FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Using Autotagging for Classification of Vocals in Music Signals

Nuno Hespanhol

DISSERTATION

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Fabien Gouyon

June 2013

Using Autotagging for Classification of Vocals in Music Signals

Nuno Hespanhol

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Gabriel de Sousa Torcato David
External Examiner: Paulo Jorge Freitas Oliveira Novais
Supervisor: Fabien Gouyon

June 2013

Abstract

Modern society has drastically changed the way it consumes music. During these last recent years, listeners have become more demanding in how many songs they want to have accessible and require to access them faster than ever before. The modern listener got used to features like automatic music recommendation and searching for songs that, for example, have “female vocals” and “ambient” characteristics. This has only been possible due to sophisticated autotagging algorithms. However, there has been an increasing belief in the research community that these algorithms often report over optimistic results. This work approaches this issue, in the context of automatic vocal detection, using evaluation methods that are rarely seen in literature.

Three methods are conducted for the evaluation of the classification model developed: same dataset validation, cross dataset validation and filtering. The cross dataset experiment shows that the concept of vocals is generally specific per dataset rather than universal as expected. The filtering experiment, which consists of iteratively applying a random filterbank, shows drastic performance drops, in some cases, from a global f-score of 0.72 to 0.27. However, these filters have been showed not to affect the human ear’s ability to distinguish vocals, by conducting a listening experiment with over 150 candidates.

Additionally, a comparison between two binarization algorithms - maximum and dynamic threshold - is performed and shows no significance difference.

The results are reported on three datasets that have been widely used within the research community, on which a mapping from its original tags to the vocals domain was performed and which is made available to other researchers.

Resumo

A sociedade moderna mudou drasticamente a maneira como consome música. Durante estes últimos anos, os ouvintes tornaram-se mais exigentes em relação ao número de músicas que querem ter acessíveis e querem acedê-las mais rapidamente do que anteriormente. O ouvinte moderno habituou-se a funcionalidades como recomendação automática de música e à possibilidade de pesquisar músicas com características, como por exemplo, "female vocals" e "ambient". Este tipo de funcionalidades só foram tornadas realidade devido a sofisticados algoritmos de autotagging. Contudo, existe uma crença pela comunidade de investigação que estes algoritmos reportam muitas vezes resultados demasiadamente otimistas. Este trabalho aborda este problema, no contexto de deteção automática de voz, usando métodos de avaliação raramente vistos na literatura.

Três métodos são realizados para a avaliação do modelo de classificação desenvolvido: validação entre o mesmo dataset, validação entre datasets e filtragem iterativa aleatória. A avaliação entre datasets mostra que o conceito de vocais é de uma maneira geral específico por dataset em vez de universal, como seria de esperar. A experiência dos filtros, que consiste em iterativamente aplicar um *filterbank* aleatório, mostra drásticas baixas na performance do sistema, em alguns casos, de um f-score global de 0.72 para 0.27. Contudo, através da realização de uma experiência perceptiva com mais de 150 candidatos, mostra-se que estes filtros não afectam a capacidade do ouvido humano de distinguir vocais.

Adicionalmente, é realizada também uma comparação entre dois métodos de binarização - máximo e limite dinâmico - que não mostra uma diferença significativa entre eles.

Os resultados são reportados em três datasets que foram largamente utilizados pela comunidade de investigação, sobre os quais é realizado um mapeamento das suas tags originais para o domínio vocal e disponibilizado para que outros investigadores possam usá-los.

Acknowledgments

I would like to thank several people that directly or indirectly made this thesis possible.

First and foremost, I must thank Prof. Fabien Gouyon for his guidance and support throughout this thesis. A special thanks to João Lobato Oliveira, that even though wasn't a formal co-supervisor to this work in many ways behaved as such providing invaluable insight and advice. I have also been assisted by members from the SMC group at INESC which I would like to thank.

Last but not least, I would also like to thank my family and closest friends for their comprehension, patience and support during the course of this work.

"I was born with music inside me. Music was one of my parts. Like my ribs, my kidneys, my liver, my heart. Like my blood. It was a force already within me when I arrived on the scene. It was a necessity for me like food or water."

Ray Charles

Contents

List of Figures

LIST OF FIGURES

List of Tables

Abbreviations

FEUP	Faculty of Engineering of the University of Porto
GMM	Gaussian Mixture Models
HMM	Hidden Markov Models
INESC	Institute for Systems and Computer Engineering of Porto
ISMIR	International Society for Music Information Retrieval
MFCC	Mel Frequency Cepstral Coefficients
MIDI	Musical Instrument Digital Interface
MIR	Music Information Retrieval
MIREX	Music Information Retrieval Evaluation eXchange
MSD	Million Song Dataset
SMC	Sound and Music Computing Group
SVM	Support Vector Machines

Chapter 1

Introduction

1.1 Motivation

Music is one of the most ancient art forms. It dates back to the prehistoric ages and has forever played a very important role in society. It is an an essential part of way of life across countries, religions and cultures. Indeed, there has been and always will be a need for society in general to be able to access and consume music.

In these last few years however, music consumption has changed drastically. There have appeared several commercial services such as iTunes, Amazon Music, Spotify and Last.fm that offer music collections in the order of the millions. In addition, these collections are constantly growing, therefore making the process of accessing and consuming them more difficult than ever before.

A way to make it easier for the listeners is to make use of metadata, that is, data about the data itself, normally called music tags. Music tags are simply keywords that are used to describe music content. Autotagging can then be defined as the process of automatically associating these kinds of tags with music content with the ultimate goal of helping future listeners to more easily find what they are looking for.

A fundamental question about this process of annotating music with tags is the implications of it being manual or automatic. Naturally, both have its advantages and disadvantages. It is clear that manual annotation will require more time, but will probably be more accurate than automatic processes.

A perfect example of the manual annotation process would be the Pandora[1] Internet Radio. It is a commercial service that offers its listeners the ability to create automatic intelligent playlists based on their musical preferences. In order for this to be possible, more than 40 musicologists have been paid to annotate musical files with over 400 attributes since 2000. Considering the ever growing number of music that are edited every day, it is easy to see how this is a big weakness of this kind of process. However, one may argue that the quality of the musical tags is superior to the ones obtained from automatic algorithms of tagging. Still, it is clear that it is an obvious limitation

[1] www.pandora.com

to annotate big collections of music such as iTunes and Amazon only manually, therefore, making automatic tagging a necessity.

1.2 Context

1.2.1 MIR Research

The work developed during the course of this dissertation is inserted in the field of Music Information Retrieval (MIR). MIR is the interdisciplinary science of retrieving information from music.

The first research works on automatic genre classification date back to 2001 [TC02]. As for music autotagging, the first research findings date back to late 2005 [ME05, TBL06]. Therefore, music autotagging is still considered a very recent research problem which is part of the research field of Music Information Retrieval (MIR). This research field most important conference is the International Society for Music Information Retrieval Conference (ISMIR) which first happened in 2000. MIR has reached a certain level of maturity and it's now entering what this society founders call it its "teen" years [DBC09], since its still in its early stages when compared to other research areas such as signal processing and speech processing [Sor12].

We would like computers to help us discover, manage, and describe the many new songs that become available every day. The goal of autotagging is not to replace humans: the best description of an album is still (and will most likely continue to be) the one of a music expert. It is impossible though for any group of experts to listen to every music piece on the Internet and summarize it in order for others to discover it. That is where autotagging comes in.

For instance, services like Pandora or Last.fm[2] both provide the feature of automatic recommendations based on listeners' musical taste. In order to achieve this, they assume that if a listener liked songs *A* and *B* and you liked *A*, you might like *B* as well. These kinds of algorithm have proven to be extremely efficient. However, it leaves two major problems:

- *cold start problem*: new songs on the music databases tend to not be very popular since they are still unknown and because of that they tend to not be tagged and ultimately never popular.
- *extreme popularity problem*: songs that eventually get popular will only tend to get more popular over time, since they are the ones being recommended all the time.

It is in these two particular cases that music autotagging algorithms can improve such services. Ultimately, we can say that the goal of music autotagging is to **do for music what *Google* and *Yahoo!* did for web documents and change the way we handle music** [BMEM10].

There have been attempts to unite the research community to join efforts in order to tackle this ultimate goal. That is how MIREX, **M**usic **I**nformation **R**etrieval **E**valuation e**X**change, was born in 2005. It consists of an annual evaluation campaign for MIR algorithms, including autotagging ones, which tries to rank these algorithms and to promote cooperation between researchers.

[2] www.last.fm

1.2.2 Research at INESC Porto

This dissertation was developed in partnership with the Sound and Music Computing Group[3](SMC) of the Institute of Engineering and Computer Systems of Porto[4](INESC). This research group combines basic research in signal processing, pattern recognition, music and human-computer interaction, and aims at contributing to make computers better understand, model and generate sounds and music.

1.2.3 Personal trajectory

The work presented in this dissertation spans over five months and it gave me the chance to apply some knowledge I have obtained during my private, professional and student life. The inclination for music started rather early having studied in music at a conservatory, where, apart from learning the guitar and piano, I also learned about composition and acoustics. Despite this I never considered working or studying in the music research area until the later years in my degree where I choose as an elective subject Automatic Music Generation. A great challenge developing this work had to deal with learning about Digital Signal Processing, in which I had no background whatsoever before starting this work in February, coming from a more Software Engineering background rather than an Electric Engineering one.

1.3 Objectives

The objectives for this dissertation were:

- to develop and thoroughly evaluate a system capable of identifying vocals in music signals;
- to propose new methods of evaluation that complement those commonly seen in literature such as *cross dataset testing* and *iterative random filterbank*;
- to do a critical study on current state-of-the-art vocals segmentation algorithms;
- to conduct a listening experiment with human candidates to validate the system and methods proposed.

1.4 Relevant Contributions

The most important contribution of this work is the systematic evaluation of an autotagging system in the context of the detection of the presence of human voice in music audio signals. Three different evaluations were applied to the system developed, two of which (*cross dataset* and *iterative random filterbank*) are rarely seen in literature and which present a new perspective on evaluation of autotagging systems. A *cross dataset* experiment showed that high performance in a

[3]http://smc.inescporto.pt
[4]http://inescporto.pt

typical cross-validation within the same dataset does not guarantee a generalization of the model to other dataset, while *iterative random filterbank* showed that such filtering has a drastic impact on a system's performance having been observed drops in global f-score from 0.72 to 0.27. Additionally, a mapping into the vocals domain of three datasets widely used within the research community was made available[5]. Lastly, some guidelines for conducting listening experiments with human candidates are described that could be of use to others researchers looking to conduct a listening experiment.

1.5 Structure of the Dissertation

This dissertation is structured in 7 chapters that somewhat resemble the chronologically way in which the present work took place. The current chapter, *Chapter 1*, contextualizes the research conducted, its motivations, aims and main contributions to the field. *Chapter 2* provides the theoretical background and explanations to many of the concepts that were of fundamental importance to the work developed, ending with a comparison of current state-of-the-art systems by other researchers. *Chapter 3* describes the process of creating the three datasets in which all the work was based. *Chapter 4* presents the framework that was used for the experiments conducted. *Chapter 5* details the experiments conducted and its results. *Chapter 6* describes the design and results of a listening experiment done with over 150 human candidates to further validate findings from the experiments reported in the previous chapter. Finally, *Chapter 7* ends with the discussion of the results, conclusions and future work.

[5] http://paginas.fe.up.pt/ ei08067/dokuwiki/doku.php.

Chapter 2

The Autotagging Ecosystem

This chapter presents the general framework of an autotagging system as well as the theoretical background of all the concepts it involves. It ends with a summary of the performance of some systems at the MIREX competition and a comparison of current state-of-the-art system for the task of voice segmentation.

2.1 Overview

A general overview of an autotagging system is given in Figure 2.1. Basically, the goal is to train a machine learning algorithm using a subset of a database of songs whose tags are already known. For this training to happen, the first step is to extract the low-level features of the sound signal. Then, the dimension reduction takes place, which tries to remove redundancy and simplify the dimensionality of the problem. That is when the machine learning algorithm is ready to learn the data. Many different algorithms can be used, which will be discussed in Section 2.5. This training will create a model that can then be used to automatically assign tags to the rest of the songs in the database. The final step is the evaluation by analyzing the performance of the system. This is possible because the data classified by the system is part of a dataset that is known beforehand. Metrics such as percentages of success and F-measures are used to access a system performance, as discussed in Section 2.6.

2.2 Feature Extraction

It would be very useful for any MIR task that music files were available under a structured and easy-to-parse format such as MIDI[1] or MusicXML[2]. However, that is not generally the case and music content is most of the times only available in a way that matters to the listeners, which is of a waveform. Therefore, the first step in virtually any MIR algorithm is to extract information that characterizes audio signals. This step is called *Feature Extraction*.

[1] Stands for Musical Instrument Digital Interface, a communication protocol introduced in 1983 that for the first time proposed a standard way to structure music data and communications between systems and applications.

[2] A standard open format to represent Western musical notation.

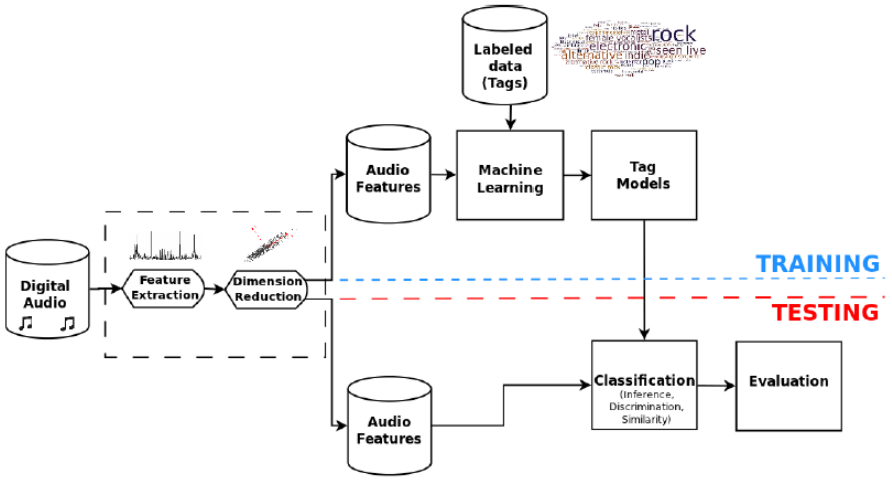

Figure 2.1: A generic autotagging system architecture. Adapted from [Sor12].

There are a vast amount of low-level features to describe signals, a few examples are Fourier Transform, Mel Frequency Cepstral Coefficients (MFCCs), chromagrams, autocorrelation coefficients, delta MFCC, double-delta MFCC, Zero Crossing Rate, the most important one being the MFCCs. In fact, the use of MFCCs is practically common to all tagging algorithms, having also been extensively used in the Speech Recognition field with great results. Additionally, it is common to do transformations of these features such as first and second-order derivatives to both create new features and increase the dimensionality of the features vector.

A brief explanation of the concepts involved in this process of Feature Extraction is given.

2.2.1 Concepts

2.2.1.1 Sampling Rate

The sampling rate is the number of samples taken per second from a continuous signal therefore turning it discrete. As an example, a sampling rate of 44.1 kHz means that for every second, 44100 samples (equally spaced in time) of a signal are recorded, which is equivalent to taking a sample every 0.0227 ms.

As Figure 2.2 clearly shows, the higher the sampling rate the closer to the original signal the representation is. Ideally, one would want to have the best representation of the signal possible, however, high sampling rates result in larger files, so the appropriate sampling rate should be set having that into account. Common sampling rates and where they are normally used are presented in Table 2.1.

2.2.1.2 Bit Depth

Bit depth is the number of bits available for recording the information of each sample. Like the sampling rate, the higher the bit depth the more information about the signal is recorded making it

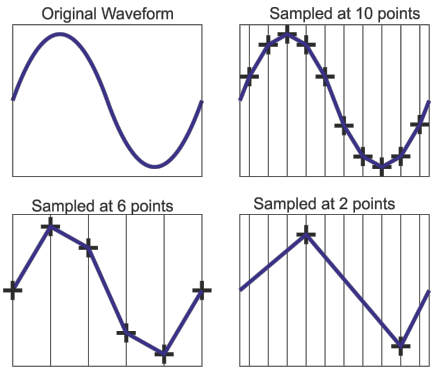

Figure 2.2: Different sampling rates for the same signal. Adapted from [Lab].

Sampling Rate	Maximum Frequency	General Use
8 kHz	3.6 kHz	Telephones
16 kHz	7.5 kHz	Modern VoIP services
22.05 kHz	10 kHz	Low quality MP3 and AM radio
44.1 kHz	20 kHz	Audio CD recordings
48 kHz	21.8 kHz	Professional Audio Equipment (i.e. mixers, EQs)

Table 2.1: Sampling rates, its maximum available frequencies and corresponding general uses [Aud].

both closer to the real continuous signal but on the other hand larger in size. As an example, a bit depth of 16 bits, allows for 65536 different amplitude values in a sample.

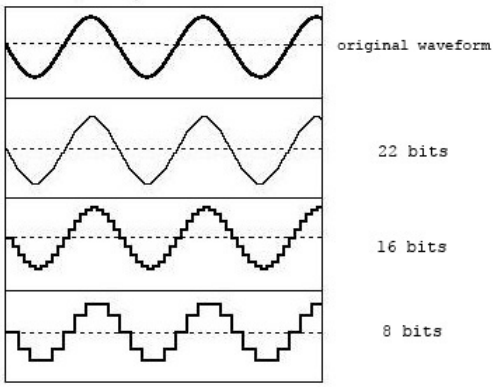

Figure 2.3: Different bit rates for the same signal. Adapted from [Gee].

Common bit depths and where they are normally used are presented in Table 2.2.

2.2.1.3 Number of channels

The number of channels are the independent sources of audio of a signal which is directly related to the spatialization perception of sound. In most of the cases, either one channel (mono or monaural)

Bit Depth	General Use
8 bits	Cassettes, FM radio, 80s video games
16 bits	Audio CD
24 bits	Professional Audio Equipment (i.e. mixers)

Table 2.2: Common bit depths and where they are normally used [Wik].

or two channels (stereo) are used. Stereo tries to simulate the impression of sound heard from two directions (left channel and right channel) as in natural hearing (left ear and right ear).

As it is common with most MIR algorithms that involve music processing, only monaural sounds were used. Since some of the datasets provided stereo files, they were converted to one channel only, using SoX[3], in a process illustrated by Figure 2.4.

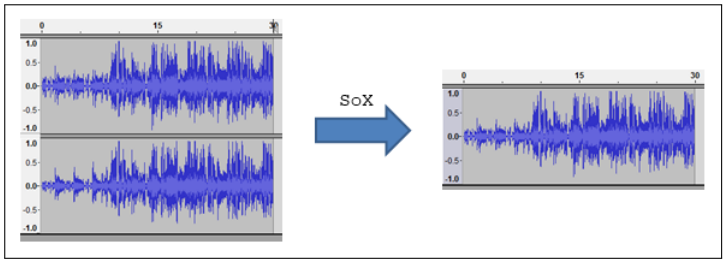

Figure 2.4: Converting a stereo signal to mono using SoX.

2.2.1.4 Window Function

Most real world audio signals are non-periodic, meaning that real audio signals do not generally repeat exactly, over any given time span. However, the math of the Fourier Transform assumes that the signal being Fourier transformed is periodic over the time span. This mismatch between the Fourier assumption of periodicity and the real world fact that audio signals are generally non-periodic, leading to errors in the transform. These errors are called "spectral leakage" and generally manifest as a wrongful distribution of energy across the power spectrum of the signal. To somewhat mitigate the "spectral leakage" errors, you can premultiply the signal by a window function designed specifically for that purpose, like for example the Hanning window function.

$$w(n) = 0.54 + 0.46\cos\left(\frac{2\pi n}{M-1}\right) \qquad 0 \leq n \leq M-1$$

2.2.1.5 Frequency Spectrum

The frequency spectrum of a signal is a representation of a time-domain signal in its corresponding frequency domain. In other words, it provides information about how much of a certain frequency

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

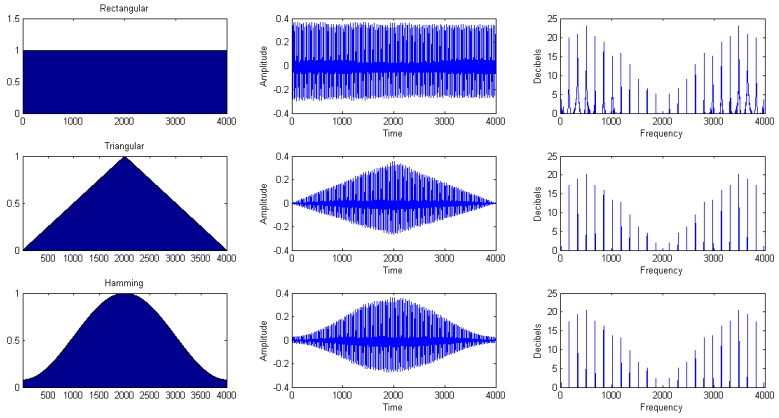

Figure 2.5: Comparison of window functions.

is present throughout the signal, which in many cases is more relevant that knowing how amplitude changes over time. It is generated by applying a Fourier Transform. As an example of the usefulness of the frequency spectrum, it is easy to tell in Figure 2.6 (bottom image) that no frequency content of 19,000 Hz or above is present in the signal, whereas such information is not directly available from the time-domain representation (top image). Many of the features used in MIR tasks are calculated over the frequency spectrum of a signal rather than its time-domain representation.

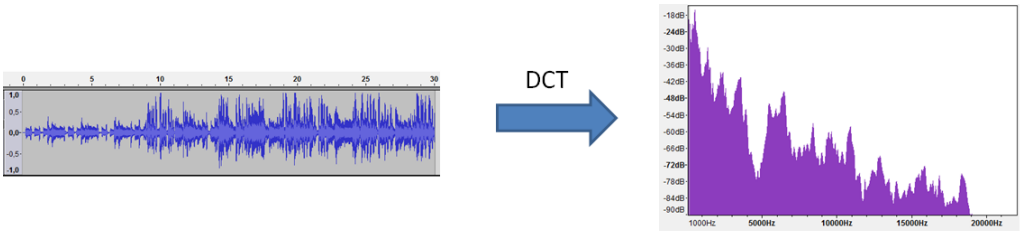

Figure 2.6: A signal and its corresponding frequency spectrum.

2.2.1.6 Analysis, Hop and Texture Window

Features are computed at regular time intervals called analysis windows that normally range from 10 to 60 ms. Most of the times these windows overlap between them by 50% (half-overlapping window), although it can vary. The offset between analysis windows is called the hop window, as can be seen in Figure 2.7. As an example, using a 10 ms half-overlapping window in a 44.1 kHz sampled signal, means that features will be calculated by considering 441 samples at each window and a offset of 220 samples between them. The concept of texture window is simply the number of analysis windows that are used to compute the averages (or standard deviations) of the features

to be extracted so that they can then be used as feature vectors in the classification phase by the machine learning algorithms (Section 2.5).

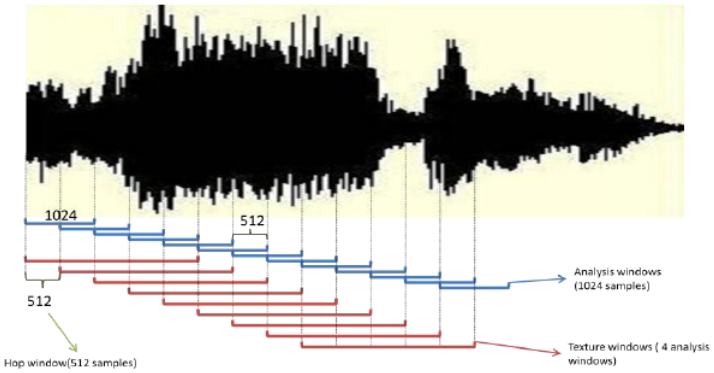

Figure 2.7: Example of a segmentation of a file into half-overlapping windows. Adapted from [All08].

However not as common, variable time intervals that coincide with musical events have been used too. For instance, [WC05] and [SZ05] adjust the time windows according to rhythm events, namely beats, as Figure 2.8 illustrates.

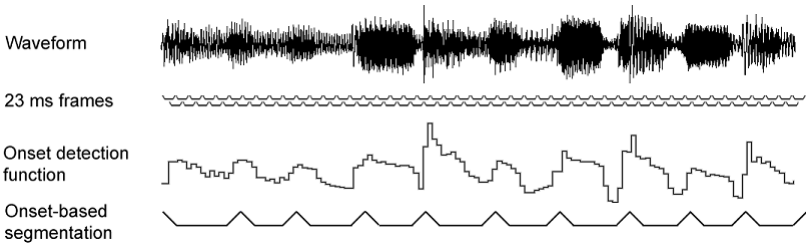

Figure 2.8: Example of variable time windows matching beats. Adapted from [WC05].

2.2.2 Zero Crossing Rate

The Zero Crossing Rate (ZCR) is the rate at which the signal changes from positive to negative and vice-versa. The ZCR of unvoiced sounds and environmental noise are usually larger than voiced sounds, which has observable fundamental periods [Jan]. Figure 2.9 shows the zero crossings for a given window of a signal, whenever the signal crosses the x-axis at zero amplitude.

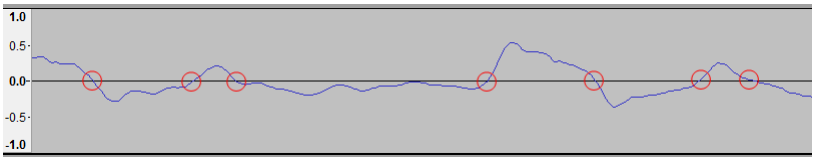

Figure 2.9: Zero crossings for a window of a signal marked in red.

2.2.3 Spectral Centroid

The spectral centroid indicates where the "center of mass" of the frequency spectrum is. It is calculated as the weighted mean of the frequencies present in the signal, determined using a Fourier transform, with their magnitudes as the weights. Perceptually, it has a robust connection with the impression of "brightness" of a sound, with higher values corresponding to brighter textures [GG78]. Figure 2.10 shows the centroid for a given window of a signal, clearly located towards the frequencies that are more present in that frame.

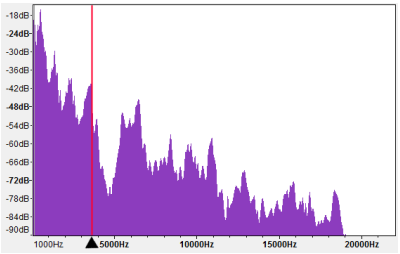

Figure 2.10: Spectral Centroid of a given frame of a signal indicated by the red line.

2.2.4 Spectral Roll-off

Spectral roll-off point is defined as the Nth percentile of the power spectral distribution, where N is usually 85% or 95%. The roll-off point is the frequency below which the N% of the magnitude distribution is concentrated. This measure is useful in distinguishing voiced speech from unvoiced: unvoiced speech has a high proportion of energy contained in the high-frequency range of the spectrum, where most of the energy for voiced speech and music is contained in lower bands [SS97]. Figure 2.11 shows the roll-off frequency for a given frame of a signal, clearly distinguishing where most of the energy of the signal is.

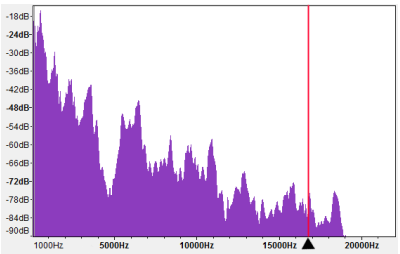

Figure 2.11: Roll-off frequency of a given frame of a signal indicated by the red line.

2.2.5 Spectral Flux

Spectral Flux is a measure of how quickly the power spectrum of a signal is changing, calculated by comparing the power spectrum for one frame against the power spectrum from the previous frame. The spectral flux is particularly useful for modeling the timbral characteristics of a signal.

As a quick summary of these spectral features, Figure 2.12 shows how there is a clear different behavior for each of them over *music*, *speech* and *noise*.

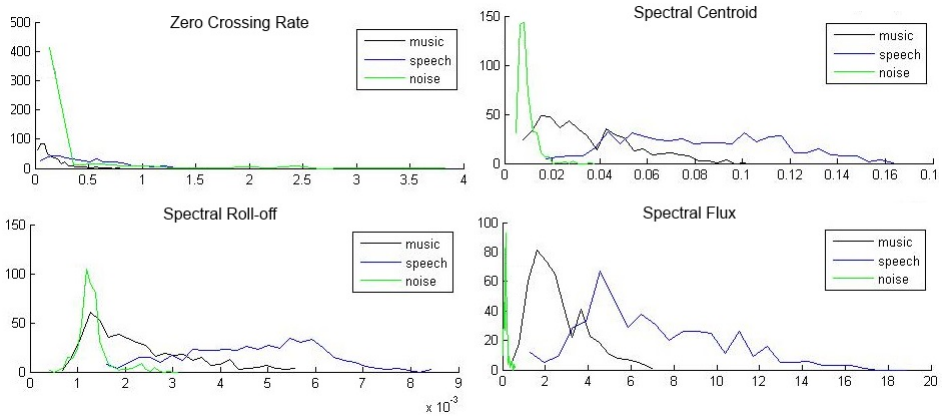

Figure 2.12: Spectral Features change in Music, Speech and Noise. Adapted from [Gia08].

2.2.6 MFCCs

The first step towards understanding the MFCCs is understanding the Mel Scale.

The Mel scale was created by [SVN37] and is widely used in tasks involving pitch perception as it is the case with many MIR applications. It can be defined as *a perceptual scale of pitches judged by listeners to be equal in distance from one another*.

This scale relates perceived frequency, or pitch, of a pure tone to its actual measured frequency. Humans are much better at discerning small changes in pitch at low frequencies than they are at high frequencies. Incorporating this scale makes our features match more closely what humans hear.

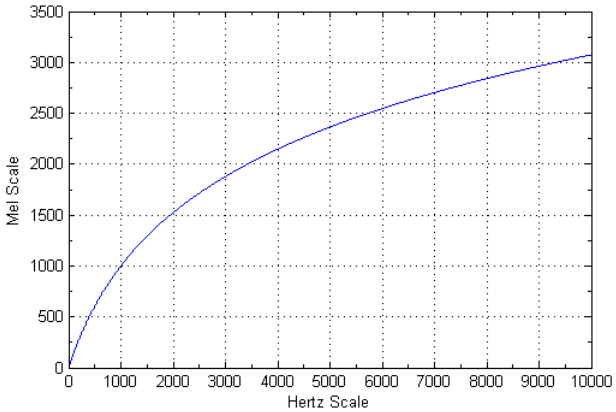

Figure 2.13: The Mel Scale.

It was built by playing experimental subjects a reference tone and asking them to adjust a second tone until it sounds twice as high or half as high in pitch. By varying the reference tone

and by testing a large number of subjects it was possible to build up a scale which relates pitch (in Hertz) to a subjective perceptual scale of pitch. 1000 Hz was arbitrarily selected and its pitch in mels was chosen to be 1000 mels. 500 mels is perceived as half the pitch of 1000 mels, whilst 2000 mels is perceived as being twice the pitch of 1000 mels.

The MFCCs are the coefficients of the Mel Frequency Cepstrum (MFC). Usually, MIR applications only make use of the first 13 coefficients. They were introduced by [DM80] in the 1980's, and have been state-of-the-art ever since [mfc]. It is also common to use the first and second order derivatives of the MFCC, which are called ΔMFCCs (delta) and ΔΔMFCCs (double delta), respectively. The process for obtaining these coefficients can be described as follows and as Figure 2.14 schematically shows:

1. Calculate the FFT of the signal, in windows.
2. Map the spectrum obtained onto the mel scale, using the Mel Filterbank (Figure 2.15).
3. Compute the logarithm of the powers at each of the mel frequencies (since the human ear perceives loudness logarithmically rather than linearly[4]).
4. Apply the Discrete Cosine Transform[5] (DCT) of the list of mel log powers.
5. The MFCCs are the amplitudes of the resulting spectrum, which is called the Mel Cepstrum.

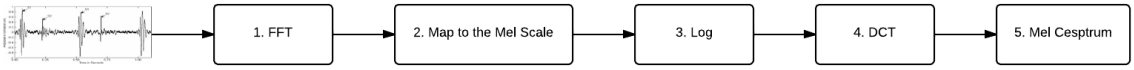

Figure 2.14: MFCC calculation process.

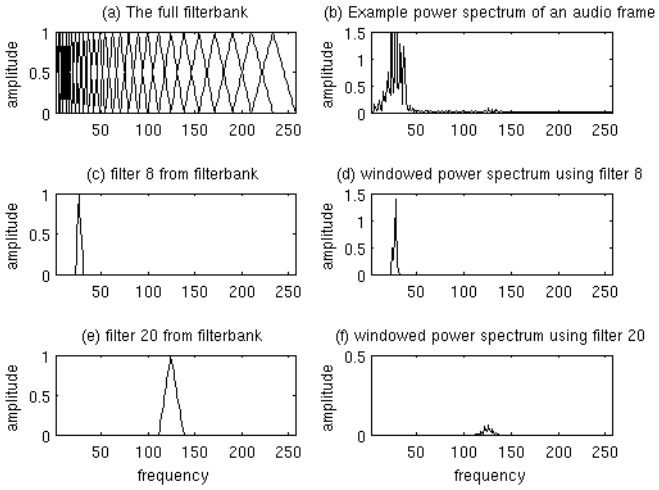

Figure 2.15: The Mel Filterbank. Adapted from [mfc].

[4] As an example, for a human ear to perceive a sound two times more loud, it is necessary to apply a eight times more energy to it

[5] It is very similarly to the DFT, but, instead of using harmonically-related complex exponential functions, it uses real-valued cosine functions.

2.3 Dimension Reduction

During the Feature Extraction process a lot of data can arise, many of which (i) might not even be relevant to the classifier or (ii) might be too much for it to be able to compute in reasonable time. Therefore, methods have been proposed to reduce the dimension of this problem, many times without loosing any information at all. Some of them are:

- Principal Component Analysis (*PCA*)
- Linear Discriminant Analysis (*LDA*)
- Independent Component Analysis (*ICA*)
- Non-negative Matrix Factorization (*NMF*)
- Relevant Component Analysis (*RCA*)

In the framework used and described in Chapter 4, considering the relatively low number of features to be used no Dimension Reduction algorithm will be used. Therefore, the description of these algorithms is out of the scope of the present dissertation.

2.4 Data

As mentioned in Section 2.4, there are many ways to obtain tags, namely through surveys, social tags, games and/or web documents. The datasets used for training the classifier are of extreme importance in the training process. It is obvious that if the dataset is poorly notated, the classifier will be badly trained and consequently it will classify many inputs wrongly. That is why there are many datasets shared among the research community such as the ones presented in Chapter 3 that took careful consideration to be built.

Music tags are simply keywords that are used to describe music content. For example, if we consider the song *Smoke on the Water* by *Deep Purple*, its music tags directly extracted from the *Last.fm* site, are shown below. They are represented under the format of a tag cloud, in which each tag is bigger according to its frequency of tagging.

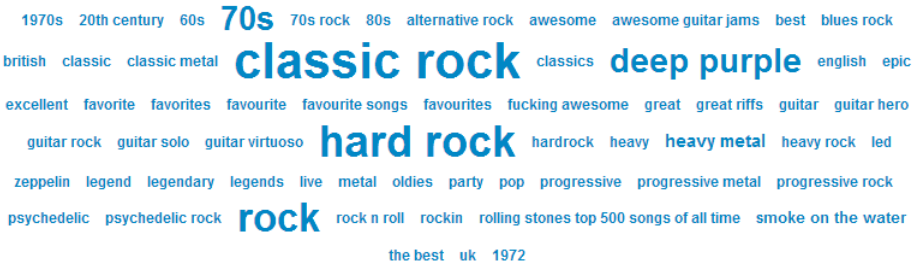

Figure 2.16: Tag cloud for the song “Smoke on the Water” by “Deep Purple” on Last.fm.

Music tags can be related to various facets of the music:

- Emotion (e.g. epic)

- Musical instruments (e.g. guitar solo, keyboards)
- Genre (e.g. rock, pop, reggae)
- Date (e.g. 70s, 80s, 90s)
- Usage (e.g. seen live, riot, war)

There are also several ways to use metadata to describe the music, but they fall into three basic categories: editorial metadata, cultural metadata, and acoustic metadata [Pac11].

- **Editorial**: data obtained directly from the editor, such as song titles, album name, date and place of recordings, composers, performers, etc.
- **Cultural**: information that is produced by the environment or culture, which typically results from the analysis of emerging patterns, categories or associations of sources like Google searches, Wikipedia articles or music databases (i.e. Allmusic[6])
- **Acoustic**: it is the entirely objective kind of metadata regarding the music content. Typical examples are beats per second, metric measure, low-level descriptors (Mel Frequency Cepstral Coefficients, MFCCs).

They can be obtained through five different methods [TBL08]:

- **Surveys**: paying people (musicologists or not) to assign tags to music files. This is how the dataset CAL500[7] was built through the payment to undergraduate students to do that job.
- **Social tags**: the most paradigmatic example of this kind of tagging is the popular site Last.fm where users voluntarily tag songs. Recently, there has been a keen interest on literature on how to better make use of this this kind of tagging. There are naturally a few obvious downsides to it, as for example, the quality of the tags being low, but, it has been show to improve a lot of autotagging algorithms.
- **Games**: there have been authors exploring the gamification of music tagging with great results, such as ListenGame [TLB07], Tag-a-Tune [LvAD07], and MajorMiner [ME07]. For instance, the Tag-a-Tune game is a two-player game where the players listen to a song and are asked to enter "free text" tags until they both enter the same tag.
- **Web documents**: this method for obtaining tags is done through web mining of music websites, artist biographies, album reviews, songs reviews, Wikipedia articles.
- **Autotags**: all previous methods require the tagging process to be done by humans. This method relies on automatic processes by computers to do so, and it consists of the system this dissertation is going to explore.

[6] www.allmusic.com
[7] http://cosmal.ucsd.edu/cal/projects/AnnRet/

2.5 Classifier

Machine Learning is the branch of Artificial Intelligence that studies the construction of systems that can learn from large amounts of data. Therefore, it uses generalization to solve many of its problems. In the particular case of music autotagging, the basic idea is to have a relatively large database of songs (see Chapter 3) from which the machine learning algorithms can construct a model that can then be applied to new songs the model has never seen before.

Machine Learning algorithms are divided into three main categories:

- Supervised learning: when there is a what is a called a **ground truth** set of elements to train the model, before actually running it on new data. Ground truth means facts that are assumed as a fundamental truth. This kind of approach to training machine learning systems is the most popular in music autotagging algorithms and is the one to have shown better results. Hence, so much effort have been put into creating ground truth databases of songs (see Chapter 3).

- Unsupervised learning: when there is no prior knowledge about the elements to be learned. This kind of learning is not very common in music autotagging, although it has been used in music genre classification, which is a particular case of the music autotagging problem.

- Semi-supervised learning: this approach consists of a mixture of the previous two - it combines the use of both labeled and unlabeled data to create a classifier. Again, in the case of music autotagging, this would mean to have both songs in the training database with music tags and others with no music tags.

This module of the autotagging system is the one responsible for automatically linking tags to audio features. There are many algorithms:

- Support vector machines (*SVM*)
- Gaussian Mixture Models (*GMM*)
- K-Nearest Neighbors (*KNN*)
- Artificial Neural Networks (*ANN*)

SVMs are one of the most widely used machine learning algorithms. Their performance as a classifier has been demonstrated and they can also handle regression, but they have the big disadvantage with its training speed, which is in the order of N^2, N being the number of audio files in the database. Since this will be the classifier to be used in framework (Chapter 4) a more detailed explanation is given below (Section 2.5.1).

Gaussian mixture is a learning algorithm that models a distribution using gaussians. Gaussian mixtures are more powerful (in terms of representation capacity) than an algorithm that only classifies because it estimates the likelihood of a data point.

K-Nearest Neighbors (KNN) is one of the most simple, and surprisingly effective, machine learning techniques [SLC07].

Neural networks have not been been used in recent papers on automatic tagging of audio, but there is no reason why it would not work efficiently [BMEM10].

2.5.1 Support Vector Machines

SVMs are a type of supervised learning approach pattern recognition algorithms. They basically binary classify an input instance as one out of two classes, having previously been trained with some labeled data. Its process can be described as follows.

1. Inputs are formulated as feature vectors (in our case using the features of the signal).
2. Mapping these vectors into a feature space by the use of a kernel function.
3. Optimally separate the classes of training vectors to build a model of classification.

There are three main types of SVM classifiers: the hard-margin, soft-margin and non-linear.

2.5.1.1 Hard Margin Linear SVM

There are lots of possible solutions for choosing the plane (or hyperplane if in a higher dimensional space) to separate two classes of data in the feature space. As can be seen in Figure 2.17 both the black and green planes separate the pink and blue classes, and there is an infinite more number of planes that do so. SVM, however, guarantees that it will find the optimum solution to maximize its margin to the feature vectors (the instances of the data that are in the dotted-lines and that directly affect the selection of the optimum solution). In Figure 2.17 it's clear how both black and green planes are a solution, but with the green providing a bigger margin and therefore yielding better classification results.

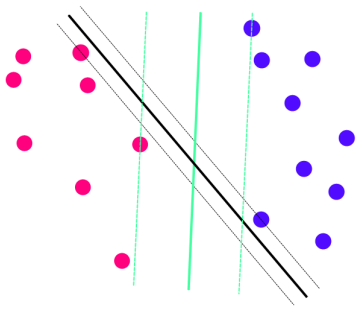

Figure 2.17: A hard-margin linear SVM example.

2.5.1.2 Soft Margin Linear SVM

There will be times however, where the data won't allow for linear separability of its classes (even in higher dimensions). A soft-margin SVM solves this problem by allowing mislabeled examples. It then tries to find the plane (or hyperplane) that splits the examples as cleanly as possible, while still maximizing the distance to the nearest clean example. The degree of allowed mislabeled

examples is controlled through the C constant called *Slack variable*. Figure 2.18 clearly illustrates how this trade-off of allowing for a few mislabeled examples (circled in dark green) for the cost of linearly separating the two classes.

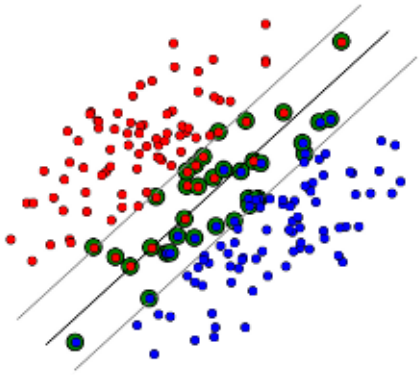

Figure 2.18: A soft-margin linear SVM example.

This kind of SVM gives a greater error on the training dataset comparing to a hard-margin SVM, but improves generalization to the test dataset, not to mention making it possible to linearly separate data which wouldn't be possible otherwise.

2.5.1.3 Nonlinear SVM

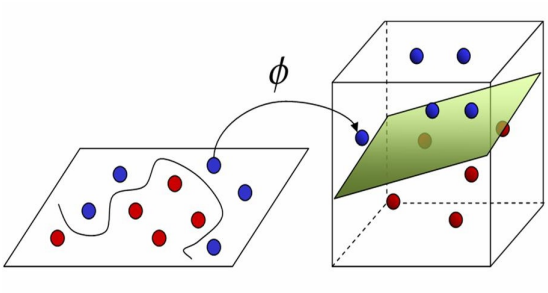

Figure 2.19: Kernel function transformation to higher dimensional space example.[8]

The idea of a nonlinear SVM is to gain linearly separation by mapping the data to a higher dimensional space. The following set on the right of Figure 2.19 can not be separated by a linear function, but can be separated by a quadratic one. A ϕ transformation is then applied to the input vectors to map them to a higher dimension space, through the means of a kernel function. The most popular choice is the radial basis function kernel, also known as RBF and is expressed by the following equation, being γ the only variable parameter:

$$K(x,x') = e^{-\gamma \|x-x'\|^2} \text{ for } \gamma > 0$$

[8] http://www.imtech.res.in/raghava/rbpred/algorithm.html

2.6 Evaluation

This part of the system consists of evaluating whether the automatic tagging was done correctly or not. Most commonly, autotagging system publications present their evaluation statistics considering two scenarios: globally (considering all tags of a system) and per tag (considering the average of the performance per tag).

In classification algorithms, as it is the case with any autotagging algorithm, it is important to bear in mind the so called *Type I* and *Type II* errors. *Type I* errors account for the False Positives (*FP*), that is the instances that are considered of a class but aren't, whereas *Type II* errors account for the False Negatives (*FN*), that is the instances that aren't considered of a class but should. A more visual explanation of this can be seen in Table 2.3.

		Predicted Class	
		True	**False**
Actual Class	**True**	True Positive (TP)	Type I Error
	False	Type II Error	True Negative (TN)

Table 2.3: Truth Table highlighting *Type I* and *Type II* errors in red.

In the same line of reasoning, a typical way of presenting results of machine learning classification methods, it's through a confusion matrix, as shown below in Table 2.4 for the context of our work.

		Predicted Class	
		Vocals	***Nonvocals***
Actual Class	***Vocals***	220	2
	Nonvocals	22	7

Table 2.4: Example of a confusion matrix.

This basically means that:

- 220 excerpts were correctly identified as having *Vocals*.
- 2 excerpts had someone singing, but the system assumed they didn't (*Type I error*).
- 22 excerpts didn't have anyone singing, but the system assumed they did (*Type II error*).
- 7 excerpts were correctly predicted as being *Nonvocals*.

While the confusion matrix might be a good way to visually present machine learning classification results, for actually benchmarking a system it is common to summarize all of these four variables (*TP, TN, FP, FN*) using the following evaluation metrics, each of them with a particular meaning on a system's performance:

Precision is the ratio of predicted classes that are relevant.

$$Precision = \frac{TP}{TP+FP}$$

Recall is the ratio of relevant classes that were predicted.

$$Recall = \frac{TP}{TP+FN}$$

F-score is a weighted harmonic mean average measure of both precision and recall.

$$Fscore = \frac{2}{\frac{1}{p}+\frac{1}{R}} = 2\frac{PR}{P+R}$$

Accuracy is the ratio of correctly predicted classes over all possible classes.

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN}$$

2.7 MIREX

The Music Information Retrieval Evaluation eXchange (MIREX) is an annual evaluation campaign for Music Information Retrieval (MIR) algorithms, coupled to the International Society (and Conference) for Music Information Retrieval (ISMIR) [15]. MIREX project tries to rank MIR algorithms in each of its categories and above all to promote cooperation between researchers. There are many categories, for the music autotagging task the ones relevant are **Audio Genre Classification**, **Audio Music Mood Classification** and in particular **Audio Tag Classification**.

The MIREX defines the Audio Tag Classification task as "This task will compare various algorithms' abilities to associate descriptive tags with 10-second audio clips of songs". The first edition of this task was in 2008 and was held ever since to 2012, the last happening of ISMIR, conference in which MIREX results were disclosed and presented below in Table 2.5 and Table 2.6.

As mentioned in Section 2.6 the main metric used for ranking the tagging systems is the F-measure.

MIREX divides the Audio Tag Classification task into two, considering they are tested against two different databases of songs, namely MajorMiner[9] and Mood[10] datasets.

The algorithm that performs the best in the Major Miner dataset is the [Ham11], which uses a Principal Mel-Spectrum Components and a combination of temporal pooling functions.

[9] http://majorminer.org/info/intro

[10] The Mood tag dataset is derived from mood related tags on last.fm

ID	Participants	F-Measure	AUC-ROC
PH2	Philippe Hamel	0.50	0.88
SSKSS1	K. Seyerlehner, M. Schedl, P. Knees, R. Sonnleitner, J. Schluter	0.49	0.89
BA2	Simon Bourguigne, Pablo Daniel Aguero	0.49	0.77
BA1	Simon Bourguigne, Pablo Daniel Aguero	0.49	0.78
GT2	George Tzanetakis	0.47	0.89
RWJ1	Jia-Min Ren, Ming-Ju Wu, Jyh-Shing Roger Jang	0.46	0.85
RWJ2	Jia-Min Ren, Ming-Ju Wu, Jyh-Shing Roger Jang	0.46	0.84
RJ1	Jia-Min Ren, Jyh-Shing Roger Jang	0.44	0.83
HN1	Jorge Herrera, Juhan Nam	0.34	0.70

Table 2.5: MIREX 2012 results for *Audio Tag Classification - Major Miner* task.

ID	Participants	F-Measure	AUC-ROC
SSKSS1	K. Seyerlehner, M. Schedl, P. Knees, R. Sonnleitner, J. Schluter	0.49	0.87
PH2	Philippe Hamel	0.46	0.84
RWJ1	Jia-Min Ren, Ming-Ju Wu, Jyh-Shing Roger Jang	0.45	0.76
RJ1	Jia-Min Ren, Jyh-Shing Roger Jang	0.43	0.74
RWJ2	Jia-Min Ren, Ming-Ju Wu, Jyh-Shing Roger Jang	0.43	0.77
BA1	Simon Bourguigne, Pablo Daniel Aguero	0.42	0.78
BA2	Simon Bourguigne, Pablo Daniel Aguero	0.41	0.77
GT2	George Tzanetakis	0.37	0.86
HN1	Jorge Herrera, Juhan Nam	0.37	0.69

Table 2.6: MIREX 2012 results for *Audio Tag Classification - Mood* task.

Chapter 3

Dataset Construction

One of the crucial steps towards having valid experiments is the use of ground truth datasets. It is strongly encouraged that these datasets be (i) as balanced as possible (similar number of instances per tag), (ii) as complete as possible (describe music excerpts in all possible ways) and (iii) shared among the scientific community (to encourage exhaustive evaluation and comparison among the different algorithms) [Sor12].

Therefore, this work makes use of datasets that already existed and have been widely used within the research community. These are the base datasets from which all experiments derived:

- CAL500: collection of 500 songs (actually 502) tagged with 174 tags by paid human labelers. 500 songs annotated using a vocabulary of 174 tags from 8 semantic categories that describe the genre (multiple and best), emotion, instruments, solos, vocal style, song characteristics and usage. Both binary (relevant / irrelevant) and affinity labels are included.

- Magtag5k[1] : processed version of the original Magnatagatune[2] dataset, a research dataset for MIR tasks, such as automatic tagging. Details about the preprocessing applied on the Magtag5k can be found in [MDLG11].

- MSD24k: processed version of the original Million Song dataset[3] (MSD), another research dataset for MIR tasks, such as automatic tagging. The audio files were obtained by crawling the Amazon web site. This dataset consists of 23740 songs and 265 tags.

[1] http://inescporto.pt/ fgouyon/data/magtag5k.zip
[2] http://tagatune.org/Magnatagatune.html
[3] http://labrosa.ee.columbia.edu/millionsong

	CAL500	**Magtag5k**	**MSD24k**
Method	paid human labelers	from a game	research created
#Songs	502	5261	23740
#Tags	174	123	264

Table 3.1: Statistics for the datasets used in the experiments.

Even though some datasets already existed within the research community for the task of identifying vocals, none provided annotated tags about other musical aspects like genre, emotion and instrumentation just to name a few. Since there was interest to tackle the identification of vocals from an autotagging point of view, it was necessary to create a ground truth dataset. For that, the songs were classified into two different classes:

- *Vocals*
- *Nonvocals*

A definition of these classes is necessary. For *Vocals*, all songs (or excerpts of songs) that contained any kind of human voice were considered. This definition covers a very large range of songs, from traditional pop/rock songs with a lead singer to someone speaking through a megaphone and even tuvan throat singing, among many others not so traditional examples that were found in the dataset. In the end, if there was a human voice (distorted or not) in the song, even if for a very short period of time, it was considered of class *Vocals*.

Naturally, this definition raises some issues (as any other one would). One may argue, for example, that a song in which the singer speaks through a megaphone - as it is the case in these datasets, more than once actually - is not very representative of a class *Vocals*. However, as it is well known and an unavoidable fact, music tagging is a field characterized by noisy information and, using this kind of definition, ground truth dataset that more closely reflects the way real users tag is being defined. So, ultimately, the definition not being perfect ends up being an asset instead of a disadvantage.

There were some cases, though, in which it was very difficult to distinguish whether a particular song had the presence of human voice or not. These particular cases, were considered *Dubious* and therefore were not included in the experiments. It didn't make sense to include songs in the ground truth dataset if not even a human was able to tell if they should be *Vocals* or *Nonvocals*. It is important to understand that these kind of *Dubious* cases are different from the issues raised in the previous paragraph regarding the definition of *Vocals*: one thing is to have a distorted human voice in a song and being able to identify it as such (previous paragraph) and another one is not being able to identify whether a song contains a human voice or not, probably because the timbre of a particular sound in that song is too unclear to allow a human listener to do that (*Dubious* case).

In general, the process of separating the datasets in the *Vocals* and *Nonvocals* classes, or in other words, creating a ground truth dataset for the vocals domain, was as follows:

1. Separate the songs through a direct mapping between their already existing tags and the vocals domain (i.e. songs tagged with tag *backing_vocals* are of class *Vocals* and songs tagged with tag *no.singing* are of class *Nonvocals*).

2. Listen to all songs and apply necessary corrections.

The first step could be basically omitted, because in the end, listening through all songs would yield essentially the same result. However, as it will be clear in the next sections, this allowed for an easier and faster way to create the new classifications required and made it easier on the listener too, since he could, for example, go through all the *Vocals* first and look for false positives there and then go through for the *Nonvocals* and do the same thing, without having to change focus at every song, which would most probably turn up to be more time consuming and more error-prone.

All of the three datasets used (CAL500, Magtag5k and MSD24k) have different taxonomies, therefore, different mappings were made for each one in step one and are described separately below. Also, the ground truth datasets created have been all manually checked, thus, its reliability is quite high.

3.1 CAL500

The CAL500 dataset has been extensively used in Music Information Retrieval before. However, surprisingly enough, some inconsistencies in the data still remain. [DM80] has addressed this concern and alterations to the dataset have been done accordingly. Some of these issues include missing/incomplete audio files and duplicate songs, among a few others.

For the separation of this dataset, the following tags were considered for the *Vocals* class. All the other songs where put into the *Nonvocals* class.

```
Backing Vocals
Female Lead Vocals
Male Lead Vocals
Aggressive
Altered With Effects
Breathy
Call - Response
Duet
Emotional
Falsetto
Gravelly
High-pitched
Low-pitched
Monotone
Rapping
Screaming
Spoken
Strong
Vocal Harmonies
Female Lead Vocals-solo
Male Lead Vocals-solo
```

Listing 3.1: CAL500 tags for *Vocals*

After this initial separation, every song was listened to, to make sure there weren't any false positives (*Vocals* that in fact do not have any vocal information) and false negatives (*Nonvocals* that in fact have vocal content).

	Vocals	**Nonvocals**	*Totals*
Before Listening	433	69	502
True	433	58	491
False	0	11	11
Dubious	0	0	0
After Listening	444	58	502

Table 3.2: Listening evolution in CAL500.

3.2 Magtag5k

The Magtag5k dataset is the one with the most compact vocabulary (see Table 3.1). Therefore, there were fewer tags to consider for the class *Vocals* in the initial phase, as can be seen in Table 3.7. Other particularity of it was the existence of the tag *no.singing*. Therefore, in this dataset, instead of considering all the songs that weren't *Vocals* as *Nonvocals* (as it was it was done in the CAL500), only those with this *no.singing* tag (Listing 3.3) were considered. This allowed for a fewer number of false negatives (*Nonvocals* that in fact have vocals content), since there is naturally a much higher confidence in this separation when compared to the other case in which it is assumed a song is *Nonvocals* only because it wasn't tagged with a *Vocals* tag. In addition, it also allowed for an easy way to reduce the number of songs to manually having to listen to (from 3665 to 797). 797 *Nonvocals* was enough for the work intended to be developed.

```
1 female.singing
2 singing
3 man.singing
```

Listing 3.2: Magtag5k tags considered for the *Vocals* class.

```
1 no.singing
```

Listing 3.3: Magtag5k tags considered for the *Nonvocals* class.

As it was done with the CAL500 dataset, the separation was manually validated and the evolution can be seen below. An example of a dataset construction log file created can be found in Appendix A.

	Vocals	**Nonvocals**	*Totals*
Before Listening	1596	797	2393
True	1541	796	2337
False	1	55	56
Dubious	0	8	8
Other	0	13	13
After listening	1627	724	2351

Table 3.3: Listening evolution in Magtag5k.

3.3 MSD24k

The MSD24k is by far the biggest dataset of the three, in both number of songs and in number of tags in vocabulary (see Table 3.1). For this reason, a more careful analysis was done to avoid having to manually listen to a large number of songs.

As a consequence of this big number of tags, this dataset has a lot of tags related with the vocals domain, namely tags for very specific vocal content (lyrics, rapping, speech, among others). Since there was a greater interest in the more general singing kind of songs, rather than more specific examples such as speech and rapping for instance, and having this option, it was chosen not to include them in the *Vocals* class, and naturally, not in the *Nonvocals* class too. Basically, only the tags that were quite representative of the presence of voice were considered for the *Vocals* class (Listing 3.4), i.e *a_female_vocal* vs. *abstract_lyrics*. However, there were some songs that used both tags from Listing 3.4 and Listing 3.5, which in that case, were considered. Only songs with tags from Listing 3.5 and with no tags from Listing 3.4 were not considered.

```
a_breathy_male_lead_vocalist
a_distinctive_male_lead_vocal
a_dynamic_female_vocalist
a_dynamic_male_vocalist
a_female_vocal
a_gravelly_male_vocalist
a_laid_back_female_vocal
a_smooth_female_lead_vocal
a_smooth_male_lead_vocalist
a_vocal-centric_aesthetic
an_aggressive_male_vocalist
an_emotional_female_lead_vocal_performance
an_emotional_male_lead_vocal_performance
jazz_vocals
```

Listing 3.4: MSD24k tags considered for the *Vocals* class.

```
a_poetic_rap_delivery
a_repetitive_chorus
a_subtle_use_of_paired_vocal_harmony
```

```
a_subtle_use_of_vocal_counterpoint
a_subtle_use_of_vocal_harmony
abstract_lyrics
ambiguous_lyrics
an_unintelligible_vocal_delivery
angry_lyrics
clear_pronunciation
consistent_rhyme_patterns
explicit_lyrics
french_lyrics
funny_lyrics
great_lyrics
heartbreaking_lyrics
heavy_use_of_vocal_harmonies
humorous_lyrics
interweaving_vocal_harmony
narrative_lyrics
offensive_lyrics
paired_vocal_harmony
political_lyrics
romantic_lyrics
sad_lyrics
southern_rap
spoken_word
upbeat_lyrics
use_of_call-and-response_vocals
vocal_duets
vocal_harmonies
vocal_samples
```

Listing 3.5: MSD24k tags related to vocals, but, for the reasons mentioned above, weren't considered for the class *Vocals*.

As already mentioned, this dataset is quite large compared to both the CAL500 and Magtag5k. In fact, after this initial separation into *Vocals* and *Nonvocals*, the *Nonvocals* class was with 20519 songs against 1178 in the *Vocals*. There were two reasons for reducing the *Nonvocals* class. Firstly, it was impossible to manually check all of the 20519 *Nonvocals* songs without distributing the work to some kind of community and that was not an option. Secondly, having so many *Nonvocals* songs wouldn't be a balanced dataset, that, as mentioned in Chapter 3 is of the utmost importance when defining a ground truth dataset.

In order to reduce the number of *Nonvocals* songs, it was done an analysis of the occurrences of each tag in the first separation of the *Vocals* and *Nonvocals* classes, as presented in the table below. The ratio of the occurrence of each tag in the *Nonvocals* on this initial separation is also presented.

$$\text{Ratio}(\text{tag}_x) = \frac{\text{number of occurrences of tag}_x \text{ in } \textit{Nonvocals}}{\text{number of occurrences of tag}_x \text{ in } \textit{Vocals} \; + \; \text{number of occurrences of tag}_x \text{ in } \textit{Nonvocals}}$$

That basically means that if the ratio was high, most of that tags occurrences was in the *Nonvocals* dataset, therefore, there was more confidence that particular tag would be a good choice

for the *Nonvocals* class. Again, the whole dataset was simply too big for manual verification, so, just the tags that looked promising were picked, both taking into account the ratios and general musical knowledge.

Tag ▲	**#Nonvocals**	**#Vocals**	**Ratio**
a_breathy_male_lead_vocalist	0	106	0.0
a_busy_bass_line	4	2	0.67
a_busy_horn_section	11	134	0.08
a_clear_focus_on_recording_studio_production	12	284	0.0
⋮	⋮	⋮	⋮
vocal_harmonies	0	240	0.0
vocal_samples	25	0	1.0
west_coast_rap_roots	1	0	1.0
western_swing	16	2	0.89

Table 3.4: Analysis via number of tag occurrences for each tag in the MSD24k dataset.

For all the tags, the ones presented in Table 3.4 were picked. The selection column indicates the percentage of songs from that tag to be selected from the dataset. This was done empirically, but with the goal of trying to obtain a relatively balanced dataset. For example, since there was considerable more songs with tag *electro* (2862 songs) than with *underground_hip_hop* (69 songs), a lower percentage was used to avoid the issue of overfitting, namely 5% instead of 100%.

Tag	**#Nonvocals**	**#Vocals**	**Ratio▼**	**Selection**
underground_hip_hop	69	0	1.0	100%
turntablism	242	0	1.0	100%
new_age_instrumental	18	0	1.0	100%
instrumental_hip_hop	90	0	1.0	100%
trance	1338	26	0.98	5%
techno	1918	38	0.98	5%
jazz_fusion	504	18	0.97	5%
drumnbass	54	2	0.96	100%
electro	2862	130	0.96	5%
industrial	969	46	0.95	5%
classical	317	20	0.94	100%
acoustic_guitar	351	24	0.94	100%

Table 3.5: MSD24k selected tags to include in *Nonvocals* class and in which percentage.

With the *Nonvocals* class now considerably reduced from 20519 songs to 1364, the manual verification process could take place. As expected, for the *Nonvocals* class, it was found a high number of False cases, since most of the songs selected for this class were a result of an educated

guess. However, it is safe to say that the False cases would have been much higher if a random selection had been made instead.

	Vocals	**Nonvocals**	*Totals*
Before Listening	1178	1372	2550
True	1152	506	1658
False	26	0	26
Dubious	5	0	5
After Listening	1147	532	1679

Table 3.6: Listening evolution in MSD24k.

3.4 Overview

To conclude this chapter, the distribution of the *Vocals* and *Nonvocals* class per dataset in from which all experiment from Chapter 5 were run are presented in Table 3.7 and Figure 3.1.

The datasets are also made available at `http://paginas.fe.up.pt/~ei08067/dokuwiki/doku.php`.

	Vocals	***Nonvocals***	**Total**	**Ratio** [4]
CAL500	444	58	502	7.66
Magtag5k	1627	724	2351	2.25
MSD24k	1147	532	1679	2.16
Total	3218	1314	4532	

Table 3.7: Distribution of *Vocals* and *Nonvocals* per dataset and globally.

Additionally, for a better visualization of the distribution of the original tags considering the three datasets for the *Vocals* and *Nonvocals* classes the following two tag clouds are presented Figure 3.2 and 3.3). Even empirically, it is clear how there are some tags that have a direct relationship with singing content i.e. *classical and strings*.

[4] Number of *Vocals* over number of *Nonvocals*.

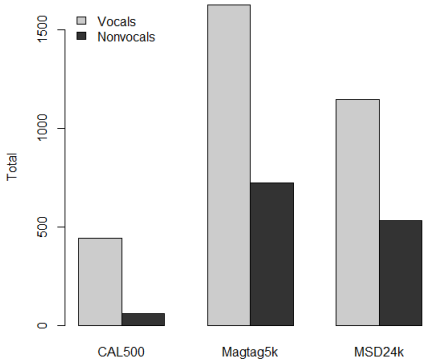

Figure 3.1: Distribution of *Vocals* and *Nonvocals* per dataset.

Figure 3.2: Tag cloud for *Vocals* for all 3 datasets.

Figure 3.3: Tag cloud for *Nonvocals* for all 3 datasets.

Chapter 4

Framework

In this chapter the overall framework used for the experiments mentioned in Chapter 5 is described. A schematic visualization of all the parts involved can be seen in Figure 4.1.

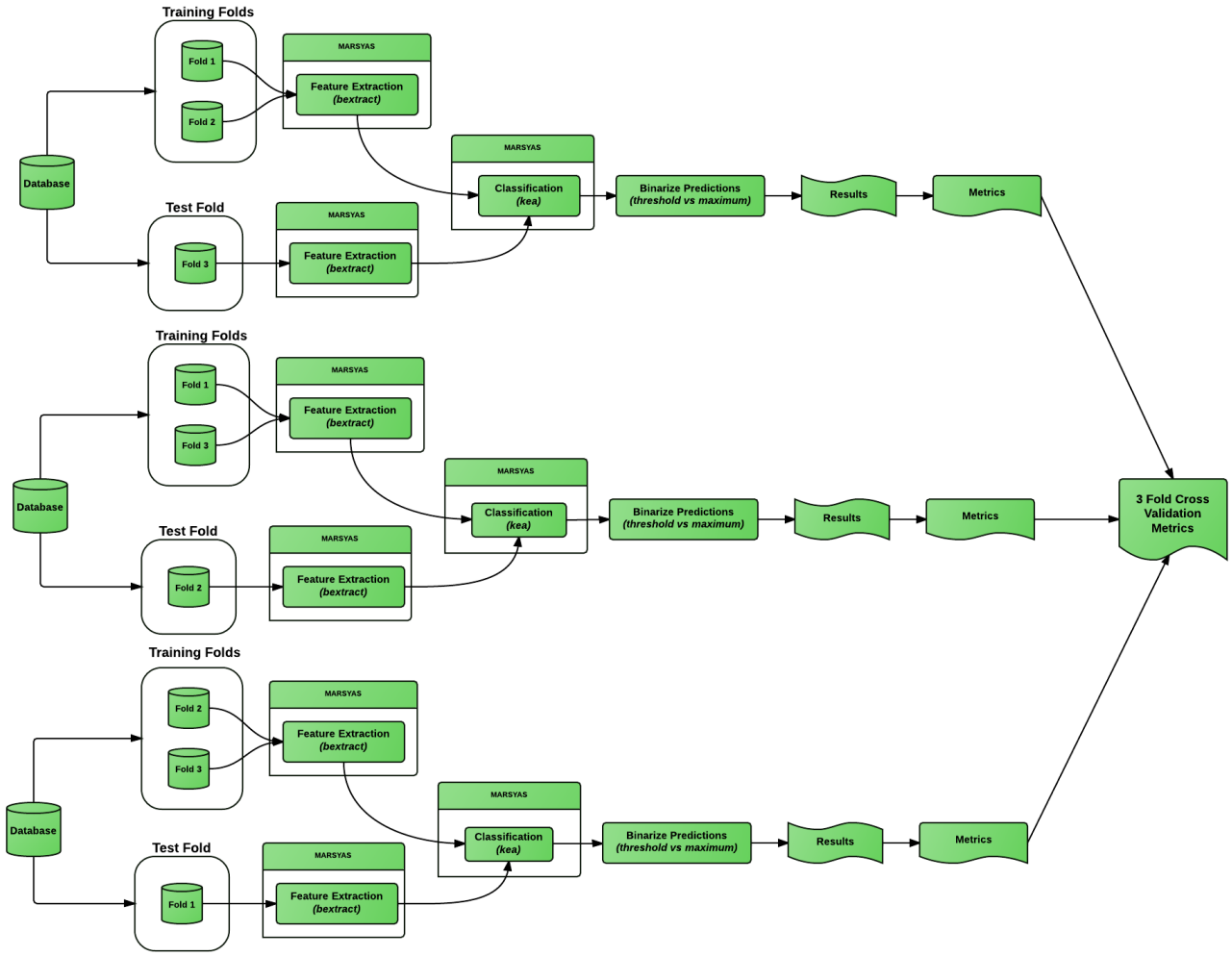

Figure 4.1: Overview of the framework.

4.1 Feature Extraction

For extracting the low-level features of the musical excerpts the Marsyas[1] (**M**usic **A**nalysis, **R**etrieval and **Sy**nthesis for **A**udio **S**ignals) software package was used. It is an open source package that provide MIR researchers and enthusiasts with a general, extensible and flexible architecture that allows for easy experimentation and fast performance. In particular, the command line application provided with the Marysas framework `bextract` was used. The basic command was run for extracting the features:

```
bextract -mfcc -zcrs -ctd -rlf -flx -ws 1024 -as 400 -sv -fe <dataset>.mf -w <dataset>.arff
```

The output of this command is an ARFF[2] file, of which there is an example in Appendix D. A brief explanation of each of its flags is given.

Features

-mfcc for extracting *MFCCs 0-12* (Section 2.2.6).

-zcrs for extracting the *Zero Crossing Rate* (Section 2.2.2).

-ctd for extracting the *Centroid Power* (Section 2.2.3).

-rlf for extracting the *Roll-off Power* (Section 2.2.4).

-flx for extracting the *Flux Power* (Section 2.2.5)

Other

-ws setting the *analysis window* to 1024 samples (Section 2.2.1.6).

-as setting the *accumulator size* to 400 analysis windows (Section 2.2.1.6).

-fe stands for *feature extraction*, meaning to only extract the features and not to train the classifier, which is done later in `kea`, another Marsyas application.

-sv stands for *single vector*, meaning to turn on single vector feature extraction where one feature vector is extracted per file.

Default

-hs stands for *hop analysis* that is set by default to 512 samples (Section 2.2.1.6).

-m stands for *memory* and sets the size of how many analysis windows make up a *texture window* (Section 2.2.1.6). Default value is 40.

Marsyas extract all these features by applying a Hamming window (described in Section 2.2.1.4). Also, for each feature, there are four output values: the average of the averages of the texture windows, the average of the standard deviations of the texture windows, the standard deviation of the averages of the texture windows and the standard deviation of the standard deviations of the texture windows.

[1] http://marsyas.info/

[2] stands for Attribute-Relation File Format and was developed as part of the Machine Learning Project at the Department of Computer Science of The University of Waikato for use with the Weka - a machine learning software.

4.2 Learning Algorithm

For the classification part of the framework, the Marsyas's command line application `kea` was used. The following command was used for all the experiments, with variables `train`, `test` and `affinities` naturally varying according to experiment.

```
kea -m tags -w <train>.arff -tw <test>.arff -pr <affinities>
```

By default `kea` uses a ***soft margin non linear SVM with RBF kernel*** with parameters $\mathbf{C} = \mathbf{1.0}$ and $\gamma = \mathbf{4}$. Even though the application `kea` of the Marsyas framework is being used, the actual software package that runs the SVM routines is `libsvm` [CL11], a popular SVM implementation package that serves as basis for many machine learning applications.

4.3 Evaluation

For the evaluation of the proposed framework the three metrics most commonly seen in literature are used: *Precision*, *Recall* and *F-Score*. A cross validation within the same dataset (*Same Dataset Experiment*) and between datasets (*Cross Dataset Experiment*) is done. Finally, a filtering experiment is also conducted.

4.4 Data

As [MDLG11] clearly shows not using *artist filtering preprocessing* can result in over optimistic results that not clearly reflect the performance of a system. Therefore, artist filtering was used when considering the separation of the datasets into folds.

Considering the size of the datasets and the number of tags per dataset (which was only two, *Vocals* and *Nonvocals*), it didn't make sense to separate the datasets into many folds, for instance 10, a common value seen in autotagging algorithms with 10+ classes. Therefore, CAL500 being a dataset with only 58 *Vocals* songs was only divided into two folds. Magtag5k and MSD24k being larger in size and having a better distribution of tags were both divided into three.

Chapter 5

Experiments

In this chapter, it will be described the experiments conducted to evaluate the system developed.

5.1 Binarization Methods

The task of classifying a song as *Vocals* or *Nonvocals* is more formally called as a *Binary Classification Problem*. This means that every instance of the data (in this case, song excerpts) has to belong to one and only one class out of the two possible. In other words, every song in the datasets will be classified as either *Vocals* or *Nonvocals*, not both of them, not none of them.

The predictions of a machine learning algorithm before being normalized (more commonly called *affinities*) are continuous values. Therefore, these values should be binarized so that a classification can in fact occur. For binarizing these continuous values, there are a few possibilities, such as using:

- a ***threshold*** value: a predefined value below which all values to binarize (*affinities*) are considered to be part of one class whereas, all other ones above it, are considered to be part of the other class. For example, all affinities above 0.5 will be considered as 0 (not representative of a class) and all equal and above will be considered 1 (representative of that class). This method is also referred to as *cutoff value*.

- a ***dynamic threshold*** value: very similar method to the previous one, with the particular difference thast the threshold value is set *dynamically* according to the distribution of the classes in the training set. That is, if the distribution in the training dataset is of, let's say, 70% *Vocals* and 30% *Nonvocals*, then the threshold value will be set to guarantee that same distribution in the test set.

- the ***maximum*** value: simply choosing the maximum affinity out of the possible classes as the predicted one. For example, if a song excerpt has an affinity of 0.3 for the class *Vocals* and 0.8 for the class *Nonvocals*, the class with the maximum affinity (in this case, the *Nonvocals*) will be the chosen one.

Being the *threshold* a relatively simpler version of the *dynamic threshold*, and the latter having reported satisfactory results in [MDLG11], it was reasonable not to consider the former. That left it open for comparing which of the *dynamic threshold* and *maximization* binarization algorithms would perform better.

Dataset	*Method*	**Vocals**			**Nonvocals**			*Averages*		
		P	R	FS	P	R	FS	P	R	FS
CAL500	Dynamic Threshold	0.93	0.93	0.93	0.46	0.46	0.46	0.70	0.70	**0.70**
CAL500	Maximum	0.91	0.99	0.94	0.70	0.21	0.32	0.80	0.60	**0.64**
Magtag5k	Dynamic Threshold	0.84	0.84	0.84	0.64	0.64	0.64	0.74	0.74	**0.74**
Magtag5k	Maximum	0.81	0.90	0.85	0.71	0.54	0.61	0.77	0.72	**0.73**
MSD24k	Dynamic Threshold	0.89	0.89	0.89	0.75	0.75	0.75	0.82	0.82	**0.82**
MSD24k	Maximum	0.86	0.91	0.89	0.78	0.70	0.73	0.82	0.80	**0.81**

Table 5.1: Comparison of *dynamic threshold* and *maximum* binarization algorithms.

As can be seen in Table 5.1, both in Magtag5k and MSD24k the results reported are very similar, although for CAL500 there is higher score using the *dynamic threshold*. One important thing to mention about the *dynamic threshold* binarization is that it binarizes the predictions for the test dataset maintaining the distribution of classes of the training dataset. With this in mind and considering the high *Vocals* to *Nonvocals* ratio of the CAL500 dataset, a higher result made all sense.

However, it can then be argued, that the *maximum* algorithm is more accurate regardless of the distribution of *Vocals* and *Nonvocals*, therefore making it more *dataset-proof*. For that reason, it was the chosen one for the following experiments.

5.2 Same Dataset

The most common evaluation procedure for an autotagging system is a *n-fold cross validation*. That is the kind of validation that is being presented here, considering same dataset folds. Each dataset is divided into *n* folds and each one of them is used as a testing set while the others are used as a training set. Figure 4.1 illustrates this process.

5.2.1 CAL500

Test Fold	*Train Fold(s)*	**Vocals**			**Nonvocals**			*Averages*		
		P	R	FS	P	R	FS	P	R	FS
1	2	0.91	0.99	0.95	0.78	0.24	0.37	0.84	0.62	0.66
2	1	0.90	0.99	0.94	0.63	0.17	0.27	0.76	0.58	0.61
Averages		0.91	0.99	0.95	0.70	0.21	0.32	**0.80**	**0.60**	**0.63**

Table 5.2: *Same Dataset* experiment results on CAL500.

On the CAL500 dataset it is interesting to note the discrepancy between classes, particularly in the *recall* variable. On one hand, it is being capable of correctly identifying nearly all of the *Vocals* from the dataset, but on the other, it can only recall 20% of the *Nonvocals*. It can then be said that the model is completely *overfitted* for the *Vocals* class. It is important to remind that CAL500 has a significant difference from the other datasets, since it is consisted of complete songs rather than 30 second excerpts. On top of that, it is also a dataset with an high ratio of *Vocals* to *Nonvocals*, two and a half more and the other two. For both that reasons, probably more due to the second one, contribute to this high performance in the *Vocals* class and very poor performance on the *Nonvocals* class. However, if considered the system as a whole, it still has a very good precision, but lacks in recall which is then naturally reflected in its *f-score*.

5.2.2 Magtag5k

Test Fold	*Train Fold(s)*	**Vocals**			**Nonvocals**			*Averages*		
		P	R	FS	P	R	FS	P	R	FS
1	2 & 3	0.77	0.91	0.84	0.67	0.40	0.50	0.72	0.65	0.67
2	1 & 3	0.83	0.88	0.86	0.70	0.60	0.65	0.77	0.74	0.75
3	1 & 2	0.80	0.92	0.86	0.72	0.49	0.58	0.76	0.70	0.72
Averages		0.82	0.90	0.86	0.71	0.54	0.62	**0.77**	**0.72**	**0.73**

Table 5.3: *Same Dataset* experiment results on Magtag5k.

Magtag5k clearly shows a more "stable" classifier when compared to CAL500. It also exhibits some lower ratings for the *Nonvocals* class, which is understandable considering is being trained with less data of that type. Overall, it gets a "decent" *f-score* of 0.73. However, it is important to mention that this is the largest dataset of the three with (2351 songs, 724 of which are *Nonvocals*). So, at least theoretically this is where it would be expected to find the best performance out of the three.

5.2.3 MSD24k

MSD24k dataset, out of the three, is the one that shows the least difference between *Vocals* and *Nonvocals* numbers, averaging a 0.81 f-score, a result at par with way more sophisticated state-of-the-art algorithms.

Test Fold	*Train Fold(s)*	**Vocals**			**Nonvocals**			*Averages*		
		P	R	FS	P	R	FS	P	R	FS
1	2 & 3	0.84	0.91	0.87	0.75	0.61	0.67	0.79	0.76	0.77
2	1 & 3	0.87	0.89	0.88	0.75	0.72	0.73	0.81	0.80	0.81
3	1 & 2	0.86	0.92	0.89	0.81	0.68	0.74	0.84	0.80	0.82
Averages		0.87	0.91	0.89	0.78	0.70	0.74	**0.82**	**0.80**	**0.81**

Table 5.4: *Same Dataset* experiment results on MSD24k.

5.3 Cross Dataset

The thought behind this evaluation experiment was to validate if it was possible to generalize the concepts of *Vocals* and *Nonvocals* a classifier was learning from a dataset to another. In other words, this experiment aims to investigate if the concepts being learned are in fact a global definition of *Vocals* and *Nonvocals* or rather a definition of *Vocals* and *Nonvocals* within just a particular dataset.

In a way, this experiment is at all very similar with the previous one. It follows the same principles, but instead of doing a cross fold within a dataset, it does a 2-fold cross validation, considering whole datasets as folds.

5.3.1 CAL500

Train Dataset	*Test Dataset*	**Vocals**			**Nonvocals**			*Averages*		
		P	R	FS	P	R	FS	P	R	FS
CAL500	Magtag5k	0.70	0.97	**0.81**	0.59	0.10	**0.16**	0.65	0.53	**0.49**
CAL500	MSD24k	0.71	0.99	**0.82**	0.83	0.10	**0.19**	0.77	0.55	**0.50**
CAL500 Cross Validation		0.91	0.99	0.95	0.70	0.21	0.32	0.80	0.60	0.63

Table 5.5: Cross Dataset Experiment training with CAL500.

As already mentioned several times, CAL500 is a dataset relatively small compared to the other two, having a considerable fewer number of *Nonvocals* songs and a high *Vocals* to *Nonvocals* ratio. Because of these characteristics, it is easy to understand the very low results in the *Nonvocals* class when its used as a training dataset. In fact, the cross validation within CAL500 itself was already poor (0.32) and using the Magtag5k and MSD24k as testing datasets only further validates this: CAL500 has a very poor concept of *Nonvocals*.

Another observation is that CAL500's concept of *Vocals* is somewhat (not much) true across datasets, with a difference of 0.14 and 0.13 in f-score between datasets. Overall, it can be said that CAL500 cross validation was already not a very good one, in fact the worst out of the three and this *cross dataset* testing proves its concepts generalize poorly to other datasets.

5.3.2 Magtag5k

Magtag5k shows the ability of generalization of concepts between datasets to both the *Vocals* and *Nonvocals* class. In fact, when using CAL500 as a test dataset even better f-score in *Vocals* is obtained. Overall, there is little to no significant difference in the f-scores results, especially in the global one, which is a clear indication of concepts successfully being applied in the other datasets. It's interesting to observe how an f-score of 0.54 is obtained for the *Nonvocals* class when using CAL500 as a test dataset, while a cross validation on it yields 0.32 (Table 5.5, a relatively lower results. This is evidence that the number of examples per class is very significant when training a dataset.

Train Dataset	*Test Dataset*	**Vocals**			**Nonvocals**			*Averages*		
		P	R	FS	P	R	FS	P	R	FS
Magtag5k	CAL500	0.94	0.93	**0.94**	0.52	0.55	**0.54**	0.73	0.74	**0.74**
Magtag5k	MSD24k	0.83	0.87	**0.85**	0.69	0.62	**0.65**	0.76	0.74	**0.75**
Magtag5k Cross Validation		0.82	0.90	0.86	0.71	0.54	0.62	0.77	0.72	0.73

Table 5.6: Cross Dataset Epxeriment training with Magtag5k.

5.3.3 MSD24k

On MSD24k, generalization of concept to other datasets is only observed in the *Vocals* class for both two other datasets. While the numbers for the *Vocals* class can be similar to MSD24k cross validation's, the same isn't true for the *Nonvocals* class. Overall, some generalization occurs for MSD24k and Magtag5k, but none for the MSD24k and CAL500.

Train Dataset	*Test Dataset*	**Vocals**			**Nonvocals**			*Averages*		
		P	R	FS	P	R	FS	P	R	FS
MSD24k	CAL500	0.96	0.76	**0.85**	0.30	0.78	**0.43**	0.63	0.77	**0.64**
MSD24k	Magtag5k	0.83	0.77	**0.80**	0.56	0.64	**0.60**	0.69	0.71	**0.70**
MSD24k Cross Validation		0.87	0.91	0.89	0.78	0.70	0.74	0.82	0.80	0.81

Table 5.7: Cross Dataset Experiment training with MSD24k.

5.4 Filters

As discussed in the previous sections, in both the *Same Dataset* and *Cross Dataset* experiments, there was evidence that the metrics being used weren't really reflecting an accurate picture of the system's performance. Therefore, another approach to test the robustness of the framework and further explore these issues was put in place. More precisely, it was decided to slightly transform (can be read as distort) the data, which was done by applying a random filterbank.

A filterbank is an array of band-pass filters, which attenuates (or amplifies) the signal in certain band of frequencies according to the filter's range and coefficients (magnitude). Figure 5.1 shows an example of a 12 band-pass filterbank all with the same magnitude, that is, all frequency bands equally affecting the input signal.

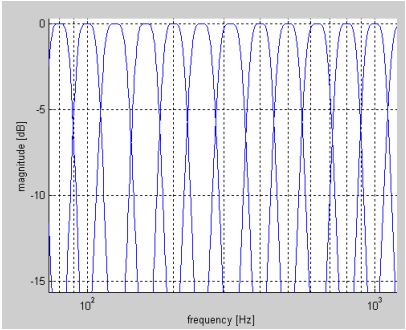

Figure 5.1: Example of a filterbank with 12 band-pass filters.

The random filterbank applied (with constant number of filters, 96, but random coefficients) was being constrained by an *abruption* variable that directly correlated with the coefficients and therefore "aggressiveness" of the filter generated. This was done to try to guarantee that the generated filterbank wouldn't distort the sound so much it would be impossible for a human ear to still detect the presence (or absence) of vocals. Also because of this, before running this experiment on the whole three datasets, an empiric sampled preliminary test was done that confirmed it was possible to still easily distinguish vocals. To even further validate this hypothesis, a listening experiment with human candidates was conducted which is described in Chapter 6.

Training a classifier after having applied a random filterbank to the data, made some predictions change when compared to when no transformation was used (*Same Dataset* Experiment). The predictions changed (or *flipped*), either from *Vocals* to *Nonvocals* or from *Nonvocals* to *Vocals*. The experiment can be more algorithmically described as follows:

1. Generate a random filterbank (considering the aforementioned constraints).
2. Apply the filterbank to the data.
3. Use the transformed signals in a way similar to the *Same Dataset* Experiment.
4. Repeat from 1. with the instances of the data that didn't change classification when compared to its untransformed version.

In other words, this experiment can be described as *N Same Dataset* experiments in which *N* is the number of filters generated.

Figure 5.2 shows the results of this experiment. Note that this experiment was only run on the subset of the data that was correctly predicted in the *Same Dataset* experiment, so, in practice, not the whole datasets is being tested. A summary of the values for the first and last iteration of each dataset is presented in Table 5.8.

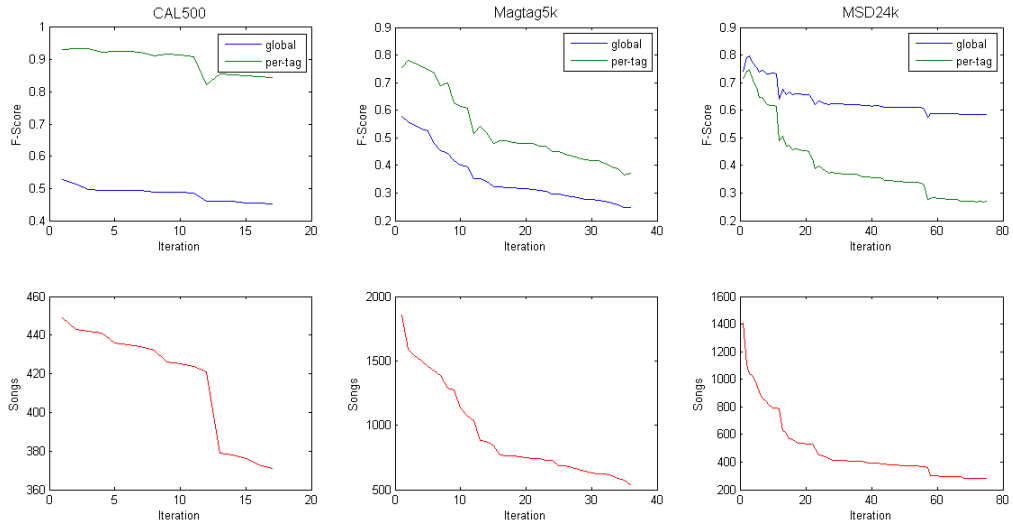

Figure 5.2: Evolution of global and per tag f-score (top) as well as number of songs that flip classification (bottom) by iteration number of random filter generated.

	CAL500		**Magtag5k**		**MSD24k**	
Iteration	1	17	1	36	1	75
Excerpts not flipped	449	371	1857	540	1406	282
F-measure per tag	0.53	0.45	0.58	0.25	0.66	0.18
F-measure global	0.93	0.84	0.76	0.37	0.72	0.27

Table 5.8: Summary of *Filters* experiment results for first and last iteration.

As a quick analysis of the graphs show, in all three datasets it is clear how there is a great number of songs that are flipping its original correct prediction, due to the fact of the filterbank being

applied. It is also clear how both the global and per tag f-score in all three datasets behave similarly, as expected, decreasing according to the increase in the amount of excerpts being misclassified.

What this experiment clearly shows, is that even though apparently the audio file is not being changed too much, the system looses all its capability of effectively detecting the presence of vocals.

Chapter 6

Listening Experiment

This chapter describes both the design and findings of the experiment conducted with human candidates, in order to further explore the findings of the *Filters Experiment* described in Section 5.4. An extensive use of the words *original* and *filtered* is made in this particular chapter, and so a clear definition is necessary.

- ***original*** excerpt: a music file directly taken from a publicly available dataset (for more details on the datasets used, see Chapter 3). For example, it can be assumed that this is the version one would get from ripping directly from the original CD recording.
- ***filtered*** excerpt: an *original* excerpt to which a randomly generated filterbank has been applied. Please note that for each *original* excerpt there are several corresponding *filtered* versions.

The script and general interface for the experiment can be found in the Annex.

6.1 Goals

In Section 5.4, it was shown how iteratively applying a filterbank to excerpts in the datasets would change its classification, that is, excerpts that were classified as *Vocals* in the ground truth were changing to *Nonvocals* and vice-versa. Furthermore, the filters being applied weren't significantly changing the sounds properties of the excerpts. This was clear evidence that supported the hypothesis of current state-of-the-art features used in Music Genre Classification not working as well as they were expected to within the domain of vocals identification. However, to fully validate this evidence, it was necessary to conduct an experiment with human subjects to evaluate if an human would exhibit the same behavior, or in other words, if an human would still be able to accurately identify the presence of vocals in the *filtered* excerpts. In addition, there was also interest in evaluating how different an human considered an *original* and its corresponding *filtered* version to be.

The goals of the experiment can be summarized as such:

1. evaluate if an human can still detect the presence (or not) of vocals in the *filtered* excerpts;

2. evaluate to which extent an human ear can guess if an excerpt is *original* or *filtered*.

6.2 Data Selection

When doing experiments with computers there is no problem using large amounts of data. However, the same isn't true for experiments that involve human participation, since time available for participation is more of a constraint. For this reason, it was necessary to carefully select some excerpts that correctly represented the whole data, or in other words, to have a good sampling of the datasets considering the experiment to conduct. That said, the following guidelines were taking into account for the selection of the excerpts to include in the experiment:

- have the same number of *Vocals* and *Nonvocals* excerpts;
- have the same number of *originals* and *filtered* excerpts;
- select the *n* most "aggressive" filters for the *filtered* excerpts, with "aggressive" meaning the iteration in which most songs *flip* its classification;
- have the same number of excerpts from Magtag5k and MSD24k.[1].

These guidelines resulted in the selection of 6 excerpts per class (*Vocals* and *Nonvocals*) per dataset (Magtag5k and MSD24k) per version (*original* and *filtered*), totaling 48 excerpts of 30 seconds each. The reason for the number 6 lies in the fact of designing a relatively short experiment, more precisely, around 10/15 minutes, since it was desirable that the candidates maintained their full attention throughout the whole experiment. As it will be clear in the next paragraphs, these 48 excerpts (of 30 seconds each) were divided into two different sets of 24, therefore making it a 12 minutes listening experiment for each of the candidates.

The selected excerpts were picked randomly given the above mentioned constraints. However, they were all listened one-by-one prior to the realization of the experiment to make sure they were in good audio quality conditions. A complete list can be found in Appendix B. Please note that the *Iteration* column is associated with the random generated band-pass filter and is presented to show how the *filtered* excerpts are obtained from applying different filters (see Section 5.4 for more details).

6.3 Design

Evaluating the first goal - evaluate if an human can still detect the presence (or not) of vocals in the *filtered* excerpts - was a relatively simple task. In fact, it was just a matter of mixing *original* with *filtered* excerpts and ask the candidate if he/she could detect the presence of vocals. On the other

[1]The CAL500 dataset was not considered here, since its song clips are not of 30 seconds duration, but rather full songs. To keep the experiment as short as possible, this dataset wasn't considered. This wasn't seen as problem though, since there was no evidence that each dataset had its own intrinsic sound properties. The other two, Magtag5k and MSD24k, were however considered, but just one would most probably have yielded similar results.

hand, for the second one - evaluate to which extent an human ear can guess if an excerpt is *original* or *filtered*- a more careful analysis had to be done. For that, the following options were considered:

1. To show the same candidate the two versions of an excerpt, the *original* and the corresponding *filtered* one without him/her knowing which one is which and ask him to guess. Even though this might seem like a possible solution, it wasn't. For instance, if the candidate was presented with a *filtered* excerpt first (even though he/she doesn't know it) and after with the corresponding *original* one, he would most likely take the first excerpt as reference to answer, therefore not providing a totally uninformed answer, which would go against the purpose of the experiment.

2. Don't show the same candidate the two versions of an excerpt to avoid the issue the previous option raises. Since the aim was to compare results between *original* and corresponding *filtered* excerpts without showing both versions to the same candidate to avoid informed answers, it was necessary to ask them to different candidates. A point can be made that this approach might not be correct, since every listener can have its own perception of the song. However, this was not posed as a problem, since identifying vocals is a relatively straightforward task and therefore such problem is not expected to happen.

For the reasons mentioned above, the second option was chosen. This implied that the data collected for the experiment had to be divided in two different disjoint sets, so that no candidate was presented with the *original* and *filtered* excerpt of the same song, in order to avoid direct comparison.

Also, to avoid any pattern that showing the same excerpts in a particular order could imply, a random ordering within the experiment set was made for each candidate.

6.4 Population

Additionally, a few considerations normally done in listening experiments were also taking into account, such as:

- only allowing the submission of an answer after the candidate has heard the full excerpt;
- only possible to hear each excerpt once;
- ask the candidate to setup their system sound level at the beginning of the experiment and leave it unaltered until its end, since altering it might have influence in timbre perception. (see Figure C.3).

Taking into account the specificity of the experiment, which dealt a lot with music listening skills, it was necessary to ask some screening questions related to music knowledge, such as:

1. If the candidate had any prior formal training in music;

2. If the candidate was an avid music listener;

3. How the candidate would be listening to the experiment.

Regarding the population of this experiment, a few numbers follows. The total number of candidates was of *154*, with an age average of *24.8*, *91* being males and *62* females. *75* of the candidates say they had any formal training in music while *79* say they don't. About being an *avid listener*, *72* candidates say they are, *55* say they don't and *27* position themselves in between. Finally, *101* of the candidates say they have no experience with sound engineering techniques, while *53* say they do.

For a complete reference of the questions in the questionnaire, please refer to Figure C.2 as Annex.

6.5 Software

For conducting this experiment, there was two options: (i) do it *in-lab*, that is bringing people to a controlled environment and conduct the experiment and (ii) distribute it over the Internet.

Considering the implications an in-lab experiment would entail, such as getting the necessary number of candidates within the time frame expected, early-on it was decided for the distributed version.

This implied designing a web page that would serve the purpose of the experiment. Considering the very specificity of the experiment itself a web page from scratch was developed. It uses the HTML5 audio capabilities to play audio in the browser. Figure 6.1 shows an overall view on how the web page works, with the yellow markers highlighting the process of choosing the candidate group in the experiment, according to how many completed versions there are already - naturally, the group with less completed answers at the moment, is the one chose for the current candidate. The web page is still accessible in `http://paginas.fe.up.pt/~ei08067/exp/`. An example answer file is also provided in Appendix E. For a more visual representation of the layout of the pages, please refer to Appendix C.

As can be seen in Figure 6.1, the web page communicates with a very simple REST API (written in PHP) that has only two possible function calls:

NEW

when: a new candidate hits the *Next* button after filling its personal details.

sends: the personal details of the candidate.

returns: the candidate number and group.

does: creates an answer file in the server with the details of the candidate (personal, browser, date, group).

ANSWER

when: a candidate hits the *Next* button after hearing an excerpt.

sends: candidate number and the answers to the current excerpt.

returns: ok if successful.

does: adds a line in the candidate's answer file with current excerpt .wav filename and answers.

The main reason for developing this kind of architecture, rather than an even simpler one, was due to the fact that it had to be possible for more than one candidate to be doing the experiment at the same time, hence the use of a candidate number id and the consequent API.

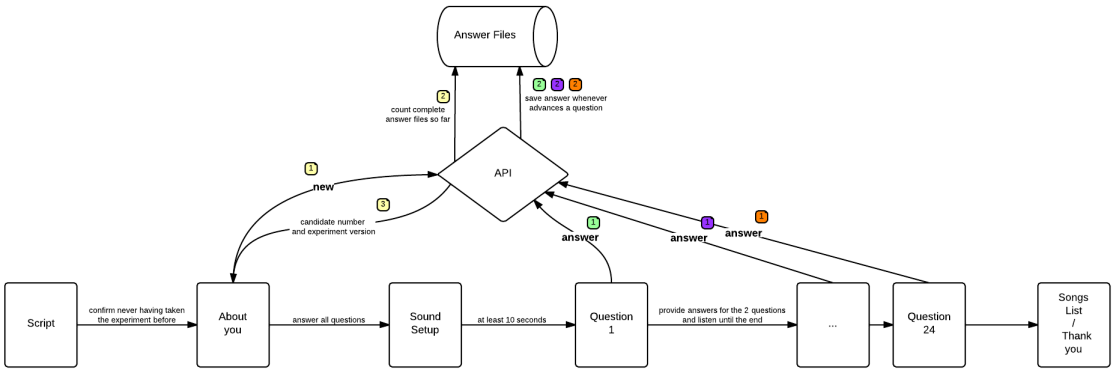

Figure 6.1: Overview of Listening Experiment Page Architecture.

6.6 Results

As mentioned in Section 6.1, the purpose of the experiment had two different distinct goals. For the first one, a more formal *hypothesis testing* was conducted, since it was of utmost importance to validate the results of the *Filters Experiment* described in Section 5.4. For the second goal, the results are present in a more intuitive way, also considering the more subjective nature of the question itself.

For the experiment there was a 251 total number of candidates that started the experiment: 251, while 154 (77 for *Experiment A* and 77 for *Experiment B*) actually finishing the experiment. Overall, there was *dropout rate* of 38.6%, which considered the time it took to fully complete the experiment was understandable.

6.6.1 First Question - Detecting *Vocals*

The first question of the listening experiment was the following:

> *Can you hear a human voice in this excerpt? (consider singing, speaking, shouting, whistling, etc.)*

The type of experiment conducted for this particular question can be described as *One-Factor Two-Levels Within-Subject* design [Lud07].

Any experiment implies the description of its *independent* and *dependent* variables. In this one, the *independent variable* is filtering the excerpts, while the *dependent variable* is the distribution of the answers on the *filtered* excerpts.

The *factor* of this experiment is *applying a filter on an excerpt*. And for that same factor, there are two possible *levels*: the *original* and the *filtered*.

It's called *within-subjects* (sometimes also referred to as *repeated measures*), since the same subjects are used in each *level* (*original* and *filtered*) of a given *factor* (the excerpts).

The *Control Group* can be described as the distribution of the answers from *original* excerpts, while the *Experiment Group* is the distribution of the answers from the *filtered* excerpts.

The *claim*, can then be described as *the human ear perception of presence of vocals is not affected by most "aggressive" filters applied on the Filters Experiment (Section 5.4)*. As remainder, the filters used on *Filter Experiment* were a 96 channel equally spaced filterbank constrained with an *abruption* set to 0.9.

When conducting hypothesis testing, both the *null hypothesis* and *alternative hypothesis* should be defined, in this case as follows:

$$H_0 : \overline{Q_1}(o) = \overline{Q_1}(f)$$
$$H_1 : \overline{Q_1}(o) \neq \overline{Q_1}(f)$$

However, there is a significant difference between this experiment formulation and a typical hypothesis testing. Normally, the interest is in proving that the alternative hypothesis should be true, considering there is significant evidence not to consider the null one. In this case, however, what is intended is to do the opposite, to show that spite of the large sampling data used, there is no evidence that shows the null hypothesis can be disregarded. The author is aware that is not viable to prove the null hypothesis, as [Arb13] clearly explains. However, given the problem at hand no other formulation was possible. What will be done throughout this section is to present how in spite of the large sample data used, considering the problem, there no evidence to discard the null hypothesis. Although, this isn't proving it it shows some clear evidence that for it to be false is very unlikely.

Figure 6.2 shows the percentage of correct answers for question 1. According to the claim that the perception of vocals isn't affected by the filters used, it was expected to see most of the bars with similar heights, which can be said to be the case. Figure 6.3 shows the same data but in difference of correct answers per excerpt. It is clear how most excerpts have similar responses, since most bars are close to the y axis. However, some excerpts originate not so consensus answers, such as excerpt number *9*, *11*, *14*, *21* and *24*. A careful listen to this excerpt shows that there is some reason for this to happen. With no exception, all of this particular excerpts are somewhat of

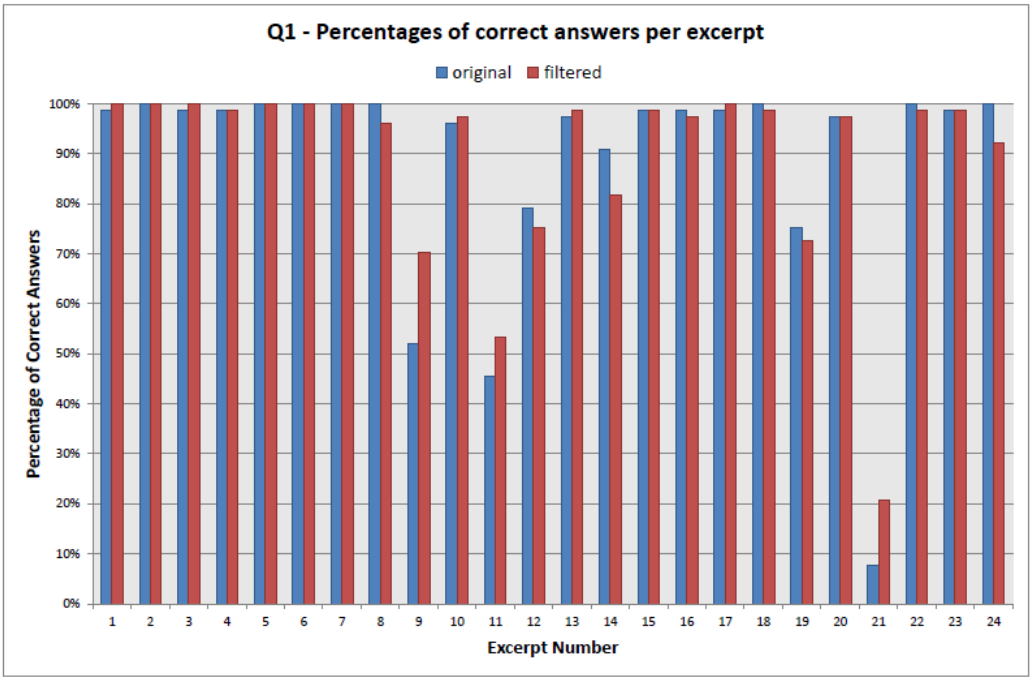

Figure 6.2: Percentage of correct answers per excerpt for question 1.

more dubious nature, having only a very short vocal presence (less than one second, for example) or have a vocal presence just at the very end of the excerpt (last second).

Still, a *a paired t-test* is performed (with a tail number of 2), which yields the following results:

Degrees of freedom	23
T-value	0.5
P-value	0.6

Table 6.1: Results of a *paired t-test* on question 1.

Results from Table 6.1, show that, at a significance level of $\alpha = 0.05$, it is completely unfeasible to reject the null hypothesis, since $p \gg \alpha$. In other words, most hypothesis test try to reject the null hypothesis, what happens when $p < \alpha$, which in this case it not true at all. Again, by not rejecting the null hypothesis, it is not being proved its truthfulness, but, considering the relatively large sampled data, it's safe to say there is several evidence to consider it not false, at the very minimum.

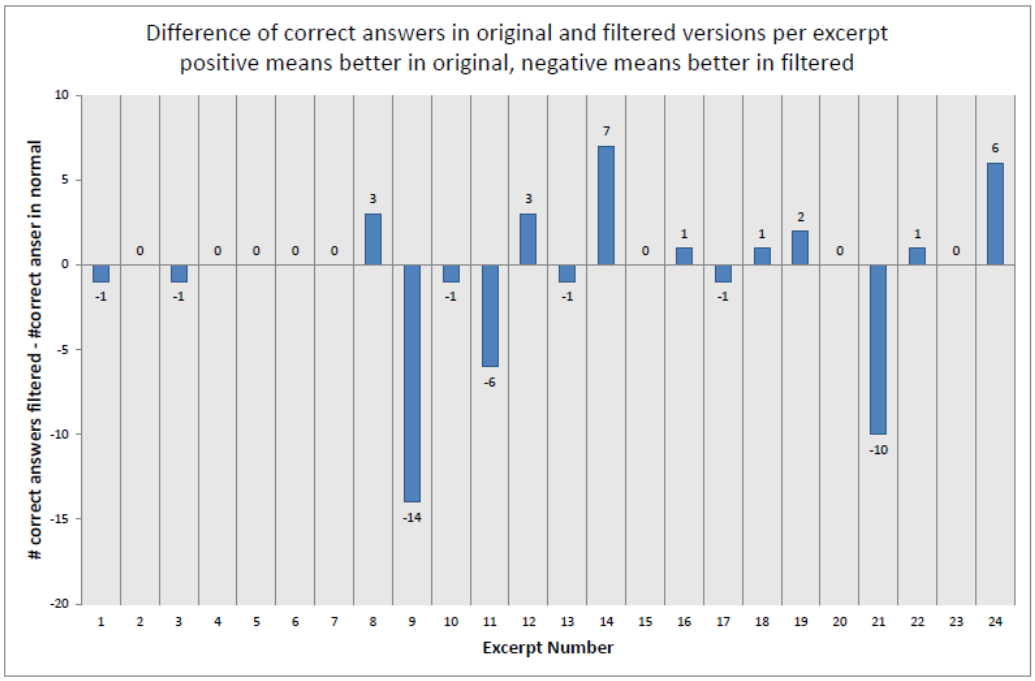

Figure 6.3: Difference in number of correct answers per excerpt for question 1.

6.6.2 Second Question - Guessing *Filtered* or *Original*

The second question of the listening experiment was:

Do you think this excerpt was digitally altered after the original published recording?

As previously mentioned, no hypothesis testing was done for this question, considering the subjective nature of the question itself. However, some interesting conclusions are provided.

The initial claim for this question was that it would be very difficult for an human ear to try and guess if an excerpt was *original* or *filtered* without making a direct comparison between them. Therefore, it was expected that this question's performance in general was averaged towards 50%, that is the equivalent of a random choice, hence proving the total inability to guess if an excerpt was *original* or *filtered*. The basis for this question was to show that if an human ear can not clearly say whether an excerpt sounds *original* or *filtered*, that is because they both seem acceptable versions of a music excerpt and, for that reason, an autotagging algorithm should still be able to correctly classify both versions the same way.

The most significant number yielded from this question was the average percentage of correct predictions for the 154 candidates, which was of **47%** (with a standard deviation of *17%*). If the answers such as *Not sure / Can't tell* aren't considered, this value goes to **52%** (with the standard deviation keeping at *17%*). In either case, it is clear how this is evidence in the line of the initial claim. Even when only assertive answers of *Yes* and *No* to the question are considered, the overall performance of guessing if an excerpt is *original* or *filtered* can be considered a random choice answer.

Additionally, the data also shows that there is no significant distinction between *Vocals* and *Nonvocals*. Guessing if a *Vocals* excerpt has been processed or not is as difficult as in a *Nonvocals* excerpt too, as Figure 6.4 shows. Notice how the standard error bars overlap, therefore proving no statistical difference between the classes.

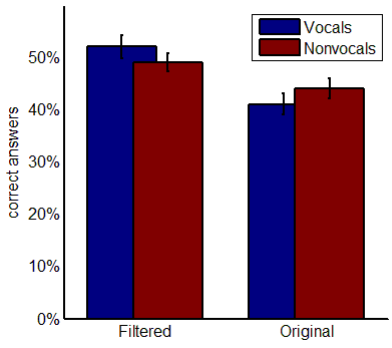

Figure 6.4: Same performance in question 2 in *Vocals* and *Nonvocals*.

Another analysis that was done on this question's data was a pivot table over the following population classification variables: *music training*, *avid listener*, *sound engineering*, and *listening method*. In other words, all, except *age* and *gender* were used, since there was no reason to

believe this two basic variables would have any influence on the results and in a way to avoid over segmentation which would make yielding results not possible.

From this pivot table only classes of at least 10 elements were considered in order to avoid an misleading average. The highest score, of **57%**, was obtained by the *music trained, avid listener, sound engineering knowledgeable, listening at high quality phones or speakers* group of candidates. It could be said this was the group to be expected to perform better in this kind of question. Still, it is interesting to notice how slightly above a random choice performance this value is, again, more evidence towards the initial claim.

On the other hand, the worst performing group, with a performance of **42%**, was the same group with the only difference of listening in poor quality headphones or speakers. In fact, this group has the same exact characteristics of the previous one, what is changing is not the population itself but rather the setting in which the experiment itself takes place. In fact, doing an analysis on all variables, but in a horizontal way rather than vertical as before, it can be seen that the variable with most impact over the overall performance is the difference of *listening method* with a difference of **5%**.

Chapter 7

Conclusions

This dissertation addressed the problem of detecting vocals in music. Its motivation derived from the increasing belief in the SMC group (and the MIR community in general) that current evaluation techniques are not evaluating autotagging system sufficiently enough.

In this final chapter, the conclusions of this work are presented as well as some guidelines for future possible work.

7.1 Conclusions

Regarding the binarization methods, it was shown a comparison of a *dynamic threshold* and *maximum* algorithms, the former one yielding best overall results, while the latter providing better results for the *Vocals* class. The difference was not very significant though.

The *Same Dataset experiment* allowed for a careful analysis of Type I and Type II errors, which showed that some very paradigmatic examples of a class weren't being correctly predicted, despite the relatively high global f-scores. For instance, this analysis showed that even though CAL500 was performing at an f-score of 0.95 for the *Vocals* class, it still wasn't able to recognize a 3-minute *a capella* song, which basically consisted of non-stop only vocal audio content.

The *Cross Dataset Experiment* showed that the concept of *Vocals* and *Nonvocals* was in generalizing in some cases from dataset to dataset. In particular, CAL500 was shown to be a rather bad dataset for training, considering it had very few instances of *Nonvocals*. MSD24k, which was the dataset on which there was better performance (global f-score of 0.81) proved some generalization to Magtag5k (0.70) but not so much to CAL500 (0.64). On the other hand, the Magtag5k dataset, yielded both very similar results for CAL500, MSD24k and its corresponding cross validation results, thus showing evidence that the concept being learned was shared among datasets and not specific per dataset.

With the *Filters Experiment* a new (or at least very rarely mentioned in literature) evaluation technique is proposed. This experiment showed how applying a *random iterative filterbank* very drastically affects the results from previous experiments. In fact, it is showed how it's possible to find filters that change from a global f-score of 0.72 to 0.27, thus proving the system totally

unusable. This evidence is shown for all three datasets, with CAL500 having full length songs, being the one that is less affected by this.

From the listening experiment conducted, two main conclusions were taken. Firstly, the filters that were being used for the *Listening Experiment* weren't affecting the human ear perception of vocals. With this, it was possible to show the datasets weren't really effectively learning the concepts of *Vocals* and *Nonvocals*. Secondly, it was also shown that without any previous knowledge an human can't say if an excerpt is *original* or *filtered*, therefore implying that the autotagging system should have performed in the same way, which wasn't the case as shown by this experiment. Therefore, evidence was shown that evaluation techniques, such as *random filterbank filtering*, should be further applied and explored, since typical evaluations such as *cross validation* aren't enough.

Another conclusion, from the revision of the state-of-the-art in vocals identification is that there is a lot of diversity on which the experiments are ran and in the way its evaluations are reported. Lot of publications use different datasets, from different sources and at different conditions, making it hard to do a direct comparison of results.

Finally, a mapping into the vocals domain of three widely used dataset within the research community was constructed and made available[1] so that other researchers can use them.

7.2 Future Work

An interesting approach to the predictions binarization process, even when there is relatively low confidence to which class an instance of data should belong, would be not to force it. From the review of literature that has been done, no such method has been proposed and the author believes that it could yield some interesting results and is worth exploring. The underlying logic for this kind of concept is that sometimes it could be much better not to provide an answer rather than providing a wrong one.

Another extension of this work would be to incorporate more features into the feature extraction process. As a matter of fact, there are several features more directly related with the singing phenomenon, in which there are published results, that haven't been used in this work. However, these publications report results within the specific task of voice segmentation. The interesting part following this work, would be to see to which extent specific singing features would impact the results and its robustness when applied in a voice identification domain.

Naturally, a future exploration that made perfect sense was too use a larger number of classifier algorithms, namely to see to which extent an experiment like the *Filter Experiment* would affect each classifier results.

Finally, it would be of interest to broaden the number of tags used by the system, namely a distinction between *male* and *female* singers.

[1] http://paginas.fe.up.pt/ ei08067/dokuwiki/doku.php

Appendix A

Dataset Construction Log File

```
[NOTVOCALS_VOCALS]
0\american_bach_soloists-joseph_haydn__masses-03-qui_tollis__adagio-0-29.mp3
0\ammonite-reconnection-06-calm-117-146.mp3
0\apa_ya-apa_ya-12-african_wedding_song-0-29.mp3
0\jeffrey_luck_lucas-what_we_whisper-08-know_my_name-117-146.mp3
0\solace-iman-08-foreshadow-30-59.mp3
1\anamar-transfado-01-eu_nao_sabia_i_did_not_knoow-0-29.mp3
2\anup-embrace-01-sweet_dissonance-0-29.mp3
3\dj_cary-eastern_grooves-02-kalimayantrahands_upon_black_earth-233-262.mp3
3\dj_cary-eastern_grooves-11-sokol_mi_letasherefe-59-88.mp3
3\emmas_mini-beat_generation_mad_trick-02-disconnected-117-146.mp3
3\panacea-songs_and_dance_music_of_europe_east_and_west-04-e_hatal-146-175.mp3
3\spinecar-autophile-04-autophile-378-407.mp3
3\very_large_array-stuff-12-everythings_fine-117-146.mp3
3\vito_paternoster-inzaffirio-06-regina_dei_cieli-0-29.mp3
4\american_bach_soloists-j_s__bach__favorite_cantatas-19-chorale__und_wenn_die_welt_voll_teufel_war-88-117.mp3
4\dj_cary-downtempo_chill_2-01-calm_ammonite-30-59.mp3
5\arthur_yoria-of_the_lovely-09-ectomorph-0-29.mp3
5\burnshee_thornside-rock_this_moon-12-took_me_by_surprise-146-175.mp3
5\dj_markitos-unreachable_destiny-09-let_me_be-262-291.mp3
5\trancevision-lemuria-03-alpha-233-262.mp3
6\grayson_wray-picassos_dream-09-lucky_star-88-117.mp3
6\mercy_machine-in_your_bed-07-a_prayer-0-29.mp3
6\norine_braun-crow-09-dreams-0-29.mp3
6\norine_braun-now_and_zen-02-now_and_zen-0-29.mp3
6\solace-satya-07-saptak_seven_notes-146-175.mp3
7\rapoon-what_do_you_suppose-11-i_dont_expect_anyone-320-349.mp3
7\roots_of_rebellion-the_looking_glass-06-amnesia-233-262.mp3
8\hybris-the_first_words-08-the_choice_i_never_had-88-117.mp3
8\magnatune-red_hat_summit_compilation-13-c__layne__just_my_luck_fourstones_net_remix-0-29.mp3
8\william_brooks-buffalo_treason-08-a_misdemeanor_or_two-0-29.mp3
9\self_delusion-happiness_hurts_me-09-christine-175-204.mp3
9\strojovna_07-iii-04-loopatchka-117-146.mp3
9\the_seldon_plan-making_circles-11-checkered_flag-0-29.mp3
9\the_strap_ons-geeking_crime-19-johnnys_motel-0-29.mp3
9\various_artists-the_2007_magnatune_records_sampler-03-in_the_middle_beight-0-29.mp3
9\ya_elah-each_of_us-04-om-175-204.mp3
a\electric_frankenstein-the_time_is_now-08-fast___furious-0-29.mp3
a\jade_leary-and_come_the_sirens-01-our_silent_ways-30-59.mp3
a\liquid_zen-magic_midsummer-01-4_oclock_sunny_and_hot-30-59.mp3
b\cargo_cult-alchemy-02-alchemy-262-291.mp3
b\magnatune_compilation-rock-07-c__layne_the_unheard_frequency-117-146.mp3
b\magnatune_compilation-world_fusion-04-shiva_in_exile_odysseia-0-29.mp3
b\seismic_anamoly-dead_mans_hand-08-tsunami-175-204.mp3
b\solar_cycle-sunlight-02-like_it_2-349-378.mp3
c\five_star_fall-automatic_ordinary-10-turn_the_light_on-233-262.mp3
d\rapoon-vernal_crossing-02-sonol-0-29.mp3
d\the_west_exit-nocturne-01-nocturne-30-59.mp3
e\atomic_opera-penguin_dust-09-watergrave-0-29.mp3
e\magnatune_com-magnatune_at_the_cc_salon-13-one_at_a_time_burnshee_thornside-59-88.mp3
e\skitzo-heavy_shit-01-curse_of_the_phoenix-88-117.mp3
f\american_bach_soloists-j_s__bach_solo_cantatas-04-bwv82__i_aria-262-291.mp3
f\asteria-le_souvenir_de_vous_me_tue-01-quant_la_doulce_jouvencelle_anon_from_oxford_can__misc__213-59-88.mp3
f\chris_juergensen-big_bad_sun-01-sweet_melissa-0-29.mp3
```

```
f\jacob_heringman_and_catherine_king-alonso_mudarra_songs_and_solos-09-o_gelosia_de_amanti-0-29.mp3
f\satori-healing_sounds_of_tibet-01-moon_night-233-262.mp3

[SPEECH_AT_END]
1\jacob_heringman-holburns_passion-27-a_toy_lute-30-59.mp3
3\jacob_heringman-siena_lute_book-09-ricercata_mlb11_da_milano-88-117.mp3
6\doc_rossi-demarzi6_sonatas_for_cetra_o_kitara-11-sonata_iii_largo-146-175.mp3
8\justin_bianco-phoenix-03-unseen_facts-88-117.mp3
9\american_baroque-dances_and_suites_of_rameau_and_couperin-13-minuets_1_2_suite_from_les_fetes_dhebe_rameau-117-146.mp3
9\janine_johnson-german_keyboard_masters-05-auf_das_heilige_pfingstfest_pachelbel-88-117.mp3
b\hanneke_van_proosdij-harpsichord_suites_of_chambonnieres-18-suite_in_c_major__courante_iris-88-117.mp3
b\jacob_heringman-jane_pickeringes_lute_book-02-a_toye-0-29.mp3
c\o_fickle_fortune-a_celebration_of_robert_burns-19-set_of_jigs-204-233.mp3
d\daniel_ben_pienaar-book_2_cd1_welltempered_clavier-21-prelude_and_fugue_no__11_in_f_major_bwv_880_praeludium-146-175.mp3
f\heavy_mellow-acoustic_abstracts-05-midnight_chimes-146-175.mp3
f\jacob_heringman_and_catherine_king-alonso_mudarra_songs_and_solos-22-romanesca_o_guardame_las_vacas_4_course_guitar-59-88.mp3
f\magnaloops-electronica_loops_1-43-osxivilion554-0-29.mp3

[VOCALS_NOTVOCALS]
5\burnshee_thornside-rock_this_moon-08-miss_your_love_forever_featuring_lilling-0-29.mp3

[DUBIOUS]
2\magnatune_compilation-electronica-06-indidginus_dusty_lands-233-262.mp3
5\domased-selection-07-wild_ride-30-59.mp3
7\rocket_city_riot-pop_killer-03-feel_alive-175-204.mp3
a\jade_leary-the_lost_art_of_human_kindness-12-earth_beyond_a_finite_thought-494-523.mp3
b\magnatune_compilation-rock-11-cargo_cult_alchemy-146-175.mp3
7\wicked_boy-the_treatment-05-strange_days-88-117.mp3
9\artemis-orbits-06-subterranean_hidden_kisses_mix__hands_upon_black_earth-262-291.mp3
e\burning_babylon-stereo_mash_up-01-7_nine_skank-88-117.mp3
```

Listing A.1: Magtag5k log file

Appendix B

Listening Experiment Data

ID	Dataset	Class	Iteration	Artist	Title
1	MSD24k	*Vocals*	12	Hole	Heaven Tonight
2	MSD24k	*Vocals*	2	David Cassidy & The Partridge Family	I Think I Love You
3	MSD24k	*Vocals*	57	Rage Against The Machine	Mic Check
4	MSD24k	*Vocals*	6	JayMay	Gray Or Blue
5	MSD24k	*Vocals*	4	Albert Hammond Jr	GfC
6	MSD24k	*Vocals*	5	Black Kids	Hurricane Jane
7	MSD24k	*Nonvocals*	14	Rodrigo y Gabriela	Hanuman
8	MSD24k	*Nonvocals*	8	Secret Garden	Song From A Secret Garden
9	MSD24k	*Nonvocals*	22	Nicolay	Fantastic
10	MSD24k	*Nonvocals*	3	Deep Dish	Deep Dish
11	MSD24k	*Nonvocals*	16	Infected Mushroom	Bombat
12	MSD24k	*Nonvocals*	13	Supervielle	Forma
13	Magtag5k	*Vocals*	9	Briddes Roune	Lutel Wot Hit Any Mon
14	Magtag5k	*Vocals*	7	Indidginus	Spiritual Spearmints
15	Magtag5k	*Vocals*	15	Jacob Heringman And Catherine King	Villancico Agora Viniesse Un Viento
16	Magtag5k	*Vocals*	10	Mercy Machine	Stark Love
17	Magtag5k	*Vocals*	2	Jami Sieber	In The Silence
18	Magtag5k	*Vocals*	4	The Kokoon	Face
19	Magtag5k	*Nonvocals*	3	Apa Ya	Apa Ya Pradha
20	Magtag5k	*Nonvocals*	14	Jacob Heringman	Waissel Polish Dance
21	Magtag5k	*Nonvocals*	6	Justin Bianco	Siren
22	Magtag5k	*Nonvocals*	5	Jacob Heringman	Newsidler Adieu Mes Amours
23	Magtag5k	*Nonvocals*	11	Seth Carlin	Sonata in Bb Kv 333 Allegretto Grazioso (Mozart)
24	Magtag5k	*Nonvocals*	8	Ehren Starks	Lines Build Walls

Table B.1: Listening Experiment Data.

Appendix C

Listening Experiment Interface

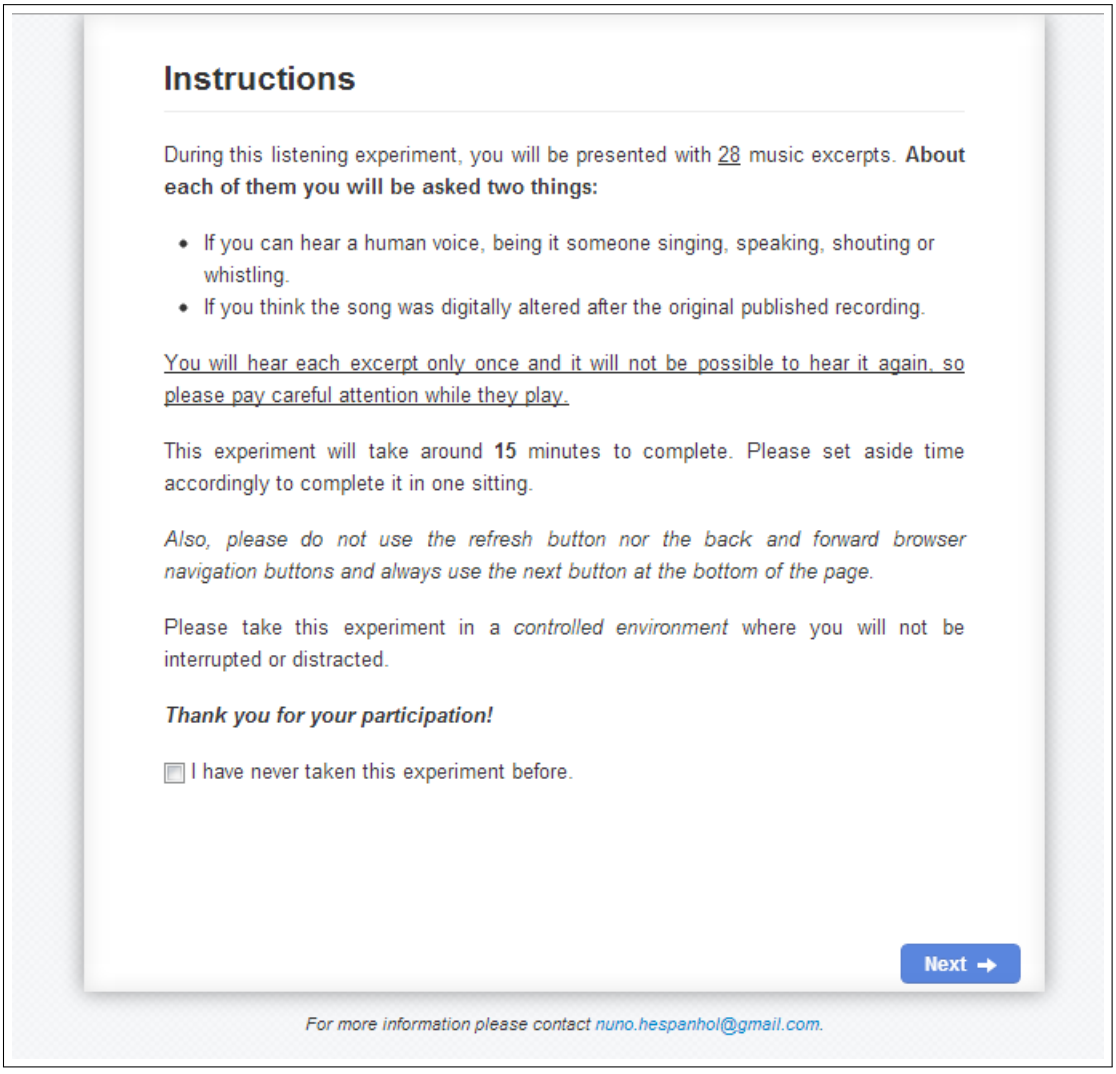

Figure C.1: First page of the questionnaire. Description of the experiment.

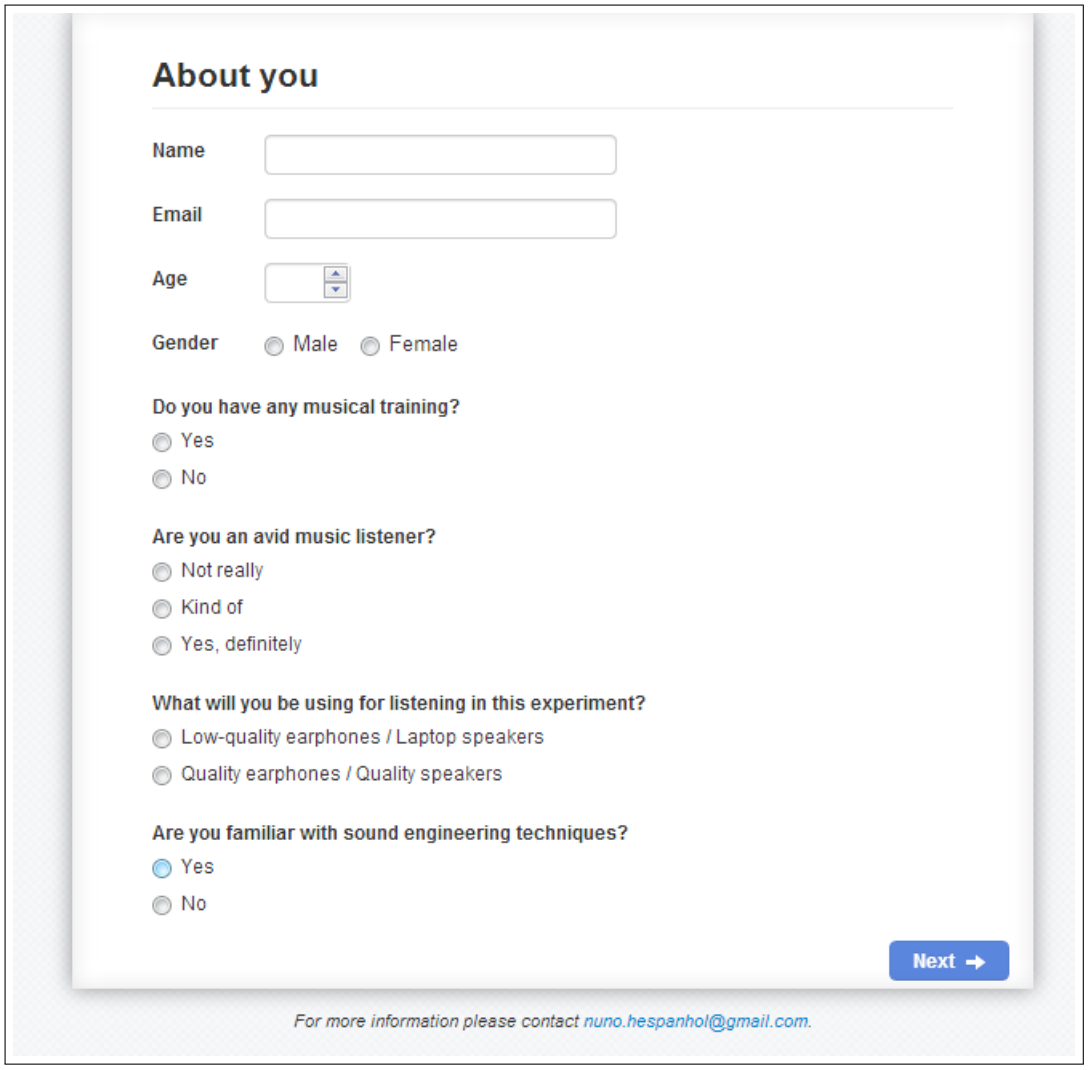

Figure C.2: Second page of the questionnaire. Screening questions.

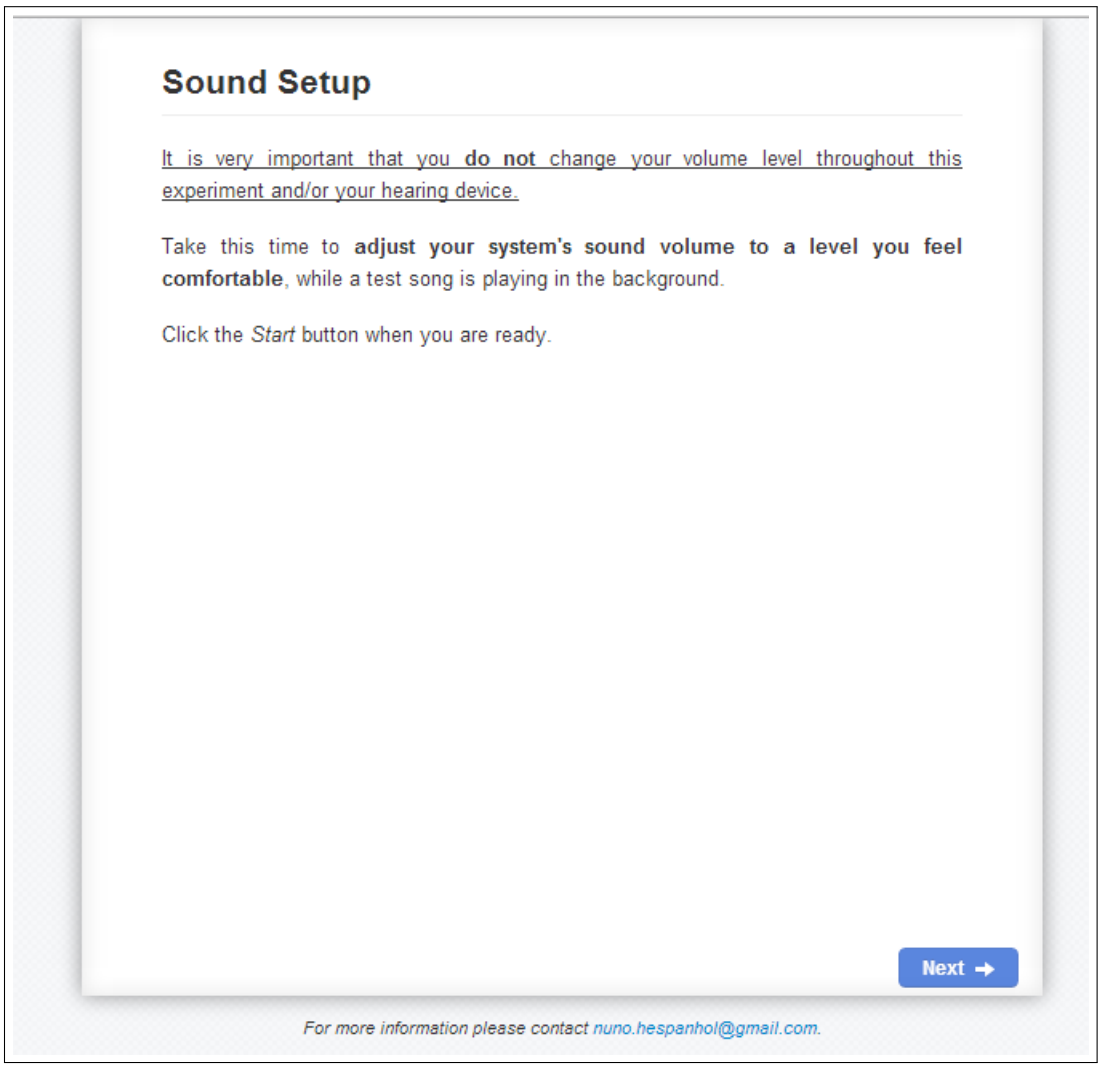

Figure C.3: Third page of the questionnaire. Sound Setup.

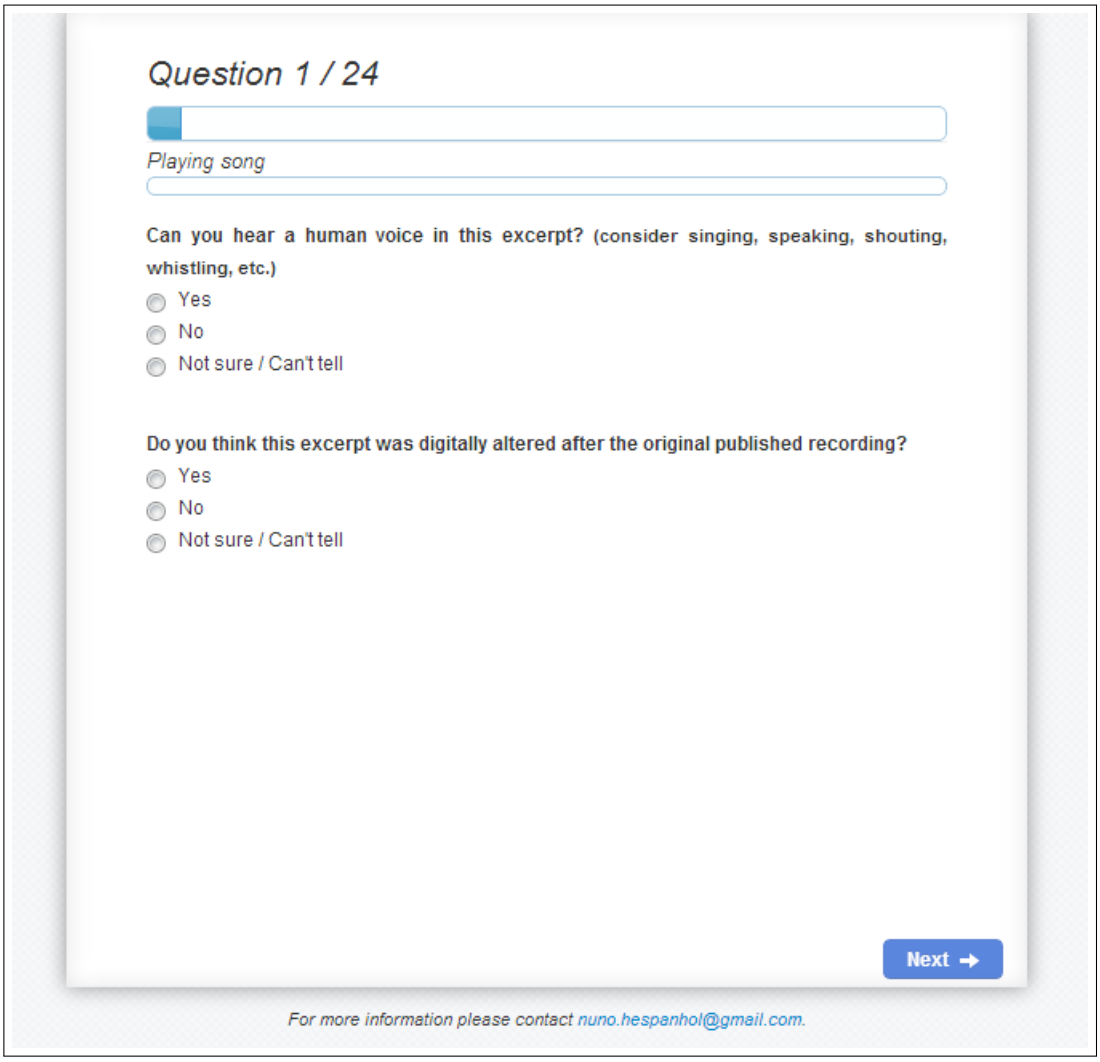

Figure C.4: The questions page of the questionnaire.

Appendix D

Arff File Example

```
% Created by Marsyas
@relation cal12_mag123_test.arff
@attribute Mean_Acc400_Mean_Mem40_ZeroCrossings_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_Centroid_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_Rolloff_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_Flux_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC0_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC1_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC2_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC3_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC4_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC5_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC6_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC7_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC8_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC9_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC10_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC11_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Mean_Mem40_MFCC12_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_ZeroCrossings_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_Centroid_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_Rolloff_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_Flux_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC0_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC1_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC2_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC3_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC4_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC5_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC6_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC7_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC8_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC9_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC10_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC11_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Mean_Acc400_Std_Mem40_MFCC12_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_ZeroCrossings_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_Centroid_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_Rolloff_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_Flux_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC0_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC1_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC2_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC3_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC4_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC5_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC6_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC7_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC8_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC9_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC10_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC11_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Mean_Mem40_MFCC12_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_ZeroCrossings_HopSize512_WinSize1024_AudioCh0 real
```

```
@attribute Std_Acc400_Std_Mem40_Centroid_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_Rolloff_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_Flux_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC0_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC1_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC2_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC3_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC4_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC5_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC6_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC7_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC8_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC9_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC10_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC11_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute Std_Acc400_Std_Mem40_MFCC12_Power_powerFFT_WinHamming_HopSize512_WinSize1024_AudioCh0 real
@attribute output {cal12_mag123_test}


@data
% filename ..\..\mag\wav\0\william_brooks-bitter_circus-01-the_gift-88-117.wav

0.074980,0.054144,0.125415,0.101078,-40.193601,2.705833,-0.543432,0.873664,-0.
135541,0.611070,0.261652,0.242124,0.340740,0.158596,0.092557,0.153308,-0.00456
2,0.018943,0.017169,0.040647,0.090665,3.545981,0.834907,0.625059,0.568093,0.45
7108,0.484083,0.509933,0.461722,0.426579,0.423766,0.465065,0.428401,0.388648,0
.019437,0.014858,0.034728,0.023312,6.402241,0.690896,0.510121,0.429690,0.21408
5,0.288941,0.376290,0.256485,0.247653,0.339819,0.148527,0.208551,0.188240,0.00
8423,0.007615,0.018011,0.018501,6.606448,0.466660,0.160579,0.210611,0.090238,0
.154525,0.165228,0.099030,0.100694,0.080730,0.124886,0.105281,0.074527,cal12_m
ag123_test

% filename ..\..\mag\wav\0\william_brooks-bitter_circus-02-try_it_like_this-88-117.wav

0.047659,0.036763,0.067530,0.128668,-40.828464,5.165160,-0.977301,0.722968,-0.
608061,0.712947,-0.248283,0.007119,-0.070354,-0.115457,-0.007304,-0.119530,0.2
38864,0.012734,0.010967,0.024505,0.106926,3.262384,0.937428,0.681022,0.489417,
0.495998,0.480388,0.500754,0.484698,0.449125,0.412358,0.441460,0.408414,0.3762
95,0.008472,0.006486,0.014740,0.031299,6.658873,1.116054,0.480703,0.276408,0.3
33904,0.380896,0.295072,0.362178,0.298321,0.297704,0.176098,0.221404,0.264504,
0.007017,0.006618,0.013151,0.019757,6.380625,0.619832,0.185431,0.107789,0.1481
80,0.114934,0.120978,0.095206,0.088286,0.109016,0.118492,0.083367,0.079322,cal
12_mag123_test

% filename ..\..\mag\wav\0\william_brooks-bitter_circus-03-seven_promises-117-146.wav

0.076671,0.052370,0.118735,0.109734,-40.126877,2.623657,0.024332,0.728177,-0.3
35325,0.247121,0.260596,0.252637,0.264246,-0.323057,0.057755,0.390968,0.234352
,0.019912,0.019929,0.038499,0.102009,3.330445,0.859786,0.647371,0.584332,0.496
104,0.473556,0.466057,0.432250,0.443897,0.436244,0.356939,0.481489,0.357760,0.
014252,0.010012,0.022072,0.024087,6.587749,0.604038,0.279371,0.231552,0.273156
,0.121427,0.316747,0.158059,0.269846,0.246839,0.250262,0.259504,0.207816,0.008
504,0.006635,0.013040,0.015428,6.108352,0.323304,0.210572,0.135321,0.098274,0.
109326,0.100174,0.080558,0.124466,0.139885,0.067613,0.125634,0.067169,cal12_ma
g123_test
```

Listing D.1: Example .arff file

Appendix E

Answer File Example

```
2013-06-04 21:17:27
Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.4 (KHTML, like Gecko) Chrome
    /22.0.1229.79 Safari/537.4
A
Rui Ribeiro,<email>,20,m,yes,yes,2,no
1,17.wav,yes,orig
2,23.wav,no,orig
3,10f.wav,no,orig
4,18f.wav,yes,orig
5,9.wav,no,orig
6,7.wav,yes,orig
7,15.wav,yes,orig
8,21.wav,yes,orig
9,14f.wav,no,orig
10,4f.wav,yes,orig
11,11.wav,cant,orig
12,22f.wav,yes,orig
13,24f.wav,no,orig
14,8f.wav,no,orig
15,12f.wav,no,filt
16,3.wav,yes,orig
17,1.wav,yes,orig
18,2f.wav,yes,orig
19,20f.wav,yes,orig
20,13.wav,no,orig
21,19.wav,yes,orig
22,6f.wav,yes,filt
23,16f.wav,yes,orig
24,5.wav,yes,orig
```

Listing E.1: Example answer file

References

[All08] P. Allegro. Singing voice detection in polyphonic music signals. Master's thesis, Faculty of Enginnering of the University of Porto, 2008.

[Arb13] Luk Arbuckle. You can't prove the null by not rejecting it, June 2013.

[Aud] Auda. Sample rates.

[BMEM10] T. Bertin-Mahieux, D. Eck, and M. Mandel. Automatic tagging of audio: The state-of-the-art. In Wenwu Wang, editor, *Machine Audition: Principles, Algorithms and Systems*. IGI Publishing, 2010.

[CL11] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[DBC09] J. Stephen Downie, Donald Byrd, and Tim Crawford. Ten years of ismir: Reflections on challenges and opportunities. In Keiji Hirata, George Tzanetakis, and Kazuyoshi Yoshii, editors, *ISMIR*, pages 13–18. International Society for Music Information Retrieval, 2009.

[DM80] Steven Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 28(4):357–366, 1980.

[FGKO10] H. Fujihara, M. Goto, T. Kitahara, and H.G. Okuno. A modeling of singing voice robust to accompaniment sounds and its application to singer identification and vocal-timbre-similarity-based music information retrieval. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(3):638–648, 2010.

[Gee] Stuffs 4 Geek. What is the bitrate? what you should know about it.

[GG78] John M. Grey and John W. Gordon. Perceptual effects of spectral modifications on musical timbres. *The Journal of the Acoustical Society of America*, 63(5):1493–1500, 1978.

[Gia08] Theodoros Giannakopoulos. Some basic audio features, 2008.

[Ham11] Philippe Hamel. Pooled features classification mirex 2011 submission. *Submission to Audio Train/Test Task of MIREX 2011*, 2011.

[HJ10] Chao-Ling Hsu and J.-S.R. Jang. On the improvement of singing voice separation for monaural recordings using the mir-1k dataset. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(2):310–319, 2010.

[HWJH12] Chao-Ling Hsu, DeLiang Wang, J.R. Jang, and Ke Hu. A tandem algorithm for singing pitch extraction and voice separation from music accompaniment. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20(5):1482–1491, 2012.

[15] International Music Information Retrieval Systems Evaluation Laboratory (IMIRSEL). Mirex home - mirex wiki, February 2013.

[Jan] Jyh-Shing Roger Jang. *Audio Signal Processing and Recognition*.

[Lab] Labtronix. About oscilloscope sample rate.

[Lud07] David Ludden. An easy introduction to experiment design and data analysis in psychology. 2007.

[LvAD07] E. L. M. Law, L. von Ahn, and R. Dannenberg. Tagatune: a game for music and sound annotation. In *ISMIR '07*, 2007.

[LW07] Yipeng Li and DeLiang Wang. Separation of singing voice from music accompaniment for monaural recordings. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(4):1475–1487, 2007.

[MDLG11] G. Marques, M. Domingues, T. Langlois, and F. Gouyon. Three current issues in music autotagging. In *Proc. of ISMIR*, pages 24–28, 2011.

[ME05] Michael Mandel and Daniel Ellis. Song-level features and support vector machines for music classification. In Joshua Reiss and Geraint Wiggins, editors, *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pages 594–599, 2005.

[ME07] M. Mandel and D. Ellis. A web-based game for collecting music metadata. In *ISMIR '07*, 2007.

[mfc] Mel frequency cepstral coefficient (mfcc) tutorial.

[MFYG11] Matthias Mauch, Hiromasa Fujihara, Kazuyoshi Yoshii, and Masataka Goto. Timbre and melody features for the recognition of vocal activity and instrumental solos in polyphonic music. In *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, 2011.

[NSW04] Tin Lay Nwe, Arun Shenoy, and Ye Wang. Singing voice detection in popular music. In *Proceedings of the 12th annual ACM international conference on Multimedia*, MULTIMEDIA '04, pages 324–327, New York, NY, USA, 2004. ACM.

[Pac11] F. Pachet. Musical metadata and knowledge management. In David G. Schwartz and Dov Te'eni, editors, *Encyclopedia of Knowledge Management*, pages 1192–1199. IGI Global, 2011.

[RH07] Martín Rocamora and Perfecto Herrera. Comparing audio descriptors for singing voice detection in music audio files. In *Brazilian Symposium on Computer Music, 11th. San Pablo, Brazil*, sep 2007.

[RP09] L. Regnier and G. Peeters. Singing voice detection in music tracks using direct voice vibrato detection. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1685–1688, 2009.

REFERENCES

[RRD08] M. Ramona, G. Richard, and B. David. Vocal detection in music with support vector machines. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 1885–1888, 2008.

[SLC07] M. Sordo, C. Laurier, and O. Celma. Annotating music collections: how content based imilarity helps to propagate labels. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR 2007).*, 2007.

[Sor12] M. Sordo. *Semantic Annotation of Music Collections: A Computational Approach*. PhD thesis, Universitat Pompeu Fabra, 2012.

[SS97] Eric Scheirer and Malcolm Slaney. Construction and evaluation of a robust multifeature speech/music discriminator. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, volume 2, pages 1331–1334. IEEE, 1997.

[SVN37] S. S. Stevens, J. Volkmann, and E. B. Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190, 1937.

[SZ05] N. Scaringella and G. Zoia. On the modeling of time information for automatic genre recognition systems in audio signals. In *Proc. ISMIR*, pages 666–671, 2005.

[TBL06] Douglas Turnbull, Luke Barrington, and Gert Lanckriet. Modelling music and words using a multi-class naıve bayes approach. In *Proc. of ISMIR*, 2006.

[TBL08] Douglas Turnbull, Luke Barrington, and Gert Lanckriet. Five approaches to collecting tags for music. In *Proceedings of the 9th International Conference on Music Information Retrieval*, pages 225–230, 2008.

[TC02] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *Speech and Audio Processing, IEEE transactions on*, 10(5):293–302, 2002.

[TLB07] D. Turnbull, R. Liu, and L. Barrington. Using games to collect semantic information about music. In *ISMIR '07*, 2007.

[VB05] Shankar Vembu and Stephan Baumann. Separation of vocals from polyphonic audio recordings. In *Proc. ISMIR*, volume 5, pages 337–344. Citeseer, 2005.

[WC05] K. West and S. Cox. Finding an optimal segmentation for audio genre classification. *Crawford and Sandler*, pages 680–685, 2005.

[Wik] Audacity Wiki. Bit depth.