigenvalues of both $\mathbf{M}\mathbf{M}^\top$ and $\mathbf{M}^\top\mathbf{M}$.

To use SVD in the context of PCA, it is more convenient to the decomposition on the following scaled and transposed version of data matrix:

$$\tilde{\mathbf{X}} = \frac{1}{\sqrt{N-1}}\mathbf{X}^\top = \mathbf{U}\Sigma\mathbf{V}^\top$$

Note that,

$$\begin{aligned}\tilde{\mathbf{X}}^\top\tilde{\mathbf{X}} &= \left(\tfrac{1}{\sqrt{N-1}}\mathbf{X}^\top\right)^\top\left(\tfrac{1}{\sqrt{N-1}}\mathbf{X}^\top\right) = \tfrac{1}{N-1}\mathbf{X}\mathbf{X}^\top = \hat{\mathbf{C}}_\mathbf{x}\\ \tilde{\mathbf{X}}^\top\tilde{\mathbf{X}} &= \left(\mathbf{U}\Sigma\mathbf{V}^\top\right)^\top\left(\mathbf{U}\Sigma\mathbf{V}^\top\right) = \mathbf{V}\Sigma\mathbf{U}^\top\mathbf{U}\Sigma\mathbf{V}^\top = \mathbf{V}\Sigma^2\mathbf{V}^\top\end{aligned}$$

From eigendecomposition, the principal components of the data matrix $\mathbf{X}$ are the eigenvectors of $\hat{\mathbf{C}}_\mathbf{x}$. If we calculate the SVD of $\tilde{\mathbf{X}}$, the columns of matrix $\mathbf{V}$ contain the eigenvectors of $\tilde{\mathbf{X}}^\top\tilde{\mathbf{X}} = \hat{\mathbf{C}}_\mathbf{x}$. Therefore the columns of $\mathbf{V}$ are the principal components of $\mathbf{X}$ and the non-zero diagonal elements of $\Sigma$ are the squared roots of the eigenvalues (the diagonal elements of $\Delta$). The projection of the data into its principal components is given as

$$\mathbf{Y} = \mathbf{V}^\top\mathbf{X}$$

In the new coordinate system, the data is uncorrelated and with individual variances equal to the eigenvalues of $\hat{\mathbf{C}}_\mathbf{x}$:

$$\hat{\mathbf{C}}_\mathbf{y} = \tfrac{1}{N-1}\mathbf{Y}\mathbf{Y}^\top = \tfrac{1}{N-1}\mathbf{V}^\top\mathbf{X}\mathbf{X}^\top\mathbf{V} = \mathbf{V}^\top\hat{\mathbf{C}}_\mathbf{x}\mathbf{V} = \mathbf{V}^\top\mathbf{V}\Delta\mathbf{V}^\top\mathbf{V} = \Delta$$

PCA is a essential tool in the fields of exploratory data analysis and model prediction. It was first proposed by Karl Pearson [Pearson, 1901] and later independently developped by Harold Hotelling [Hotelling, 1933], who named it after himself. Therefore PCA is also known as the Hotelling transform, but depending on the field of applicaton, other names are used, such as the Karhunen-Loéve transform in signal processing, proper orthogonal decomposition in mechanical engineering. Typically only $k$ with $k \ll d$ eigenvectors are kept, corresponding to the top eigenvalues. This is the case in our work, where we used PCA as a dimensionality reduction technique for data visulaziation and for converting high-dimensional feature sets into more manegeable, low-dimensional ones.

### A.2.2 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is supervised linear projection, rather than a discriminative functional. It is a generalization of the two-class Fisher linear discriminative methods [Fisher, 1936, Rao, 1948]. Its purpose is to find a transformation that at the same time maximizes the class separation (between-class scatter) while minimizing the variance within each class (within-class scatter). Formalizing, let $\mathbf{X}$ be a $d\times N$ matrix containing the data where each data vector has a known class label assiated with it: $\mathbf{x} \in \gamma_i$ and with $\gamma_i \in \Gamma$, where $|\Gamma|$ is the total

number of classes. The within-class scatter matrix, $\mathbf{S}_\text{w}$ is given by:

$$\begin{aligned} \mathbf{S}_\text{w} &= \sum_{i=1}^{|\Gamma|} \mathbf{S}_{\gamma_i} \\ \text{where} \quad \mathbf{S}_{\gamma_i} &= \sum_{\mathbf{x} \in \gamma_i} (\mathbf{x} - \mu_{\gamma_i}) (\mathbf{x} - \mu_{\gamma_i})^\top \\ \text{and} \quad \mu_{\gamma_i} &= \tfrac{1}{N_i} \sum_{\mathbf{x} \in \gamma_i} \mathbf{x} \end{aligned}$$

where $N_i$ is the number of vectors in the class $\gamma_i$, and $\sum_i N_i = N$. The between-class scatter matrix, $\mathbf{S}_\text{b}$ is:

$$\begin{aligned} \mathbf{S}_\text{b} &= \sum_{i=1}^{|\Gamma|} N_i (\mu_{\gamma_i} - \mu) (\mu_{\gamma_i} - \mu)^\top \\ \text{where} \quad \mu &= \frac{1}{N} \sum_{\forall \mathbf{x}} \mathbf{x} = \frac{1}{N} \sum_{i=1}^{|\Gamma|} N_i \mu_{\gamma_i} \end{aligned}$$

We can now define the projected vectors, $\mathbf{y}$, in terms of the original ones, and a projection matrix, $\mathbf{W}$, of size $d \times (|\Gamma| - 1)$ (the reason why this matrix has this dimensions will be explained shortly):

$$\mathbf{y} = \mathbf{W}^\top \mathbf{x}$$

The within and between scatter matrices of the projected data are defined has:

$$\begin{aligned} \tilde{\mathbf{S}}_\text{w} &= \sum_{i=1}^{|\Gamma|} \sum_{\mathbf{x} \in \gamma_i} (\mathbf{y} - \tilde{\mu}_{\gamma_i}) (\mathbf{y} - \tilde{\mu}_{\gamma_i})^\top = \mathbf{W}^\top \mathbf{S}_\text{w} \mathbf{W} \quad \text{where} \quad \tilde{\mu}_{\gamma_i} = \tfrac{1}{N_i} \sum_{\mathbf{y} \in \gamma_i} \mathbf{y} \\ \tilde{\mathbf{S}}_\text{b} &= \sum_{i=1}^{|\Gamma|} N_i (\tilde{\mu}_{\gamma_i} - \tilde{\mu}) (\tilde{\mu}_{\gamma_i} - \tilde{\mu})^\top = \mathbf{W}^\top \mathbf{S}_\text{b} \mathbf{W} \quad \text{where} \quad \tilde{\mu} = \tfrac{1}{N} \sum_{\forall \mathbf{y}} \mathbf{y} \end{aligned}$$

The maxtrix $\mathbf{W}$ is the projection that maximizes the speration between class (the between-class scatter) and minimizes the variance in each class (the within-class scatter), which is obtained by maximizing the following objective function:

$$\mathcal{J}(\mathbf{W}) = \frac{|\hat{\mathbf{S}}_\text{b}|}{|\hat{\mathbf{S}}_\text{w}|} = \frac{|\mathbf{W}^\top \mathbf{S}_\text{b} \mathbf{W}|}{|\mathbf{W}^\top \mathbf{S}_\text{w} \mathbf{W}|}$$

where $|\mathbf{A}|$ is the determinant of matrix $\mathbf{A}$. It can be shown that the optimal projection is the matrix $\mathbf{W}_\text{otp}$ whose columns are the eigenvectors corresponding to the largest eigenvalues of the matrix $\mathbf{S}_\text{w}^{-1} \mathbf{S}_\text{b}$. Like in PCA, this matrix can be otained via eigendecomposition or via SVD. Note that the matrix $\mathbf{S}_\text{b}$ is the sum of $|\Gamma|$ matrices with rank $\leq 1$. Furthermore, there is an additional constraint since the sum of the class means $\mu_{\gamma_i}$ (weighted by the class percentages) must be equal to global data mean $\mu$. Therefore $\mathbf{S}_\text{b}$ has rank $|\Gamma| - 1$ or less - there is, at most, $|\Gamma| - 1$ non-zero eigenvalues.

## A.3 Information Theoretic Measures

Claude Shannon in his seminal paper [Shannon, 1948] analysed the limits of compression, data stroge and communication over a noisy channel, and established the basis of information theory as we know it today. Next we a brief summary of some fundamental concepts in this field, namely concept of entropy and mutual information, their mathematical formulation and respective properties. We will address only the case of discrete variables since this is the case used in the context of this thesis. We should stress that information theory is an active research field with many intricassies and subtleties, and although the original formulation was for discrete variables, it also generalizes to continous ones. For in depth study on this matters the reader can consult [Cover and Thomas, 1991, MacKay, 2000, Papoulis, 1984].

### A.3.1 Entropy

Entropy is a measure of uncertainty or information content associated with a random variable. Uncertainty and information are "two sides of the same coin" since the uncertainty about a certain outcome vanishes after its realization, and therefore information is aquired when it happens. Let $x$ be a discrete random variable with a set of possible outcomes $x \in \{x_1, \ldots, x_n\}$ having probabilities[1] $0 \leq P(x_i) \leq 1$ with $\sum_i P(x_i) = 1$. The entropy of $x$ is given by:

$$\mathbf{H}(x) = -\sum_i P(x_i) \ln P(x_i) = -\mathbf{E}\left[\ln P(x_i)\right]$$

Amongst some of its properties are:

- $\mathbf{H}(x) \geq 0$ with equality for deterministic "random variables" (i.e. $P(x_k) = 1$ and $P(x_i) = 0$ for $k \neq i$.
- $\mathbf{H}(x)$ is maximum for equiprobable random variables (i.e. $\mathbf{H}(y) \leq \mathbf{H}(x)$ for $P(x_i) = \frac{1}{n}$).
- Entropy is invariant to permutations of the probabilities $P(x_i)$.
- Shannon-Gibbs enequality

$$\mathbf{H}(x) = -\sum_i P(x_i) \ln P(x_i) \leq -\sum_{i=1}^{n} P(x_i) \ln Q(x_i)$$

  with $\sum_i Q(x_i) = 1$ and $0 \leq Q(x_i) \leq 1 \ \ \forall i$.

[1] Here we will use the term "probabilities" to designate *probability mass functions*.

### A.3.2 Joint, Conditional and Marginal Entropies

The **joint entropy** of two discrete random variables $x \in \{x_1, \ldots, x_n\}$ and $y \in \{y_1, \ldots, y_m\}$ is

$$\mathbf{H}(x, y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} P(x_i, y_j) \ln P(x_i, y_j)$$

where $P(x, y)$ is the joint probability of $x$ and $y$.
The individual, or **marginal entropies** of $x$ and $y$ can be expressed in terms of the joint probability $P(x, y)$:

$$\begin{aligned}
\mathbf{H}(x) &= -\sum_{i=1}^{n} P(x_i) \ln P(x_i) = -\sum_{j=1}^{m}\sum_{i=1}^{n} P(x_i, y_j) \ln P(x_i) \\
\mathbf{H}(y) &= -\sum_{j=1}^{m} P(y_j) \ln P(y_j) = -\sum_{j=1}^{m}\sum_{i=1}^{n} P(x_i, y_j) \ln P(y_j)
\end{aligned}$$

The conditional entropy of $y$ given $x{=}x_i$ is

$$\mathbf{H}(y|x{=}x_i) = -\sum_{j=1}^{m} p(y_j|x{=}x_i) \ln P(y_j|x{=}x_i)$$

The **conditional entropy** of $y$ given $x$ is the expected value of $\mathbf{H}(y|x = x_i)$. This measure represents the uncertainty left in $y$ after observing $x$:

$$\mathbf{H}(y|x) = \sum_{i=1}^{n} P(x_i)\mathbf{H}(y|x_i) = -\sum_{i=1}^{n}\sum_{j=1}^{m} P(x_i, y_j) \ln P(y_j|x_i)$$

From the previous equations result the following properties:

1. $\mathbf{H}(x) \geq 0$, $\mathbf{H}(y) \geq 0$, $\mathbf{H}(x, y) \geq 0$, $\mathbf{H}(x|y) \geq 0$, and $\mathbf{H}(y|x) \geq 0$.
2. $\mathbf{H}(x, y) = \mathbf{H}(x|y) + \mathbf{H}(y) = \mathbf{H}(y|x) + \mathbf{H}(x)$.
3. $\mathbf{H}(x|y) \leq \mathbf{H}(x)$ with equality if and only if $x$ and $y$ are independent random variables.
4. $\mathbf{H}(x, y) \leq \mathbf{H}(x) + \mathbf{H}(y)$ with equality if and only if $x$ and $y$ are independent random variables.

### A.3.3 Mutual Information

The mutual information, $\mathcal{I}(x, y)$, between variables $x$ and $y$ is the amount of information that $x$ conveys about $y$, or vice versa. In other words, it measures the average reduction in uncertainty

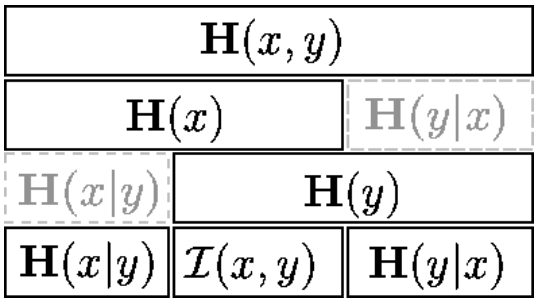



Figure A.1: Graphic representation of the properties of the joint, marginal, and conditional entropy of two discrete random variables $x$ and $y$. This representation was inspired by (i.e. copied from) [MacKay, 2000].

about $x$ that results from observing $y$.

$$\begin{aligned}\mathcal{I}(x,y) = \mathcal{I}(y,x) &= \mathbf{H}(x) - \mathbf{H}(x|y) \\ &= \mathbf{H}(y) - \mathbf{H}(y|x) \\ &= \mathbf{H}(x) + \mathbf{H}(y) - \mathbf{H}(x,y) \\ &= \mathbf{H}(x,y) - \mathbf{H}(x|y) - \mathbf{H}(y|x)\end{aligned}$$

It follows from these relations that the mutual information between two independent variables is zero. Figure A.1 is a graphic representation of the relations between joint, conditional, and marginal entropies and the mutual information.

### A.3.4 Kullback-Leibler Divergence

The Kullback-Leibler (KL) divergence [Kullback and Leibler, 1951] is a measure of the difference between two distributions. For discrete probability distributions $P$ and $Q$ the Kullback-Leibler divergence is defined as

$$\mathrm{KL}(P\|Q) = \sum_i \ln\left(\frac{P(i)}{Q(i)}\right) P(i) \qquad \text{(Equation 3.23)}$$

and for continuous probability density funcitons, $p(x)$ and $q(x)$ the expression is

$$\mathrm{KL}\left(p(x)\|q(x)\right) = \int_{-\infty}^{+\infty} \ln\left(\frac{p(x)}{q(x)}\right) p(x)dx$$

The KL-divergence is not a proper distance because it is not simetric $(\mathrm{KL}(P\|Q) \neq \mathrm{KL}(Q\|P))$ and does not fullfil the triangular inequality. It is common, though, to make simple adjustments in order to make the divergence simetric. Two symetric divergences based on the KL-divergence are:

- Kullback-Leibler: $\mathrm{dist}_{\mathrm{KL}}(P,Q) = \frac{1}{2}\mathrm{KL}(P\|Q) + \frac{1}{2}\mathrm{KL}(P\|Q)$
- Jensen-Shannon: $\mathrm{dist}_{\mathrm{JS}}(P,Q) = \frac{1}{2}\mathrm{KL}\left(P\|\frac{P+Q}{2}\right) + \frac{1}{2}\mathrm{KL}\left(Q\|\frac{P+Q}{2}\right)$

$\mathrm{dist}_{\mathrm{KL}}$ is a symmetric version of the Kullback-Leibler divergence, and $\mathrm{dist}_{\mathrm{JS}}$ is the Jensen-Shannon divergence. Both of these divergences are not strictly a distance, since they do not fulfill the triangular inequality (although the squared root of $\mathrm{dist}_{\mathrm{JS}}(P,Q)$ does [Endres and Schindelin, 2003]).

Note that for Gaussian densities, the Kullback-Leibler divergence has a closed-form solution. For two $d$-dimensional Gaussians, $\mathcal{N}_1$ and $\mathcal{N}_2$, with means $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ and covariance matrices $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$, the Kullback-Leibler divergence is:

$$\mathrm{KL}(\mathcal{N}_1\|\mathcal{N}_2) = \frac{1}{2}\left(\mathrm{trace}\left(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1\right) + (\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)^\top\boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1) - d - \ln\left(\frac{\det\boldsymbol{\Sigma}_1}{\det\boldsymbol{\Sigma}_2}\right)\right)$$

# Appendix B

# VQMM Related Experiments

The purpose of this appendix is to report and summarize the experiments that were not included in the main body of this thesis, since we felt that they were not essential to the presentation. Nevertheless, these experiments are an important part of our work and they complement the previously reported results. Additionally, the experiments (and corresponding evaluation scores) reported in this appendix were obtained throughout the course of our research, and are simply taken out of the several informal documents we compiled to keep track and described the many experiments we conducted. In this sense, this contrasts with the ones that are presented in the chapters of the thesis which were derived from more rigorous evaluation methodologies (i.e. results in the thesis main body were obtained through tests performed in a systematic way, with several test runs, and on many occasions, were re-done for the writing of the thesis).

## B.1 Genre Classification

The experiments here described pertain to the task of genre classification. We investigated how the VQMMs perform with different types of low-level features, and how they are affected by the length of the audio excerpts.

### B.1.1 Low-Level Features

In this section we report on experiments we conducted on low-level features. The tests were performed in the early on in our research with the objective of familiarizing ourselves with some basic aspects of inferring genre based on low-level features. Table B.1 shows the accuracies obtained with the VQMMs on genre classification with the ISMIR04 data set. The experiments where divided in three parts, and in each part we investigated a particular aspect of low-level

| | Win | Hop | T1 | T2 | T3 | Mean+Std |
|---|---|---|---|---|---|---|
| Frame length and hop size | 512 | 256 | 80.2 | 80.0 | 79.8 | 80.0±0.20 |
| | 1024 | 512 | 80.7 | 80.7 | 79.8 | 80.4±0.52 |
| | 2048 | 1024 | 80.4 | 81.5 | 81.2 | 81.0±0.57 |
| | 4096 | 2048 | 80.4 | 81.3 | 80.1 | 80.6±0.62 |
| | 4096 | 3072 | 80.4 | 81.6 | 80.9 | 81.0±0.60 |
| | 8192 | 2048 | 79.6 | 81.2 | 80.1 | 80.3±0.82 |
| | 8192 | 4096 | 80.0 | 79.6 | 80.0 | 79.9±0.23 |
| | 16384 | 8192 | 79.8 | 80.8 | 78.3 | 79.6±1.26 |
| | 16384 | 16384 | 77.8 | 77.4 | 78.3 | 77.8±0.45 |
| Features | MFCCs | | 80.4 | 81.5 | 81.2 | 81.0±0.57 |
| | Mel | | 80.9 | 81.8 | 82.9 | 81.7±1.00 |
| | Cent | | 73.0 | 71.1 | 70.6 | 71.6±1.27 |
| | Log-Cent | | 82.2 | 80.5 | 81.9 | 81.5±0.91 |
| | PCPs | | 72.0 | 70.8 | 71.6 | 71.5±0.61 |
| | Log-PCPs | | 72.4 | 70.3 | 71.5 | 71.4±1.05 |
| Projections | Log-Cent + PCA | | 80.1 | 81.1 | 81.8 | 81.0±0.85 |
| | Log-Cent + whitening | | 75.6 | 75.4 | 76.4 | 75.8±0.53 |
| | Log-Cent + ICA | | 69.0 | 68.3 | 68.4 | 68.6±0.38 |

Table B.1: Genre classification performances on the ISMIR04 data set. The tests we performed with the objective of studying three aspects related to low-level features. The top part of the table different window and hop-size lengths were considered. In the middle part we investigated the use of different timbral and chroma related features. In the bottom part of the table are the results obtained with different type of linear transformations (see text for details).

representations.

The top part of the table shows accuracies for different values of frame length and hop sizes. The features chosen for the tests were the all-purpose MFCCs. We used 13 coefficients, including the zero order MFCC, obtained form the DCT transform of the logarithmic Mel-spectrum (a bank of 50 filters was used). The original audio is a 22050 Hz mono signal with 16 bits per sample. The results show that accuracies remain unchanged for a wide range of frame and hop sizes. For very large frame lengths (i.e. 8192 and 16384 - which correspond to audio durations of $\approx 0.37$ and $\approx 0.74$ seconds respectively) the accuracies shows a small decrease in values. In our perspective this is a bit counter intuitive, since the signal portions encompassed by this durations may not by the stationary.

The second part of the table shows performances obtained with different type of timbral and chroma related features. The features are:

1. The MFCCs (12 total) - the same as the ones used for the top part of the table.
2. The Mel features (40 total) which are the log-square root power in each filter of the Mel filter bank.
3. The Cent features (72 total) were obtained from the magnitude of the cent spectrum, with 12 semi-tones per octave, and 6 octave bands total (from 130 Hz to 8.7 kHz).
4. The Log-Cent features (72 total) - logarithmic version of the Cent features.
5. The PCP features (12 total) are the folded version of the Cent features: i.e. each coefficients corresponds to the mean value of each semi-tone for the 6 octave bands.
6. The Log-PCP features (12 total) - logarithmic version of the PCP features.

The results show two modes in the achieved performances. The VQMMs using as inputs the MFCCs, the Mel features and the Log-Cent features obtained accuracies comparable to the ones reported in Chapter 3. However, the Cent features did not perform well, which is an indication that the logarithmic scaling is an important aspect in spectral representations. The PCPs and Log-PCPs also showed poor results and is a sign that the folding process applied in each octave band may be too destructive, a least for genre classification purposes.

The bottom part of the table shows the VQMMs performances with linear transformations of the Log-Cent features. In all three transformations, we first reduced the dimensionality of the features to twelve (the same cardinality as the MFCCs). The dimensionality reduction was done via principal component analysis. The line (Log-Cent + PCA) corresponds to the features obtained with this transformation. The second line of the last part of the table (Log-Cent + whitening), the features are normalized to have unit variance in each dimension, after the PCA transformation. Note that this normalization has a negative effect on performances. The last line, the features were obtained via Independent Component Analysis (ICA), a transformation

| | Song maximum duration (in seconds) | | | | | |
|---|---|---|---|---|---|---|
| | 15 | 30 | 45 | 60 | 75 | 90 |
| Train & Test | 58.2±2.7 | 64.7±2.0 | 68.4±0.7 | 69.3±1.7 | 70.0±0.9 | 69.9±1.6 |
| Train Only | 63.3±2.2 | 64.4±2.0 | 67.2±2.3 | 68.2±1.0 | 69.1±1.1 | 70.3±0.9 |
| Test Only | 64.7±3.8 | 69.2±2.4 | 70.1±2.1 | 71.6±1.7 | 71.6±2.1 | 71.4±1.9 |

Table B.2: VQMMs accuracies on the LMD data set (plus or minus one standard deviation), restricting the length of the songs in the training and test sets. In the first line are the results obtained with all the songs in the set limited to a predefined number of seconds (columns). The second line corresponds to restricting the length of the songs from the training set only. The last line, the restriction is applied to test songs only.

that tries to make the data in each dimension as independent from the other dimensions as possible. This method obtained the worst results, suggesting that this type of transformations are not well suited for low-level feature representations in the context of automatic genre recognition. For more information about ICA we refer to [Hyvärinen et al., 2001].

### B.1.2 Audio Signal Duration

A limitation with the VQMMs is the fact that the audio excerpt duration has to be long enough; otherwise classification accuracies start to deteriorate. In order to determine the minimum signal length necessary for the VQMMs to function properly, we conducted tests with the LMD data set (clean version), restricting the songs durations. We imposed this restriction in three distinct ways:

1. Limit the length of all the songs in the set.
2. Limit the length of the songs used for training.
3. Limit the length of the songs used for testing.

The results are reported in Table B.2 (via 3-fold cross validation). We used a codebook size of $k_2 = 200$, and the accuracies are the average of the 3 test folds. The impact of the songs duration is noticeable particularly when we restrict the length of the audio excerpts in the training phase of the VQMMs: for clips lengths below 60 seconds the performances start to degrade. The length of the songs used for testing also as an impact on performances, although in this case, it only happens when the audio duration is below 30 seconds.

| | Per-Tag | | | Global | | | |
|---|---|---|---|---|---|---|---|
| Models | Prec. | Recall | F-score | Prec. | Recall | F-score | AROC |
| BLF | - | - | 30.61 | - | - | 50.15 | - |
| $BLF_{PCA}$ | - | - | 29.08 | - | - | 49.65 | - |
| Aff-SVM | - | - | 49.8 | - | - | - | 89.0 |
| MD | 60.6 | 21.2 | 31.4 | - | - | - | - |
| FMSV | 63.7 | 12.1 | 20.3 | - | - | - | - |
| MA-FMSV | 58.8 | 20.6 | 30.7 | - | - | - | - |
| $VQMMs_{10F}$ | 26.70 | 31.50 | 28.80 | 47.79 | 49.43 | 48.54 | - |
| $VQMMs_{10F}\star$ | 36.34 | 31.50 | 33.62 | 47.79 | 49.43 | 48.54 | - |
| $SVM_2$ | 27.19 | 27.35 | 27.27 | 47.00 | 45.78 | 46.38 | - |

Table B.3: CAL500: Tag scores with binary matrices. In first two lines are the scores with the trivial classifiers (all tags on). BLF and $BFL_{PCA}$ are the results in [Seyerlehner et al., 2010b] (2-fold cross validation). The Aff-SVM are reported in [Ness et al., 2009] (2-fold cross validation). MD, FMSV and MA-FMSV are the results in [Zhao et al., 2010] (2-fold cross validation). Here they do not take in account tags that were not chosen (with 0 hits in classification). This can skew the per-tag precision and F-scores - the VQMMs$\star$ scores were estimated in this fashion, compared to averaging with a "0" score. The last line are the results for the $SVM_2$ from Section 5.2.

## B.2 Autotagging

In this section we report the performances obtained with the VQMMs for the autotagging tasks using the CAL500 data. The results are summarize in Tables B.3, B.4, and B.5, and serve to complement the ones presented in Chapter 3 (Table 3.5).

| Category | A/\|Θ\| | Model | Precision | | Recall | | Fscore |
|---|---|---|---|---|---|---|---|
| All words | 10 / 174 | Random | 14.4 | (0.4) | 6.4 | (0.2) | 8.9 |
| | | UpperBnd | 71.2 | (0.7) | 37.5 | (0.6) | 49.1 |
| | | GMM | 26.5 | (0.7) | **15.8** | (0.6) | **19.8** |
| | | MFCCΔ | 28.1 | (6.6) | 13.1 | (1.9) | 17.9 |
| | | afeats | 26.6 | (7.8) | 9.4 | (1.8) | 13.9 |
| | | bfeats | 29.1 | (10.5) | 8.9 | (3.4) | 13.6 |
| | | VQMM | **30.44** | (12.28) | 14.29 | (3.47) | 19.43 |
| Emotion | 4 / 36 | Random | 27.6 | (1.2) | 11.3 | (0.4) | 16.0 |
| | | UpperBnd | 95.7 | (0.5) | 39.6 | (1.0) | 56.0 |
| | | GMM | 42.4 | (0.8) | 19.5 | (0.4) | 26.7 |
| | | MFCCΔ | 44.4 | (2.5) | 19.2 | (1.6) | 26.8 |
| | | afeats | 43.3 | (3.0) | 17.1 | (1.1) | 24.5 |
| | | bfeats | 41.8 | (5.3) | 15.6 | (3.7) | 22.7 |
| | | VQMM | **46.15** | (8.18) | **19.83** | (2.27) | **27.72** |
| Genre | 2 / 31 | Random | 5.5 | (0.5) | 7.9 | (0.8) | 6.5 |
| | | UpperBnd | 56.2 | (2.6) | 77.7 | (1.8) | 65.2 |
| | | GMM | 17.1 | (0.9) | **24.2** | (1.9) | 20.0 |
| | | MFCCΔ | 15.4 | (2.4) | 16.8 | (2.1) | 16.1 |
| | | afeats | 17.3 | (4.8) | 13.4 | (3.3) | 15.1 |
| | | bfeats | 14.7 | (2.7) | 16.0 | (4.5) | 15.3 |
| | | VQMM | **19.92** | (8.40) | 21.75 | (8.53) | **20.72** |
| Instrumentation | 4 / 24 | Random | 14.1 | (0.9) | 19.5 | (1.4) | 16.4 |
| | | UpperBnd | 60.1 | (1.5) | 86.8 | (1.8) | 71.0 |
| | | GMM | 26.7 | (0.8) | 32.0 | (2.2) | 29.1 |
| | | MFCCΔ | 26.7 | (4.7) | **36.3** | (2.1) | 30.8 |
| | | afeats | 29.4 | (7.3) | 27.5 | (7.4) | 28.4 |
| | | bfeats | **32.9** | (6.5) | 28.9 | (8.4) | 30.8 |
| | | VQMM | 30.32 | (8.20) | 34.53 | (10.05) | **32.10** |
| Solo | 1 / 9 | Random | 3.1 | (0.7) | 15.5 | (3.5) | 5.2 |
| | | UpperBnd | 19.7 | (1.9) | 76.0 | (5.2) | 31.3 |
| | | GMM | 6.0 | (1.2) | 26.1 | (5.0) | 9.8 |
| | | MFCCΔ | 5.4 | (0.2) | **37.4** | (3.5) | 9.4 |
| | | afeats | 4.5 | (0.2) | 27.8 | (7.8) | 7.8 |
| | | bfeats | 4.2 | (0.2) | 31.2 | (9.4) | 7.4 |
| | | VQMM | **9.57** | (2.73) | 25.79 | (14.25) | **13.05** |
| Usage | 2 / 15 | Random | 7.3 | (0.8) | 15.4 | (1.6) | 9.9 |
| | | UpperBnd | 36.3 | (1.4) | 81.4 | (3.1) | 50.2 |
| | | GMM | 12.2 | (1.2) | **26.4** | (2.7) | **16.7** |
| | | MFCCΔ | 12.2 | (1.1) | 23.9 | (2.8) | 16.2 |
| | | afeats | 10.3 | (1.0) | 18.8 | (5.4) | 13.3 |
| | | bfeats | 10.6 | (1.0) | 20.9 | (6.6) | 14.0 |
| | | VQMM | **12.59** | (3.25) | 22.46 | (10.07) | 15.84 |
| Vocal | 2 / 16 | Random | 6.2 | (0.7) | 15.3 | (1.8) | 8.8 |
| | | UpperBnd | 32.1 | (1.7) | 78.8 | (1.9) | 45.6 |
| | | GMM | 13.4 | (0.5) | **33.5** | (2.1) | 19.1 |
| | | MFCCΔ | 11.6 | (1.1) | 25.2 | (2.9) | 15.9 |
| | | afeats | 13.0 | (3.0) | 19.8 | (5.0) | 15.7 |
| | | bfeats | 13.3 | (1.7) | 21.2 | (4.6) | 16.4 |
| | | VQMM | **17.03** | (6.26) | 28.60 | (11.29) | **21.09** |

Table B.4: CAL500 - ***Annotation*** results obtain via ranking with fixed annotation length $A$, and a vocabulary of size $|\Theta|$. GMM are the results in Turnbull et al. [2008b], the MFFCΔ, afeats, bfeats are the results in Bertin-Mahieux et al. [2008]. The experiments were all performed via 10-fold cross validation (in parenthesis are the standard deviation for the test runs). The VQMMs results were obtained with a codebook size of $k_2 = 200$, and using only a positive model.

.

| Category | $\|\Theta\|$ | Model | MeanAP | | MeanAROC | |
|---|---|---|---|---|---|---|
| All words | 174 | Random | 23.1 | (0.4) | 50.3 | (0.4) |
| | | GMM | **39.0** | (0.4) | 71.0 | (0.4) |
| | | MFCC$\Delta$ | 30.5 | (5.7) | 67.8 | (1.5) |
| | | afeats | 32.3 | (9.2) | 62.2 | (1.3) |
| | | bfeats | 34.0 | (12.4) | 66.2 | (2.0) |
| | | $CBA_{500}$ | **39.0** | (0.8) | **75.9** | (0.7) |
| | | VQMM | 38.01 | (6.47) | 69.19 | (3.16) |
| Emotion | 36 | Random | 32.7 | (0.6) | 50.4 | (0.3) |
| | | GMM | 50.6 | (0.8) | **71.0** | (0.4) |
| | | MFCC$\Delta$ | 50.3 | (3.1) | 70.2 | (0.5) |
| | | afeats | 46.9 | (2.6) | 65.2 | (0.5) |
| | | bfeats | **56.5** | (4.8) | 68.6 | (0.6) |
| | | VQMM | 49.95 | (3.38) | 69.32 | (1.71) |
| Genre | 31 | Random | 13.2 | (0.5) | 50.0 | (0.5) |
| | | GMM | **32.9** | (1.2) | 71.9 | (0.5) |
| | | MFCC$\Delta$ | 9.4 | (1.3) | 70.5 | (1.3) |
| | | afeats | 8.8 | (1.1) | 62.6 | (1.0) |
| | | bfeats | 11.8 | (3.2) | 69.3 | (1.5) |
| | | VQMM | 31.02 | (6.62) | **72.62** | (3.75) |
| Instrumentation | 24 | Random | 22.1 | (0.7) | 50.2 | (0.4) |
| | | GMM | **39.9** | (1.8) | 71.9 | (0.6) |
| | | MFCC$\Delta$ | 13.7 | (2.2) | 70.7 | (0.5) |
| | | afeats | 18.2 | (3.3) | 65.8 | (1.5) |
| | | bfeats | 14.0 | (3.2) | 72.0 | (1.5) |
| | | VQMM | 38.96 | (5.93) | **73.17** | (2.07) |
| Solo | 9 | Random | 10.6 | (1.4) | 50.2 | (0.4) |
| | | GMM | 18.0 | (2.8) | **71.2** | (0.6) |
| | | MFCC$\Delta$ | 5.2 | (0.2) | 56.5 | (2.5) |
| | | afeats | 5.0 | (0.2) | 58.2 | (0.9) |
| | | bfeats | 4.2 | (0.3) | 51.9 | (2.6) |
| | | VQMM | **21.09** | (6.16) | 61.16 | (6.11) |
| Usage | 15 | Random | 16.9 | (1.2) | 50.1 | (0.5) |
| | | GMM | **24.0** | (1.6) | **70.7** | (0.4) |
| | | MFCC$\Delta$ | 12.0 | (0.9) | 62.1 | (2.2) |
| | | afeats | 13.3 | (2.2) | 53.8 | (1.6) |
| | | bfeats | 10.1 | (1.4) | 56.3 | (3.4) |
| | | VQMM | 22.04 | (3.59) | 62.45 | (4.03) |
| Vocal | 16 | Random | 13.7 | (0.6) | 50.2 | (0.4) |
| | | GMM | 26.0 | (1.8) | **70.5** | (0.5) |
| | | MFCC$\Delta$ | 11.1 | (1.2) | 65.2 | (1.8) |
| | | afeats | 9.0 | (0.7) | 58.6 | (1.4) |
| | | bfeats | 12.8 | (1.8) | 63.0 | (2.0) |
| | | VQMM | **29.17** | (7.24) | 67.32 | (5.57) |

Table B.5: CAL500 - ***Retrieval*** results for the same authors as in table B.4. Possibly the results in [Bertin-Mahieux et al., 2008] have a bug for the last five subsets, since the scores are below the random baseline.

# Appendix C

# Data Sets

## C.1 ISMIR04

This dataset was created for the genre classification contest organized during the ISMIR 2004 conference [Cano et al., 2006]. The data is organized in six genres, with a total of 729 music pieces for training and the same number of music pieces for testing. The number of songs per genres is: 320 Classical, 115 Electronic, 26 JazzBlues, 45 MetalPunk, 101 RockPop, and 122 World. The dataset was used in the same fashion as in the original ISMIR 2004 contest, and therefore, the data set does not account for artist filtering between both sets.

## C.2 LMD

The Latin Music Database [Silla et al., 2008] is composed of 3160 songs from 543 artists, divided into ten Latin music genres (in parenthesis are the number of songs belonging to that genre): Axé (304), Bachata (308), Bolero (302), Forró (315), Gaúcha(306), Merengue (307), Pagode (301), Salsa (303), Sertaneja (310), and Tango (404). Song selection and genre attributions were performed by trained Latin dance music teachers - a laborious process that took over a year to accomplish. This data set has the characteristic of having a strong rhythmic content, which is also a main discriminative factor for identifying the genres present in the set.

### LMD - (Clean)

For our experiments, we created a subset of 900 music pieces of the The Latin Music Database, which are divided in three folds of equal size (30 pieces per class). The music pieces are

uniformly distributed over the 10 genres. We used an artist filter (see Section 2.3.4) so that the music pieces from a specific artist are present in one and only one of the three folds. We also added the constraint of the same number of artists per fold.

## C.3 GTZAN

The GTZAN [Tzanetakis and Cook, 2002] data set consists of 1000 audio excerpts, 30 seconds long, divided into ten genres (100 excerpt per genre): Blues, Classical, Country, Disco, Hiphop, Jazz, Metal, Pop, Reggae, and Rock. According to the authors, the files were collected from a variety of sources including personal CDs, radio, and microphone recordings. This is one of the first available data sets for genre classification and probably the most referenced data set in MIR literature. However, the variety of recording conditions and issues such as duplications and mislabelings constitute an impediment for the evaluation of genre recognition systems. For an analysis of this data set and the problems associated with it, please refer to [Sturm, 2012c, 2013b].

## C.4 ARTIST20

ARTIST20 [Ellis, 2007] is a data set used for artist identification. The data is composed of 6 albums by 20 different artists making a total of 1413 tracks. It is based on the 18 artist set used on [Mandel and Ellis, 2005] (drawn from "uspop2002") with some additions and enhancements, to correct a number of issues such repeated tracks, live recordings, and others.

## C.5 CAL500

The Computer Audition Lab 500 (CAL500) [Turnbull et al., 2008b] is one of the most popular music data sets used in the context of autotagging. It consists of 502 popular songs from different artists annotated with pre-defined semantic concepts from a fixed vocabulary (the tag names are listed by category in Table C.1). The labeling process was done by paid undergraduate students, and based on the participants annotations the tag vectors were derived ensuring user agreement on the tags. This set is characterized by a high number of annotations per song (26 on average). Figure C.1 shows the tag frequencies by musical facet.

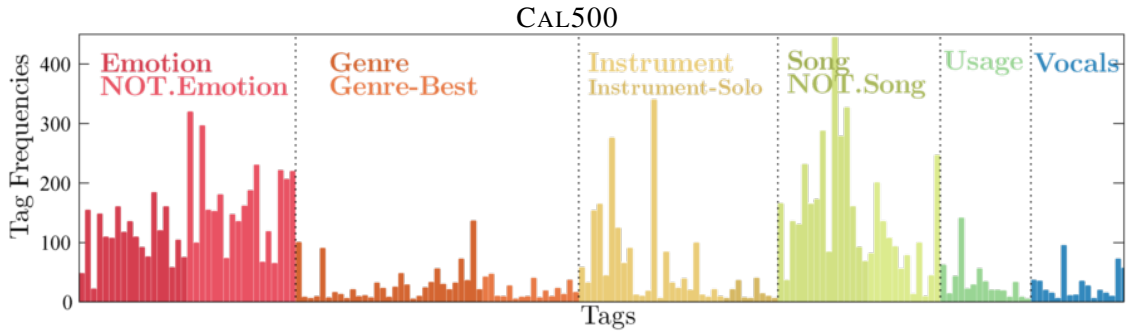



Figure C.1: Tag frequencies for the CAL500 data set. The CAL500 is consist of 502 songs annotated with a set of 174 possible tags. The tags are divided into six broad categories: emotions, genres, instruments, acoustic song qualities, usage, and vocal characteristics. Table C.1 lists all the tags ordered by category. The total number of annotations (assigned tags) is 13074 which results 26 annotations per song, on average.

## C.6 Magnatagatune - MTT

The Magnatagatune data set (http://tagatune.org/Magnatagatune.html), which for simplicity will be refered as MTT, consists of 21642 excerpts of length $\approx$ 30 seconds from 230 different artists. Excerpts annotations are among a set of 188 tags.

The MTT data set reveals a significant number of problems with annotation: (1) **synonymy**: we merged a number of tags (e.g. "clasical", "classical" and "classic"), (2) **trivial cases**: we removed excerpts with tags such as e.g. "silence", (3) **antonymy**: we removed tag attributions of an excerpt when they were not compatible (e.g. having both "drums" and "no-drums" tags, or "fast" and "slow"), (4) **extreme sparseness**: we removed excerpts with no tags, and (5) **duplication**: many excerpts in the MTT data set are segments of the same original piece and have different tag annotations, we kept those segments with the maximum number of tags and removed the other segments.

## C.7 MAGTAG5k

After pre-processing the MTT data set as detailed above, the remaining data, referred henceforth as MAGTAG5k, consists of 137 tags among 5259 excerpts from 230 artists. The tag names are reported in Table C.2), where we divided them into 8 musical facets. We did this categorization with the intent to give the reader a general perspective of the tag characteristics present in this data set. However, we must point out that these facets are themselves very broad and this

| Facet | # Tags | Tag Names |
|---|---|---|
| Emotion | 18 | Angry.Aggressive, Arousing.Awakening, Bizarre.Weird, Calming.Soothing, Carefree.Lighthearted, Cheerful.Festive, Emotional.Passionate, Exciting.Thrilling, Happy, Laid-back.Mellow, Light.Playful, Loving.Romantic, Pleasant.Comfortable, Positive.Optimistic, Powerful.Strong, Sad, Tender.Soft, Touching.Loving |
| NOT.Emotion | 18 | NOT.Angry.Aggressive, NOT.Arousing.Awakening, NOT.Bizarre.Weird, NOT.Calming.Soothing, NOT.Carefree.Lighthearted, NOT.Cheerful.Festive, NOT.Emotional.Passionate, NOT.Exciting.Thrilling, NOT.Happy, NOT.Laid-back.Mellow, NOT.Light.Playful, NOT.Loving.Romantic, NOT.Pleasant.Comfortable, NOT.Positive.Optimistic, NOT.Powerful.Strong, NOT.Sad, NOT.Tender.Soft, NOT.Touching.Loving |
| Genre | 31 | Alternative, AlternativeFolk, Bebop, BritPop, ClassicRock, ContemporaryBlues, ContemporaryR&B, CoolJazz, CountryBlues, DancePop, ElectricBlues, Funk, Gospel, Metal.HardRock, Punk, RootsRock, Singer.Songwriter, SoftRock, Soul, Swing, Bluegrass, Blues, Country, Electronica, Folk, HipHop.Rap, Jazz, Pop, R&B, Rock, World |
| Genre-Best | 16 | Alternative, ClassicRock, Metal.HardRock, Punk, Soft.Rock, Soul, Blues, Country, Electronica, Folk, HipHop.Rap, Jazz, Pop, R&B, Rock, World |
| Instrument | 24 | AcousticGuitar, AmbientSounds, BackingVocals, Bass, DrumMachine, DrumSet, ElectricGuitar(clean), ElectricGuitar(distorted), FemaleLeadVocals, HandDrums, Harmonica, HornSection, MaleLeadVocals, Organ, Piano, Samples, Saxophone, Sequencer, StringEnsemble, Synthesizer, Tambourine, Trombone, Trumpet, Violin.Fiddle |
| Instrument-Solo | 9 | AcousticGuitar, ElectricGuitar(clean), ElectricGuitar(distorted), FemaleLeadVocals, Harmonica, MaleLeadVocals, Piano, Saxophone, Trumpet |
| Song | 12 | Catchy.Memorable, ChangingEnergyLevel, FastTempo, HeavyBeat, HighEnergy, Like, PositiveFeelings, Quality, Recommend, Recorded, Tonality, VeryDanceable |
| NOT.Song | 12 | NOT.Catchy.Memorable, NOT.ChangingEnergyLevel, NOT.FastTempo, NOT.HeavyBeat, NOT.HighEnergy, NOT.Like, NOT.PositiveFeelings, NOT.Quality, NOT.Recommend, NOT.Recorded, NOT.Tonality, NOT.VeryDanceable |
| Song-Texture | 3 | Acoustic, Electric, Synthesized |
| Usage | 15 | AtAParty, AtWork, CleaningTheHouse, Driving, Exercising, GettingReadyToGoOut, GoingToSleep, HangingWithFriends, IntenselyListening, Reading, Romancing, Sleeping, Studying, WakingUp, WithTheFamily |
| Vocals | 16 | Aggressive, AlteredWithEffects, Breathy, Call&Response, Duet, Emotional, Falsetto, Gravelly, HighPitched, LowPitched, Monotone, Rapping, Screaming, Spoken, Strong, VocalHarmonies |

Table C.1: Tags names (a total of 174) from the CAL500 data set, divided into six semantic concepts.

arrangement may not be consensual, at least for some labels. For example, we placed the tag "celtic" in "Language/Geography" facet, but we could as well had put it in the "Genre" facet, and other similar cases could also be pinpointed.

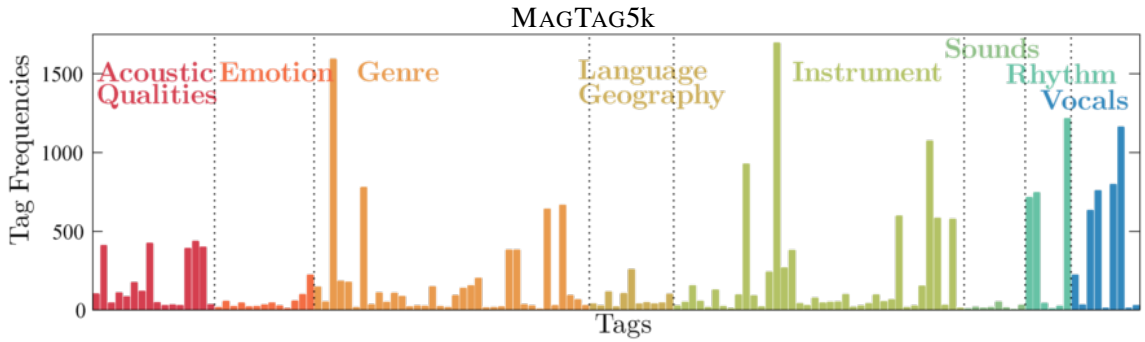



Figure C.2: Tag frequencies for the MAGTAG5k data sets. The MAGTAG5k data set consists of 5259 songs annotated with a set of 137 possible tags. We chose to group the tags into eight broad categories to give a general perspective of the tag characteristics present in the set: acoustic qualities/styles, emotions, genres, languages/geography, instruments, sounds, rhythmic and vocal qualities (see Table C.2 for details). The MAGTAG5k has a total of 25572 annotations, which gives, on average, a number of 5 annotations per song.

| Facet | # Tags | Tag Names |
|---|---|---|
| Acoustic Qualities/Style | 16 | acoustic, ambient, calm, electric, funky, instrumental, jazzy, loud, low, old, operatic, plucking, quiet, soft, solo, spacey |
| Emotion | 13 | airy, dark, deep, different, eerie, happy, light, mellow, sad, scary, strange, upbeat, weird |
| Genre | 36 | baroque, blues, classical, country, dance, disco, electro, female.opera, folk, funk, hard.rock, heavy.metal, hip.hop, house, industrial, jazz, jungle, male.opera, medieval, metal, modern, new.age, not.classical, not.opera, not.rock, opera, pop, punk, rap, reggae, rock, soft.rock, techno, trance, tribal, world |
| Language/Geography | 11 | arabic, celtic, eastern, english, foreign, indian, irish, middle.eastern, not.english, oriental, spanish |
| Instrument | 38 | acoustic.guitar, banjo, bass, bells, bongos, cello, chimes, clarinet, classical.guitar, drums, electric.guitar, fiddle, flutes, guitars, harp, harpsichord, horns, keyboard, lute, no.drums, no.flutes, no.guitars, no.piano, no.strings, no.violins, oboe, orchestra, organ, percussion, piano, piano.solo, sax, sitar, strings, synth, trumpet, violins, woodwind |
| Sounds | 8 | birds, clapping, drone, echo, noise, space, water, wind |
| Rhythm | 6 | beats, fast, fast.beat, no.beats, repetitive, slow |
| Vocals | 9 | chant, duet, female.singing, man.singing, monks, no.singing, singing, soprano, talking |

Table C.2: Tags names (a total of 137 total) from MAGTAG5k data set divided into 8 semantic concepts.

# Bibliography


P. Agarwal, H. Karnick, and B. Raj. A comparative study of indian and western music forms. In *Proc. of the $14^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 29–34, Curitiba, Brazil, 2013.

C. C. Aggarwal. Re-designing distance functions and distance-based applications for high dimensional data. *ACM SIGMOD Record*, 30(1):13–18, 2001.

C. C. Aggarwal, A. Hinneburg, and D. A. Keim. On the surprising behavior of distance metrics in high dimensional spaces. In *Proceedings of the 8th International Conference on Database Theory*, pages 420–434, London, UK, UK, 2001. Springer-Verlag.

J. Ahn, J. S. Marron, K. M. Muller, and Y.-Y. Chi. The high-dimension, low-sample-size geometric representation holds under mild conditions. *Biometrika*, 94(3):760–766, 2007.

P. Ahrendt. *Music Genre Classification Systems - A Computational Approach*. PhD thesis, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Lyngby, 2006.

P. Ahrendt and A. Meng. Music genre classification using multivariate ar feature integration model. In *MIREX*, 2005.

X. Amatriain, J. M. Pujol, and N. Oliver. I like it... i like it not: Evaluating user ratings noise in recommender systems. In *User Modeling, Adaptation, and Personalization*, pages 247–258. Springer, 2009.

C. Anderson. *The long tail: Why the future of business is selling less of more*. Hachette Digital, Inc., 2006.

P. Annesi, R. Basili, R. Gitto, A. Moschitti, and R. Petitti. Audio feature engineering for automatic music genre classification. In *RIAO*, Pittsburgh, 2007.

J.-J. Aucouturier. *Ten Experiments on the Modelling of Polyphonic Timbre*. PhD thesis, University of Paris 6, Paris, 2006.

J.-J. Aucouturier. Sounds like teen spirit: Computational insights into the grounding of everyday musical terms. In *(J. Minett and W. Wang eds) Language, Evolution, and the Brain*. Academia Sinica Press, 2009.

J.-J. Aucouturier and F. Pachet. Music similarity measures: What's the use? In *Proc. of the $3^{rd}$ Int. Conf. on Music Information Retrieval (ISMIR)*, France, October 2002.

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

J.-J. Aucouturier and F. Pachet. Improving timbre similarity: How high is the sky? *Journal of Negative Results in Speech and Audio Sciences*, 1(1), 2004.

J.-J. Aucouturier and F. Pachet. The influence of polyphony on the dynamical modelling of musical timbre. *Pattern Recognition Letters*, 28(5):654–661, 2007.

J.-J. Aucouturier and E. Pampalk. From genres to tags: A little epistemology of Music Information Retrieval research. *Journal of New Music Research*, 32(7), 2008.

M. Babbitt. The use of computers in musicological research. *Perspectives of New Music*, 3(2): 74–83, 1965.

G. Ball and D. Hall. Isodata, a novel method of data analysis and pattern classification. Technical report, Stanford Research Institute, 1965.

L. Barrington, M. Yazdani, D. Turnbull, and G. Lanckriet. Combining feature kernels for semantic music retrieval. In *Proc. of the $9^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Philadelphia, USA, 2008.

L. Barrington, A. Chan, and G. Lanckriet. Modeling music as a dynamic texture. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(3):602–612, March 2010.

L. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

J. Bello and A. J. Pickens. A robust mid-level representation for harmonic content in music signals. In *Proc. of the $6^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, London, UK, 2005.

A. Berenzweig. *Anchors and Hubs in Audio-Based Music Similarity*. PhD thesis, Columbia University, New York, 2007.

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

J. Bergstra, N. Casagrande, and D. Eck. Two algorithms for timbre and rhythm-based multiresolution audio classification. MIREX, 2005.

J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl. Aggregate features and AdaBoost for music classification. *Machine Learning*, 65(2-3):473–484, 2006.

T. Bertin-Mahieux, D. Eck, F. Maillet, and P. Lamere. Autotagger: a model for predicting social tags from acoustic features on large music databases. *Journal of New Music Research*, 37(2), June 2008.

T. Bertin-Mahieux, D. Eck, and M. Mandel. Automatic tagging of audio: The state-of-the-art. In W. Wang, editor, *Machine Audition: Principles, Algorithms and Systems*. IGI Publishing, 2010.

C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

D. Bogdanov, J. Serrà, N. Wack, P. Herrera, and X. Serra. Unifying low-level and high-level music similarity measures. *IEEE Transactions on Multimedia*, 13(4):687 – 701, August 2011.

D. Brackett. *Interpreting Popular Music*. University of California Press, 1995.

A. P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

E. Britannica. Rythm, 2014. URL http://www.britannica.com/EBchecked/topic/501914/rhythm. [Online; accessed February-2014].

C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

J. J. Burred and A. Lerch. A hierarchical approach to automatic musical genre classification. In *in Proc. Of the 6 th Int. Conf. on Digital Audio Effects (DAFx*, pages 8–11, 2003.

G. Cabral, J.-P. Briot, and F. Pachet. Impact of distance in pitch class profile computation. In *Proc. of the* $10^{th}$ *Brazilian Symposium on Computer Music (SBCM)*, 2005.

P. Cano, E. Gomez, F. Gouyon, P. Herrera, M. Koppenberger, B. Ong, X. Serra, S. Streich, and N. Wack. Ismir 2004 audio description contest. In *MTG Technical Report MTG-TR-2006-02*, 2006.

C. Cao and M. Li. Thinkit submissions for mirex2009 audio music classification and similarity tasks. MIREX, 2009.

M. Casey, C. Rhodes, and M. Slaney. Analysis of minimum distances in high-dimensional musical spaces. *IEEE Trans. on Audio, Speech, and Language Processing*, 16(5):1015–1028, July 2008a.

M. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, 96 (4):668–696, April 2008b.

O. Celma and X. Serra. Foafing the music: Bridging the semantic gap in music recommendation. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(4):250–256, 2008.

Ó. Celma, P. Cano, and P. Herrera. Search sounds: An audio crawler focused on weblogs. In *Proc. of the $7^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Victoria, Canada, 2006a.

Ó. Celma, P. Herrera, and X. Serra. Bridging the music semantic gap. In *Proceedings of $3^{rd}$ European Semantic Web Conference (ESWC)*, 2006b.

C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, May 2011.

A. R. Chase. Music discriminations by carp (cyprinus carpio). *Animal Learning & Behavior*, 29(4):336–353, 2001.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

C. H. L. Costa, J. D. V. Jr., A. L. Koerich, and R. L. Koerich. Automatic classification of audio data. In *IEEE Transactions on Systems, Man, and Cybernetics*, pages 562–567, 2004.

Y. M. Costa, L. Oliveira, A. L. Koerich, F. Gouyon, and J. Martins. Music genre classification using LBP textural features. *Signal Processing*, 92(11):2723–2737, 2012.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., New York, U.S.A., 1991.

E. Coviello, A. Chan, and G. Lanckriet. Time series models for semantic music annotation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 19(5):1343–1359, 2011.

A. Craft, G. Wiggins, and T. Crawform. How many beans make five? the consensus problem in music-genre classification and a new evaluation method for single-genre categorisation systems. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2007.

S. J. Cunningham, D. Bainbridge, and J. S. Downie. The impact of mirex on scholarly research (2005-2010). 2012.

I. Daubechies. Orthonormal bases of compactly supported wavelets. *Communications on pure and applied mathematics*, 41(7):909–996, 1988.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

M. Deza and E. Deza. *Encyclopedia of Distances*. Springer, 2009.

S. Dhande and B. Tiple. A survey of mood-based music classification. *International Journal of Engineering Research & Technology*, 2(5):1745–1750, 2013.

J. S. Downie. The music information retrieval evaluation exchange (2005/2007): A window into music information retrieval research. *Acoustical Science and Technology*, 29(4):247–255, 2008.

J. S. Downie, D. Byrd, and T. Crawford. Ten years of ismir: Reflections on challenges and opportunities. In *Proc. of the $10^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2009.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. John Wiley & Sons, 2012.

D. Eck, T. Bertin-Mahieux, and P. Lamere. Autotagging music using supervised machine learning. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Vienna, Austria, 2007.

Y. EL-Manzalawy and V. Honavar. *WLSVM: Integrating LibSVM into Weka Environment*, 2005. Software available at `http://www.cs.iastate.edu/~yasser/wlsvm`.

A. Elisseeff and J. Weston. A kernel method for multi-labelled classification. In *Advances in neural information processing systems*, pages 681–687, 2001.

D. Ellis. Classifying music audio with timbral and chroma features. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2007.

D. Ellis and G. Poliner. Indentifying cover songs with chroma features and dynamic programming beat tracking. In *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, Havai, U.S.A., 2007.

D. Ellis, B. Whitman, A. Berenzweig, and S. Laurence. The quest for the ground truth in musical artist similarity. In *Proc. of the $3^{rd}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Paris, France, 2002.

K. Ellis, E. Coviello, A. Chan, and G. Lanckriet. A bag of systems representation for music auto-tagging. *Audio, Speech, and Language Processing, IEEE Transactions on*, 21(12):2554–2569, Dec 2013.

D. Endres and J. Schindelin. A new metric for probability distributions. *IEEE Transactions on Information Theory*, 49(7):1858–1860, 2003.

L. Ertoz, M. Steinbach, and V. Kumar. A new shared nearest neighbor clustering algorithm and its applications. In *Workshop on Clustering High Dimensional Data and its Applications at $2^{nd}$ SIAM International Conference on Data Mining*, pages 105–115, 2002.

F. Fabbri. Browsing music spaces: Categories and the musical mind. In *Proc. of the International Association for the Study of Popular Music Conference (IASPM)*, 1999.

T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27, 2006.

M. Filippone, F. Camastra, F. Masulli, and S. Rovetta. A survey of kernel and spectral methods for clustering. *Pattern recognition*, 41(1):176–190, 2008.

R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7 (2):179–188, 1936.

A. Flexer. A closer look on artist filters for musical genre classification. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2007.

A. Flexer, E. Pampalk, and G. Widmer. Hidden Markov models for spectral similarity of songs. In *DAFx*, 2005.

A. Flexer, D. Schnitzer, M. Gasser, and T. Pohle. Combining features reduces hubness in audio similarity. In *Proc. of the $11^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2010.

A. Flexer, D. Schnitzer, and J. Schluter. A mirex meta-analysis of hubness in audio music similarity. In *Proc. of the $13^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2012.

R. Foucard, S. Essid, M. Lagrange, and G. Richard. Multi-scale temporal fusion by boosting for music classification. In *Proc. of the $12^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 663–668, Miami, USA, 2011.

Y. Freund and R. Schapire. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

Z. Fu, G. Lu, K. Ting, and D. Zhang. A survey of audio-based music classification and annotation. *IEEE Transactions on Multimedia*, 13(2):303–319, 2011.

T. Fusjishima. Realtime chord recognition of musical sounds: a system using common lisp music. In *ICMC*, 1999.

E. Gauss. *Automatic Content Processing for Automatic Music Genre Classification: Descriptors, Databases, and Classifiers*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2009.

R. Gjerdingen and D. Perrott. Scanning the dial: The rapid recognition of music genres. *Journal of New Music Research*, 37(2):93–100, 2008.

S. Godbole and S. Sarawagi. Discriminative methods for multi-labeled classification. In *Advances in Knowledge Discovery and Data Mining*, pages 22–30. Springer, 2004.

E. Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2006.

E. Gómez and P. Herrera. The song remains the same: identifying versions of the same piece using tonal descriptors. In *Proc. of the $7^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Victoria, Canada, 2006.

M. Goto. A chorus-section detecting method for musical audio signals. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03)*, volume 5, pages 437–440, 2003.

F. Gouyon. A review of automatic rhythm description systems. *Computer Music Journal*, 29 (1):34–54, 2005a.

F. Gouyon. *A Computational Approach to Rhythm Description*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2005b.

F. Gouyon, P. Herrera, E. Gómez, P. Cano, J. Bonanda, A. Loscos, X. Amatrian, and X. Serra. Content processing of music audio signals. In P. Polotti and D. Rocchesso, editors, *Sound to Sense - Sense to Sound: A state of the art in Sound and Music Computing*, pages 83–160. Logos Verlag, 2008.

S. J. Green, P. Lamere, J. Alexander, F. Maillet, S. Kirk, J. Holt, J. Bourque, and X.-W. Mak. Generating transparent, steerable recommendations from textual descriptions of items. In *Proceedings of the third ACM conference on Recommender systems*, pages 281–284. ACM, 2009.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11(1), 2009.

P. Hall, J. Marron, and A. Neeman. Geometric representation of high dimension, low sample size data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(3): 427–444, 2005.

P. Hamel, S. Wood, and D. Eck. Automatic identification of instrument classes in polyphonic and poly-instrument audio. In *Proc. of the $10^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2009.

P. Hamel, M. E. P. Davies, K. Yoshii, and M. Goto. Transfer learning in mir: Sharing learned latent representations for music audio classification and similarity. In *Proc. of the $14^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 9–14, Curitiba, Brazil, 2013.

T. Heittola, A. Klapuri, and T. Virtanen. Musical instrument recognition in polyphonic audio using source-filter model ofr sound separation. In *Proc. of the $10^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2009.

J. Herre, E. Allamanche, and O. Hellmuth. Robust matching of audio signals using spectral flatness features. In *Applications of Signal Processing to Audio and Acoustics, 2001 IEEE Workshop on the*, pages 127–130, 2001.

P. Herrera, A. Yeterian, and F. Gouyon. Automatic classification of drum sounds: A comparison of feature selection methods and classification techniques. In *Music and Artificial Intelligence*, volume 2445 of *Lecture Notes in Computer Science*, pages 69–80. Springer Berlin Heidelberg, 2002.

P. Herrera, G. Peeters, and S. Dubnov. Automatic classification of musical instrument sounds. *Journal of New Music Research*, 32, 2003.

A. Hicklin, B. Ulery, and C. Watson. Myth of goats: How many people have fingerprints that are hard to match. Technical report, US Dept. of Commerce, National Institue of Standards and Technology, 2005.

M. Hoffman, D. Blei, and P. Cook. Easy as CBA: A simple probabilistic model for tagging music. In *Proc. of the $10^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2009.

A. Holzapfel and Y. Stylianou. A statistical approach to musical genre classification using non-negative matrix factorization. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 2, pages II–693, 2007.

H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, (24):417–441, 1933.

J. Hu, S. G. Lim, and M. K. Brown. Writer independent on-line handwriting recognition using an HMM approach. *Pattern Recognition*, 33(1):133 – 147, 2000.

N. Hu, R. B. Dannenberg, and G. Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *In Proc. of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2003.

P.-S. Huang, R. Mertens, A. Divakaran, G. Friedland, and M. Hasegawa-Johnson. How to put it into words-using random forests to extract symbol level descriptions from audio content for concept detection. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 505–508. IEEE, 2012.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley & Sons, 2001.

A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8): 651–666, 2010.

F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, 1997.

J. H. Jensen. *Feature Extraction for Music Information Retrieval*. PhD thesis, Aalborg University, Aalborg, Denmark, 2009.

J. H. Jensen, D. P. Ellis, M. G. Christensen, and S. H. Jensen. Evaluation distance measures between gaussian mixture models of mfccs. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 107–108, Vienna, Austria, September 2007.

J. H. Jensen, M. G. Christensen, D. P. Ellis, and S. H. Jensen. Quantitative analysis of a common audio similarity measure. *Audio, Speech, and Language Processing, IEEE Transactions on*, 17(4):693–703, 2009.

D.-N. Jiang, L. Lu, H.-J. Zhang, J.-H. Tao, and L.-H. Cai. Music type classification by spectral contrast feature. In *ICME*, 2002.

C. Joder, S. Essid, and G. Richard. A comparative study of tonal acoustic features for a symbolic level music-to-score alignment. In *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pages 409–412, March 2010.

I. Karydis, M. Radovanović, A. Nanopoulos, and M. Ivanović. Looking through the "glass ceiling": A conceptual framework for the problems of spectral similarity. In *Proc. of the $11^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2010.

M. Kassler. Toward musical information retrieval. *Perspectives of New Music*, pages 59–67, 1966.

Y. Kim, D. Williamson, and S. Pilli. Towards understanding and quantifying the "album effect" in artist identification. In *Proc. of the $7^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Victoria, Canada, 2006.

Y. Kim, E. M. Schmidt, R. Migneco, B. G. Morton, P. Richardson, J. J. Scott, J. A. Speck, and D. Turnbull. Music emotion recognition: A state of the art review. In *Proc. of the $11^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 255–266, Utrecht, Netherlands, 2010.

A. Klapuri and M. Davy, editors. *Signal Processing Methods for Music Transcription*. Springer-Verlag, 2006.

P. Knees and M. Schedl. A survey of music similarity and recommendation from music context data. *ACM Trans. Multimedia Comput. Commun. Appl.*, 10(1):2:1–2:21, Dec. 2013.

P. Knees, T. Pohle, M. Schedl, D. Schnitzer, and K. Seyerlehner. A document-centered approach to a natural language music search engine. In *Advances in Information Retrieval*, pages 627–631. Springer, 2008.

A. L. Koerich. Improving the reliability of music genre classification using rejection and verification. In *Proc. of the $14^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 511–516, Curitiba, Brazil, 2013.

T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69, 1982.

S. Kullback and R. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

P. Lamere. Social tagging and music information retrieval. *Journal of New Music Research*, 37 (2):101–114, 2008.

T. Langlois and G. Marques. A music classification method based on timbral features. In *Proc. of the $10^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Kobe, Japan, 2009a.

T. Langlois and G. Marques. Automatic music classification using a hierachical clustering and a language modeling approach. In *Proc. of the $1^{st}$ Int. Conf. on Advances in Multimedia (MMEDIA)*, Colmar, France, 2009b.

T. Langlois, T. Chambel, E. Oliveira, P. Carvalho, G. Marques, and A. Falcão. Virus: Video information retrieval using subtitles. In *Proceedings of the 14th International Academic MindTrek Conference: Envisioning Future Media Environments*, MindTrek, pages 197–200, New York, NY, USA, 2010. ACM.

C. Laurier. *Automatic Classification of Musical Mood by Content-Based Analysis*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2011.

E. Law, L. von Ahn, R. Dannenberg, and M. Crawford. Tagatune: a game for music and sound annotation. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Vienna, Austria, 2007.

K. Lee and M. Slaney. Automatic chord recognition from audio using an HMM with supervised learning. In *Proc. of the $7^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Victoria, Canada, 2006.

A. Lerch. *An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics*. Wiley/IEEE Press, 2012.

P. Leveau, D. Sodoyer, and L. Daudet. Automatic instrument recognition in a polyphonic mixture using sparse representations. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Vienna, Austria, September 2007.

M. Levy and M. Sandler. A semantic space for music derived from social tags. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Vienna, Austria, September 2007.

M. Li and R. Sleep. A robust approach to sequence classification. In *ICTAI*, 2005.

T. Lidy and A. Rauber. Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In *Proc. of the $6^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 34–41, 2005.

T. Lidy, A. Rauber, A. Pertusa, and J. Iñesta. Improving genre classification by combination of audio and symbolic descriptors using a transcription system. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2007.

T. Lidy, C. N. S. Jr., O. Cornelis, F. Gouyon, A. Rauber, C. A. Kaestner, and A. L. Koerich. On the suitability of state-of-the-art music information retrieval methods for analyzing, categorizing and accessing non-western and ethnic music collections. *Signal Processing*, 90(4): 1032 – 1048, 2010.

J. Lin. Scalable language processing algorithms for the masses: A case study in computing word co-occurrence matrices with mapreduce. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 419–428. Association for Computational Linguistics, 2008.

H. B. Lincoln. Some criteria and techniques for developing computerized thematic indices. *Electronische Datenverarbeitung in der Musikwissenschaft. Regensburg*, 1967.

S. Lippens, J. Martens, and T. De Mulder. A comparison of human and automatic musical genre classification. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on*, volume 4, pages iv–233–iv–236 vol.4, May 2004.

B. Logan and A. Salomon. A music similarity function based on signal analysis. In *ICME*, 2001.

D. MacKay. *Information Theory, Inference and Learning Algorithms*. Web Page, 2000. Draft2.2.0.

G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Džeroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9):3084 – 3104, 2012.

J. Makhoul, T. Starner, R. Schwartz, and G. Chou. On-line cursive handwriting recognition using speech recognition methods. In *In ICASSP*, pages 125–128, 1994.

M. Mandel and D. Elis. A web-based game for collecting music metadata. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Vienna, Austria, 2007.

M. Mandel and D. Ellis. Song level features and support vector machines for music classification. In *Proc. of the $6^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2005.

M. Mandel, R. Pascanu, H. Larochelle, and Y. Bengio. Autotagging music with conditional restricted boltzmann machines. 2011. Online article: http://arxiv.org/abs/1103.2832.

B. S. Manjunath, P. Salembier, and T. Sikora. *Introduction to MPEG 7: Multimedia Content Description Language*. Ed. Wiley, 2002.

C. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, 2002.

G. Marques and T. Langlois. A similarity measure for music signals. In *Proc. of the $10^{th}$ Int. Conf. on Enterprise Information Systems (ICEIS)*, Barcelona, Spain, 2008.

G. Marques and T. Langlois. A language modeling approach for classification of audio music. In *DMIN*, Las Vegas, U.S.A., 2009.

G. Marques, M. Lopes, M. Sordo, T. Langlois, and F. Gouyon. Additional evidence that common low-level features of individual audio frames are not representative of music genre. In *Proc. of the $7^{th}$ Sound and Music Computing Conf. (SMC)*, Barcelona, Spain, 2010.

G. Marques, M. Domingues, T. Langlois, and F. Gouyon. Three current issues in music autotagging. In *Proc. of the $12^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Maimi, U.S.A., 2011a.

G. Marques, T. Langlois, F. Gouyon, M. Lopes, and M. Sordo. Short-term feature space and music genre classification. *Journal of New Music Research*, 40(2):127–137, 2011b.

MARSYAS. Music analysis, retrieval and synthesis for audio signals. Website. URL http://marsyas.sness.net/.

L. G. Martins, J. J. Burred, G. Tzanetakis, and M. Lagrange. Polyphonic instrument recognition using spectral clustering. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 213–218, 2007.

B. Matityaho and M. Furst. Neural network based model for classification of music type. In *Electrical and Electronics Engineers in Israel, 1995., Eighteenth Convention of*, pages 4.3.4/1–4.3.4/5, March 1995.

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

R. Mayer, A. Rauber, P. J. Ponce de León, C. Pérez-Sancho, and J. M. Iñesta. Feature selection in a cartesian ensemble of feature subspace classifiers for music categorisation. In *Proceedings of 3rd International Workshop on Machine Learning and Music*, MML '10, pages 53–56. ACM, 2010.

J. McClelland and D. Rumelhart. *Parallel distributed processing: explorations in the microstructure, vol. 2: psychological and biological models*. 1986.

C. McKay. Using neural networks for musical genre classification.

C. Mckay. *Automatic Music Classification with jMIR*. PhD thesis, McGill University, Montreal, Canada, 2010.

C. McKay and I. Fujinaga. Automatic music classification and the importance of instrument identification. In *Proc. of the Conference on Interdisciplinary Musicology*, 2005.

C. McKay and I. Fujinaga. Musical genre classification: Is it worth pursuing and how can it be improved? In *Proc. of the $7^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Victoria, Canada, 2006.

A. Meng. *Temporal Feature Integration for Music Organization*. PhD thesis, Technical University of Denmark, 2006.

A. Meng, P. Ahrendt, and J. Larsen. Improving music genre classification by short time feature integration. In *Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on*, volume 5, pages v–497. IEEE, 2005.

J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A*, 209 (441-458):415–446, 1909.

R. Miotto, L. Barrington, and G. Lanckriet. Improving auto-tagging by modeling semantic co-occurences. In *Proc. of the $11^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2010.

T. M. Mitchell. *Machine Learning*. McGraw Hill series in computer science. McGraw-Hill, 1997.

F. Moerchen, I. Mierswa, and A. Ultsch. Understandable models of music collections based on exhaustive feature generation with temporal statistics. In *Proceedings of the international conference on Knowledge discovery and data mining*, pages 882–891. ACM, 2006.

B. Moore, editor. *Hearing – Handbook of Perception of Hearing*. Academic Press, San Diego, California, 1995.

M. Müller and S. Ewert. Chroma toolbox: Matlab implementations for extracting variants of chroma-based audio features. In *Proc. of the $12^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Miami, USA, 2011.

K. Murphy. Hidden Markov models toolbox for matlab. Website. URL http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html.

Y. Nam and K. Wohn. Recognition of space-time hand-gestures using hidden markov model. In *ACM Symposium on Virtual Reality Software and Technology*. ACM, 1996.

S. R. Ness, A. Theocharis, G. Tzanetakis, and L. G. Martins. Improving automatic music tag annotation using stacked generalization of probabilistic svm outputs. In *Proc. of the 17th ACM Int. Conf. on Multimedia (MM -09*, New York, U.S.A., 2009. ACM.

T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):971–987, Jul 2002.

A. Oppenheim, A. Willsky, and S. Nawab. *Signals and systems*. Prentice-Hall signal processing series. Prentice Hall, 1997.

N. Orio. Music retrieval: A tutorial and review. *Foundations and Trends in Information Retrieval*, 1(1):1–90, 2006.

P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

F. Pachet and P. Roy. Improving multilabel analysis of music titles: A large-scale validation of the correction approach. *IEEE Transactions on Audio, Speech, and Language Processing*, 17 (2), 2009.

E. Pampalk. Islands of music: Analysis, organization, and visualization of music archives, 2001.

E. Pampalk. Speeding up music similarity. MIREX, 2005.

E. Pampalk. Audio-based music similarity and retrieval: Combining a spectral similarity model with information extracted from fluctuation patterns. MIREX, 2006a.

E. Pampalk. *Computational Models of Music Similarity and their Application in Music Information Retrieval*. PhD thesis, Vienna University of Technology, Austria, 2006b.

E. Pampalk, A. Rauber, and D. Merkl. Content-based organization and visualization of music archives. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 570–579. ACM, 2002.

E. Pampalk, A. Flexer, and G. Widmer. Improvements of audio-based music similarity and genre classification. In *Proc. of the $6^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2005.

Y. Panagakis, C. Kotropoulos, and G. Arce. Music genre classification using locality preserving non-negative tensor factorization and sparse representations. In *Proc. of the $10^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2009.

R. Panda and R. P. Paiva. Music emotion classification: Dataset acquisition and comparative analysis. In *15th International Conference on Digital Audio Effects (DAFx-12)*, 2012.

A. Papoulis. *Signal analysis*. McGraw-Hill electrical and electronic engineering series. McGraw-Hill, 1977.

A. Papoulis. *Random Variables, and Stochastic Processes*. McGraw-Hill, New York, 1984.

J. Paulus, M. Müller, and A. Klapuri. State of the art report: Audio-based music structure analysis. In *Proc. of the $11^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 625–636, 2010.

S. Pauws. Musical key extraction from audio. In *Proc. of the $5^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2004.

K. Pearson. On lines and planes of closest fit to systems of points in space. *Phylosophical Magazine*, 2(11):559–572, 1901.

G. Peeters. A generic system for audio indexing: Application to speech/music segmentation and music genre recognition. In *DAFx*, 2007.

G. Peeters and X. Rodet. A large set of audio feature for sound description (similarity and classification) in the cuidado project. Technical report, Ircam, Analysis/Synthesis Team, 1 pl. Igor Stravinsky, 75004 Paris, France, 2004.

F. Pereira. *Comunicações Audiovisuais: Tecnologias, Normas e Aplicações*. IST PRESS, 2009.

C. Pérez-Sancho. *Stochastic language models for music information retrieval*. PhD thesis, Universidad de Alicante, Spain, 2009.

T. Pohle. *Automatic Characterization of Music for Intuitive Retrieval*. PhD thesis, Johannes Kepler University, Linz, Austria, January 2010.

T. Pohle, E. Pampalk, and G. Widmer. Evaluation of frequently used audio features for classification of music into perceptual categories. In *CBMI*, 2005.

T. Pohle, D.Schnitzer, M. Schedl, P. Knees, and G. Widmer. On rhythm and general music similarity. In *Proc. of the $10^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2009.

D. Porter and A. Neuringer. Music discriminations by pigeons. *Journal of Experimental Psychology: Animal Behavior Processes*, 10(2):138, 1984.

M. F. Porter. An algorithm for suffix stripping. *Program: electronic library and information systems*, 14(3):130–137, 1980.

L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, 1993.

L. Rabiner and R. Schafer. *Theory and Applications of Digital Speech Processing*. Prentice Hall, 2010.

L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286, 1989.

M. Radovanović, A. Nanopoulos, and M. Ivanović. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal Machine Learning Research*, 11:2487–2531, 2010.

D. M. Randel, editor. *The New Harvard Dictionary of Music*. Belknap Harvard University Press, 1986.

R. C. Rao. The utilization of multiple measurements in problems of biological classification. *Journal of the Royal Statistical Society - Series B*, 10(2):159–203, 1948.

M. Rossing. *The Science of Sound*. Addison Wesley, 1990.

Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

J. Salamon, B. Rochay, and E. Gomez. Music genre classification using melody features extracted from polyphonic music signals. In *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2012.

N. Scaringella and G. Zoia. On the modeling of time information for automatic genre recognition systems in audio signals. In *Proc. of the $6^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2005.

M. Schedl and P. Knees. Context-based music similarity estimation. In *Proc. of the 3rd International Workshop on Learning Semantics of Audio Signals (LSAS)*, pages 59–74, 2009.

A. Schindler and A. Rauber. Capturing the temporal domain in echonest features for improved classification effectiveness. In *Adaptive Multimedia Retrieval*, Lecture Notes in Computer Science, Copenhagen, Denmark, October 2012. Springer.

D. Schnitzer. *Indexing Content-Based Music Similarity Models for Fast Retrieval in Massive Databases*. PhD thesis, Johannes Kepler University, Linz, 2011.

D. Schnitzer, A. Flexer, M. Schedl, and G. Widmer. Using mutual proximity to improve content-based audio similarity. In *Proc. of the $12^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2011.

D. Schnitzer, A. Flexer, M. Schedl, and G. Widmer. Local and global scaling reduce hubs in space. *Journal of Machine Learning Research*, 13:2871–2902, 2012.

J. Serrà, E. Gómez, P. Herrera, and X. Serra. Chroma binary similarity and local alignment applied to cover song identification. *IEEE Transactions on Audio, Speech and Language Processing*, 16:1138–1151, 2008.

K. Seyerlehner. *Content-Based Music Recommender Systems: Beyond simple Frame-Level Audio Similarity*. PhD thesis, Johannes Kepler University, Linz, 2010.

K. Seyerlehner, G. Widmer, and P. Knees. Frame level audio similarity - a codebook approach. In *DAFx*, 2008.

K. Seyerlehner, G. Widmer, and T. Pohle. Fusing block-level features for music similarity estimation. In *DAFx*, 2010a.

K. Seyerlehner, G. Widmer, M. Schedl, and P. Knees. Automatic music tag classification based on block-level features. In *SMC*, Barcelona, Spain, 2010b.

C. Shannon. A mathematical theory of communication. *Bell Syst. Tech. Journal*, 27:379–423, 623–656, 1948.

S. Sigurdsson, K. Petersen, and T. Lehn-Schiøler. Mel frequency cepstral coefficients: An evaluation of robustness of MP3 encoded music. In *Proc. of the $7^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2006.

C. Silla, A. Koerich, and C. Kaestner. The latin music database. In *Proc. of the $9^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2008.

A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, 2004.

H. Soltau, T. Schultz, M. Westphal, and A. Waibel. Recognition of music types. In *ICASSP*, 1998.

M. Sordo. *Semantic Annotation of Music Collections: A Computational Approach*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2011.

M. Sordo, M. Celma, M. Blech, and E. Guaus. The quest for musical genres: Do the experts and the wisdom of crowds agree? In *Proc. of the $9^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2008.

M. Sordo, F. Gouyon, and L. Sarmento. A method for obtaining semantic facets of music tags. In *1st Workshop On Music Recommendation And Discovery (WOMRAD)*, Barcelona, Spain, 2010.

M. Stein, B. M. Schubert, M. Gruhne, G. Gatzsche, and M. Mehnert. Evaluation and comparison of audio chroma feature extraction methods. In *Audio Engineering Society Convention 126*, May 2009.

S. S. Stevens, J. Volkman, and E. Newman. A scale for the measurement of the psychological magnitude pitch. *Journal of the Acoustical Society of America*, 8(3):185–190, 1937.

B. L. Sturm. Two systems for automatic music genre recognition: What are they really recognizing? In *Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies*, pages 69–74. ACM, 2012a.

B. L. Sturm. A survey of evaluation in music genre recognition. *Proc. Adaptive Multimedia Retrieval, Copenhagen, Denmark*, 2012b.

B. L. Sturm. An analysis of the gtzan music genre dataset. In *Proceedings of the second international ACM workshop on Music information retrieval with user-centered and multimodal strategies*, pages 7–12. ACM, 2012c.

B. L. Sturm. Evaluating music emotion reconition: Lessons from music genre recognition? In *International Conference on Multimedia and Expo*, 2013a.

B. L. Sturm. The gtzan dataset: Its contents, its faults, their effects on evaluation, and its future use. *CoRR*, abs/1306.1461, 2013b.

B. L. Sturm. Classification accuracy is not enough. *Journal of Intelligent Information Systems*, 41(3):371–406, 2013c.

D. Tingle, Y. E. Kim, and D. Turnbull. Exploring automatic music annotation with "acoustically-objective" tags. In *Int. Conf. on Multimedia Information Retrieval*, 2010.

G. Tsoumakas, I. Katakis, and I. Vlahavas. Mining multi-label data. In O. Maimon and L. Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer US, 2010.

D. Turnbull, L. Barrington, and G. Lanckriet. A game-based approach for collecting semantic annotations of music. In *Proc. of the $8^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 535–538, Vienna, Austria, 2007.

D. Turnbull, L. Barrington, and G. Lanckriet. Five approaches to collecting tags for music. In *Proc. of the $9^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 225–230, Philadelphia PA, USA, 2008a.

D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Semantic annotation and retrieval of music and sound effects. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2), 2008b.

G. Tzanetakis. *Manipulation, Analysis and Retrieval Systems for Audio Signals*. PhD thesis, Princeton University, NJ, USA, June 2002.

G. Tzanetakis and P. Cook. Musical genre classification of audio singals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

G. Tzanetakis, G. Essl, and P. Cook. Automatic musical genre classification of audio signals. In *Proc. of the $2^{nd}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Bloomington, Indiana, 2001.

V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, U.S.A., 1995.

N. Vasconcelos. On the complexity of probabilistic image retrieval. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 400–407. IEEE, 2001.

D. Ververidis and C. Kotropoulos. Information loss of the mahalanobis distance in high dimensions: Application to feature selection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(12):2275–2281, 2009.

H. Vinet, P. Herrera-Boyer, and F. Pachet. The CUIDADO project. In *Proc. of the $3^{nd}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 197–203, Paris, France, 2002.

U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 4(17), 2007.

G. Voronoi. Nouvelles applications des paramètres continus à la théorie de formes quadratiques. *Journal für die reine und angewandte Mathematik*, (133):97–178, 1908.

A. Wang. An industrial strength audio search algorithm. In *Proc. of the $4^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2003.

D. Wang and G. Brown, editors. *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications*. Wiley-IEEE Press, 2006.

T. Welch. A technique for high-performance data compression. *Computer*, 17(6):8–19, 1984.

F. Weninger, B. Schuller, C. C. Liem, F. Kurth, and A. Hanjalic. *Music Information Retrieval: An Inspirational Guide to Transfer from Related Disciplines*, volume 3 of *Dagstuhl Follow-Ups*, pages 195–216. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2012.

K. West. *Novel Techniques for Audio Music Classification and Search*. PhD thesis, University of East Anglia, Norwich, UK, 2008.

K. West and S. Cox. Features and classifiers for the automatic classification of musical audio signals. In *Proc. of the $5^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, 2004.

K. West and P. Lamere. A model-based approach to constructing music similarity functions. *Journal on Advances in Signal Processing*, 2007.

K. West, S. Cox, and P. Lamere. Incorporating machine-learning into music similarity estimation. In *AMCMM*, pages 89–96, CA, U.S.A., 2006.

B. Whitman and D. Ellis. Automatic record reviews. In *Proc. of the $5^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, Barcelona, Spain, 2004.

G. Widmer, S. Dixon, P. Knees, E. Pampalk, and T. Pohle. From sound to sense via feature extraction and machine learning: Deriving high-level descriptors for characterising music. In P. Polotti and D. Rocchesso, editors, *Sound to Sense - Sense to Sound: A state of the art in Sound and Music Computing*, pages 161–194. Logos Verlag, 2008.

G. Wiggins. Semantic gap?? Schemantic schmap!! Methodological considerations in the scientific study of music. In *Proc. of the IEEE Int. Symposium on Multimedia*, pages 477–482, 2009.

N. Yager and T. Dunstone. The biometric menagerie. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(2):220–230, 2010.

Y.-H. Yang, D. Bogdanov, P. Herrera, and M. Sordo. Music retagging using label propagation and robust principal component analysis. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 869–876, 2012.

T. Yoshioka, T. Kitahara, K. Komatani, T. Ogata, and H. G. Okuno. Automatic chord transcription with concurrent recognition of chord symbols and boundaries. In *Proc. of the $5^{th}$ Int. Conf. on Music Information Retrieval (ISMIR)*, pages 100–105, 2004.

G. Yule and M. Kendal. *An Introduction to the Theory of Statistics*. Charles Griffin & Co., 1968.

M.-L. Zhang and Z.-H. Zhou. A k-nearest neighbor based algorithm for multi-label classification. In *Granular Computing, 2005 IEEE International Conference on*, volume 2, pages 718–721. IEEE, 2005.

M.-L. Zhang and Z.-H. Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *Knowledge and Data Engineering, IEEE Transactions on*, 18(10):1338–1351, 2006.

Z. Zhao, X. Wang, Q. Xiang, A. M. Sarroff, Z. Li, and Y. Wang. Large-scale music tag recommendation with explicit multiple attributes. In *ACM Multimedia*, 2010.

J. Ziv and A. Lempel. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3):337–343, 1977.

J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536, 1978.

E. Zwicker. Subdivision of the audible frequency range into critical bands. *Journal of the Acoustical Society of America*, 33(2):248–248, 1961.

E. Zwicker and E. Terhardt. Analytical expressions and critical bandwidth as a function of frequency. *Journal of the Acoustical Society of America*, 68(5):1523–1525, 1980.