POLITECNICO DI MILANO
Facoltà di Ingegneria dell'Informazione
Corso di Laurea in Ingegneria e design del suono
Dipartimento di Elettronica e Informazione

MUSICAL GENRE CLASSIFICATION AND TRACKING BASED ON CLUSTERING-DRIVEN HIGH-LEVEL FEATURES

Supervisor: Prof. Augusto SARTI
Assistant supervisor: Dr. Massimiliano ZANONI

Master graduation thesis by:
Daniele Ciminieri, ID 736364

Academic Year 2009-2010

POLITECNICO DI MILANO
Facoltà di Ingegneria dell'Informazione
Corso di Laurea in Ingegneria e design del suono
Dipartimento di Elettronica e Informazione

CLASSIFICAZIONE E TRACKING DEL GENERE MUSICALE BASATI SU DESCRITTORI DI ALTO LIVELLO DERIVATI DA CLUSTERING

Relatore: Prof. Augusto SARTI
Correlatore: Dr. Massimiliano ZANONI

Tesi di Laurea di:
Daniele Ciminieri, matricola 736364

Anno Accademico 2009-2010

"All of old. Nothing else ever.
Ever tried. Ever failed.
No matter. Try again.
Fail again. Fail better."

Samuel Beckett

A nonna Antonietta.

Abstract

The coming of the digital era and the large scale diffusion of internet made multimedia one of the main means of information and comunication of these days. The fruition of audio and video digital content has grown exponentially, thanks to their creation, distribution and sharing.

Talking specifically about music, a great quantity of services and software dedicated to its production, analysis, reproduction and research has been developed, dedicated to all kinds of users (professional, amateur and common ones). The field of research alllowing these technologies is called Music Information Retrieval (MIR). Among this interdisciplinary science, one of the most challenging tasks is the automatic musical genre classification; the aim of this problem is to automatically categorize audio excerpts according to a taxonomy of classes.

Usual works in this direction use low-level features to describe a training dataset and give the resulting data points distribution to a classifier, which classifies unlabeled data according to a supervised learning process. Such low-level descriptors have no immediate meaning to humans (experts or common users), but can be used by a machine to discriminate between classes. Lately, though, part of the research is focused on the implementation of systems adopting high-level features to describe audio signals; using a higher abstraction level, these descriptors acquire an intelligible music meaning; they could therefore be used by experts and common users, opening new implementation possibilities.

An issue in using these type of features is their definition, which is usually done in a subjective way by selecting the low-level descriptors which by guess should resemble a high-level characteristic if combined.

This thesis proposes an objective, example-based method to define a set of high-level features and use them to perform genre classification. The definition is obtained by finding the set of low-level features best clusterizing

a training dataset, and modeling the resulting clusters through a statistical model using a GMM classifier. The classification step is instead performed with an SVM classifier, using only the derived high-level features.
Two approaches are implemented: one, following a bottom-up fashion, aims to find out if compact and well separated clusters deliver a more efficient classification. The other, following a top-down fashion, aims at finding the subset of low-level features forming the clusters that achieve the best classification. Experimental results show promising results, especially for the top-down approach.

Sommario

L'avvento dell'era digitale e la diffusione su larga scala di internet hanno reso i contenuti multimediali i principali veicoli di informazione e comunicazione di oggi. La fruizione di audio e video è cresciuta esponenzialmente, grazie alla facilità con cui sono creati, distribuiti e condivisi.

Parlando nello specifico dell'ambito musicale, sono stati sviluppati moltissimi servizi e software dedicati alla sua produzione, analisi, riproduzione e ricerca, dedicati a tutti i tipi di utenti (professionisti, amatori e utenti comuni). Il campo di ricerca che permette l'implementazione di queste tecnologie è chiamato Music Information Retrieval (MIR). Uno dei problemi più ostici di questa disciplina è il riconoscimento automatico del genere musicale, il cui obiettivo è quello di categorizzare automaticamente brani audio rispetto ad una tassonomia di classi.

La maggior parte dei lavori svolti in questa direzione usa descrittori di basso livello per descrivere un dataset di training e fornisce la distribuzione di punti di dati risultante ad un classificatore, che decide il genere musicale di dati senza etichetta attraverso un processo di apprendimento supervisionato. I descrittori di basso livello non hanno un significato semantico intellegibile, ma possono essere usati da sistemi informatici per la discriminazione tra classi. Ultimamente, tuttavia, parte della ricerca è mirata all'implementazione di sistemi che adottino caratteristiche di alto livello per descrivere un segnale audio; usando un livello di astrazione più elevato, i descrittori acquistano un significato musicale comprensibile anche all'utente finale, e possono aprire nuove possibilità di implementazione.

Un problema riguardante questo tipo di descirittori è la loro definizione, che è generalmente svolta in modo soggettivo, selezionando i descrittori di basso livello che intuitivamente dovrebbero definire una caratteristica musicale se combinati.

Questa tesi propone un metodo oggettivo e bassato su esempio per definire

un insieme di feature di alto livello e le usa per la classificazione del genere musicale. La definizione dei descrittori di alto livello è ottenuta trovando il sottoinsieme di caratteristiche di basso livello che meglio clusterizza un dataset di training, e modellando i cluster risultanti attraverso un modello statistico usando un classificatore GMM. La fase di classificazione è invece svolta tramite un classificatore SVM, usando unicamente i descrittori di alto livello precedentemente implementati
Sono stati implemntati due approcci: il primo, seguendo un percorso bottom-up, è mirato a capire se cluster compatti e ben separati forniscano una classificazione più efficiente. Il secondo, seguendo un percorso top-down, è mirato a trovare il sottoinsieme di descrittori di basso livello che formi i cluster che portano alla accuracy più alta possibile.
I risultati ottenuti mostrano risultati promettenti, specialmente per quanto riguarda l'approccio top-down.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter introduces the thesis, stating its context and motivations as well as a brief description of the system developed to implement the proposed approach. A brief description of the structure of the thesis is also provided.

1.1 Motivations and goals

With the start of the digital era, and even more after the wide diffusion of internet, multimedia content has become one of the main realities of communication and information means. From entertainement to journalism, audio and video data are nowadays widely diffused, thanks also to the facility of their creation, distribution and sharing.
Talking about music, a great quantity of software dedicated to its production, analysis, reproduction and research has been developed, dedicated to all kinds of users (professional, amateur and common ones). Many services have also been delivered to assist the client in many tasks, defining new fields of interest. As an example, in the last years music recommendation systems have become popular and widely used, allowing the user to be guided through a musical selection covering his personal tastes. As another example, music identification systems allow a user to get the title of a song from an excerpt of even few seconds. These kinds of software and services are massively used and widely found in any environment, and we may interact with them every day, even if we don't notice it.
All these tasks are included in a main research field called Music Information Retrieval (MIR), which aims to retrieve various kinds of information from

music in order to use them for developing new ways of musical content discovery and fruition. Many problems are covered by this large interdisciplinary science; one of the most challenging is the recognition of the musical genre of an audio excerpt or piece, which is known as musical genre classification.
Part of the difficulty of this task lies in the ambiguity of the definition of musical genre: sometimes the characteristics of a particular music style are based on cultural references (a Gamelan orchestra piece of music may be considered sacred by a javanese, while catalogued as world music in Europe) or specific historical periods (i.e. baroque music, which contains many styles and different sets of instruments).
Crossovers between genres (e.g. fusion, which is a mixture of jazz and rock) and splitting of a single genre in several dinstinct subgenres (e.g. rock, divided in glam rock, hard rock, post rock, and so on) contribute to the overall confusion in defining an analytical, unambiguous method for classifying musical pieces into genres.
Pachet and Cazaly [1] shown that a general agreement on genre taxonomies does not exist. Taking the example of well known Web sites like Allmusic[1] (531 genres), Amazon[2] (719 genres), and Mp3[3] (430 genres), they only found 70 terms common to the three taxonomies. They notice that some widely used terms like rock or pop denote different sets of songs and those hierarchies of genres are differently structured from one taxonomy to the other.
Lippens, Martens and De Mulder [2] compared automatic and human categorization and showed two important facts: first, human music categorization is inherently subjective and therefore perfect results can not be expected neither from automatic nor human classification; second, automatic genre recognition is still far from reaching human expertise.
Such issues may be descouraging when approaching this field or research; however, automatically extracting music information is gaining importance as a way to structure and organize the increasingly large numbers of music files and radios available digitally on the Web. Genre hierarchies, typically created manually by human experts, are currently one of the ways used to structure music content on the Web. The use of tag assigned by professionals or by crowdsourcing is actually the most adopted method to categorize audio files and streams.

[1] `http://www.allmusic.com`
[2] `http://www.amazon.com`
[3] `http://www.mp3.com`

This method is often referred as context-based, because it relies on metadata related to the audio signal (artist, track number, album, genre, etc). While widely used, such approach presents a series of practical and theoretical limitations: the first one is that tagging a large database, e.g. composed of hundreds of thousands of musical pieces, can be extremely time-consuming and costly. Not all platforms or architectures are based on crowdsourcing, and not all those relying in their users for this issue have a sufficiently large customer base required to obtain an efficient tagging.

Another limitation is given by the subjectivity of this approach. As stated before, the definition of genre can be extremely ambiguous; as a consequence, not even a tagging conducted by human experts can be much objective. Finally, context-based classification is not suitable for the tracking of musical features inside a song or stream, as manually performing this task would be too costly in many aspects.

This is a major limitation, since information on feature tracking would enable new analysis and research possibilities: continuously observing the evolution of the musical characteristics of an audio excerpt would allow a continuous classification, not based on a fixed, pre-determined taxonomy. In the case of genre classification, this means that an audio clip characterized by a mixture of genres could be placed in a middle point between two or more main categories, and see how much and when it gets nearer to one of them.

For these reasons, many current researches aim to extract relevant information about music files and streams directly from the raw digital signal. This approach is called content-based, and its goal is to allow an automatic, objective and efficient musical genre classification. A system succesfully succeeding in this task can potentially automate the process of music categorization and description, and provide an important component for a complete music information retrieval system for audio signals.

The goal of this thesis is to achieve genre classification (in its most strict definition, i.e. classification of music signals into a single unique class based on computational analysis of music feature representations) in real time, i.e. to develop a system capable of classifying raw audio segments of few seconds (3 in this case) atomically, without taking into consideration the whole signal.

The proposed method is based on the creation of high-level features describing the audio signal. High-level features differ from the ones typically used in MIR for its higher abstraction level, discriminating the observations on a more intelligible basis. Usually these descriptors are derived from a com-

bination of low-level ones, choosing the raw signal characteristics that best resemble a musical property when put togheter.
In this work such features, in opposition to other works trying this approach, are created in an objective way, overcoming the issue of subjectivity in their definition. More specifically, once obtained a low-level features dataset from a collection of audio files, the best combination of clusters given a fixed number of features that best subdivides the dataset is found through a genetic algorithm; the resulting clusters are used as high-level features.
The implementation of high level features can improve classification systems not only in their performance, but also in the way users interact with them; in example, using semantically consistent characteristics of audio derived from low level descriptors could help non experts in tuning content-based similarity search systems.
The expected result of the research is an efficient classification system, suitable for real time usage (e.g. labeling of web radios and online streaming nusical channels).

1.2 Overview

As mentioned before, the goal of this work is to classify raw digital audio files or streams among a strict set of genres through the definition of a small set of high level features. The first part of the system is dedicated to this task, and it is composed by two main steps: the first one is the low-level features extraction from the training dataset, performed by collecting a large number of descriptors; the second step is to derive an example-based set of high-level features by choosing clusters combinations of the training observations, derived by optimizing the result of different objective functions. In order to efficiently model these clusters, GMMs are used.
The second part is the classification process itself, using only the obtained high-level features as discriminative characteristics. In this case, non-linear SVMs are used by training them with the high-level feature vectors and by using the resulting models to predict a label for unclassified observations.
Two approaches have been followed to accomplish this task. The first is a bottom-up approach: once the low-level features have been extracted from the training dataset, the resulting data points gets clusterized according to the K-means algorithm. In order to choose the best clustering a genetic algorithm is used, adopting a cluster validation index as fitness value (as

discussed later, the silhouette coefficient satisfied the needs of this research). Since the clustering algorithms requires the number of clusters as input, a grid search is performed varying the number of clusters to use and the number of features to be taken as subset w.r.t. the d-dimesional space of total features. Once clusters are obtained, they are taken as triples and used as training data for a GMM. The resluting mixture model is a high-level feature upon which classify the test set data. In order to compute such features for the test input, it is sufficient to extract the features subset found in the clustering phase and compute the log likelihood of the resulting low-level feature vector w.r.t. the mixture models obtained from the GMM training. Finally, an SVM is used for classification in a one-vs-one fashion.
This approach is aimed to investigate the relationship between audio excerpts having similar semantic properties and the way in which they get clusterized. The starting guess is that such excerpts, representing the same high-level musical characteristic (e.g. dynamicity), could also share similar low-level features values, hence being closely grouped in the dataset distribution. Therefore, if such guess is true, it should be possible to derive an example based definition of high-level features using naturally derived clusters from the datset.
The top-down approach follows the opposite path, finding the best clustering directly from the classification outcome: the fitness value used to evaluate the genetic algorithm execution is the error rate ϵ, defined as $1 - a$, with a being the classification accuracy. In this case there is no need to specify the number of features to be taken as subset of the whole features set, while the number of clusters is still a tuneable parameter. Classification is once again performed with a SVM, in the same way of the bottom-up approach.
Experiments show promising reults, especially for the top-down approach. In particular the best defined genres (classical and punk-metal) are always properly recognized, with a top value of 96% for classical music.

1.3 Thesis Outline

The thesis is structured in the following way: in chapter 2 the state of the art related to the arguments and goals of this work are analyzed, including the most relevant researches on musical genre classification and on high level features creation.
In chapter 3 the mathematical tools and methodologies used in our work are

shown; this includes the description of the low-level features used to create the data points, the clustering algorithms and the genetic algorithms used to find the clusters to be used as high level features and the mathematical formulation of Gaussian Mixture Models and Support Vector Machines, used for classification.
In chapter 4 the system framework is shown, divided in the two approaches followed: bottom-up and top-down. For each one a detailed description is provided.
In chapter 5 the experimental results of the research are shown, discussed and confronted with the current state of the art.
Chapter 6 concludes this work, presenting some considerations on the thesis and possible future work to be done in order to advance in the field using this approach.

Chapter 2

State of the art

In this chapter the current state of the art in music genre classification is described. In particular, in the first section the general problem is discussed and the most prominent studies in this field are described.
In the second section, the attention is focused on solutions and methodologies for such problem implementing high-level features. This approach is still largely unexplored; the main works adopting it are presented.

2.1 Automatic genre classification

As defined previously, music genre recognition is about classifying music into certain pre-defined categories. The goal is therefore to develop a system in which a music signal in digital form is provided as input, i.e. audio files from a local music collection, web radio streamings and so on, with the aim of finding its categorization in a taxonomy of pre-defined musical genres. The system's objective is to provide information about the input signal, describing how much it pertains to the various music genre classes and selecting the most relevant one. As an example, an audio player software could use this technique to offer a service for the automatic subdivision of a user's music collection. The design of the black box that generates this output from the input signal is the subject of music genre recognition research.
In order to accomplish such task, various methods have been proposed; typically, two main phases can be recognized in this process, as illustrated in figure 2.1 :

- Feature Extraction

- Classification

The feature extraction phase is aimed at giving a formal description of an input dataset, i.e. provide numerical values of the characteristics of its audio files. In the classification phase, a software tool called classifier is trained to discriminate musical genres by learning what values are most representative of each class. Once the learning step is completed, the classifier will be able to recognize the musical genre of an unlabeled input audio file by analyzing its features.

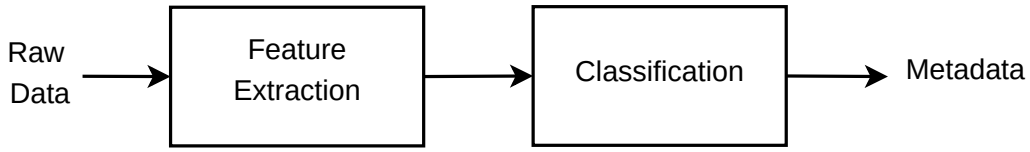

Figure 2.1: Block diagram of the genre classification process

2.1.1 Feature Extraction

The goal of the first step, feature extraction, is to get the essential information out of the input data. Features may be related to the main dimensions of music including melody, harmony, rhythm, timbre and spatial location. Usually, each audio file is segmented into frames of variable length, and for each frame different audio features are computed, with no relevance regarding their order. This approach is called bag of frames, recalling the bag of words technique applied in textual information retrieval.
Most of the features are derived from the DFT transform, hence describing the spectral shape of the audio file. Most of them are formally defined in the MPEG-7 standard [3]. However, descriptors based on the perceptual model of the human ear are gaining importance for this task. Mel frequency spectral coefficients (MFCC), first used for speech recognition, have been adopted for genre classification for the first time in [4], obtaining good results. Since tht time most of the systems implemented used this descriptors, in some cases coupled with their delta values (an example is given by [5]).
Harmonic descriptors have also been widely used: the chroma features, first introduced in [6], are essential for chord and key estimation and symbolic transcription. Perez Sancho et al. [7] perform genre classification using the

tonal harmony as discriminant factor, achieving good results for harmonically distant genres. Tao et al. [8] use the chroma features combined with MFCC and spectral descriptors to obtain a frame feature vector, successively merged in a larger time window. Statistics over such window are succesively computed and used as musical information for the audio files.
Rhythmic features are also largely used: in [4], the beat histogram is used, computed through discrete wavelet transform. Gouyon et al. [9] evaluate a set of 74 rythmic descriptors, divided into 3 main categories: metrical hierarchy derived, periodicity histogram derived and inter-onset interval histogram (IOIH) derived. Combining the most relevant of such features with 15 MFCC, they achieved 90% accuracy on their custom dataset.

2.1.2 Classification

In the field of multimedia information retrieval, classification is the problem of identifying the class to which new observations belong, where the identity of the class is unknown, on the basis of a training set of data containing observations whose class is known. Thus the requirement is that new individual items are placed into groups based on quantitative information on one or more measurements, traits or characteristics, and based on the training set in which previously decided groupings are already established. A system performing this task is called a classifier.
Two main approaches can be used: supervised and unspervised classification. In the supervised case the data given as input for the training of the classifier has been previously labeled, and the taxonomy of the classes is known in advance. The goal of the classifier is to learn what values of the features describing the training dataset best fit in each class, and choose the most likely pertaining one for an unlabeled input. In the unsupervised case the taxonomy of classes is not known, and the goal of the classifier is to determine how the input data are organized in order to define a grouping.
Classifiers are extremely useful in the field of musical genre classification, as they perfectly fit to the proposed problem, representing the most suitable way to categorize an unlabeled audio signal. As long as such field is concerned, supervised classification methods are preferred over unsupervised ones. The major interest of this approach is that one does not need to explicitly describe musical genres: the classifier attempts to form automatically relationships between the features of the training set and the related categories.

A description of the most used classifiers follows: K-Nearest Neighbor (KNN) is a non-parametric classifier based on the idea that a small number of neighbors influence the decision on a point. More precisely, for a given feature vector in the target set, the K closest vectors in the training set are selected (according to some distance measures) and the target feature vector is assigned the label of the most represented class in the K neighbor (there is actually no other training than storing the features of the training set).

An example of usage can be found in [10], [11] and [12]. Although widely diffused, K-NN is often used as comparison with other machine learning approaches, and performs generally worse than more robust methods such as SVMs, which will be described later in this chapter and more thoroughly in section 3.5.

Gaussian mixture models (GMM) model the distribution of feature vectors. For each class, it is assumed the existence of a probability density function expressible as a mixture of a number of multidimensional Gaussian distributions. The iterative expectation maximization (EM) algorithm is usually used to estimate the parameters for each Gaussian component and the mixture weights. GMMs have been widely adopted in the music information retrieval community; they can be used as classifiers by creating a mathematical model of the training dataset based on a mixture of gaussian components, and then applying a maximum likelihood criterion on an unlabeled feature vector to find the model best suited to represent it, hence achieving genre classification. Examples of this approach are [4] and [13], in which GMM are used to directly describe musical genres. Scaringella et al. [14] modeled each genre by a single 4-state HMM, with each state characterized by a GMM with 3 mixture components.

Hidden Markov Models (HMM) can be used for classification purposes. The Hidden Markov Model is a finite set of states, each of which is associated with a (generally multidimensional) probability distribution. Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state an outcome or observation can be generated, according to the associated probability distribution. It is only the outcome, not the state visible to an external observer and therefore states are "hidden" to the outside; hence the name Hidden Markov Model. They have been extensively used in speech recognition because of their capacity to handle time series data. In the context of genre classification, this technique is often used to describe the harmonic progression of a musical piece; in [15], the chord of

the current segment is estimated via N-gram, and a Hidden Markov Model is used to model the chord progression following a language modeling process. Support vector machines (SVMs) are a set of related supervised binary classification methods that analyze data and recognize patterns. Given training examples labeled either "class 1" or "class 2", a maximum-margin hyperplane splits the "yes" and "n" training examples, such that the distance from the closest examples (the margin) to the hyperplane is maximized. They are based on two properties: margin maximization (which allows for a good generalization of the classifier) and nonlinear transformation of the feature space with kernels (as a data set is more easily separable in a high dimensional feature space). First used in the field of genre classification in [16], they have rapidly become one of the most used tools, due to their distinctive performance. Tsanetakis et al. [17] use timbral features combined with fundamental rhythmic patterns extracted for each genre, using SVMs for classification. Mayer et al. [18] performed genre classification using lyrics and rhyme information. Various classifiers were used for comparison, with SVMs achieving the best results. Pachet et al. [19] developed a system similar to the one presented in this thesis in the feature selection process: a set of analytical features for the current classification problem is derived from a larger set of low level ones, using genetic algorithms. Classification was attempted with various classifiers, and also in this case SVM outperformed the others.

2.1.3 MIREX

The Music Information Retrieval Evaluation eXchange (MIREX)[1] represents an annual community-based MIR/MDL evaluation event where techniques and algorithms tailored to a variety of tasks are submitted by research laboratories from all over the world to the International Music Information Retrieval Systems Evaluation Laboratory (IMIRSEL)[2]. These algorithms are then run along with standardized datasets, and evaluated using community-defined evaluation metrics. The evaluation results are published on a year basis and they are presented at the International Conference on Music Information Retrieval[3]. A set of music classification tasks are present among the

[1] http://www.music-ir.org/mirex/wiki/MIREX_HOME
[2] http://www.music-ir.org/
[3] http://www.ismir.net

others; as an example, in the MIREX 2010 edition the following classification tasks were proposed:

- Audio Artist Identification
- Audio US Pop Genre Classification
- Audio Latin Genre Classification
- Audio Music Mood Classification
- Audio Classical Composer Identification

The standardized datasets are fairly important in classification, as they provide a common ground truth upon which test and compare the submitted systems. This is a key concept in the field on musical genre classification, as the definition of genre is not objective and often based on abritrary, ambiguous characteristics. The only way to have an objective basis on which evaluate the research is therefore the adoption and sharing of a common, example-based definition of the genres. MIREX is also an excellent tool to estimate the state of the art in the genre classification problem. Analyzing the outcome of the competition over the latest years, a valuable review of the used methods and the achieved results can be drawn.
In the 2008 edition, Tsanetakis et al. [20] ranked first in mixed genre audio classification using the software framework MARSYAS [4]. The system computed a single feature vector for each 30 second long clip using spectral centroid, rolloff, flux and MFCC as features. To capture the feature they computed a running mean and standard deviation over the past M frames (where $M = 40$, corresponding to approximately a texture window of 1 second)), resulting in a feature vector of 32 dimensions. The sequence of feature vectors is collapsed into a single feature vector representing the entire audio clip by taking again the mean and standard deviation across the 30 seconds (the sequence of dynamics features) resulting in the final 64-dimensional feature vector per audio clip. A linear SVM was used as classifier.
In 2009, Cao et al. [21] obtained the best classification accuracy; their system is based on a Gaussian Super Vector followed by Support Vector Machine (GSV-SVM) framework. A set of acoustic features such as MFCC, rhythm pattern (RP) are extracted from every music excerpt in the front-end part;

[4] `http://marsyas.info/`

frame-level features are used to adapt a Gaussian mixture model (GMM) from a music universe background model (UBM) in our system. Then a super vector representing the adapted GMM model is obtained and utilized to train models of different genres. Similarly, genre labels of test music pieces are decided by SVM classification result on the super vector feature of the test music.
Finally, in the 2010 edition Seyerlehner et al. [22] achieved the best results using block-level features extracted from the cent-scaled magnitude spectrum of the training set samples, focusing the attention on rythm (spectral, delta-spectral and fluctuation patterns) and harmony (Correlation and spectral contrast patterns). Once again, classification is obtained through SVM.

2.2 High-level features

The features commonly used in the field of MIR are typically low-level; they are extracted from the digital representation of the audio signal and have little or no semantic meaning for a human. Instead, they throw into relief basic characteristics of the analyzed sound to be used by a machine in order to achieve similarity or classification tasks. This holds even for features derived from the human auditory system, like the MFCC: while their abstraction level is based on our perceptual model and is higher than other descriptors (e.g. statistics over the spectrum of a signal), they have no immediate meaning to a consumer.
Hence, the implementation of high-level features can be useful not only to improve the performance of classification systems (although desirable), but also to provide an intelligible description of the signal and discriminate and classify musical pieces on a semantically consistent basis. Little research has been done in this direction; the following is a resume of the most important works.
Zhu et al. [23] proposed three new features based on the recognition of the musical instruments used in a musical piece: the distribution of instruments is proposed to represent the percentage that each group of instruments is played in the music performance, and the means and standard deviations of notes in each instrument group are extracted to represent the instrument-based melody and the statistical features on some time range. A dictionary of instruments' spectrum is constructed in order to get enough information on different instruments. The classification step is performed through a GMM,

which models the training data with a mixture of 16 gaussian components for each genre and computes the maximum likelihood for each unlabeled observation in order to categorize it. Their experimental results showed that such features have a 4.5% higher average classification accuracy than that of MFCC with energy term.

Tsuchihashi et al. [24] implemented a system to extract features from the bass lines of musical pieces. Bass parts play an important role for two (rhythm and harmony) of the three basic elements of music; moreover, bass line styles can often be associated to a specific genre (e.g. an upbeat, dynamic and syncopated bass line is associable with the funk genre). The method used is first described by Goto et al. [25]: it detects the pitch of the bass part, specifically the most predominant pitch in a low-pitch range, for each frame. Using this information, features of pitch variability and features of pitch motions are obtained. The Mahalanobis distance between the feature vector of each frame and the feature distribution of each genre is then computed. Coupling such features with classical low-level features, an average accuracy of 62,7% was achieved using the ISMIR04 dataset.

A similar approach can be found in [26]: the goal here is to design transcription-based high-level features that enable a better characterization of the bass track in different songs. Furthermore, the work aims to develop a general method to translate musicological knowledge into rules on feature values that can be easily evaluated, in order to facilitate the design of an instrument-related classifier. Starting from melodic, rhytmic and structural features, different bass playing styles are modeled by applying a generic framework to translate known musicological properties into explicit restrictions on feature values. Every style is directly linked to a musical genre (e.g. Walking bass style to swing). The system achieves mixed results, ranging from an accuracy of 70% for latin music to 6% for pop.

Finally, Prandi et al. [27] developed the system upon which this work is based, which performs visualization and classification task by means of three high-level, semantic features extracted computing a reduction on a multi-dimensional low-level feature vector through the usage of Gaussian Mixture Models. The low-level feature extraction block receives the music stream and performs a continuous extraction of sets of low-level features from consecutive small parts of the signal. Each resulting low-level feature vector is fed into the next high-level feature extraction module, which implements three high-level semantic features by means of Gaussian Mixture Models (GMMs).

Each feature is mapped on a single GMM, which performs a non-linear reduction of the low-level feature vector to a single scalar number. The three values related to the three semantic descriptors make up the high-level feature vector, which is used both for visualization and genre classification. The classification exploits a set of one-against-one three-dimensional Support Vector Machines trained on some target genres.
The difference between this approach and the one presented in this work is the high-level feature creation: while in the first case such task is done by arbitrarly selecting musical pieces representing the chosen high-level features (e.g. dynamicity is composed by upbeat dance songs, bebop jazz and so on), here the definition of such features is derived by the clustering obtained from low-level descriptors extracted from the dataset.

Chapter 3

Tools

In this section the main tools used in the implementation of the thesis are listed and explained. First, the low level features used are analyzed, subdivided into basic, timbric, harmonic and rythmic features.

Secondly, a description of genetic algorhitms is presented. They are an heuristics for search problems that mimic the process of natural evolution, used in this work for clustering optimization w.r.t. an objective funciton. The section starts with a general coverage of this method, successively going in depth in the tuning parameters used in this work. Hence the initialization, selection, reproduction and mutation functions are explained.

Next, an in-depth explanation of the applied clustering techniques is provided. This includes the clustering algorithms used to partition our dataset, and the clustering validation indexes applied to find the best clusterings.

Finally, the implemented classification methods are presented, with emphasys on their mathematical modeling. This includes a comprehensive analysis of Gaussan Mixture Models (GMM) and Support Vector Machines (SVM).

3.1 Audio Features

Most audio classification systems combine two processing stages: feature extraction followed by classification. A variety of signal features have been used for this purpose; in this work, a large set of descriptors is used. They are subdivided into timbric, harmonic and rythmic descriptors.

3.1.1 Basic features

Zero-crossing rate

The zero-crossing rate is the rate of sign-changes along a signal, i.e., the rate at which the signal changes from positive to negative or back.

$$zcr = \frac{1}{T-1}\sum_{t=1}^{T-1}|sgn(s(t)) - sgn(s(t-1))|\frac{F_s}{N} \tag{3.1}$$

where T is the number of samples is $s(t)$ and F_s is the sampling frequency.

Root mean square energy

The root mean square energy is a measure of the energy of the signal; it is computed by taking the root average of the squared signal x:

$$rms = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \tag{3.2}$$

3.1.2 Timbric features

Spectral centroid

The spectral centroid is the baricenter of the spectrum; it is calculated as the weighted mean of the frequencies present in the signal, with their magnitude as the weights:

$$sc = \frac{\sum_{n=0}^{N-1} f(n)x(n)}{\sum_{n=0}^{N-1} x(n)} \tag{3.3}$$

where $x(n)$ represents the weighted frequency value of bin n, and $f(n)$ is the central frequency of that bin.

Spectrum spread

Also called instantaneous bandwidth, it is defined as the second central moment (i.e. the variance)of spectrum. Being the squared deviation of the distribution from its mean value, the variance is always positive and is a measure of the dispersion or spread of the spectrum. It is computed thorugh the following equation:

$$sspread = \sum_{n=0}^{N-1} x(n)(f(n) - sc)^2 \tag{3.4}$$

where sc is the spectral centroid.

Spectral skewness
The skewness gives a measure of the asymmetry of the spectrum around its mean value. It is computed from the 3rd order of the moment:

$$sskew = \frac{\sum_{n=0}^{N-1} x(n)(f(n) - sc)^3}{\sigma^3} \tag{3.5}$$

Values of skewness equal to 0 indicate a symmetric distribution of the spectrum, positive values indicate more energy on the left, negative values indicate more energy on the right

Spectral kurtosis
The kurtosis gives a measure of the flatness of the spectrum around its mean value. It is computed from the 4th order of the moment:

$$skurt = \frac{\sum_{n=0}^{N-1} x(n)(f(n) - sc)^4}{\sigma^4} \tag{3.6}$$

Values of kurtosis below 3 indicate a flatter distribution of the spectrum, while values above 3 indicate a peaker one.

Spectral rolloff
The spectral rolloff frequency is defined as the frequency below which 85% of the accumulated spectrum magnitude is concentrated:

$$\sum_{k=0}^{K_{roll}} |S(k)| = 0.85 \sum_{k=0}^{N_{Ft}/2} |S(k)| \tag{3.7}$$

Spectral brightness
The spectral brightness is defined as the amount of energy in the spectrum starting from a specified cutoff frequency (in this case 1500 Hz), divided by the total energy of the signal spectrum:

$$\frac{\sum_{k=K_c}^{K} |S(k)|}{\sum_{k=1}^{K} |S(k)|} \tag{3.8}$$

a graphical illustration of the spectral brightness is provided in figure 3.1

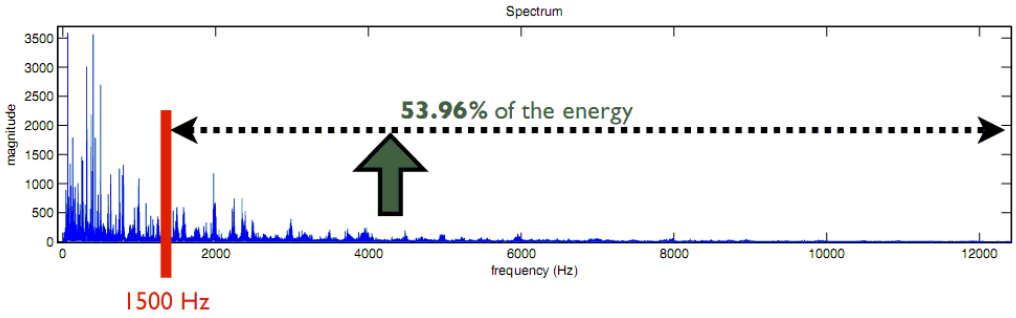

Figure 3.1: Graphical illustration of the spectral brightness

Spectral entropy

Spectral entropy is computed as the Shannon's entropy of the signal spectrum $S(k)$:

$$se = -\frac{\sum_{k=1}^{K} S(k)log(S(k))}{log(N)} \tag{3.9}$$

the entropy is divided by the logarithm of the length N of the spectrum in order to have a result which is independent from the windowing length.

Spectral roughness

The roughness r of a signal is related to the beating phenomena between the peaks of its spectrum; in paerticular, the roughness of a spectrum having two sinusoidal components with frequencies f_1, f_2 and amplitudes A_1, A_2, where $f_{min} = min(f_1, f_2)$, $f_{max} = max(f_1, f_2)$, $A_{min} = min(A_1, A_2)$, $A_{max} = max(A_1, A_2)$, is:

$$r = X^{0.1} * 0.5(Y^{3.11}) * Z \tag{3.10}$$

$$X = A_{min} * A_{max} \tag{3.11}$$

$$Y = \frac{2A_{min}}{(A_{min} + A_{max})} \tag{3.12}$$

$$Z = e^{-b_1 s(f_{max} - f_{min})} - e^{-b_2 s(f_{max} - f_{min})} \tag{3.13}$$

where $b_1 = 3.5$, $b_2 = 5.75$, $s = \frac{0.24}{(s_1 f_{min} + s_2)}$, $s_1 = 0.0207$ and $s_2 = 18.96$. An estimation of the total roughness can be obtained by computing the peaks of the spectrum, and taking the average of all the dissonance between all possible pairs of peaks.

Irregularity

The irregularity of a spectrum is the degree of variation of the successive peaks of the spectrum. It is computed as the sum of the square of the difference in amplitude between adjoining partials:

$$irr = \frac{\sum_{k=1}^{N}(a_k - a_{k+1})^2}{\sum_{k=1}^{N} a_k^2} \tag{3.14}$$

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. More precisely, it is usually calculated as the 2-norm (also known as the Euclidean distance) between the two normalised spectra:

$$sf = \|S_n - S_{n-1}\| = \sqrt{(S_n - S_{n-1})^2} \tag{3.15}$$

Spectral Flatness

Spectral flatness, measured in decibels, provides a way to quantify how tone-like a sound is, as opposed to being noise-like. The meaning of tonal in this context is in the sense of the amount of peaks or resonant structure in a power spectrum, as opposed to flat spectrum of a white noise. The spectral flatness is calculated by dividing the geometric mean of the power spectrum by the arithmetic mean of the power spectrum:

$$sf = \frac{\sqrt[K]{\prod_{k=1}^{K-1} S(k)}}{\frac{\sum_{k=1}^{K-1} S(k)}{K}} \tag{3.16}$$

Mel frequency cepstral coefficient

Mel-Frequency Cepstral Coefficients (MFCC) originate from automatic speech recognition, where they have been used with great success. They have become very popular in the MIR society where they have been used successfully for music genre classification and for categorization into perceptually relevant groups such as moods and perceived complexity.

The MFCCs are to some extent created according to the principles of the human auditory system, but also to be a compact representation of the amplitude spectrum and with considerations of the computational complexity. They are commonly derived as follows:

- Take the Fourier transform of (a windowed excerpt of) a signal.
- Map the powers of the spectrum obtained above onto the mel scale, using triangular overlapping windows.

$$Pitch(Mel) = 1127.0148 \log\left(1 + \frac{f}{700}\right) \tag{3.17}$$

- Take the logs of the powers at each of the mel frequencies.
- Take the discrete cosine transform of the list of mel log powers, as if it were a signal.

$$c_i = \sum_{j=1}^{N_f} \log\left(E_j\right) \cos\left[i\left(j - \frac{1}{2}\right)\frac{\pi}{N_f}\right], \quad 1 \leq i \leq N_c \tag{3.18}$$

where c_i is the ith-order MFCC, E_j is the spectral energy measured in the critical band of the jth mel filter and N_f is the total number of mel filters (typically $N_f = 24$). N_c is the number of cepstral coefficients c_i extracted from each frame (in this case $N_c = 13$). The derivation process of MFCCs is illustrated in figure 3.2

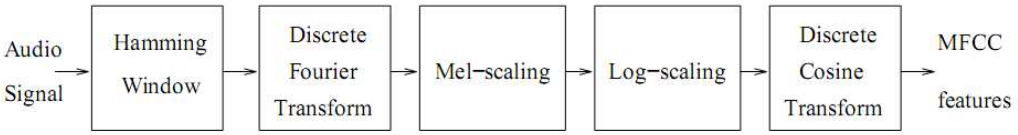

Figure 3.2: MFCC derivation block diagram

3.1.3 Harmonic features

Inharmonicity

Inharmonicity estimates the amount of partials that are not multiples of the fundamental frequency; it is computed by considering the ideal locations of harmonics vs. the actual harmonics in a spectrum where f_0 is the fundamental and f_n refers to subsequent harmonics:

$$h_n = \frac{|f_n - nf_0|}{nf_0} \tag{3.19}$$

Chroma features

Chroma features are a powerful representation for music audio in which the entire spectrum is projected onto 12 bins representing the 12 distinct semitones (or chroma) of the musical octave. Since, in music, notes exactly one octave apart are perceived as particularly similar, knowing the distribution of chroma even without the absolute frequency (i.e. the original octave) can give useful musical information about the audio.
A pitch class is defined to be the set of all pitches that share the same chroma. For example, the pitch class corresponding to the chroma C is the set $\{..., C0, C1, C2, C3, ...\}$ consisting of all pitches separated by an integral number of octaves. To obtain the energy amount for each pitch class, an audio signal is decomposed into 88 pitch subbands,the Short-Time Mean-Square Power (STMSP) for each subband are computed, and then all STMSPs that belong to the same pitch class are added up. This results in an STMSP for the respective chroma.

A chromagram is a temporal sequence of vectors of chroma features, often used for harmony analysis. An example of chromagram is shown in figure 3.3.

Chromatic Flux

In this work a novel feature is defined, called chromatic flux; it is computed as the euclidean distance between the chroma features vectors of two successive segments of the audio signal:

$$cf = \|C_i - C_{i-1}\| = \sqrt{(C_i - C_{i-1})^2} \tag{3.20}$$

where C_i is a vector composed by the chroma features of the ith segment of the audio signal. Chromatic flux can be useful to detect the harmonic

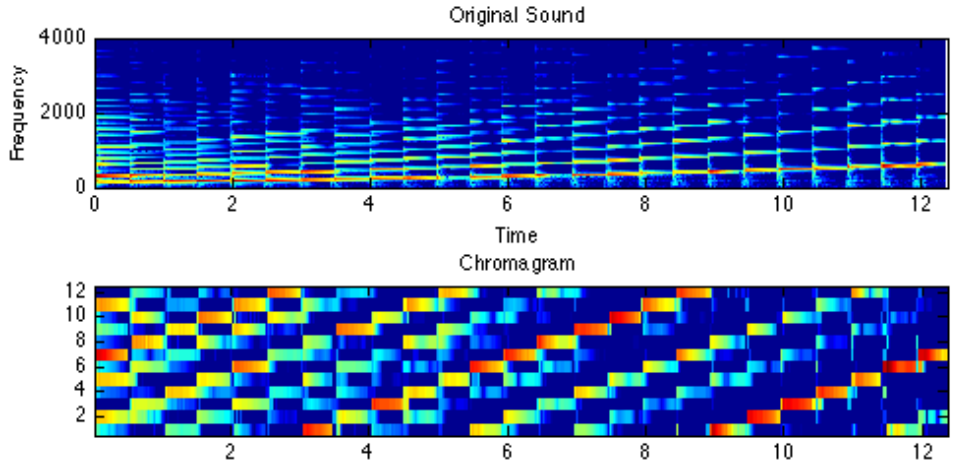

Figure 3.3: Chromagram of the Shepard tone

variations in a musical piece; moreover, the value of the distance determines how much the harmonic structure is changing from segment to segment. In harmonically dynamic genres like jazz or classical music, the harmonic flux is expected to be relatively high during the entire duration of the audio file, while in more structurally simple genres like electronic it is expected to be fairly lower.

Harmonic change detection function

Similarly to the previous feature, the Harmonic Change Detection Function (HCDF) aims to detect the harmonic variations of a musical piece. The implementation is fairly complex: at the lowest level there is a Constant-Q spectral analysis, which is a particular kind of Fourier transform, defined as a bank of filters having geometricaly spaced center frequencies. Adeguately configuring the filters parameters, center frequencies can be placed in correspondence with the musical notes frequencies. This analysis is followed by a 12-semitone Chromagram decomposition. A harmonic Centroid transform is then applied to the Chroma vectors: this process returns a 6-dimensional feature vector based derived from the principal harmonic relations within the 12 tones of an octave (fifths, major thirds and minor thirds). Such vector is then smoothed with a Gaussian filter before the distance measure is calculated. More details on its implementation can be found in [3].

3.1.4 Rythmic features

Bpm estimation

There exist several ways to estimate the bpm of a musical segment. In this work, it is found through the autocorrelation function of an onset detection curve following the amplitude envelope that can be easily constructed by rectifying and smoothing (i.e., low-pass filtering) the signal:

$$E_0(n) = \frac{1}{N} \sum_{m=-\frac{N}{2}}^{\frac{N}{2}-1} |x(n-m)|w(m) \tag{3.21}$$

where is an N-point window or smoothing kernel, centered at $m = 0$. The displacement of the peaks in the autocorrelation function is a widely used measure for estimating the tempo.

Event density

Event density estimates the average number of acoustic events, i.e. the number of note onsets per second. This can be trivially derived by counting the onsets found in the onset detection curve.

Pulse Clarity

This descriptor measures the sensation of pulse in music. Pulse is here seen as a fluctuation of musical periodicity that is perceptible as "beats", i.e. changes in some characteristics of the musical piece that take place in regular interval of times. This can be of any musical nature (melodic, harmonic or rhythmic) as long as it is perceived by listeners as a fluctuation in time.

There are several ways to describe the clarity of the audio pulses, a detailed description of the implementations of this feature can be found in [4]. In this work the maximum of the autocorrelation of the onset detection curve is used as a reference in order to normalize the autocorrelation function. The onset detect curve is a function describing the number and position of musical events in the analyzed audio segment. In this way, the actual values shown in the autocorrelation function correspond uniquely to periodic repetitions, and are not influenced by the global intensity of the total signal.

3.2 Genetic algorithms

There exists a large class of problems for which no computationally efficient algorithms have been developed. Many of these problems contain globally optimal solutions within a large search space.One approach in dealing with such problems is the use of genetic algorithms, which are based on the principles of natural evolution.
Genetic algorithms (GAs) are stochastic-optimization methods first proposed by James Holland in 1975. GAs encode each point in a solution space into a string called a chromosome. Unlike other data mining samples, which are often predefined and not likely to change as the data mining algorithm is carried out, samples in GAs change or evolve during their execution.
The features of interest are located in different portions of the string. The string structures in the chromosomes undergo operations similar to the natural evolution process to yield increasingly optimal solutions. The quality of the solution is based on a fitness value, which relates to the objective function for the optimization problem.
GAs consist of five major phases: schema encoding, fitness evaluation, parent selection, crossover, and mutation. Each of these phases is described in the next sections.

3.2.1 Schema encoding

The first phase of a genetic algorithm is to design a representation of a solution for the given problem. Although solution forms vary by application, the most common representation is a string of characters used for feature representation. The characters are members of a fixed alphabet whose size determines the number of features that can be represented by each character. In this work the most suitable encoding is associating binary strings to the features set; in particular, each chromosome is a set of features, i.e. the subset of dimensions for which the dataset gets clusterized.
A set of chromosomes forms a collection called a population. The initial population is created by selecting a random set of chromosomes. The size of the population also has an impact on GA performance as well as the quality of the resulting solution. If the initial population is too small, the GA may converge quickly, thus finding only a local optimum. Conversely, if the initial population is too large, the GA may waste computational resources and will

most likely require more time to converge. Regarding this work, a population number of 50 individuals has proven to be a good choice.

3.2.2 Fitness evaluation

After creating a population, a fitness value is calculated for each member in the population because each chromosome is a candidate solution. The fitness of a solution is a comparative measure for judging the quality (optimality) of candidate solutions. The choice of an appropriate fitness function depends significantly on the given problem. In this work, each of the presented aprroaches has its own fitness function.
In the bottom-up approach, where the goodness of each clustering instance must be evaluated, the Silouette index is used. Being its range [-1,1], such index is inverted and incremented by 1, so that its new range is [0,2]. Now a value of 0 indicates an optimal clustering, while a value of 2 stands for a bad one.
In the top-down approach the fitness value is represented by the error rate of the entire classification process. The error rate can be directly obtained by the confusion matrix resulting from the SVM label prediction. First, we introduce the Correct Classification Rate (CCR), which is defined as:

$$CCR = \frac{\sum_{i=1}^{D} C_f(i,i)}{N} \tag{3.22}$$

where C_f is the confusion matrix, N is the total number of elements in C_f and D is the number of classes. The error rate can now be defined as:

$$ER = 1 - CCR \tag{3.23}$$

Its range goes from 0, indicating a perfect classification, to 1, indicating a totally wrong one.

3.2.3 Selection

The selection function chooses parents for the next generation based on their fitness values. An individual can be selected more than once as a parent, in which case it contributes its genes to more than one child.
The method used in this work, called stochastic uniform, lays out a line in which each parent corresponds to a section of the line of length proportional to its fitness value. The algorithm moves along the line in steps of equal size. At each step, the algorithm allocates a parent from the section it lands on.

3.2.4 Crossover

The core of genetic algorithms involves exchanging chromosome information between highly-fit individuals. To exploit similarities between chromosomes, a crossover function is used. The process of crossover is analogous to the evolutionary process of exchanging gene segments between two parents and then passing them on to their children. This process allows children to inherit desirable traits from their increasingly-optimal parents.
A usual strategy implies a one-point segmentation: an integer n between 1 and the number of features D is chosen, and then:

- vector entries numbered less than or equal to n are selected from the first parent;
- vector entries numbered greater than n are selected from the second parent;
- the obtained entries are concatenated to form a child vector.

The same method can be implemented segmenting the parents in 2 points, and so on. The function used in this thesis, called *scattered crossover*, is a generalization of this procedure: first, it creates a random binary vector of length D; then, it selects the genes where the vector is a 1 from the first parent, the genes where the vector is a 0 from the second parent, and combines the genes to form the child.

3.2.5 Mutation

Even though crossover exploits the potential of existing chromosomes, the population may not contain all the encoded information needed to solve a given problem. To address this issue, a mutation operator capable of generating new "genetic" information is introduced. The most common method of implementing mutation on binary strings is to invert a given bit based on a probability-based mutation rate.
A mutation operator is effective in preventing any single bit from converging to a value throughout the entire population. More importantly, the mutation operator can prevent the population from converging and stagnating at local optima. The mutation rate is typically set to a low probability such that good chromosomes obtained from crossover are not lost. High mutation rates cause

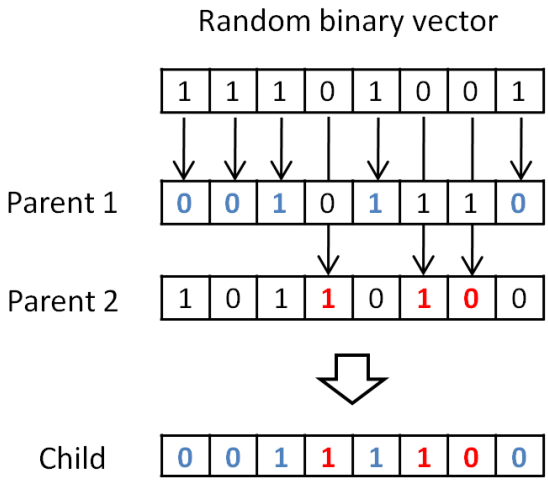

Figure 3.4: The scattered crossover function

the GA performance to approach that of a random search. A typical value for the mutation rate, used also in this work, is 0.1

3.2.6 Stopping criteria

The process of evaluating fitness, selecting parents, and applying crossover and mutation operators occurs until a given stopping criterion is met. In this case, two criteria have been defined:

- the algorithm reaches 600 iterations;
- the average change of the fitness value in 350 iterations is smaller than 10^{-6}.

3.3 Clustering

In this work, clustering is used to derive a set of high-level features from a dataset of low-level ones. In the bottom-up approach, the idea is to find the most natural clustering according to the current observations and features composing the dataset through genetic algorithm optimization. The fitness function for such optimization method should be a measure of the goodness of the clustering; for this reason, the silhouette coefficient was adopted.

In the top-down approach no validation indexes are needed, since the fitness value is the error rate of the final classification. In both cases the number of clustering iterations is high, hence k-means was chosen as clustering algorithm for its low computational cost.

3.3.1 K-means

In statistics and machine learning, k-means clustering is a method of cluster analysis which aims to partition n observations into k clusters S_0, ..., S_k ($1 < k < n$) in which each observation belongs to the cluster with the nearest mean, so as to to minimize the within-cluster sum of squares (WCSS):

$$\arg\min_{S} \sum_{i=1}^{k} \sum_{x_j \in S_i} \|x_j - \mu_i\|^2 \tag{3.24}$$

where μ_i is the mean of points in S_i. The basic implementation of the algorithm is the following:

Algorithm 3.1 K-means algorithm

1. Select k points as initial centroids
2. **repeat**
3. form k clusters by assigning all points to the closest centroid
4. recompute the centroid for each cluster
5. **until** Centroids don't change

As it is a heuristic algorithm, there is no guarantee that it will converge to the global optimum, and the result strongly depends on the initial clusters. Due to its computational speed, though, it is common practice to run the algorithm several times with different starting conditions and choose the best output (i.e. the clustering minimizing the $WCSS$)
This method presents two major issues: the first one is the parameter k, i.e. the number of clusters. Such parameter must be provided by the user, and knowing in advance the best number of clusters in which a dataset should be partitioned is not always possible. In order to overcome this problem an optimization routine is frequently used, clustering the distribution with varying values of k and evaluating the output with validation indexes such as the Silhouette coefficient (see next subsection).

The second issue is the weakness of the algorithm towards non globular clusters: k-means is based on Euclidean distances, hence it segments the data space employing linear hyper-planes; therefore, it will fail if nonlinear separation boundaries are required. This is well shown in figure 3.5, where the original distribution presents non globular data partitioning that can't be relieved by the k-means method.

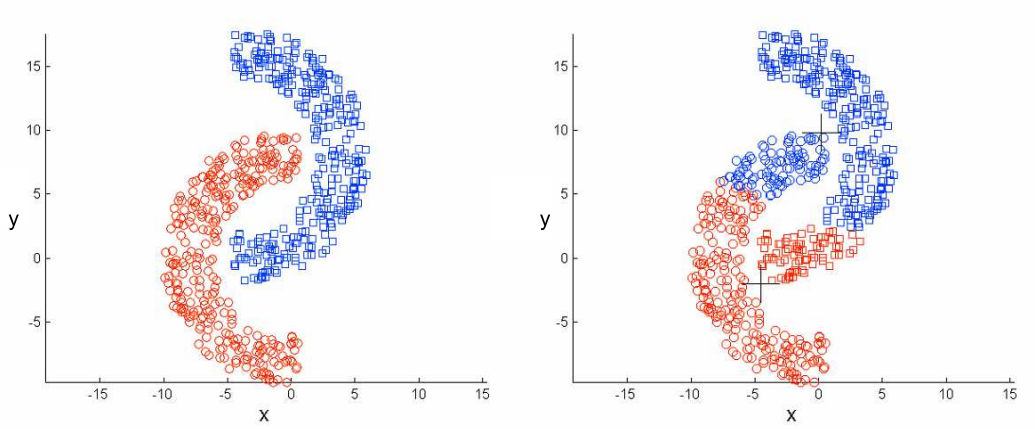

Figure 3.5: K-means weakness on non-globular clusters. On the left: original distribution. On the right: K-means clustering

3.3.2 Silhouette index

The Silhouette index of a clustering is a robust measure of the quality of the clustering introduced in [4]. The Silhouette index $SI(x)$ of each observation x is defined as follows: If x is the only point in its cluster, then $SI(x) = 0$. Denote by $a(x)$ the average distance between x and all other points of its cluster. For any other cluster C, let $d(x, C)$ be the average distance between x and all points of C. The minimum $b(x) = \min\{d(x, C) : x \notin C\}$ is the distance from x to its nearest cluster C to which x does not belong. Finally, put:

$$SI(x) = \frac{b(x) - a(x)}{\max\{a(x), b(x)\}} \tag{3.25}$$

The Silhouette index of the whole clustering is found as the average index over all observations. The Silhouette index always belongs to $[-1:1]$. The partition with highest Silhouette index is regarded as optimal.

3.4 Gaussian mixture models

Gaussian Mixture Models (GMM) have been widely used in a vaste range of classification applications; their capability to model arbitrary probability densities and to represent general spectral features motivates their use. The GMM approach assumes that the density of an observed process can be modelled as a weighted sum of component densities given by:

$$p(x|\lambda) = \sum_{m=1}^{M} c_m b_m(x) \tag{3.26}$$

where x is a d-dimensional random vector,, M is a number of mixture components,c_m is the weight associated to the component and $b_m(x)$ is a Gaussian density function, parameterized by a mean vector μ_m and the covariance matrix Σ_m.
In particular, we can define $b_m(x)$ as:

$$b_m(x) = \frac{1}{2\pi^{D/2}|\Sigma^{1/2}|} \exp\left[-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)\right] \tag{3.27}$$

The Gaussian distribution is usually a quite good approximation for a class model shape in a suitably selected feature space. It is a mathematically sound function and extends easily to multiple dimensions.

3.4.1 Maximum likelihood

Assume that there is a set of independent samples $X = \{x_1, \ldots, x_N\}$ drawn from a single distribution described by a probability density function $p(x|\lambda)$. The likelihood function

$$\mathcal{L}(x|\lambda) = \prod_{n=1}^{N} p(x_n|\lambda) \tag{3.28}$$

tells the likelihood of the data x given the distribution or, more specifically, given the distribution parameters λ. The goal is to find $\hat{\lambda}$ that maximizes the likelihood:

$$\hat{\lambda} = \arg\max_{\lambda} \mathcal{L}(x|\lambda) \tag{3.29}$$

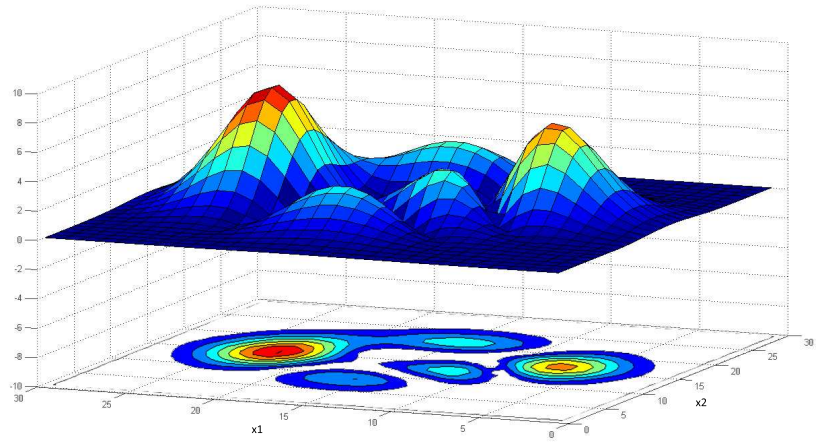

Figure 3.6: An example surface of a two-dimensional Gaussian mixture PDF.

Usually this function is not maximized directly but the logarithm

$$L(x|\lambda) = \ln \mathcal{L}(x|\lambda) = \sum_{n=1}^{N} \ln p(x_n|\lambda) \tag{3.30}$$

called the log-likelihood function is used, being it analytically easier to handle. Because of the monotonicity of the logarithm function the solution to Eq.(3.29) is the same using $\mathcal{L}(x|\lambda)$ or $L(x|\lambda)$.

3.4.2 The Figueredo-Jain algorithm

A useful algorithm for estimating the parameters of a GMM that maximize the likelihood of a set of n data vector is the Expectation Maximization (EM) algorithm. The algorithm works by iteratively updating the parameters according (in the case of diagonal covariance matrices) to the following equations:

$$\mu_m^{new} = \frac{\sum_{i=1}^{n} p(m|x_i, \lambda) x_i}{\sum_{i=1}^{n} p(m|x_i, \lambda)} \tag{3.31}$$

$$\Sigma_M^{new} = \frac{\sum_{i=1}^{n} p(m|x_i, \lambda)(x_i - \mu_m)'(x_i - \mu_m)}{\sum_{i=1}^{n} p(m|x_i, \lambda)} \tag{3.32}$$

$$c_m^{new} = \frac{1}{n} \sum_{i=1}^{n} p(m|x_i, \lambda) \tag{3.33}$$

where the value $p(m|x_i, \lambda)$ can be computed as:

$$p(m|x_i, \lambda) = \frac{c_m b_m(x_i)}{\sum_{j=1}^{M} c_j g_j(x_i)} \tag{3.34}$$

The flaws of this algorithm are that the outcome depends from an initial guess of the parameters to be maximized, and that the number of components M must be provided by the user in advance (much like the number of clusters in the k-means algorithm, which is a direct derivation of the EM method).
The Figueredo-Jain algorithm tries to overcome these issues by adjusting the number of components during estimation by annihilating components that are not supported by the data. FJ also allows to start with an arbitrarily large number of components, which tackles the initialization issue with the EM algorithm. The initial guesses for component means can be distributed into the whole space occupied by training samples, even setting one component for every single training sample.
In this case a component-wise EM algorithm (CEM) is adopted. CEM updates the components one by one, computing the E-step after each component update, where the basic EM updates all components 'simultaneously'. When a component is annihilated its probability mass is immediately redistributed strengthening the remaining components. More details about the Fugeredo-Jain algorithm can be found in [5].

3.5 Support Vector machines

Support Vector Machine (SVM) is a binary classifier that learns the boundary between items belonging to two different classes. It works by searching a suitable separation hyperplane between the different classes in the feature space. The best separation hyperplane maximizes its distance from the closest training items.
More formally: we have N training points, where each input $x_i = (x_1, x_2, \ldots, x_D)$ is a vector in D-dimension space $\mathbb{R}^D$. Each training point has its own class y_i, $y_i = +1$ denotes the positive class and $y_i = -1$ denotes the negative class. Assuming that the data is linearly separable, it is possible to draw a line to separate two classes of datapoints. The hyperplane has the form

$$w^T x + b = 0 \tag{3.35}$$

where w is the normal vector of the plane, b is the offset, $\frac{b}{\|w\|}$ is the perpendicular distance from the hyperplane to the origin. For an input x_i, there are two possible situations:

- if $w^T x_i + b \geq +1$, x_i belongs to the positive class and $y_i = +1$
- if $w^T x_i + b \leq -1$, x_i belongs to the negative class and $y_i = -1$

The two hyperplanes H_1 and H_2 that go through support vectors form the gap between two classes for datapoints. The distances from H_1 and H_2 to the separating plane are called SVM margin, $d_1 = d_2$. It is desirable to adjust the separating plane so that it is as far as possible from both sides of the two classes, this is the same as maximizing the margin d_1, d_2.
The margin can be calculated as $\frac{1}{\|w\|}$, so maximizing the margin is similar to minimizing $\|w\|$ or minimizing $\frac{1}{2}\|w\|^2$. The process of training SVM is equivalent to solving the optimization problem to find w and b:

$$\min \frac{1}{2}\|w\|^2 \quad \text{subject to} \quad y_i(w^T x_i + b) - 1) \geq 0, \quad \forall i \tag{3.36}$$

such optimization problem is solved through Lagrange multiplier method.

3.5.1 Soft margin

In real-world examples, data normally contain noises. This problem can not be solved by using the previous solution because the constraint of (3.36) is not satisfied. To address this problem, a positive slack variable is introduced ξ_i, $i = 1, \ldots, N$.

$$\begin{aligned} w^T x_i + b &\geq +1 - \xi_i, \quad y_i = +1 \\ w^T x_i + b &\leq -1 + \xi_i, \quad y_i = -1 \\ \xi_i &\geq 0, \quad \forall i \end{aligned} \tag{3.37}$$

The problem of training SVM is converted to solving the below optimization problem:

$$\min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \xi_i \quad \text{subject to} \quad y_i(w^T x_i + b) - 1 + \xi_i \geq 0, \quad \xi_i \geq 0 \forall i \tag{3.38}$$

This is called 1-norm soft margin problem. Here is a regularization parameter that controls the trade-off between maximizing the margin and minimizing

the training error. Small C tends to emphasize the margin while ignoring the outliers in the training data, while large C may tend to overfit the training data.

3.5.2 Nonlinear classification

In practical applications, data is not always separatable in the input space. If the border between classes is not linear, it is not possible to use a hyperplane to separate them. The method used to overcome this issue is to switch from a non-linear problem to a linear problem by using *Kernel Functions* $x \rightarrow \phi(x)$. When applying a kernel function, the input data is projected into a new space called feature space. In this new space, the data are linear separable. It is always possible, for a finite point set, to find a dimension where all

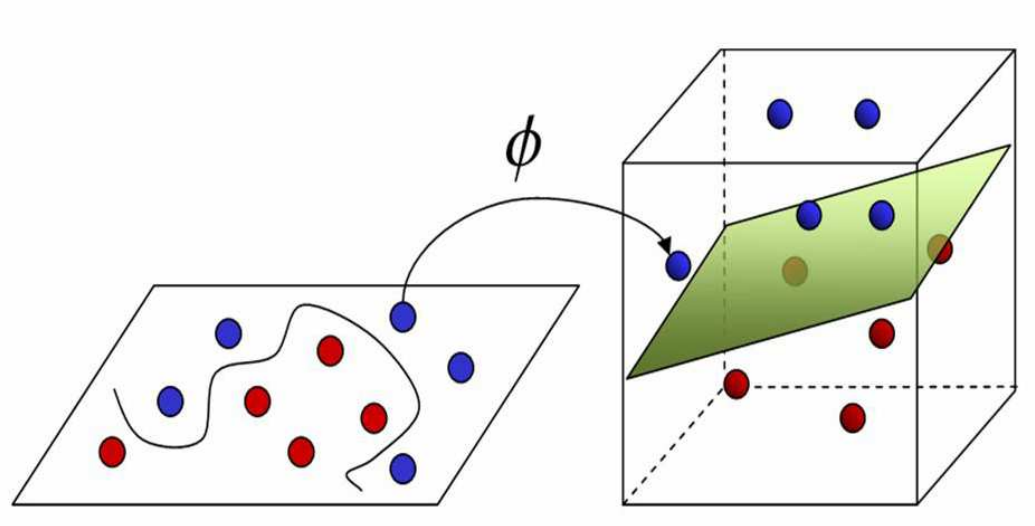

Figure 3.7: A Kernel function projects the input data into a new space where a hyperplane can be used to separate the classes

possible separations of the point set can be achieved by a hyperplane. Hence, mapping the datapoints to a space of sufficiently high dimension it is possible to use the linear SVM training algorithm described above to find the optimal separating hyperplane. If ϕ maps points from an n-dimensional space to an m-dimensional space ($m > n$), then by replacing $x_i \cdot x_j$ by $\phi(x_i) \cdot \phi(x_j)$ in all the equations of the Lagrange optimisation formulation it is possible to find the hyperplane normal vector w in $\mathbb{R}^m$.

The kernel is related to the transform $\phi(x)$ by the equation:

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j) \tag{3.39}$$

Different types of kernels can be used to achieve a robust classification; in this work, we used a radial basis function, defined as:

$$K(x_i, x_j) = exp(-\gamma \|x_i - x_j\|^2) \tag{3.40}$$

3.5.3 Parameters tuning

The effectiveness of SVM depends on the selection of kernel, the kernel's parameters, and soft margin parameter C. In the case of a radial basis function, the parameter γ must be estimated. A usual technique, implemented also in this work, is to select such parameters by a grid-search with exponentially growing sequences of C and γ, for example $C \in \{2^{-5}, 2^{-3}, \ldots, 2^{13}, 2^{15}\}$, $\gamma \in \{2^{-15}, 2^{-13}, \ldots, 2^{1}, 2^{3}\}$. Each combination of parameter choices is checked using cross validation, and the parameters with best cross-validation accuracy are picked. The final model, which is used for testing and for classifying new data, is then trained on the whole training set using the selected parameters.

Chapter 4

System architecture

This chapter presents the proposed method to reach our goals, i.e. automatic musical genre classification using a restricted set of high-level features derived by the natural clusterization of training data, supporting a real-time behaviour. This work investigated two approaches; the first sections describe their shared operations (the used datatbase and the extraction of the low-level features), while the remaining ones show in detail each approach.

4.1 Database

The database used to develop our thesis is the Ismir 2004 dataset for genre recognition. It is based on the entries of the site Magnatune[1], an independent record label whose licensing scheme allow the use of the audio content for research. The data are divided into a training set and a test set. The training set consists of:

- classical: 320 samples
- electronic: 115 samples
- jazz - blues: 26 samples
- metal - punk: 45 samples
- rock - pop: 101 samples
- world: 122 samples

[1] `http://www.magnatune.com`

The test set is composed by 700 tracks in the genres (classes) mentioned above with a similar distribution.

4.2 The overall scheme

The goal of this work is to achieve automatic genre classification using a set of high-level features as discriminative characteristics. The adoption of high-level features is desired not only to improve the system performance with respect to the classification outcome, but also to categorize musical files or streams through semantically consistent descriptors; such descriptors, representing intelligible musical characteristics, can be used for a variety of applications, e.g. implement a music reccomendation system that allows the user to specify the features they want in the suggested songs (for example highly dinamic, happy and upbeat).

Such features are defined through an example-based process by clustering the distribution of data points described by a wide set of low-level features; the resulting clusters will be related to audio segments having similar musical characteristics, therefore describing the distribution at a higher abstraction level. In order to use these clusters to define a set of high-level features, a mixture of gaussians for each one is computed by using them as input for a GMM training. In this way, a statistical model of each cluster is created.

Once the high-level feature models are computed, it is possible to proceed to the classification process, which is executed through a SVM. The first step is the training of the classifier with a new training set, for which the derived high-level features are computed for every audio segment. Being the SVM a binary classifier, the training phase will be done in a one-vs-one fashion. The second and final step is the classification of unlabeled audio segments by computing their high-level features and using them as input for the SVM prediction, which returns the most likely musical genre for the current observation. The block scheme of this process is showed in figure 4.1.

Two main approaches have been followed: the first one, following a bottom-up fashion, finds the subset of low-level features best clusterizing the GMM training set; as a consequence, the goodness of the clusterization is used as a mesaure of the optimization process. This approach aims at validating the guess that compact and well separated clusters define high-level features that provide the best classification accuracy.

The second approach, following a top-down fashion, aims at finding the subset of low-level features defining the high-level features that achieve the best final classification. As a consequence, the overall classification accuracy is used as a measure of the optimization process. This approach aims at improving the performance of the system using the proposed method.

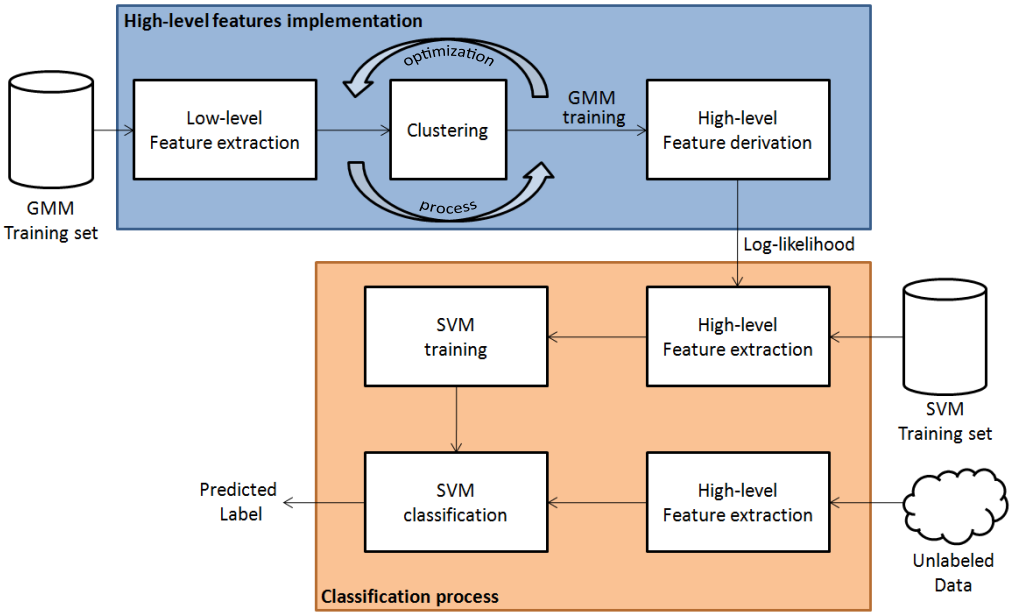

Figure 4.1: The block scheme of the overall system

4.3 High-level features definition

As said before, the adoption of high-level features has several advantages: the main one is that a higher level of abstraction means a more intelligible way to categorize music excerpts. This means that such descriptors could be used not only by experts, but also by common users. In both cases, this approach can bring many benefits; in the first case, high-level features could be used as a starting step for further levels of abstraction, or as tools to analyze musical databases. In the second case, the final user could select the wanted characteristics and their relevance in a song selection, or discover new artists and genres by starting from a set of desired musical features.

An issue regarding the high-level features usage is their implementation; there are few works in the actual state of the art going in this direction, and in all of them the derivation of semantically consistent descriptors is made through a subjective criterion: usually a set of low-level features is chosen as basis for a wanted musical characteristic. This approach has the flaw of being based on a guess, without the possibility to prove that the used combination of low-level features is effectively the best one describing that particular music characteristic.
The present work tries to overcome this problem by deriving high-level features through an example-based definition process. The method consists in describing a training dataset with a wide choice of low-level descriptors, and then finding the subset that best clusterize the data points distribution through an optimization process. Intuitively, points representing audio segments with similar high-level musical characteristics will be close in the distribution of the data set; hence, the resulting clusters could be a basis to offer an example-based definition of such characteristics.
This procedure implies a method to formally describe the clusters derived from the optimization process. For this reason GMMs have been used, as they provide a thorough statistical model for a data point distribution by using a mixture of gaussian components. Hence, a model for the derived high-level musical caracteristics is obtained by using the realtive clusters as input for the training phase of the GMM.
A new dataset can be described by the derived high-level features by extracting the low-level features best clusterizing the dataset used for the GMM training, and then computing the log-likelihood of the resulting feature vector with each gaussian mixture model describing the clusters previously found.
Following this procedure, the high-level features implementation is fully objective and derived by the natural distribution of points related to audio segments rather than by an initial guess, providing a more solid approach. In this case, though, the semantics of the found musical characteristics must be determined by listening to the audio segments associated with the points clustering togheter.

4.4 Low-level feature extraction

The aforementioned dataset was used to extract the audio features described in section 3.1. Each audio file has been segmented using its dissimilarity

matrix (a graphical illustration is shown in figure 4.2), which is convolved along its diagonal in order to obtain the novelty coefficient [28]. Applying peak detection to the curve, it is possible to get a segmentation based on the self similarity of the signal. This process is implemented in order to take the most homogeneous and invariant segments from the currently analyzed audio clip.
From such segmentation 3 seconds long windows w_i called texture windows were taken in order to obtain a set of training datapoints (D_{3s} from now on). Most of the features were computed over windows of 0.23 seconds with a 50% overlap factor, and the mean value of the overall texture window was used; the rhythmic features were instead computed on the overall length of each texture window.

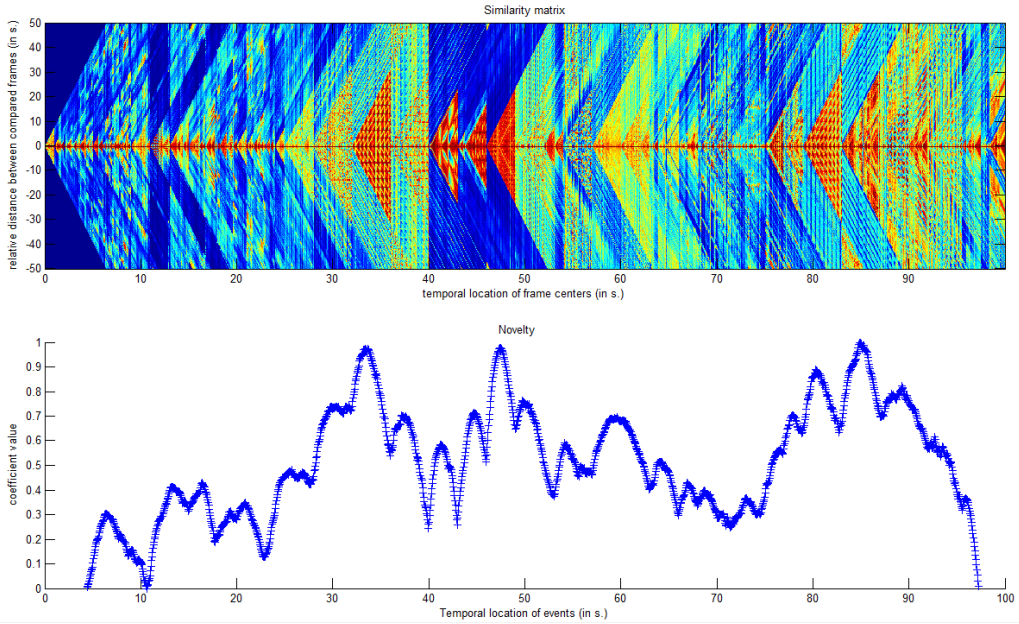

Figure 4.2: An example of (horizontal) similarity matrix and its related novelty coefficient curve

The datasets are $M \times N$ matrices, where M is the number of observations and N is the number of features. The Chroma features turned out to be heavily influential on the characterization of the low-level feature vectors; furthermore, they are highly correlated between themselvesFor this reasons, considering smaller values of the number of features to take into account for clustering they could cover most of the dimensions, leading to unbalanced

clusters characterizzation. In order to overcome this bias a further dataset was created leaving out the chroma feature vector; anyway, information about the chroma features is preserved through the introduction of the chromatic flux, which is explained in detail in section 3.1.3. Therefore a total of 2 distinct datasets are used for training:

- D_{3s}: Texture window of 3 seconds, no chroma features. $M = 7200, N = 32$
- $D_{3sCHROM}$: Texture window of 3 seconds, chroma features included. $M = 7200, N = 44$

The data were successively normalized, dividing each column for its maximum; as shown in figure 4.1, the overall procedure is composed by two separated classification steps: one used for the creation of the high-level features and one performed to achieve genre classification. Hence, to avoid the overfitting problem, the training set is split in two equal parts. The first one, D_{GMM}, will be used for the training of the GMM; the second one, D_{SVM} will instead be used for the training of the SVM.
Up to this point, the operation were common to both the approaches proposed in this work. The following sections describe each approach separately in detail.

4.5 Bottom-up approach

In the bottom-up approach, the goal is to find the subset of low-level features best clusterizing the input dataset and use the resulting clusters as high-level features. The aim of this approach is to determine if compact and well separated clusters lead to a better classification. As a consequence, this approach is characterized by a focus on the goodness of the clusterization. The optimization process is aimed at finding the subset of low-level features that best clusterize the dataset. An index of the goodness of clusterization is given by the silhouette coefficient, which is used as objective function. An illustration of the block scheme of the approach is visible in figure 4.3.

4.5.1 Clustering execution and evaluation

Finding the best clustering for a dataset is a large optimization problem. As stated in 3.2, genetic algorithms have been adopted to solve this prob-

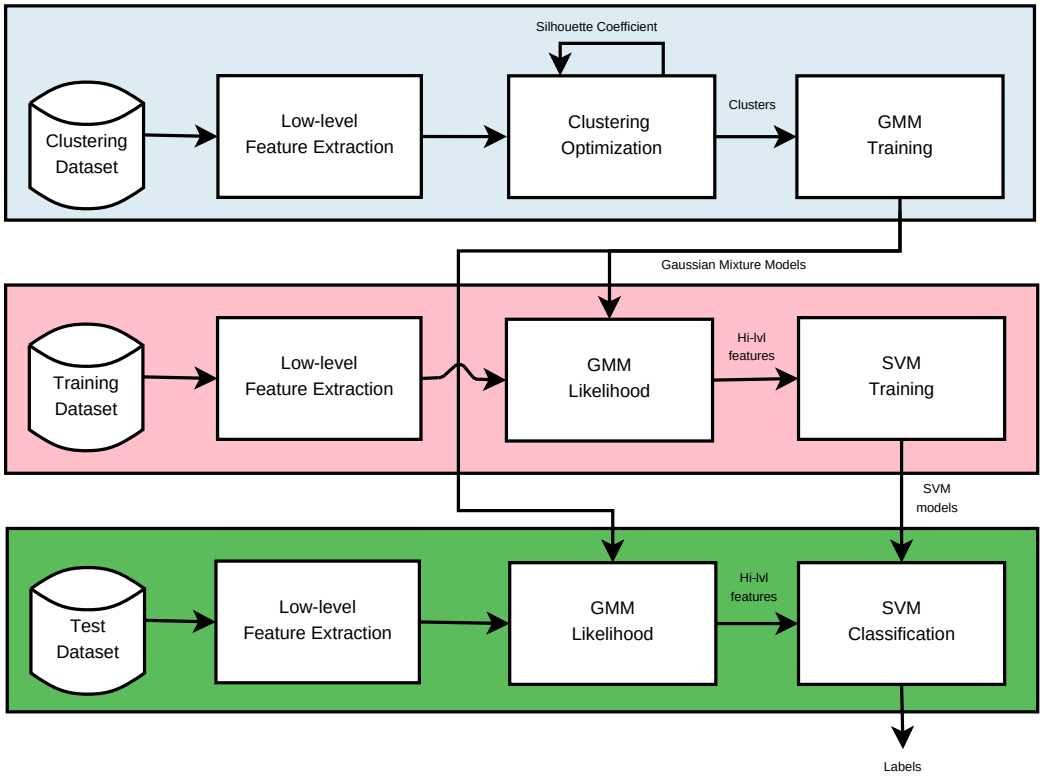

Figure 4.3: Block scheme for the bottom-up approach

lem, using a reversed, shifted version of the silhouette coefficient as fitness value. The aim of using the genetic algorithm is to find the subset of low-level features that best clusterize the GMM training dataset D_{GMM}. Every chromosome is a string of bits of length d representing a combination of the entire set of features.

Two main parameters must be taken into consideration: the number k of clusters to use and the number f of features to select as a subset of the complete d-dimensional features space. The first one is necessary since the algorithm chosen for clustering requests it as an input from the user. Other algorithms, such as the DBSCAN [29], automatically choose the best number of clusters according to the data distribution. Due to the high number of iterations to perform, though, K-means was preferred for its low computational cost, which is $O(mkt)$ with m being the number of data points, k the nuber of clusters and t the number of repetitions in order to avoid local minima.

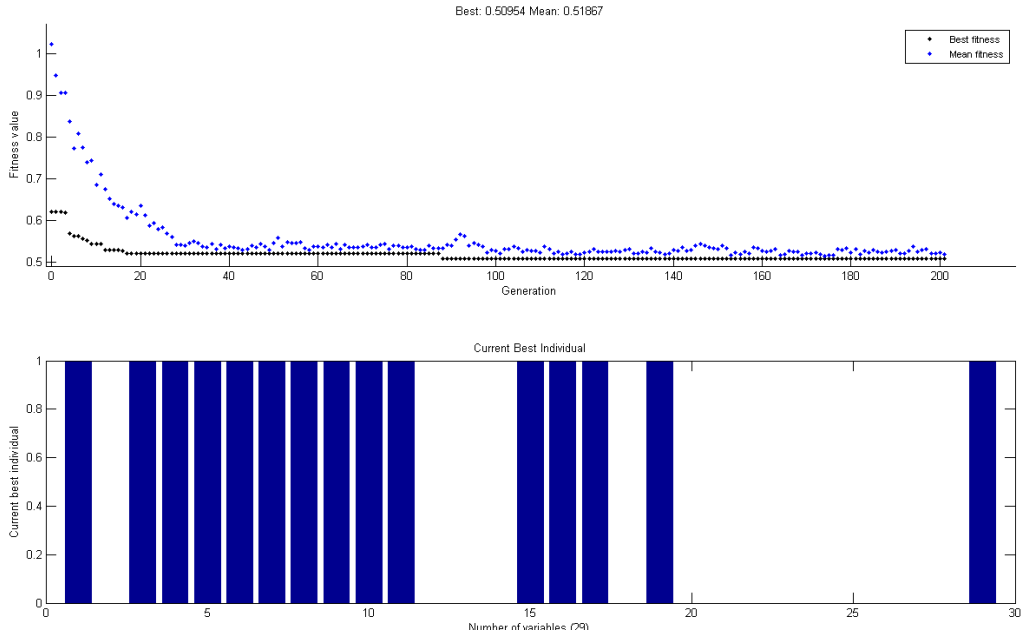

Figure 4.4: An example of clustering optimization through genetic algorithm

The second parameter is a consequence of the optimization process applied to clustering: trivially, the best clusterization for k clusters starting from d features is attained by selecting the set of features which divides the best way the data in k groups (i.e. having the lowest entropy). Adding more features results in increasing the entropy of the distribution for the current k, hence receiving a worse score from the clustering validation index. For this reason, a penalty factor p was added to the silhouette coefficient in order to contain the research among f features. the penalty factor p is computed as:

$$p = \frac{|f_c - n|}{d - n} \tag{4.1}$$

where f_c is the current number of selected features used as subset for clustering. Every set of features is therefore evaluated by the sum of the silhouette coefficient (stating how good the clusterization is) and the penalty factor p (stating if and how much the number of features used as subset is distant from the current desired f). Trivially, if the current number of features used as subset $\hat{n}$ is equal to f, the penalty factor p is equal to 0. The higher the distance between $\hat{f}$ and f is, the higher will be the value of p.
As the computational cost of the optimization problem is very cumbersome in

terms of time, a grid search approach was decided, defining a set of values for k and f to investigate. The set for the number of clusters is $K = \{2, 3, 4, 5, 6\}$. The higher bound is due to the insufficient amount of data for the GMM training in case of higher number of clusters.

The set for the number of features is $F = \{9, 12, 15, 18\}$. The lower bound is due to the features used in this work: this ensures that the features selected as subset for the clusterization do not pertain to a single characteristic of music (e.g. a selection of rhytmic features only) or to a single group of descriptors (e.g. chroma features); selecting a suitable minimum number f of features for which evaluate the clusterization guarantees the avoidance of biases in a specific direction. The higher bound and the step are necessary for the high amount of time requested for each single optimization.

Results from the grid search, furtherly discussed in chapter 5 showed that increasing the number of clusters k, the validation coefficient presented lower values, meaning a weaker clustering. This is clearly visible in figure 4.5: the 3-dimensional plot for the dataset D_{3s} tends to lower as the number of clusters increase. Similar results were obtained for the other datsets.

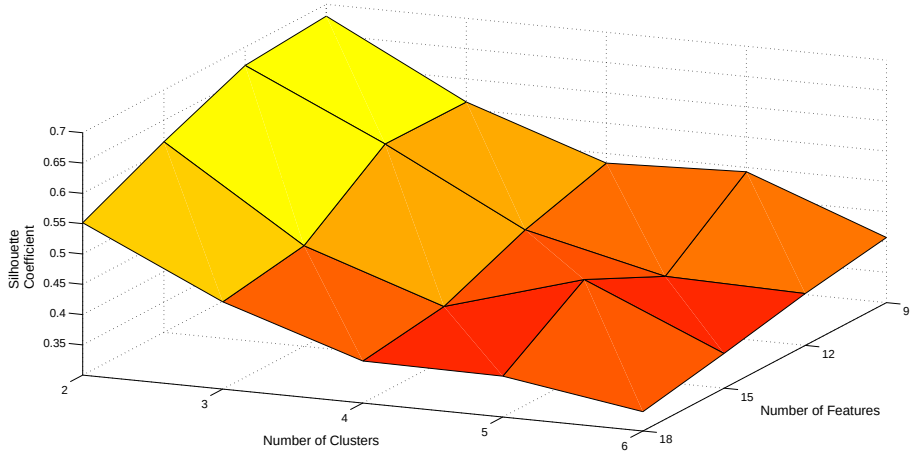

Figure 4.5: silhouette coefficient for varying numbers of clusters and features; the dataset is D_{3s}

As a result from these observations, clusterings with $k > 3$ were not explored.

4.5.2 High-level features implementation

The core of the system is the implementation of the high-level features. Once the optimization process for clustering is over, the data points are grouped in clusters according to their euclidean distance; hence groups are composed by points having similar values for the features used, forming analytically derived properties of the distribution. As an example, highly dynamic songs will be clustered togheter due to their characterization deriving from the low-level descriptors (e.g. event density, spectral flux and other dynamics-related descriptors will have reasonably high values).

Therefore, such clusters can be used to describe the distribution reaching a higher abstraction level; in order to be used as features, though, they must be modeled. For this reason, every cluster was supplied as training for a GMM using the Figueredo-Jain algorithm described in section 3.4.2. The result is the set $B = \{b_1, b_2, \ldots, b_j\}$ a mixture model b for each cluster obtained in the optimization process.

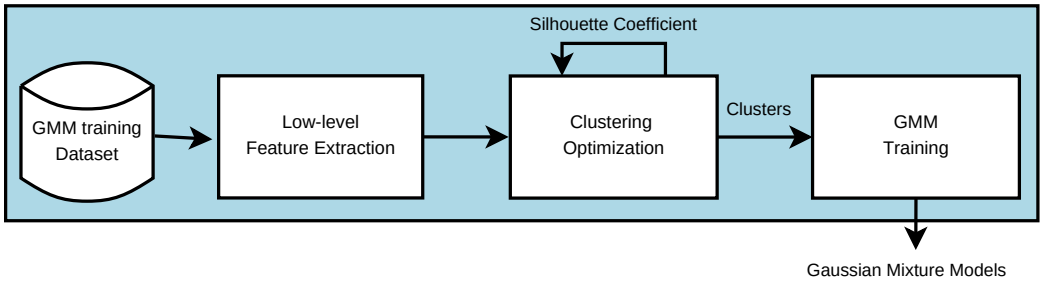

Figure 4.6: High-level feature implementation block scheme - Bottom-up approach

4.5.3 Classification through SVM

The statistical models describing the clusters obtained in the previous phase can be used to define a set of high-level features. Such definition enables the description of the SVM training dataset D_{SVM} through these new features and the training of the clssifier using them as discriminative characteristics. Once obtained the SVM models in a one-vs-one fashion, audio clips from a test dataset are classified and the resulting overall accuracy is computed.

SVM training

Once computed the high-level features models, it is possible to build a training dataset D'_{SVM}, starting from D_{SVM}. This last dataset, obtained from the feature extraction process, is composed of the whole set of low-level descriptors; hence, it is necessary to discard the features pruned out by the clustering optimization process, retaining only the subset used to obtain the final clusters.

The resulting dataset will be composed of feature vectors f_i of length f, with i being the number of observations in the dataset. These low-level feature vectors are transformed into high-level features ones by computing the log-likelihood with each gaussian mixture model $b_i \in B$. The log-likelihood is the probability of the observation outcome given the model parameters, and gives a measure of how much the current values of the low-level features represent every single high-level feature.

The outcome of this operation is a feature vector f'_i of length k. In all the configurations tried during this research, $k = 3$; in facts, clusters have always been taken as triples. Therefore, the new feature vector f'_i will be a 3-dimensional data point. The new dataset D'_{SVM} will therefore be a matrix of $M \times 3$, with M being the total number of observations.

The final step of the system is the use of a classifier. SVMs have been chosen due to their higher performance in comparison with other methods, especially using non-linear kernels on non-separable distributions. The training process, as explained in section 3.5 is done by dividing the training dataset w.r.t. its genre labels, hence having the observations separated according to their pertaining musical genre, and adopting a one-vs-one fashion.

Moreover, a grid search through cross validation is performed in order to find the best combination of parameters C and γ, associated respectively with the soft margin implementation and the usage of a radial basis function ϕ as kernel. Values investigated for C are $\{2^{-5}, 2^{-3}, \ldots, 2^{13}, 2^{15}\}$, those for γ are $\{2^{-15}, 2^{-13}, \ldots, 2^{1}, 2^{3}\}$.

This are the complete training procedre steps:

Algorithm 4.1 SVM training

1. **for all** combinations of couples of genres g_i, g_j, $i \neq j$ **do**
2. subdivide data genres D_{gi} and D_{gj} in training ($\frac{9}{10}$ of total points) and test ($\frac{1}{10}$ of total points) parts
3. **for all** all combinations of values of C and γ **do**
4. Train SVM with the training subset of the current genres
5. Use the test subset of the current genres for classification
6. **end for**
7. Retain the couple of values of C and γ achieving the best classification
8. **end for**

The outcome of the training procedure is a SVM model for each couple of genres, to be used during the classification process. This stage of the system is illustrated in figure 4.7

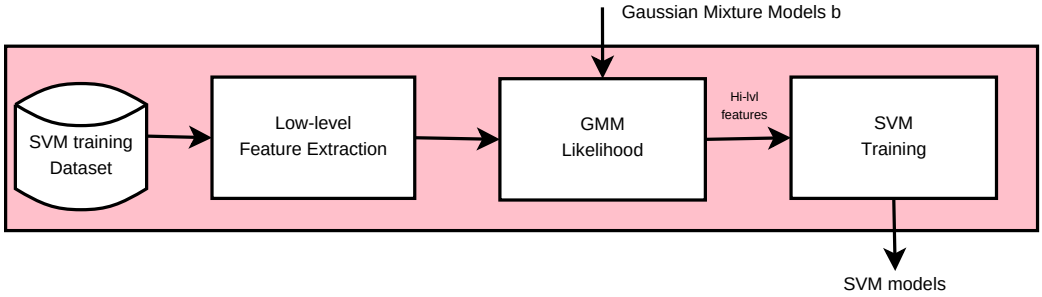

Figure 4.7: SVM training block scheme - Bottom-up approach

SVM test

The data to be classified is the test set provided by the ISMIR04 dataset. More than evaluating entire songs, as usually done for classification tasks, in this case a segment classification is performed; this decision was taken in order to resemble a real-time behaviour, where every chunk of the analyzed signal must be classified per se, without knowledge of the genre estimation of the previous samples.
The feature extraction for the test set is the same described before for the training set, hence implying the computation of the low-level features used

in the best clustering obtained in the optimization process and their normalization w.r.t. the maximum values found for the training dataset. The high-level features derivation is once again executed by applying the log-likelihood between the low-level feature vectors f and the high-level models $b_j \in B$.

Finally, the 3-dimensional high-level feature vectors f' are given as input to the SVM classifier, using the models computed in the training phase. Each model will discriminate between 2 genres and decide a winner for the current test observation. The overall winner is the genre with more wins; in case of a tie, the overall winner is the winner of the direct confrontation between the two candidates. In case of 3 or more genres with the same score, one is randomly picked.

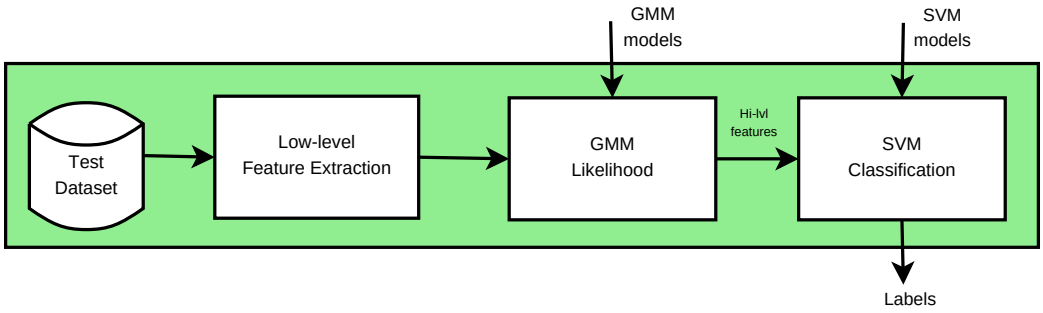

Figure 4.8: SVM test block scheme - Bottom-up approach

4.6 Top-down approach

In the top-down approach, the goal is to find the clustering leading to the high-level feature set that best classifies the test set. The block scheme of the system is visible in figure 4.9. In this approach, the aim is to find the subset of low-level features determining the clusters that best classify the audio clips extracted from the test dataset. For this reason, the error rate of the final classification is used as fitness value for the genetic algorithm optimizing the clustering.

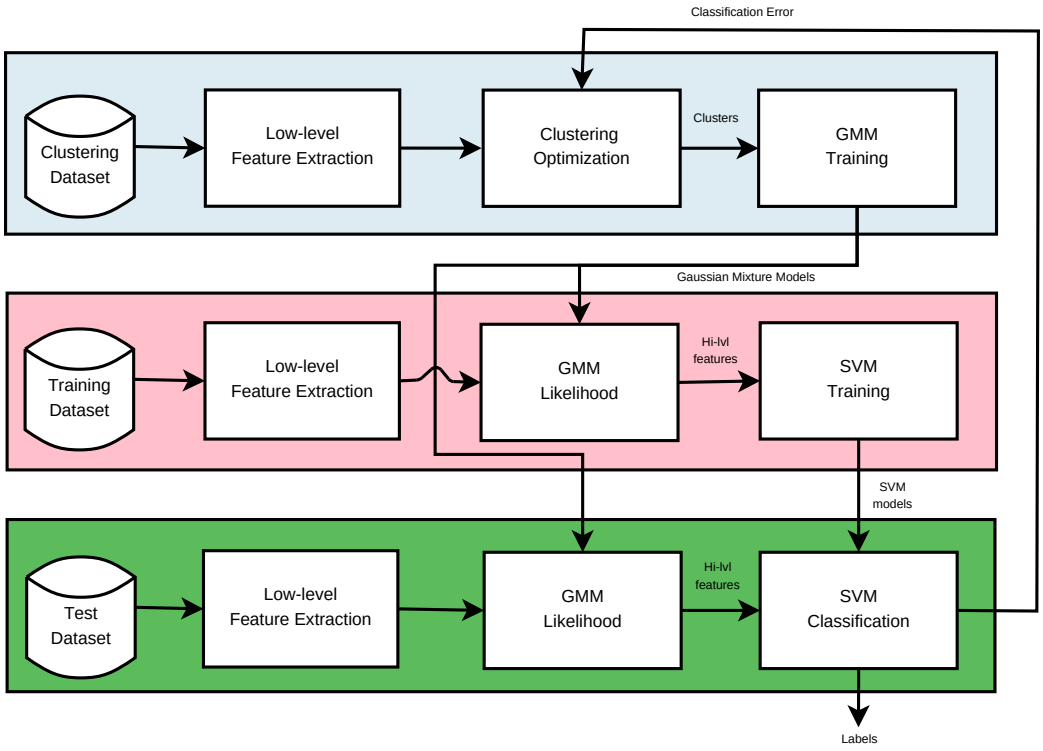

Figure 4.9: Block scheme for the top-down approach

4.6.1 Classification-driven clustering optimization

In the bottom-up approach, the aim is to find the best clustering according to its internal configuration, evaluated by means of the silhouette coefficient. In this approach, instead, the best clustering is found in relation with the classification outcome. To accomplish this goal, the fitness value of the genetic algorithm will not be a measure of the goodness of the clustering, but an index stating the overall quality of the final classification. As stated in section 3.2.2, the error rate ϵ was chosen rather than accuracy, due to the genetic algorithm optimization process which operates by minimizing the bojective function. Minimizing the error rate means increasing the accuracy, hence obtaining a better classification result.

The only tuneable parameter in this case is the number k of clusters; it is no more necessary to specify a number of features to use, since the internal configuration of the distribution is no more taken into consideration; in this approach, in facts, no clustering evaluation is performed through validation

indexes, and is therefore no more necessary to bound the number of features taken as subset to a minimum value in order to avoid a singular feature selection. Experiments have been conducted for all the datasets with values of $k = 3$ and 4. Higher values of k could not be investigated due to insufficient data for the training of the GMM.
More in detail, the procedure is the following: as in the previous approach, every chromosome composing the populations of the genetic algorithm is a string of bit of length d, representing a selection of the copmlete set of low-level features. For every combination, the clustering is performed through k-means and the resulting clusters are taken as triples. In case of $k = 4$, the 3 largest clusters are taken in order to have enough point for the subsequent GMM training step.
As in the previous case, the obtained clusters represent the data distribution from a more abstract level w.r.t. the low-level descriptors, being agglomerations of points sharing similar values. Their usage as high-level features is realized through their modeling by means of the GMM. Each cluster is separately used as input for the classifier, which returns a mixture model. This process is illustrated in figure 4.10

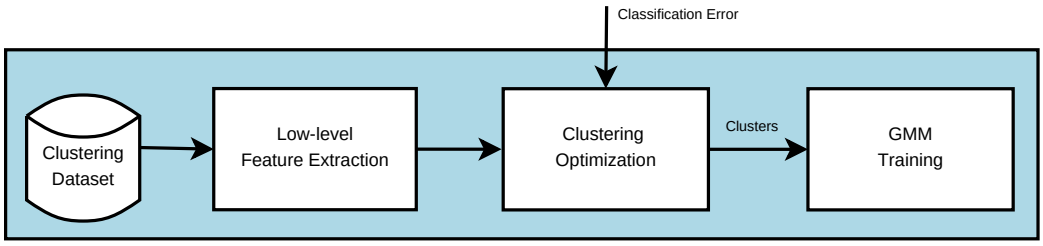

Figure 4.10: High-level features implementation block scheme - Top-down approach

4.6.2 Classification through SVM

Similarly to the other approach, the GMM models are used to implement the high-level features. The SVM training dataset D_SVM is described by such features and used as input for the SVM training. Finally, the same high-level feature extraction process is executed for the test set and genre classification is performed through one-vs-one SVM label prediction. The accuracy of the classification in this approach is used to minimize the clustering process.

SVM training

Once obtained a model for each high-level feature, for each genre the respective subset of the training dataset D_{SVM} is transformed from a set of low-level features vectors f_i to a set of high-level features ones f_i' computing the log-likelihood w.r.t. each model previously obtained. Therefore, the resulting dataset D'_{SVM} will be a 3-dimensional data distribution. Such dataset will be suplied as input for the SVM training stage. The classification step is performed in the same way as the previous approach, dividing the training dataset with respect to its genre labels, hence having the observations separated according to their pertaining musical genre, and adopting a one-vs-one fashion. The detailed procedure is explained in the algorithm 4.1 and shown in figure 4.11. At the end of such process, a SVM model will be created for each couple of genre.

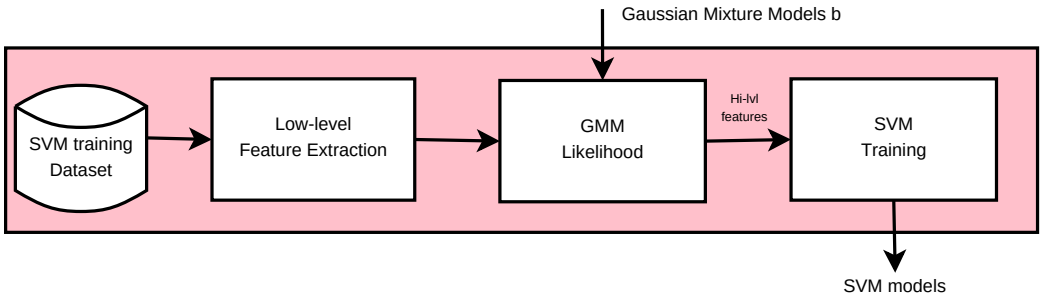

Figure 4.11: SVM training block scheme - Top-down approach

SVM test

Finally, the low-level features selected in the clustering phase are extracted from the testing dataset, taking into consideration audio segments of 3 seconds. The high level-features vector is computed for such observations through log-likelihood computation with respect to the gaussian mixture models of the k clusters previously obtained, and given as input to the SVM for genre prediction. Again, being this task executed in a one-vs-one trend, the overall winning genre is the genre with more wins among the total confrontations; in case of a tie, the overall winner is the winner of the direct confrontation between the two candidates. In case of 3 or more genres with the same score, one is randomly picked.

The error rate ϵ of the classification step is computed through the confusion matrix resulting from the classification outcome. This value is associated from the initial chromosome taken into account. Every chromosome of the current population will have its own associated ϵ. The chromosomes having the lowest ϵ are used as parents for the generation of the next population, as described in section 3.2. This process is graphically described in figure 4.12.

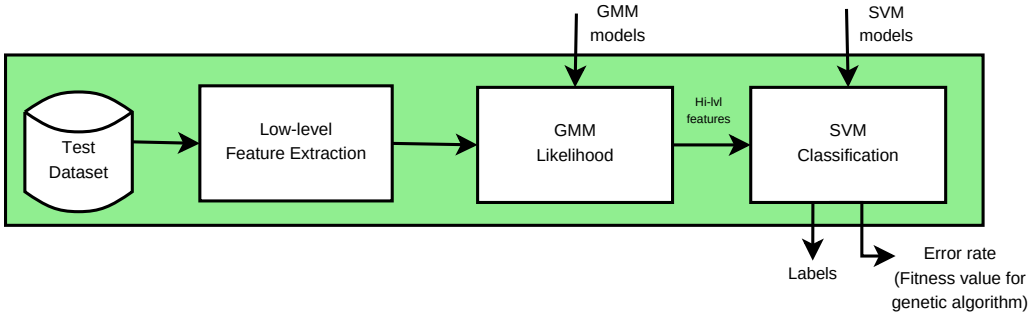

Figure 4.12: SVM test block scheme - Top-down approach

At the end of the optimization process, the chromosome having the lowest fitness value will be taken as best individual. The associated confusion matrix is the final result of the approach for the given dataset.

Chapter 5

Experimental results

In this chapter, the results of the experiments carried out in this research are showed. Clustering and classification results will be reported for the bottom-up approach, with a related dissertation. Classification results only will be reported for the top-down approach, since no evaluation has been performed over the clustering phase in this case.
For each approach, best and worst outcomes and parameters choice will be discussed, including the features subset used for clustering, the number of clusters used as input for the k-means algorithm and their related scores.

5.1 Bottom-up approach results

Concerning the bottom-up approach, the aim is to determine if compact and well separated clusters found during the optimization process of the clustering phase lead to a better classification score. Reminding that two datasets were proposed and that different numbers of clusters k and of features f were used, a total of 40 clusterizations were computed. The curve of the silhouette coefficient for varying numbers of clusters and features is reported as a result. The next step is the classification through SVM, using the clusters previously found as high-level features. Accuracy and confusion matrices are presented as result.

5.1.1 Clustering results

The clustering optimization performed by the genetic algorithm amis at finding the combination of features leading to the best clustering of the input

training test D_{GMM}. The silhouette coefficient was used as cluster validation index and given as fitness value for the genetic algorithm. Results of this process are shown in figures 5.1 and 5.2.

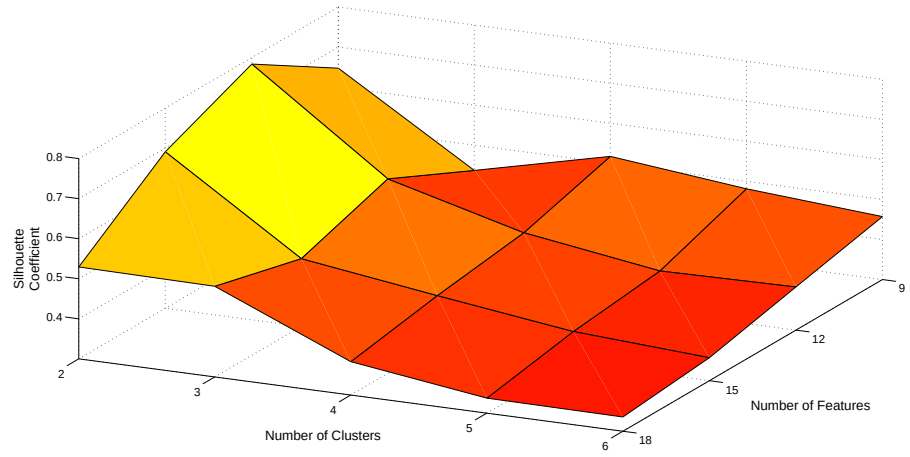

Figure 5.1: Clustering validation with varying numbers of clusters and features. Dataset $D_{3sChrom}$

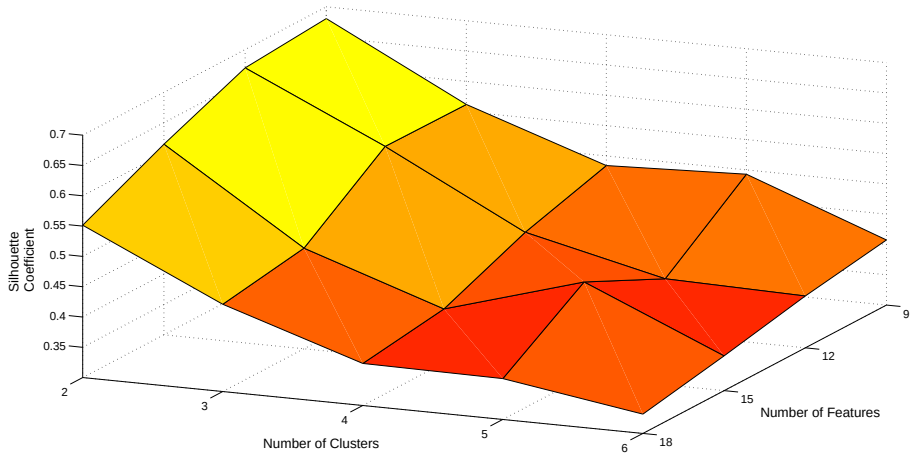

Figure 5.2: Clustering validation with varying numbers of clusters and features. Dataset D_{3s}

It is possible to see that the goodness of the clustering is heavily influenced by the number of clusters k used to subdivide the training dataset. More specifically, for growing values of k the Silhouette coefficient returns lower values, indicating a worse clustering (spread and badly separated clusters). Such result seems to indicate that the dataset is homogeneous, and that data points tend to form a chaotic distribution.
Being the aim of this approach to investigate the best clusterizations, data obtained with values of $k > 3$ have been discarded. Therefore, for each datset D_{3s} and $D_{3sCHROM}$, 8 complete classification processes were performed, with $k = 2$ and 3 and varying number of dimensions f, corresponding to low-level features.
The tables 5.1, 5.2, 5.3 and 5.4 show, for each dataset, the selected low-level features for the clustering at the end of the optimization process. Results are divided by the number f of selected features.

Table 5.1: Clustering results for subset of dimension $f = 9$

k=2		**k=3**	
D_{3s}	$D_{3sCHROM}$	D_{3s}	$D_{3sCHROM}$
RMS	zero crossing rate	zero crossing rate	spectral centroid
spectral skewness	spectral centroid	spectral centroid	spectral brightness
spectral kurtosis	spectral brightness	spectral brightness	spectral entropy
spectral flatness	spectrum spread	spectrum spread	MFCC 7
roughness	spectral rolloff	spectral entropy	Chroma C#
irregularity	spectral entropy	spectral flatness	Chroma E
spectral flux	roughness	roughness	Chroma F#
event density	MFCC 2	event density	Chroma G
pulse clarity	Chroma B#	MFCC 2	Chroma A

Looking at the low-level features used for the best clustering found, it can be seen that spectral features are the most present in the results; this is especially for the statistics-oriented ones (flux, spread, skewness and kurtosis), as well as for spectral entropy and flatness. All of these descriptors are classified as timbral; concerning the same class, MFCCs are always present with at least one component.
Rhytmic descriptors are differently taken into account: bpm estimation has never been selected for clustering, while event density and pulse clarity appear in most of the combinations. In general rhytm does not characterize the distribution of points; it must be noted, though, that the number of its

Table 5.2: Clustering results for subset of dimension $f = 12$

k=2		**k=3**	
D_{3s}	$D_{3sCHROM}$	D_{3s}	$D_{3sCHROM}$
zero crossing rate	zero crossing rate	zero crossing rate	zero crossing rate
spectral centroid	spectrum spread	spectral centroid	spectral centroid
spectral brightness	roughness	spectral brightness	spectrum spread
spectrum spread	irregularity	spectrum spread	spectral rolloff
spectral kurtosis	spectral flux	spectral rolloff	spectral entropy
spectral rolloff	MFCC 1	spectral entropy	spectral flatness
spectral entropy	pulse clarity	spectral flatness	roughness
spectral flatness	Chroma C	roughness	spectral flux
roughness	Chroma C#	irregularity	MFCC 2
spectral flux	Chroma D	spectral flux	MFCC 10
MFCC 1	Chroma F	MFCC 1	pulse clarity
MFCC 2	Chroma A	MFCC 2	Chroma F

Table 5.3: Clustering results for subset of dimension $f = 15$

k=2		**k=3**	
D_{3s}	$D_{3sCHROM}$	D_{3s}	$D_{3sCHROM}$
zero crossing rate	zero crossing rate	RMS	spectral brightness
RMS	spectral centroid	spectral centroid	spectral rolloff
spectral centroid	spectrum spread	spectral brightness	spectral flatness
spectral brightness	spectral entropy	spectrum spread	spectral flux
spectrum spread	roughness	spectral skewness	event density
spectral kurtosis	irregularity	spectral kurtosis	MFCC 13
spectral rolloff	spectral flux	spectral rolloff	pulse clarity
spectral entropy	MFCC 1	spectral entropy	Chroma C
spectral flatness	MFCC 12	spectral flatness	Chroma D
roughness	pulse clarity	roughness	Chroma D#
irregularity	Chroma C#	spectral flux	Chroma F#
spectral flux	Chroma D	event density	Chroma G
MFCC 1	Chroma F	MFCC 2	Chroma A
MFCC 2	Chroma F#	MFCC 5	Chroma B
pulse clarity	Chroma A	pulse clarity	Chroma B#

related features is fairly inferior with respect to the other categories. Finally, harmonic descriptors are rarely present in these results, exception made for the chroma features. Their presence tend to increase as f increases; they are usually selected in a close range of pitch classes, e.g. in table 5.4 8

Table 5.4: Clustering results for subset of dimension $f = 18$

k=2		**k=3**	
D_{3s}	$D_{3sCHROM}$	D_{3s}	$D_{3sCHROM}$
zero crossing rate	zero crossing rate	zero crossing rate	zero crossing rate
RMS	spectral centroid	RMS	spectral centroid
spectral centroid	spectral brightness	spectral centroid	spectral brightness
spectral brightness	spectral kurtosis	spectrum spread	spectrum spread
spectrum spread	spectral rolloff	spectral skewness	spectral rolloff
spectral kurtosis	spectral entropy	spectral kurtosis	spectral entropy
spectral rolloff	spectral flatness	spectral rolloff	spectral flatness
spectral entropy	spectral flux	spectral entropy	roughness
spectral flatness	event density	spectral flatness	spectral flux
roughness	MFCC 2	roughness	MFCC 2
irregularity	Chroma C	irregularity	pulse clarity
spectral flux	Chroma C#	spectral flux	Chroma C
event density	Chroma D#	event density	Chroma C#
MFCC 2	Chroma E	HCDF	Chroma D
MFCC 12	Chroma F#	MFCC 1	Chroma D#
MFCC 13	Chroma G#	MFCC 2	Chroma A
Chromatic flux	Chroma B	Chromatic flux	Chroma B
pulse clarity	Chroma B#	pulse clarity	Chroma B#

chroma features are present for the dataset $D_{3sCHROM}$, ranging contigously from C to B#.
The prevalence of timbric descriptors could be explained by the analysis procedure, wich is focused on the single segments more than on a whole song or clip. Hence, an harmonic or rhythmic description is not as accurate as the timbre definition for short time periods (3 seconds in this case). The number of descriptors in this category, fairly superior with respect to the others, is another detemrinant factor.

5.1.2 Classification results

As stated before, the clusters resulting from the best clusterizations have been modeled through a mixture of gaussians and used as high-level features for the classification step, which was performed with a SVM. The most important result for this approach is to confront the silhouette coefficient value of a clustering with the accuracy of its related classification, i.e. the classification carried out by using its clusters as high-level features.

Table 5.5: Evaluation of clusterings and their related classifications

# Clusters	# features	Dataset	Silhouette	Accuracy
2	9	3s Chrom	0,647	0,368
		3s	0,681	0,264
	12	3s Chrom	0,784	0,388
		3s	0,670	0,384
	15	3s Chrom	0,691	0,382
		3s	0,614	0,277
	18	3s Chrom	0,529	0,394
		3s	0,551	0,303
3	9	3s Chrom	0,437	0,416
		3s	0,562	0,398
	12	3s Chrom	0,542	0,392
		3s	0,563	0,402
	15	3s Chrom	0,469	0,370
		3s	0,466	0,318
	18	3s Chrom	0,527	0,429
		3s	0,444	0,323

Table 5.5 relates these values. Such results indicate that the starting guess behind the bottom-up approach is not verified: higher values for the silhouette coefficient do not correspond to higher accuracy scores in the classification phase. This means that using compact and well separated clusters for the high-level features definition does not lead to a set of strongly discriminating descriptors.

Hence, optimizing the clustering step is not useful for the performance of the classification step. The same comparison for the top-down approach will confirm this result.

Table 5.5 shows a second important result to be analyzed, i.e. the accuracy of the classification step using this approach. The achieved values are modest if compared with those found in literature, ranging from 26,4% to 40,2% for the D_{3s} dataset, and from 36,8% to 42,9% for the $D_{3sCHROM}$. There is a noticeable improvement of the classification including the chroma features, indicating their suitability for this task. A lower accuracy for this approach with respect to the top-down one was expected, as it is not directly focused

on the final outcome of the system, but on a possibly related issue.
The last results for the bottom-up approach are the single per-genre accuracies. In order to display such information, confusion matrices are used. Each confusion matrix has the genres g to be recognized on its rows and columns. Cells g_{ij} for which $i = j$ contain the correctly classified entries, while those for which $i \neq j$ contain the entries of genre i misclassified for genre j. These results will be provided for the combination of k and f obtaining the best accuracy values. Such values are:

- for the D_{3s} dataset:
  - $k = 2,\ f = 12$
  - $k = 3,\ f = 18$
- for the $D_{3sCHROM}$ dataset:
  - $k = 2,\ f = 12$
  - $k = 3,\ f = 18$

The related tables are 5.6, 5.7, 5.8 and 5.9

Table 5.6: Confusion matrix for dataset D_{3s}, $k = 2$, $f = 12$

	Classical	Electronic	Jazz - Blues	Metal - punk	Pop - rock	World
Classical	**60**	4	23,5	4	0	8,5
Electronic	11,5	**34**	29,5	17	3	5
Jazz - Blues	11	15	**60,5**	2,5	8	3
Metal - punk	0,5	18,5	12,5	**61**	7	0,5
Pop - rock	6,5	20,5	34	24	**11,5**	3,5
World	34	13,5	35,5	10	3,5	**3,5**

Table 5.7: Confusion matrix for dataset $D_{3sCHROM}$, $k = 2$, $f = 18$

	Classical	Electronic	Jazz - Blues	Metal - punk	Pop - rock	World
Classical	**55,5**	9	30,5	0	0,5	4,5
Electronic	15	**45**	21	6,5	2	10,5
Jazz - Blues	18,5	25	**51,5**	2	2	1
Metal - punk	1	13,5	7	**71**	7	0,5
Pop - rock	8,5	21,5	31	32	**5**	2
World	34,5	24,5	26,5	4	2	**8,5**

Table 5.8: Confusion matrix for dataset D_{3s}, $k = 3$, $f = 12$

	Classical	Electronic	Jazz - Blues	Metal - punk	Pop - rock	World
Classical	**67**	5,5	15,5	0	0	12
Electronic	12,5	**32,5**	16,5	8,5	10,5	19,5
Jazz - Blues	11,5	18,5	**35**	8	16	11
Metal - punk	1	7	11	**69,5**	7	4,5
Pop - rock	8,5	19,5	15	32	**20**	5
World	36,5	16,5	17	7,5	5,5	**17**

Table 5.9: Confusion matrix for dataset $D_{3sCHROM}$, $k = 3$, $f = 18$

	Classical	Electronic	Jazz - Blues	Metal - punk	Pop - rock	World
Classical	**96,5**	0	0	0	0	3,5
Electronic	51	**15,5**	23,5	3	2	5
Jazz - Blues	17,5	6	**76**	0	0,5	0
Metal - punk	21	3	11	**60**	4,5	0,5
Pop - rock	51,5	6	21	16,5	**5**	0
World	59,5	5	30	1	0	**4,5**

The confusion matrices show that the accuracy for each specific genre is largely heterogeneous. Classical music is the genre achieving the best overall result: 96,5% of its test audio excerpts were correctly detected, using the $D_{3sCHROM}$ dataset, $k = 3$ and $f = 18$. The worst result is instead 55%, which is however above the average score. In general, classical music is the best recognized genre, with an average accuracy of 69,75%.
Good results are obtained also for the metal-punk genre; ranging from 60% to 71% accuracy on the total test observation provided, it is the second class in the average accuracy ranking. Jazz-blues is instead the third one, achieving a maximum score of 76%, followed by electronic music; this class is not often recognized, achieving a maximum accuracy of 45%.
The worst performing genres are pop-rock and world, having similar results. There is a great gap between these two classes and the previously analized ones, being their average score well below the total average accuracy. This is probably due to the broad musical categories they represent; 'world' music is a culturally derived label, not related to a specific genre but to the geographical provenience of the artist. Such category may include chinese opera works and latin american songs, without distinguishing their playing styles, the instruments used in the excution, etc.
Pop-rock is also a very wide definition, which may range from synth-pop (in which electronics is heavily used) to rock and roll (which is definitely near

to blues, often using the same structure and insturments). Such discussion over these two classes can be confirmed by the top-down approach results, which share with this one the low scores of world and pop-rock with respect to the general average accuracy.

5.2 Top-down approach

As stated before, in the top down-approach is to drive the clustering optimization process to the best final classification, by using the accuracy as objective function. In this case, the clustering goodness is not taken into account and no analysis is made on the distribution of data points during the high-level features definition process. Hence, the only relevant results reported here are the classification outcomes. Consequently, as explained in section 4.6.1, there is no need to specify the parameter f. The number of clusters, though, is still a tuneable parameter affecting the performance of the system. Therefore, for every dataset 3 experiments were performed, with $k = 3$, 4 and 5 clusters respectively.

5.2.1 Classification results

It is useful to report, togheter with the overall accuracy obtained for each experiment, the related silhouette coefficient. Note that such value is computed at the end of the entire process and only for the clustering leading to the best classification score. As stated before, there is no clustering validation in the optimization process.
Table 5.10 shows the aforementioned results.
As a confirmation of the considerations on the previous section, the accuracy seems to be totaly unrelated to the silhouette coefficient of the clustering. This validates the idea that the initial guess for the bottom-up approach may be rejected.
The second important result is the fairly improved value of the scores: there is an average increase of aproximately 12% in the overall accuracy. This increment is given by the structuring of the top-down approach, which is directly focused on the outcome of the classification. The values obtained show that the high-level features defined by an example-based process effectively have a discriminative power on the training dataset D_{SVM} of the SVM. This

Table 5.10: Silhouette coefficient and average accuracy for the top-down experiments

# Clusters	Dataset	Silhouette	Accuracy
2	3schrom	0,354436	0,54
	3s	0,377664	0,486667
3	3schrom	0,183423	0,5675
	3s	0,281212	0,505833
4	3schrom	0,224253	0,540833
	3s	0,242222	0,5025

is especially true for the majority of genres included in this work, as showed in table 5.11 - 5.16.
Another consideration regards the number of clusters k used for clustering: the best accuracy values are obtained for $k = 3$; note anyway that the classification for $k = 4$ was performed by selecting the 3 largest clusters of the 4 found for each individual analyzed by the genetic algorithm. This choice was forced by the relatively small database used (e.g. the pop-rock category featured only 26 songs); the number of data points for each cluster was too low to perform a training of a GMM in the case of $k = 4$, for most individuals evaluated by the genetic algorithm.
Tables 5.11 - 5.16 show the confusion matrices for the experiments proposed for the top-down approach.

Table 5.11: Confusion matrix for $k = 2$, dataset $D_{3sCHROM}$

	Classical	Electronic	Jazz - Blues	Metal - punk	Pop - rock	World
Classical	**91,5**	0	2,5	0	0	6
Electronic	7	**62**	17,5	1,5	6	6
Jazz - Blues	8,5	11	**75,5**	0	2	3
Metal - punk	1	13	8	**62**	15,5	0,5
Pop - rock	9	30	23,5	13,5	**20,5**	3,5
World	44,5	14	27,5	0	1,5	**12,5**

Results for the single genres resemble those found for the bottom-up approach, but with an increased accuracy: classical music is still the most recognized class, ranging from 71% to 91.5%, with an average value of 77%. Notice that for the dataset $D_{3sCHROM}$ this genre always get better scores, meaning that adding information about the harmony is useful to discriminate

Table 5.12: Confusion matrix for $k = 2$, dataset D_{3s}

	Classical	Electronic	Jazz - Blues	Metal - punk	Pop - rock	World
Classical	**74**	6,5	19	0	0	0,5
Electronic	3,5	**56,5**	20,5	5,5	13	1
Jazz - Blues	10	12,5	**63**	0,5	11,5	2,5
Metal - punk	0	15,5	7	**59,5**	17,5	0,5
Pop - rock	3,5	23	21	16	**35**	1,5
World	33,5	24,5	28	1	9	**4**

Table 5.13: Confusion matrix for $k = 3$, dataset $D_{3sCHROM}$

	Classical	Electronic	Jazz - Blues	Metal - punk	Pop - rock	World
Classical	**79,5**	0,5	7,5	0,5	0	12
Electronic	7	**51,5**	10,5	7	9,5	14,5
Jazz - Blues	6,5	3	**78**	1	2	9,5
Metal - punk	1	9	0	**72**	12,5	5,5
Pop - rock	5,5	18,5	12	28	**24,5**	11,5
World	34	12	12,5	1,5	5	**35**

Table 5.14: Confusion matrix for $k = 3$, dataset D_{3s}

	Classical	Electronic	Jazz - Blues	Metal - punk	Pop - rock	World
Classical	**74,5**	10,5	3	0	0,5	11,5
Electronic	6,5	**66**	4	5	4,5	14
Jazz - Blues	16,5	17	**44,5**	10	3,5	8,5
Metal - punk	0	10,5	0	**79,5**	6	4
Pop - rock	4,5	21,5	5,5	38,5	**12**	18
World	35	27,5	5	2	3,5	**27**

Table 5.15: Confusion matrix for $k = 4$, dataset $D_{3sCHROM}$

	Classical	Electronic	Jazz - Blues	Metal - punk	Pop - rock	World
Classical	**74,5**	0,5	10	0	0	15
Electronic	6	**44,5**	19,5	9	8	13
Jazz - Blues	5,5	3,5	**83,5**	0	3	4,5
Metal - punk	0,5	2,5	1	**73**	18	5
Pop - rock	5,5	16	16	26	**30**	6,5
World	42	10	18	0,5	10,5	**19**

Table 5.16: Confusion matrix for $k = 4$, dataset D_{3s}

	Classical	Electronic	Jazz - Blues	Metal - punk	Pop - rock	World
Classical	**71**	4,5	20	0,5	0,5	3,5
Electronic	4,5	**69,5**	12	6,5	4,5	3
Jazz - Blues	6	10,5	**57**	10	11,5	5
Metal - punk	0,5	10	8	**77**	4	0,5
Pop - rock	4	20	14	41	**17**	4
World	33,5	22	21	4,5	9	**10**

this class.
The second best recognized genre is metal-punk, which ranges from 59.5% to 79.5% with an average score of approximately 70%. The high performance when considering this class is probably due to its characteristic loud, distorted and saturated sound, which leads to a recognizable spectrum distribution.
Jazz-blues is also generally well recognized, even if in the worst case (44,5%), the score is below the average accuracy. Electronic music has similar scores, although lower in some cases (especially for the best case, which is 69.5% against the 83.5% of jazz-blues).
As found in the previous approach, pop-rock and world have the worst accuracy values, far below the average. Compared with the bottom-up results, though, these ones show an improvement of roughly 10%. The considerations made for these two genres in the previous section are confirmed by such scores.

5.2.2 Classification without the *world* genre

As noted before, the definition of the world class is ambiguous and too broad. Indeed, it is based on the geographical provenience of the artist rather than on musical features; due to its definition it contains a variety of different genres, from japanese traditional music to flamenco. Such genres are obviously strongly unrelated: there is no similarity at all in the choice of instruments, playing style, composition, harmonical structure, or any other musical attribute.
This issue has been noted in other works, e.g. in [30], encouraging the test of the system presented in this thesis with a reduced database, where the world genre has been excluded. Such test has been performed for the top-down approach only, and using the combination of dataset and number of clusters k returning the best overall accuracy in the previous cases, i.e. dataset

$D_{3sCHROM}$ and $k = 3$.
The test with a reduced set of genres was performed in this case only, in order to preserve the choerence with respect to the database choice and obtain comparable results with the literature. A future development of this work could investigate more cases and collect more results.
The experiment resulted in a highly increased overall accuracy (67.5%), with an improvement of 11% with respect to the best case obtained with the entire genre set. The resulting confusion matrix is showed in table 5.17:

Table 5.17: Confusion matrix for the reduced genre set

	Classical	**Electronic**	**Jazz - Blues**	**Metal - punk**	**Pop - rock**
Classical	**95,5**	4	0,5	0	0
Electronic	15,5	**56,5**	12	4,5	11,5
Jazz - Blues	15	4	**75,5**	0,5	5
Metal - punk	4,5	6,5	1	**71,5**	16,5
Pop - rock	16,5	13	9,5	22,5	**38,5**

The genre ranking according to their accuracy values is the same found in the previous results, with the classical music being recognized in the 95,5% of the cases. The worst score is obtained for the pop-rock genre, which is anyway furtherly increased with respect to its accuracy values found in the complete genres set experiments.

5.3 Semantics definition

The previous section showed that the proposed system successfully derived high-level features having a discriminant factor in a classification process; an accuracy of 67.5% was infact achieved using only 3 descriptors. Now it is important to understand if these features represent a well defined musical characteristics, and which ones in case of a positive feedback.
In order to accomplish this task, the human internvention is necessary; the only way to find and define a semantics for these features is to listen to the audio segments associated with the clusters used for the feature derivation. Such procedure has been carried out on the combinations of clusters returning the best classification results in the top-down approach, i.e. $k = 3$, for each dataset.
Listening to the entire set of audio segments realted to each cluster would be confusing, since very long audio files should be listened and even the

points laying on the borders of the clusters would be included. In order to perform a more efficient and precise feature labeling, the most relevant 20 audio segments per cluster were listened, i.e. the ones related to the 20 closest data points with respect to the cluster centroid. The listening process presented very clear results, with similar outcomes for the two evaluated clusterings: each one of the 3 features can be strongly associated with a particular musical characteristic.

In both cases, the first cluster presented a high percentage of classical and world music. The principal attribute of the its audio clip is the absence of electric or electronic instruments: the wast majority of segments presented strings instruments, i.e. guitars, violins, violoncellos and oriental music instruments such as the sitar. A minor set of samples featured piano and wind instruments. Apparently, there was no other link between the excerpts. A musical characteristic that could be associated with the high-level feature could therefore be the type of used instruments, ranging from acoustic to distorted or electronic. The proposed name for such feature could hence be **classicity**.

The audio clip of the second cluster presented a higher variety of musical genres, from jazz to electronic. The type of instruments adopted were also heterogeneous, from guitars (acoustic and electric) to synthetizers. In this case this kind of information is therefore unrelevant. A common factor in the audio segments was instead the drum pattern which was very often much articulated. Syncopated rhythms were frequently found, determining a "catchy" style of playing. This feature could hence be a measure of the rhythmic emphasis of an excerpt. A possible name for the feature is **groove**. Again, the two clips obtained from both the adopted datasets resulted to be very similar.

Finally, the third cluster presented a majority of rock, hard rock, metal and punk songs. All the audio clips were upbeat and characterized by electric guitars riffs and loud drums, sometimes used as background for the singer. Hence, this feature could represent the level of vivacity and aggressiveness of a music piece. A descriptive name for the feature could be **roughness**.
The characteristics of the audio clips, the discriminant factors of the high-level features and the proposed names for each one are resumed in table 5.18.
Due to time reasons, the listening test has been executed only by the author of this thesis. As discussed in section future, a future development will provide

Table 5.18: High-level features semantics

Audio characteristics	Discriminant factor	Name
Strong presence of classical and ethnic music Acoustic instruments only, mainly strings Little or no rhythmic section	Usage of unplugged instruments	Classicity
Strong rhythmic characterization Catchy drum patterns	Rhythmic emphasis	Groove
Heavy use of distorted guitars Loud playing style	Vivacity and aggressiveness	Roughness

a more objective definition of the features semantics through a validation performed by several people.
The results, though, should not be far from this first observation, since the high-level descriptors found are strongly characterized.

Chapter 6

Conclusions and future work

In this work a novel method for real time automatic music genre classification and tracking has been presented, based on the implementation and usage of clustering-driven high-level features. Unlike most of the research adopting macro descriptors, in this system the definition of such descriptors is derived through the natural clustering of musical audio segments described by a set of low-level features. This method implies an example-based definition of the high-level descriptors, overcoming the issue of ambiguity in their implementation, which is often raised for other works following this direction.

Using high-level features in content-based systems can open new scenarios for music analysis and fruition: tracking the evolution of the musical characteristics of a song or a stream means having a continuous description, which is not achievable with the context based systems used today. In the case of genre recognition, this could be useful to follow the stream of a web-radio and label it in real time, or to graphically visualize the position of the current song with respect to the genre space.

In order to reach this goal, two different approaches were adopted: the first, following a bottom-up fashion, aims to determine if clusters resulting from a good clustering (i.e. having compact and well separated clusters) form high-level features with a strong discriminative power in the classification process.

Such approach was implemented by using an optimization process to find the combination of low-level features returning the best clustering. Genetic algorithms were used for this task, adopting the silhouette coefficient as cluster validation index. Various combinations of number of clusters and number of low-level features were tested in this phase to find the best clustering. The

clusters resulting from the optimization process have been then modeled through a mixture of gaussians, using a GMM. Finally, a standard classification procedure was performed, using the high-level features previously found to describe a test set and a SVM classifier.
In the second approach, following a top-down fashion, the aim is to find the high-level features that lead to the highest possible classification performance. In this case, the clustering optimization process was therefore directly driven by the goodness of the final classification outcome. Each individual evaluated by the genetic algorithm is a subset of low-level features used for clustering the training dataset.
The resulting clusters are modeled through GMM as in the previous approach, and a standard classification process is performed using the high-level features previously found to describe a test set and a SVM classifier. The rsulting error rate is used as fitness value of the genetic algorithm for the current individidual. Differently from the first approach, the only parameter tuned is the number of cluster in which subdivide the training data distribution.
The results showed that the bottom-up approach did not perform as well as the other one, indicating that compact, well separated clusters do not necessarily form highly discriminative high-level features, i.e. determining such features by optimizing the clustering with respect to its internal configuration does not imply a better accuracy in the classification task.
This idea seems to be confirmed by the outcomes of the top-down approach, which instead has proven to be more efficient. The clustering optimization process driven by the overall classification accuracy returned promising results, especially for a number $k = 3$ of clusters. Single genre scores strongly vary: classical music was the most recognized genre, with a highest accuracy of 96%; pop-rock and world were instead the most misclassified ones.
Since the world music genre was considered too broad and ambiguous, a test excluding it from the classes has been performed. The resulting scores are significantly higher, achieving a 67,5% average accuracy. This value is comparable with related works in the literature.
Concerning the semantic definition of the implemented high-level features, a preliminary evaluation has been developed: a significative sample of each cluster obtained in the top-down approach has been associated with its respective audio segment, and their combination has been listened in order to find relevant musical characteristics. It has been noted that the high-level

features indeed have a well defined semantics; combinations for $k = 3$ were evaluated, finding that the 3 related high level features could be respectively associated to the degree of classicity of the played instruments, the groove of the rhytmic section and the roughness/vivacity of the mood.

6.1 Future works

This thesis explores a still largely uncovered direction in literature. Therefore many improvements can be proposed, touching different sides of the research. A first important future work would be a more objective definition of the semantics related to the high-level descriptors found by the system. This task could be performed by a psychoacoustical experiment. Given a set of audio files representatives of the high-level features, a listening session is prepared. Every participant is invited to listen to the audio excerpts, and for each one choose an adjective describing it from a list of possible choices; every abjective is a musical characteristic that could possibly be associated with the features. In this way, a more objective validation of the semantics of the descriptors could be found.
A second limitation of this work which could be solved in a future development is the exploration of clustering configurations with a higher number of clusters. There is actually a higher bound for this parameter, given by the low quantity of available data provided by the ISMIR04 database. Clustering the dataset in more than 4 clusters was not possible due to the insufficient data points given as training to the GMM.
The ISMIR04 database was chosen as it is one of the few adoptable sets of audio files freely usable for research purposes. Anyway, it should be useful to find a larger and updated database, in order to enable the exploration of new configurations and to compare the resulting work with the current state of the art. Note that this choice would also solve the problem of ambiguously defined musical genres present in the ISMIR04 database.
A third proposal is to implement a visualization system capable of displaying the evolution of the high-level features in real time. Such system could track the evolution of the musical characteristics of an audio stream, visualizing interesting information about the current position of the stream in the features space with respect to the genres cdntroids. As an example, a musical excerpt of jazz rock would display values floating in the space between the jazz and rock centroids, returning a continuous classification, not vinculated

by discrete class taxonomies.
The dimensionality of the visualization system is obviously bound by the number of the high-level features used in the classification process. Since 3 descriptors seem to have good discriminative power, a 3-dimensional space could be used.

Bibliography

[1] Franois Pachet Daniel and Daniel Cazaly. A taxonomy of musical genres. In *In Proc. Content-Based Multimedia Information Access (RIAO)*, 2000.

[2] S. Lippens, JP Martens, and T. De Mulder. A comparison of human and automatic musical genre classification. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, volume 4. IEEE, 2004.

[3] MPEG-7. Information technology - multimedia content description interface. part 4 audio, 2002.

[4] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Trans. on Speech and Audio Processing*, 10(5):293–302, 2002.

[5] P. Ahrendt, A. Meng, and J. Larsen. Decision time horizon for music genre classification using short time features. In *Proc. of EUSIPCO*, volume 14, pages 1040–1043. Citeseer, 2004.

[6] M.A. Bartsch and G.H. Wakefield. To catch a chorus: Using chroma-based representations for audio thumbnailing. *Ann Arbor*, 1001:48109–2110.

[7] C. Perez-Sancho, D. Rizo, J.M. Inesta, P.J.P. de Leon, S. Kersten, and R. Ramirez. Genre classification of music by tonal harmony. *Intelligent Data Analysis*, 14(5):533–545, 2010.

[8] R. Tao, Z. Li, Y. Ji, and EM Bakker. Music Genre Classification Using Temporal Information and Support Vector Machine. ASCI Conference, 2010.

[9] F. Gouyon, S. Dixon, E. Pampalk, and G. Widmer. Evaluating rhythmic descriptors for musical genre classification. In *Proc. 25th International AES Conference*, pages 196–204. Citeseer, 2004.

[10] C.H.L. Costa, J.D. Valle Jr, and A.L. Koerich. Automatic classification of audio data. In *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, volume 1, pages 562–567. IEEE, 2005.

[11] C.N. Silla, C.A.A. Kaestner, and A.L. Koerich. Automatic music genre classification using ensemble of classifiers. In *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, pages 1687–1692. IEEE, 2007.

[12] CN Silla, A.L. Koerich, and C. Kaestner. Feature Selection in Automatic Music Genre Classification. In *Multimedia, 2008. ISM 2008. Tenth IEEE International Symposium on*, pages 39–44. IEEE, 2009.

[13] A. Holzapfel and Y. Stylianou. Musical genre classification using nonnegative matrix factorization-based features. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(2):424–434, 2008.

[14] 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. Citeseer, 2005.

[15] H.T. Cheng, Y.H. Yang, Y.C. Lin, I.B. Liao, and H.H. Chen. Automatic chord recognition for music classification and retrieval. In *Multimedia and Expo, 2008 IEEE International Conference on*, pages 1505–1508. IEEE, 2008.

[16] C. Xu, N.C. Maddage, X. Shao, F. Cao, and Q. Tian. Musical genre classification using support vector machines. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03). 2003 IEEE International Conference on*, volume 5. IEEE, 2003.

[17] E. Tsunoo, G. Tzanetakis, N. Ono, and S. Sagayama. Audio genre classification using percussive pattern clustering combined with timbral features. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 382–385. IEEE, 2009.

[18] R. Mayer, R. Neumayer, and A. Rauber. Rhyme and style features for musical genre classification by song lyrics. In *ISMIR 2008: proceedings of the 9th International Conference of Music Information Retrieval*, page 337. Lulu. com, 2008.

[19] F. Pachet and P. Roy. Exploring billions of audio features. In Eurasip, editor, *Proceedings of CBMI 07*, pages 227–235, Bordeaux, France, 2007.

[20] G. Tzanetakis. Marsyas submissions to MIREX 2007. *MIREX 2007*.

[21] C. Cao and M. Li. Thinkits submissions for MIREX 2009 audio music classification and similarity tasks. In *MIREX abstracts, International Conference on Music Information Retrieval*, 2009.

[22] K. Seyerlehner, M. Schedl, T. Pohle, and P. Knees. USING BLOCK-LEVEL FEATURES FOR GENRE CLASSIFICATION, TAG CLASSIFICATION AND MUSIC SIMILARITY ESTIMATION.

[23] J. Zhu, X. Xue, and H. Lu. Musical genre classification by instrumental features. In *Int. Computer Music Conference, ICMC*, volume 2004, pages 580–583. Citeseer, 2004.

[24] Y. Tsuchihashi, T. Kitahara, and H. Katayose. Using bass-line features for content-based mir. In *Proc. 9th Int. Conf. Music Inf. Retrieval (ISMIR 2008*, pages 620–625. Citeseer, 2008.

[25] M. Goto. A real-time music-scene-description system: predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication*, 43(4):311–329, 2004.

[26] J. Abeßer, H. Lukashevich, C. Dittmar, and G. Schuller. "Genre classification using bass-related high-level features and playing styles". In *Proc. of the Int. Society of Music Information Retrieval (ISMIR Conference), Kobe, Japan*, 2009.

[27] G. Prandi, A. Sarti, and S. Tubaro. Music Genre visualization and Classification Exploiting a Small set of High-level Semantic Features. In *Proceedings of the 12th International Conference on Digital Audio Effects (DAFx09), Como, Italy, September 1–4, 2009.*

[28] J. Foote and M. Cooper. Media segmentation using self-similarity decomposition. In *Proceedings of SPIE*, volume 5021, pages 167–175. Citeseer, 2003.

[29] Martin Ester, Hans peter Kriegel, Jörg S, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

[30] CN Silla and A.A. Freitas. Novel top-down approaches for hierarchical classification and their application to automatic music genre classification. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 3499–3504. IEEE, 2009.