Towards Supervised Music Structure Annotation: A Case-based Fusion Approach.

Giacomo Herrero
MSc Thesis, Universitat Pompeu Fabra

Supervisor: Joan Serrà, IIIA-CSIC

September, 2014

Abstract

Analyzing the structure of a musical piece is a well-known task in any music theory or musicological field. However, in recent years, trying to find a way of performing such task in an automated manner has experienced a considerable increase in interest within the music information retrieval (MIR) field. Nonetheless, up to this day, the task of automatically segmenting and analyzing such structures remains an open challenge, with results that are still far from human performance. This thesis presents a novel approach to the task of automatic segmentation and annotation of musical structure by introducing a supervised approach that can take advantage of the information about the music structure of previously annotated pieces. The approach is tested over three different datasets with varying degrees of success. We show how a supervised approach has the potential to outperform state-of-the-art algorithms assuming a large and varied enough dataset is used. The approach is evaluated by computing standard evaluation metrics in order to compare the obtained results with other approaches. Several case studies that are considered relevant are as well presented, along with future implications.

Acknowledgements

I would like to first thank my supervisor, Joan Serrà, for all the invaluable help and support throughout this whole process. I would like to thank as well all the MTG researchers and professors for sharing their incredible knowledge and especially Xavier Serra for making me part of it. Of course I couldn't have done any of this without the inconditional support from my parents, and for that I will be forever grateful. Last, but not least, big thanks to all my colleagues and classmates for a fantastic year together.

Thank you all!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The task of automatic segmentation and annotation of musical pieces is a common field of study in the music information retrieval (MIR) area. In the last years, the task has experienced an increase in interest, with a considerable amount of different approaches dealing with it. This study thus focuses on discussing said approaches and introducing a new alternative to the task.

The ability to automatically extract information about the musical structure of a song with an acceptable accuracy is a task that can yield an important number of direct and indirect applications. From a musicological standpoint, a system able to provide the relative position of structural elements and the repeating similarities across them could greatly improve the understanding of numerous concepts related to musical form and simplify the large-scale analysis of music. Moreover, it would facilitate the automated analysis of massive corpora, which is currently an unexplored area. Knowledge about the structure of a song can also be useful as a preprocessing stage to other common MIR tasks, such as chord detection [26] or version identification [42]. In a more direct way, there are various applications in which such a system could be useful [6]: automatic chorus detection, intra-piece navigation in music players (i.e. allowing users to jump directly to different sections of a song), playlist and mash-up generation, or automatic extraction of representative audio clips as used in most online music stores and streaming services to offer previews for potential customers.

Nowadays, with the advent of big digital audio and music databases and their ever-increasing growth, new ways of analyzing music arise. In the case of automatic segmentation and annotation, the availability of large music collections of annotated musical data will increasingly allow for new approaches that could not have been undertaken some years ago. In this context, a supervised learning ap-

proach to this task can be considered reasonable, as we expect the availability of annotated musical datasets to continue to increase in the near future, being either from musicological studies or classic MIR datasets such as the Mazurka[1] [41] or SALAMI[2] [47] projects, or even from record labels themselves. Additionally, and in contrast to some of the algorithms that will be reviewed in Chapter 2, a supervised approach is conceptually easy to implement and intuitive to understand.

We believe a case-based approach, comparatively similar to how humans are able to recognize particular song structures, would be an adequate fit to the task. In the same way a person is only able to recognize a structure in a musical piece bacause they have had prior experience with similar or contrasting structures, a machine, although far from providing an accuracy as high as a human would, should be able to integrate past knowledge as well in order to determine future structures.

1.2 Musical Structure

Musical structure is a concept that deals with the formal organization of musical pieces. There is, to date, not a clear consensus on what exactly makes us humans perceive the differences in the structure of a song, although it is possibly a combination of several factors such as changes in rhythm, changes in melody, changes in harmony and, from a linguistic standpoint, changes in lyrics [3].

In musicology, the concept of structure in music is traditionally referred to as *musical form*. Musical form can be described on many different levels, from large-scale or high-level musical structures to micro-structures within them corresponding to brief motifs or passages. Furthermore, even if a consensus on a definition of 'form' were reached, it would not translate from one genre to another [46].

In this study, however, we will refer as musical form or structure to the higher level of music structural organization that can be found in most Western music. This structure refers to the different alternating or repeating segments that compose a musical piece. For instance, in popular music, these would correspond to 'intro', 'chorus', 'verse' or 'bridge'. For simplicity, this concept will henceforth be referred to as *musical structure*. An example can be seen in Figure 1.1. Nonetheless, even with the mentioned definitions, different annotators can annotate the structure of a song in a contrasting manner. This will be shown in Chapter 5 when discussing the corpora used in this study.

In order to characterize these structures, at least one element is necessary: the location of the boundaries that form an established "section" of a piece, which we will henceforth refer to simply as boundaries. These boundaries, however, while

[1] http://www.mazurka.org.uk

[2] http://ddmal.music.mcgill.ca/research/salami

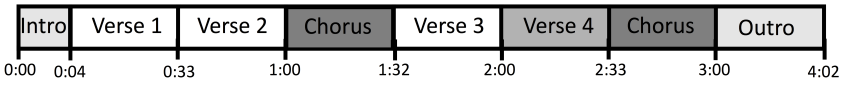


Figure 1.1: Basic structure of the song *There is a light that never goes out* by The Smiths.

providing information about *changes* in the piece, do not explicitly convey anything about the *repetitions* that occur within it. In order to provide information about repetitive structures, there needs to be what we will call "labels". In our case, said labels will not carry semantic information about each section (i.e. "intro", "verse", "chorus") but will merely be an indicator of what sections are repeated over the piece (i.e ABCABA or, in this case, '0','1','2','0','1','0'). This is common practice in all MIR approches so far, with some exceptions (see [31]). As we will see in the state of the art review (Chapter 2), depending on whether the goal is to assess the boundaries and/or the labels, the approach will vary greatly. In this study, however, the objective is to perform both a *segmentation* and an *annotation* of the piece, and finding both boundary and label elements, therefore, will be imperative.

1.3 Goals

The main objective of this thesis is to explore new alternatives to current algorithms for automatic segmentation and annotation of songs. In particular, it aims at exploring supervised approaches that contrast to current unsupervised ones. To do so, we follow a number of necessary steps:

- An extensive review of past and state-of-the-art approaches must be performed in order to establish the context where this thesis is situated, gather possible ideas for our approach and elements to consider or avoid, as well as establish what is considered standard practice in the evaluation procedure.
- Starting with a collection of datasets, the next step is to develop a simple supervised learning algorithm for the retrieval of similar structures (or previously annotated cases).
- A subsequent step must be implemented in order to refine the results obtained and adapt them to the piece being annotated. In this case three different variants of postprocessing have been implemented.
- A special effort has been made in order to provide a concise and significant evaluation of the algorithm. Standard evaluation metrics have been used in

order to allow for comparisons between the aforementioned state-of-the-art algorithms and this thesis.

- The final step will be a commentary and discussion of the approach and the results presented, and the proposal of potential future research lines.

1.4 Outline of the Thesis

This document is presented as follows: in Chapter 2, a review of the state of the art in the particular task of automatic segmentation and annotation is presented. Chapter 3 includes the methodology followed in this study, explaining the feature extraction and supervised learning stages, with three different approaches. In Chapter 4 the results of this study are presented and discussed. In Chapter 5 some conclusions are drawn and a general discussion about the contributions of this study is presented, along with suggestions for future research lines.

Chapter 2

State of the Art

As anticipated in the previous Chapter, structural form is a term that most of the time is not clearly defined. As a result, the different approaches that can be taken to handle the problem are varied, depending mainly on the final goal one wants to achieve. However, apart from their differences, some common ground between all methods can be observed. Table 2.1 offers a summary of all the described methods, techniques and features, with information about evaluation procedures and corpora used.

When it comes to automatically analyzing the structure of a musical piece, the process can be divided into two different stages that are considerably independent of each other and which provide different information about the piece. The first stage could be viewed as the segmentation stage, where a musical piece is divided into parts that are considered different from each other, while the second stage aims at assigning labels to segments that are conceptually similar. Figure 2.1 shows the diagram of a very general take on the segmentation and annotation problem. Most of the techniques employed in past and current works adopt to some degree these steps included in the diagram: a feature extraction stage (often accompanied of a pre-processing step), a measure of similarity between feature vectors, and a final segmentation step.

Author(s), Year	Ref	Basic Technique	Descriptors	Corpus	Evaluation
Logan and Chu, 2000	[22]	HMM + Clustering	MFCC	18 beatles songs	User tests
Foote and Cooper, 2003	[12]	SSM + Clustering	MFCC	one example	Visual evaluation
Bartsch and Wakefield, 2005	[2]	SSM + correlation	Chroma	93 pop songs	Precision and Recall
Ong, 2005	[29]	SSM	Timbre + Dynamics	54 Beatles songs	Precision, Recal and F-Measure
Goto, 2006	[15]	SSM	HPCPs	RWC-P (100 songs)	Precision, Recall and F-measure
Chai, 2006	[4]	HMM	Tonality (Key and Mode)	26 Beatles songs + 10 class piano	Precision, Recall and F-measure
Peeters, 2007	[35]	SSM+ Maximun Likelihood	MFCC, Spectral Contrast and PCP	11 popular songs	Own measure
Eronen, 2007	[9]	SSM beat-synced	MFCC + Chroma	206 pop rock songs	Precision, Recall and F-measure
Turnbull et al., 2007	[48]	Supervised (BSD)	Timbre + Harmony + Melody + Rhythm	RWC-P (100 songs)	Precision, Recall, F-Measure + true-to-guess and guess-to-true dev.
Jensen ,2007	[18]	SSM	Timbre + Chroma + Rhythm	21 Chinese songs + 13 electronica songs + 15 varied songs	Precision, Recall and F-Measure
Peiszer et al., 2008	[38]	Clustering	MFCC	94 pop songs	Precision and Recall + edit distance for labels
Levy and Sandler, 2008	[20]	Clustering	Audio Spectrum Envelope	60 songs varied genres	Precision, Recall and F-measure + pairwise F-measure for labels
Mauch et al., 2009	[26]	SSM beat-synced	Chroma	125 Beatles songs	Only reported for chord detection
Cheng et al., 2009	[5]	Clustering	Audio Spectrum Envelope + Chord Sequence [1]	13 Chinese and Western songs	Precision, Recall and F-measure
Paulus and Klapuri, 2009	[31]	Fitness Measure	Chroma + Timbre + Rhythmogram	TUTStructure07 (557 songs) + Beatles (174 songs) + RWC-P (100 songs)	Precision, Recall, F-Measure + Over- and under-segmentation
Peeters. 2010	[36]	Clustering	Timbre + Chroma	MIREX collections	Precision, Recall, F-Measure (+ MIREX submission)
Barrington et al., 2010	[1]	DTM (HMM)	Timbre + Melody + Rhythm	RWC-P + 60 pop songs	Precision, Recall, F-Measure + true-to-guess and guess-to-true dev.
Weiss and Bello, 2011	[50]	SI-PLCA beat-synced	Chroma	Beatles (180 songs)	Precision, Recall and F-measure
Rocha et al., 2012	[40]	SSM tempo-adjusted	Timbre	35 EDM songs + RWC-P + Eurovision (124 songs)	Precision, Recall and F-measure
Kaiser et al., 2013	[19]	SSM	Timbre	MIREX collections	Precision, Recall, F-Measure (+ MIREX submission)
Serrà et al., 2014	[44]	SSM	HPCP	Beatles (180 songs) + RWC-P + Mazurka (2972)	Precision, Recall and F-measure
McFee and Ellis, 2014	[27]	SSM	Chroma + Timbre	Beatles (179 songs) + SALAMI (253 songs)	Precision, Recall and F-measure
Ullrich et al., 2014	[49]	CNN	Spectrogram	SALAMI (487 songs) + 733 songs	Precision, Recall and F-mesure

Table 2.1: Summary table of related works.

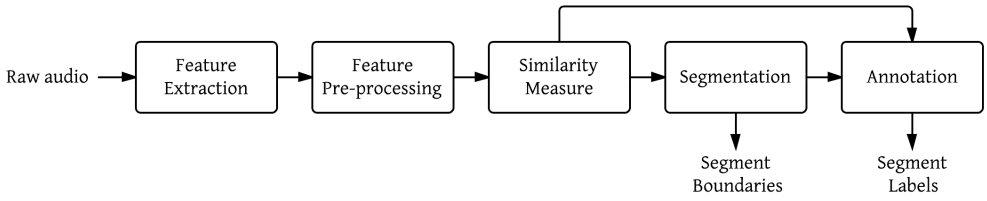


Figure 2.1: Schematic diagram of the general process for structural segmentation.

There have also been some attempts to categorize the different techniques into well-defined taxonomies. A first attempt at categorizing existing approaches was done by Peeters [34]. This taxonomy is called the *sequences* and *states* hypothesis, where the former aims at looking for feature vectors that form recurring patterns while the latter aims at detecting sections of the piece that are homogeneous with respect to a certain feature. From this taxonomy, Paulus et al. [32] then tried to provide a more semantic definition by introducing three basic principles to tackle this task: approaches based on novelty, approaches based on repetition, and approaches based on homogeneity. As it can be seen, it draws from the states and sequences hypothesis while introducing a new element: the novelty-based approach, in which the task is, as the name suggests, to seek for elements of the piece that are contrasting between successive parts.

Of course, more often than not, there is not a clear distinction between said principles in the same approach, and boundaries between them become fuzzy. Some examples can be seen in [31], where the proposed system combines both sequence- and state-based approaches not explicitly, and [44], where the novelty detection was inferred by the structural repetitions in the piece. These combinations, as can be seen in the following sections, are fairly frequent in the literature.

In Section 2.1, a review of the descriptors used is offered (corresponding to the first block of the diagram in Figure 2.1). Section 2.2 introduces some of the most common preprocessing stages (second block in the diagram). Section 2.3 describes the different techniques used in the segmentation stage (last two blocks of the diagram). Finally, in Section 2.5 an overview of the available datasets and evaluation procedures is offered.

2.1 Feature Extraction

A possible categorization is according to the type of descriptors used: timbral, rhythmic, harmonic or melodic (both often included under the unifying term

chroma). In general, across different parts of a piece, these (or at least a combination of such) characteristics will vary. For instance, a chorus section is likely to have a different chord progression, with different instruments and perhaps a different melodic line, than the verse section. Some of the methods described below use a combination of several features, although as this increases computational cost, most algorithms focus on single descriptors or combinations of two. In an automatic structure segmentation task, as in any other MIR-related task, the selection and extraction of the features that define the element to be characterized is crucial.

When trying to characterize a piece according to its texture, or particularly for the case of segmentation tasks, its *change* in texture, most researchers make use of Mel-Frequency Cepstrum Coefficients (MFCCs) [21], which has been demonstrated to be of relevance for this purpose. Depending on the number of coefficients used, some pitch information is expected to be captured as well, which can be useful or in some cases a hindrance, depending on the final objective one wants to achieve. Some examples of the use of MFCCs can be seen in [22], which is one of the first approaches dealing with segmentation of songs to extract key phrases. Foote and Cooper [12] describe a method for media segmentation that also makes use of MFCCs. In more recent works, these features can be seen as well in more content-specific tasks, such as the work described in [40], that deals with segmentation focused only in electronic dance music, in which the timbre characteristics are very relevant, and in [38], where they are used to segment the structure in popular music.

Peeters [35] and Eronen [9] both use a combination of MFCCs with Pitch Class Profiles to find melody repetitions in songs and chorus sections, respectively. Pitch class profiles (first used by Fujishima [13]) are a representation of the spectrogram of an audio piece, in which the energy content of the frequencies are mapped into the 12 pitch classes of the Western chromatic scale (although depending on the use there can be a higher number of bins [14]). An enhanced version of pitch class profiles, Harmonic Pitch Class Profiles (HPCPs) [14], incorporate as well information about the harmonic content. This type of features, commonly called chroma features, is considered extremely useful in cases where the majority of changes in a song occur in the harmony or melody, e.g. chord or voice changes [32].

Some other structure-related uses of chroma features can be found in [2], where a set of chroma features was used to extract audio thumbnails (i.e. representative excerpts of a song) from popular music. Mauch et al. [26] used them to enhance chord detection by applying structural information in the algorithm. In the last years it has been used for structure-based fingerprinting in music retrieval tasks [17], as well as in classic segmentation and annotation tasks [44] [32].

The last category of descriptors that is also used (although to a lesser extent)

are descriptors based on rhythm. Rhythm is often a characteristic of music that has also been found to be useful in certain circumstances, especially in popular music, where transitions between sections of the song are often led by drum fills or even different rhythmic patterns occur in contrasting parts. Some examples of structure annotation approaches using rhythmic feature can be found in [48] and [39], using both what it is called *fluctuation patterns* (FP), a measure of the modulation in loudness for a particular series of frequency bands [30]. Some research has been done as well to combine other forms of data with audio in order to improve the results of the segmentation. Particularly, Cheng et al. [5] use information provided by the lyrics plus timbral and harmonic descriptors to segment the audio.

2.2 Preprocessing Stage

One of the most common preprocessing techniques (see [9], [26] and [50]) is to perform a synchronization of the feature vectors with beat information, so that the boundaries of the segments are forced to rely on estimated beat positions. This preprocessing technique is useful to obtain higher accuracies, since normally sections' boundaries of a song correspond to beat positions. However, it requires more computational time and normally beat-tracking algorithms are not yet completely accurate, which could add to the segment detection to fail. Thus, in the end, the higher accuracy of beat-based segmentation algorithms remains an open issue.

Eronen [9] uses beat synchronous MFCCs and chroma features in his work on chorus detection by means of a two-fold self-distance matrix, one per feature vector. Beat-tracked features help synchronize both matrices and refine the location of the chorus boundaries. Mauch et al. [26] use beat positions to perform structure segmentation and measure the similarity between chord segments in order to improve chord detection. Chord segments are compared in harmonicity and in beat length. Weiss and Bello [50] employ beat-synchronous chromagrams in order to discover recurrent harmonic motifs in songs. In a more recent study, McFee and Ellis [27] use as well a beat-synchronous feature vector to develop a structural feature called *latent structural repetition*. Another common preprocessing stage is first normalizing the feature vector, usually performed over chroma features to account for changes in the signal dynamics [44].

2.3 Segmentation and Annotation Approach

A different taxonomy can also be established by considering the most common techniques used for the core part of the task: segmentation and annotation (last

two blocks in Figure 2.1). For an extended overview, the reader is referred to the surveys developed by Smith [46], Paulus et al. [32] and Dannenberg and Goto [6]. In the following sections, some main approaches are reviewed: self-similarity matrices, hidden Markov models, and clustering techniques.

Methods based on self-similarity matrices

A similarity matrix or recurrence plot [8] is a representation of the local similarity between two data series. In the case of self-similarity matrices (SSM), it represents the similarity between different parts of the same series. By its nature, this way of visualizing helps to find recurrent patterns in an audio excerpt. Figure 2.2 shows an example of an ideal self-similarity matrix for the song *Yesterday* by The Beatles. The letters indicate the different sections of the song while the diagonal lines indicate the repetition of said sections.

The seminal work of Foote [10] used self-similarity matrices in order to visualize audio and music. In this first work, MFCCs were used to assess the similarity between frames, since the author mainly used this visualization for drum patterns. Foote also introduced a measure of similarity between segments by computing the correlation between feature vectors over a specific window. In later approaches, Foote used the same techniques to perform automatic segmentation of musical pieces using cosine [11] and Kullback-Leibler [12] distance to measure similarity.

In his work on chorus sections detection, Goto [15] implemented a segmentation method based on SSM and using pitch class information to analyze relationships between different sections. This approach had the novelty of accounting for key changes that often occur between chorus sections of the same song. The system starts by extracting the 12-pitch-class vectors from audio and then calculates the pairwise similarity between them, after which the pairs of repeating sections are integrated into larger groups that are pitch shifted to account for modulations in key (this step was later refined by Müller and Clausen [28]). The process ends by selecting the group that possesses the highest average similarity. In more recent studies, Peeters [35] goes one step further and uses second- and third-order SSMs in order to reduce noise by reinforcing the diagonals, and a Maximum-likelihood approach to reveal the most representative section.

In [17], a method for extracting what they introduced as *structure fingerprints* is implemented by means of SSMs and chroma features. Using these structure fingerprints, or structure features, Serrà et al. [44] propose a novelty detection system based on SSM and HPCPs, that yielded fairly good results in MIREX 2012. In that case, they measured similarity by using simply Euclidean distance. The algorithm [44] makes use of a time-lag representation of SSMs to compute a novelty curve that represents changes in the structure. The peaks on that curve are used again on the SSM in order to infer non-semantic labels from it. The study

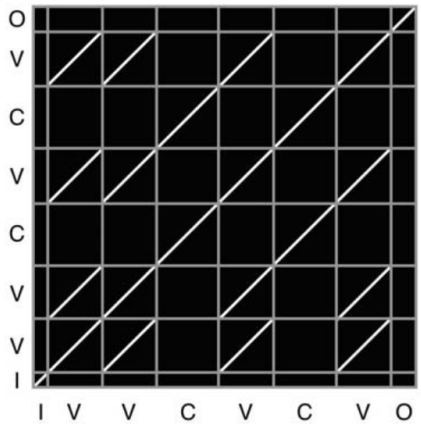


Figure 2.2: Ideal sequence representation for the SSM of "Yesterday". The letters I, V, C and O stand for "intro", "verse", "chorus" and "outro", respectively. Gray lines indicate annotated boundaries and diagonal white lines indicate repetitions. (Taken from [46].)

by Serrà et al. is in fact the research that serves as basis for this thesis.

Methods based on clustering

Some of the most common approaches resort to methods based on clustering [51] for the segmentation part. Normally, clustering techniques work by first segmenting a feature vector into fixed-length frames, which are then assigned a label. A bottom-up clustering technique would then find similarities and/or distortions between said frames and cluster together segments that follow a certain rule. This process usually iterates until a prestablished condition is met.

An early work in automatic structure analysis was done by Logan and Chu [22], who found the use of clustering techniques on MFCC feature vectors to be successful in user experiments to find key phrases in a song. The system used bottom-up clustering to find repeating structures within a piece and select the one that repeats the most as a key phrase.

Foote [12] also used clustering techniques, coupled with SSMs, that helped reveal the structure of the piece after determining its segments' boundaries. Obtaining the boundaries from the SSM beforehand reduces the computation cost of performing the clustering algorithm, since clusters can be assumed to be within said boundaries. In [38] several clustering techniques (based on k-means) are tested with MFCCs in order to assign labels to segments detected by means of self-similarity matrices, and [20] used as well clustering after predicting section labels with a hidden Markov model approach (see below).

Methods based on hidden Markov models

A hidden Markov model is a statistical model in which the system to be modeled is a Markov process with states that are inaccesible to the observer. A Markov process is a stochastic process where predictions on futures states can be made solely by observing its most recent states. In automatic structure detection, such a system is used in general to predict changes in a pre-established feature, such as chroma, key or timbre. By being able to succesfully predict those changes, inferring the structure of a song becomes possible. Some of the studies mentioned above use hidden Markov models to enhance or perform a preprocessing step for the segmentation (see [22]). The approach in [4] uses key and mode as features to build an HMM in order to automatically segment and extract a summarization of the piece. The HMM is trained by empirically assigning the transition and observation probabilities according to musical theory. Levy and Sandler [20], instead of using HMMs over the feature vector directly, first employ a clustering technique to create more robust states for the HMM. A similar approach is taken in [1], where a *Dynamic Texture Model* (DTM) is used as a sequence of states to predict changes in timbre and rhythm. A DTM behaves similarly to an HMM, but while in the latter the states are discretized, the DTM uses a continuous state space that is able to account for smooth state transitions. Both [20] and [1] train the HMMs and DTMs respectively by solely analyzing the song to be annotated, introducing a basic element of supervision. However, no information about other songs and/or different annotations from the song being considered in the annotation process is used.

Supervised approaches

To the best of our knowledge, only three attempts have been made so far to employ explicitly supervised techniques to the task of audio segmentation. The approach in [48] is based on the AdaBoost algorithm, and uses a combination of timbral, harmonic, melodic and rhythmic low-level features. The system starts by creating a set of *difference features* by means of a sliding window over the low-level features and comparing the information in the first half of the window with the second half. These resulting features are then smoothed and combined into a high-dimensional feature vector. Samples corresponding to a boundary are labeled as such and a supervised boosted decision stump classifier is trained to classify the two different classes (boundary or non-boundary). The system however does only identify boundaries, and does not go beyond to label annotations.

In [27], a latent structural repetition descriptor that is based on self-similarity matrices is used to facilitate learning. First, a beat-synchronous feature vector is employed to determine an SSM that, in a similar fashion as in [44], is circular-

shifted, filtered and limited to a fixed dimension. The resulting matrix is called *latent structural repetition*, which is a descriptor of the repetitions of a song. The features used to compute said descriptor are weighted by a variant of Fisher's linear discriminant analysis in order to optimize the output of a clustering algorithm by giving more importance to some features according to the statistics of the training data. Labels are then assigned by anlyzing the individual songs. Thus, no supervision is used in that stage.

Ullrich et al. [49] used Mel spectograms to train a convolutional neural network (CNN) with a subset of the SALAMI dataset in order to detect the boundaries of a piece. At query time, a boundary probability curve is computed over which a peak-picking algorithm extracts the exact location of each boundary. While all three approaches use a different supervised approach for the task of structure segmentation, so far no attempts have been made to use a supervised approach to combine both boundary and label detection.

2.4 Music Corpora

Up until recently the use of standard corpora across studies, which would enable easy comparison between algorithms and results, was fairly uncommon. It was not until 2007, when MIREX (MIR Evaluation eXchange)[2] [7] started including tasks of automatic audio segmentation, that researchers started to push for unified datasets. Nowadays, there is quite a more established system, although some researchers still use their own corpus, mainly because the existing ones do not fit their scope (see [40] as an example). This section briefly describes the main datasets used nowadays in the task of automatic audio segmentation: The Beatles[3] dataset, the Real World Computing[4] (RWC) dataset, the Mazurka Project[5] dataset and the recent SALAMI (Structural Analysis of Large Amounts of Music Information) dataset[6].

Beatles

The Beatles dataset has been evolving from its early stages when musicologist Allan Pollack started analyzing the entire Beatles catalog in terms of melody, structure, key, style, etc. The Universitat Pompeu Fabra (UPF) did a preliminary stage of annotating timing information according to Pollack's study and the Tampere

[2]http://www.music-ir.org/mirex
[3]http://isophonics.net/content/reference-annotations-beatles
[4]https://staff.aist.go.jp/m.goto/RWC-MDB
[5]http://www.mazurka.org.uk
[6]http://ddmal.music.mcgill.ca/research/salami

University of Technology (TUT) performed an editing step to said annotations. Independently, the Centre for Digital Music (C4DM) at Queen Mary University of London also undertook the task of annotating musical form, key, chords and beat locations for the entire Beatles discography [25].

Nowadays both C4DM's (180 songs) and TUT's (177 songs) annotations are commonly used for evaluation, although the divergence in style and timings between both makes sometimes the evaluation process inaccurate. Some studies in how the disparity between annotations can affect the results and evaluation can be found in [44], where both datasets were used to cross-evaluate each other. In Section 2.5 some discussion about these issues is presented.

Real World Computing

The Real World Computing database [16] (RWC) is a common dataset built for audio processing and MIR tasks. It is divided by genres, with 100 songs for popular music, 50 songs for jazz, 50 songs for classical music and a set of 100 mixed genres songs. For this database, two different annotations were performed: the IRISA (Institut de Recherche en Informatique et Systèmes Aléatoires) annotations, which offer structural annotations as defined in [3], only for the pop music RWC database, and the original AIST (National Institute of Advanced Industrial Science and Technology) annotations, which offer beat-synced structural annotations for the entire RWC database. AIST also provides the audio material for research purposes.

Mazurka Project

The Mazurka Project dataset [41] is one of the largest annotated databases of classical music available. It consists of a collection of 49 of Chopin's Mazurkas, assembled by the Mazurka Project[7], including several performances of each piece, which amounts to a total of 2792 recordings that were mostly manually annotated by a human expert. However, the annotations for musical structure were performed automatically with a later validation from an expert. See [44] for information about the creation and validation process.

SALAMI

The SALAMI dataset [47] is the only dataset built exclusively for audio structure analysis and segmentation tasks. It has been in ongoing development by McGill University in partnership with the University of Illinois at Urbana-Champaign and

[7] http://www.mazurka.org.uk

the University of Southampton, and it offers a total of more than 350,000 recordings from various sources, although the annotated data that is publicly available is a subset of only 779 recordings. This subset provides structural information including small- and large-scale structure, and musical form information. However, due to copyright laws, a large majority of the audio is not publicly available, making it unsuitable for this thesis. Although some of the audio files can be obtained legally, there is no guarantee that the annotations will correspond exactly to the audio files, be it because of differences in the mastering or simply the timestamps of the audio do not match the ones in the annotations.

2.5 Evaluation Metrics

In order to evaluate the performance of each algorithm, several measures have been proposed in. All these aim at comparing the result yielded by the algorithm with the ground truth annotations. In most of the tasks, the evaluation process is divided into two separate procedures: (a) establishing the accuracy of the algorithm in segmenting the audio piece and (b) evaluating the labeling of repetitions. For a detailed review of evaluation metrics used in structure segmentation and annotation see [46] and the MIREX website.

2.5.1 Segment boundary evaluation

Evaluating any MIR-related algorithm can be a difficult task. Oftentimes, researchers do not conform to a standard evaluation metric (even if there is a fairly common one, they do not always rely on it) and this makes the comparison between algorithms much more arduous. These issues translate as well to the task of music segmentation. However, as can be observed in Table 2.1, there is a tendency to employ three standard measures developed in the information retrieval context [24]: precision, recall and f-measure. As a way to compare with previous and future algorithms, these measures were also used in this thesis.

Particularly, precision in this task means the number of estimated segments' boundaries that fall within a temporal threshold or time interval of the corresponding ground truth out of the total number of estimated boundaries. Conversely, recall amounts to the number of estimated values that fall within the threshold out of all ground truth boundaries.

$$P_B = \frac{|ES \cap GT|}{|ES|} \tag{2.1}$$

$$R_B = \frac{|ES \cap GT|}{|GT|} \tag{2.2}$$

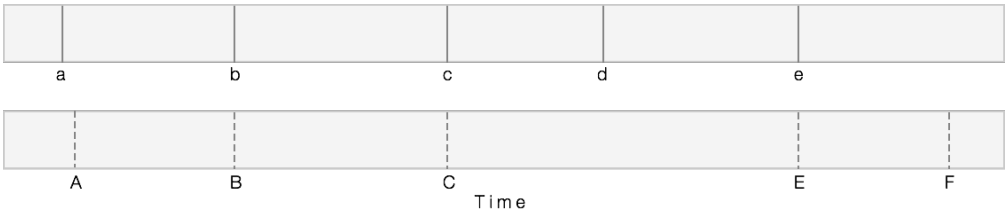


Figure 2.3: Example diagram of a comparison between ground truth (top) and estimated (bottom) segment boundaries.

where ES denotes the number of estimated boundaries and GT the number of boundaries in the ground truth.

As a way to avoid cases of over-segmentation (high precision and low recall) and under-segmentation (low precision and high recall) an f-value [24] (harmonic mean) is often introduced as a way to enforce high values of both.

$$F_B = 2 \cdot \frac{P_B \cdot R_B}{P_B + R_B} \tag{2.3}$$

In Figure 2.3 we can see a simple example of comparison between a segmentation result and its corresponding ground truth. Boundary `A` is estimated with a slight offset, which depending on the threshold used (see Section 2.5.3 for common values) could yield both a false negative and a false positive [29]. Not detecting boundary `d` could yield a false negative and estimating `F` yields a false positive.

The aforementioned metrics however only provide a binary "hit or miss" information and do not provide information about the deviation (in time usually) from estimated boundaries. To this end, precision and recall are often accompanied by measures that indicate how incorrect the estimated boundaries are from the ground truth and vice versa. This is commonly called median true-to-guess deviation in the case of the median of all minimum distances between each ground truth boundary and any estimated boundary, and guess-to-true deviation in the case of all minimum distances between each estimated boundary and any ground truth boundary [48].

2.5.2 Label evaluation

While evaluating the detected boundaries might seem fairly simple and intuitive, evaluating the estimation of the labels assigned to each segment is not. Assigning the labels "ABCABC" to a piece that is annotated as "BCABCA" should still yield a perfect match, since the repetitions are identical. In order to consider this, most approaches rely on clustering metrics, since the task of detecting repetitions

is precisely grouping segments together. One of the most common ways to compute this is applying precision and recall in a pairwise fashion to all labels [20] so that:

$$P_L = \frac{|M_{ES} \cap M_{GT}|}{|M_{ES}|} \tag{2.4}$$

$$R_L = \frac{|M_{ES} \cap M_{GT}|}{|M_{GT}|} \tag{2.5}$$

where M_{GT} are all pairwise matches in the ground truth and M_{ES} all pairwise matches in the estimation. As for boundaries, an f-measure (F_L) is often included as well.

Lukashevich [23] introduced another metric of over- and under-segmentation by using a measure of normalized conditional entropy. This metric accounts for the fact that the difficulty of randomly assessing the structure of a piece increases with the number of segments of which it is composed.

2.5.3 Music Information Retrieval Evaluation eXchange

The Music Information Retrieval Evaluation eXchange (MIREX) is an international evaluation campaign for MIR algorithms, partnered with the International Society for Music Information Retrieval conference (ISMIR), and hosted by the University of Illinois at Urbana Champaign. It has been organized since 2005 and aims at setting standards for the evaluation of MIR algorithms.

In 2009 the task of "Structural Segmentation" was introduced and while the dataset to be used has been evolving, the evaluation metrics have been stagnant throughout the years, in order to allow comparisons between algorithms from different years. The current dataset employed in the MIREX evaluation process includes a collection of 297 songs (from The Beatles and other pop songs), the RWC-P dataset (100 pop songs as mentioned in the above section) and a last collection of 1000 songs of mixed genres, which has been mostly annotated by two independent experts.

As for the evaluation metrics used in MIREX, they include all measures mentioned in Section 2.5: precision, recall, f-measure and median deviations for the segmentation task, and pairwise precision, recall and f-measure for the labeling task. The threshold for boundaries' hit-rates in the segmentation is double: a fine threshold of 0.5 seconds (as in [48]) and a coarse threshold of 3 seconds (as in [20]), which correspond to windows of 1 and 6 seconds around boundaries, respectively. Furthermore, they include a measure for over- and under-segmentation: normalized conditional entropies [23], which measure both the amount of information of the estimation that is missing from the ground truth and the amount of *spurious* information that is present in the estimation.

Chapter 3

Case-based Supervised Annotation of Musical Structure

In order to determine the suitability of a supervised approach to the task at hand, several variants of a straightforward method have been implemented in the course of this thesis. In this section, the methodology followed to achieve that is explained. A general view of the proposed supervised algorithm can be seen in Figure 3.1. The complete process is divided into two main blocks: the feature extraction stage (Section 3.1), performed over both the training set and the query data, and the supervised learning algorithm used in order to find the most similar matches (Section 3.3). The first task is to extract the so-called *structure features* [44], which are going to represent the similarities and repetitions within each song. Said features are then used by the supervised algorithm to find the matches of a query with the most similar structure in the database of structure features. The general task therefore entails tackling a two-part problem: first, the structure features must be representative enough of the musical structure of a piece, and second, a way of obtaining information from the resulting neighbors must be implemented. This last step is represented as *Neighbors Fusion* in the diagram of Figure 3.1.

3.1 Feature Extraction

The feature extraction process (as detailed in [17]) begins with extracting the HPCPs (see Section 2.1) of the raw audio data. The main parameters used in the extraction of the descriptors are: 12 pitch classes, a window length of 209 ms and a hop size of 139 ms, as described in [44], [43] and [45]. A routine called delay coordinate embedding is then employed over the resulting feature vector to include information of its recent past as a way to discount for quick changes in the HPCP vector. This technique has already been proved to be useful in this

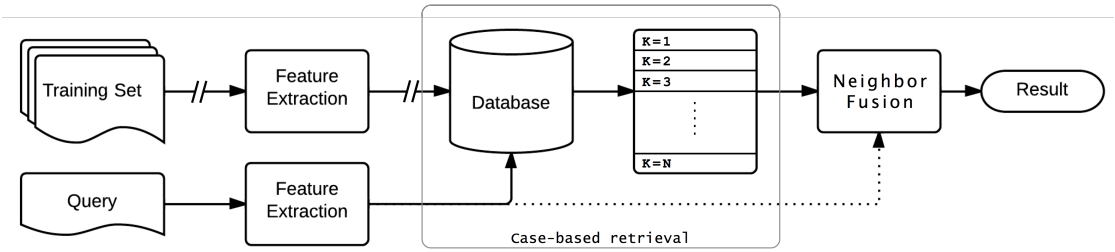


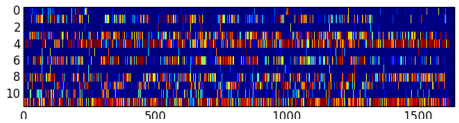
Figure 3.1: Diagram of the general process of the methodology followed.

particular task [44] and the parameters ($m = 3\,s$ and $\tau = 1\,s$) are the same as in [44]. A self-similarity matrix is next built from the resulting delayed vector, which accounts for similarities and repetitions within the signal. In this particular case this is done by assessing the similarity for every sample in a pair-wise fashion by means of a simple k-Nearest Neighbors algorithm with k set to a fraction of the length of the signal ($k = \kappa N$ where $\kappa = 0.03$ as detailed in [44]). In order to easily compute the similiarities and repetitions, the SSM is next circular-shifted along its rows and a Gaussian blur is applied by convolving the shifted SSM with a Gaussian window with a kernel of size $lhor \times lver$ (where $lhor = 215$ and $lver = 2$ samples, as indicated in [44] (Figure 3.2c)). In order to easily store and compute the data, the resulting matrix is next downsampled to 100x100 samples. Finally, as the information in the original matrix is duplicated due to the SSM symmetry, only half of the matrix is stored, creating a one-dimensional vector.

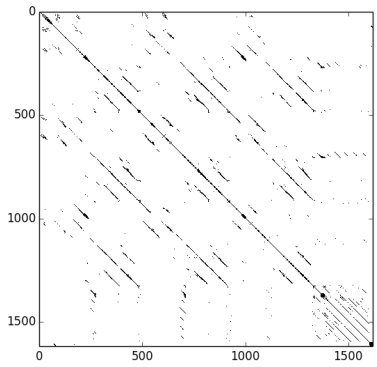
3.2 k-Nearest Neighbors

The first step is common for all methods implemented and it consists on obtainig the k nearest neighbors of an input query. The algorithm chosen to perform the structural features retrieval is a simple k-NN algorithm. A k-NN algorithm is one of the simplest machine learning algorithms. The training data, in this case all structural feature vectors are distrubuted along a multidimensional space. By inputting a new data sample (our query) it will return the k nearest data samples from the training data according to the distance metric provided.

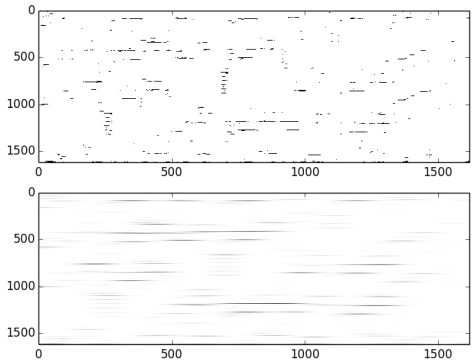
Specifically in this thesis, a k-dimensional (KD) tree was used. A KD-tree is a binary tree structure which is often used as a substitute of a brute-force algorithm when the number of samples N is large and the dimensionality of the features D is relatively low (in which case the cost is approximately $O[Dlog(N)]$). A *leaf size* parameter is set to 20 operations, after which the algorithm switches to brute force. We use the Euclidean distance as after testing with several, more complex distances



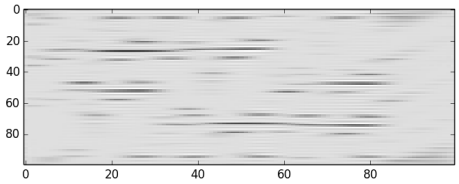
(a)



(b)



(c)



(d)

Figure 3.2: Feature extraction process for the song *Across the Universe* by The Beatles. (a) HPCPs, (b) self-similarity matrix, (c) circular-shifted and smoothed SSM, and (d) downsampled version.

(Manhattan, Minkowski or Mahalanobis) the results were not significantly better. In the tests, the module `neighbors` from the scikit-learn library v0.14 [33] for Python was used and all methods explained in the next sections were first tested with $k = 5$ neighbors. All other paremeters were left as default. After obtaning a list of the k nearest neighbors of the query, a method for combining the information from all the neighbors is required.

3.3 Neighbor Fusion

In the third stage of the system the goal is to obtain information about the nearest neighbors of the query and perform a neighbor fusion. Generally, this can be done either by performing a fusion over the extracted features (early fusion) or over the semantic information (late fusion). In our case, semantic information refers to the annotation information, that is, performing the neighbor fusion once the annotations for boundaries and segments are obtained from the neighbors. In this thesis, examples of both approaches are taken (early fusion represented in Figure 3.1 as a dotted line).

Three different methods of fusion are thus presented in this thesis, which are explained in sections 3.3.2 and 3.3.3 (late fusion) and 3.3.4 (early fusion). A preliminary approach is as well described in section 3.3.1. The results obtained with all methods are presented in Chapter 4.

3.3.1 Method 0

The first and most simple method, which serves the purpose of a baseline for the subsequent methods is to simply select the first neighbor. The process, which is shared with the ensuing methods, can be summarized as:

1. Obtaining the filename of the first neighbor (excluding the query).
2. Removing possible duplicates of the same song. In the case of the mazurka database we remove performances of the same song and artist for different years, e.g., for the file `Chopin_Op006No1_Hatto-1993_pid610003-03.mp3` the rest of files with `Chopin_Op006No1_Hatto` in their filename are removed.
3. Obtaining the annotation information of said neighbor
4. Uniformly rescaling the neighbor's song duration to match the duration of the queried song.
5. Assign such rescaled annotation to the query.

3.3.2 Method I

The next step is to merge the information about the annotations from more than one neighbor obtained with the method described in 3.3.1. In this method, after obtaining the annotations from the results of the query, each result is processed as follows: a Gaussian window is placed over every boundary of each retrieved song. All resulting Gaussians of all songs are then summed over the k results. A peak-picking algorithm chooses the number of peaks corresponding to the median number of boundaries for all songs. Labels are then calculated as the mode of all songs between said boundaries. In a step-by-step process, the algorithm works as follows:

1. Method 0 (except step 5).

2. The boundaries locations are converted to miliseconds to maintain an acceptable resolution throughout the process.

3. For each boundary in each song a Gaussian window is calculated with the following parameters: the left part of the window has $\sigma_l = 0.1N_L$ where N_L is the length of the section that precedes the boundary and the right part of the window has $\sigma_r = 0.1N_R$, where N_R is the length of the section that follows the boundary. The length of the window is fixed at M=23000. The assymetry in the Gaussian window is necessary in order to avoid the overlap of two windows when two boundaries are close together. The resulting time series of Gaussians will then have a length of $L = length(query)$.

4. The resulting k time series of Gaussian windows are summed together and the resulting time series is normalized to the range 0-1.

5. A peak-detection algorithm is used over the resulting time series. The algorithm will consider a peak all local maxima inside a 500 ms window.

6. To avoid oversegmentation, out of all peaks, only the $\tilde{x}_k$ maximum values are selected, where $\tilde{x}_k$ corresponds to the median number of boundaries of the k neighbors.

7. The peak locations will correspond to the boundaries of the resulting annotation.

8. Each label of each neighbor is multiplied by the duration of its corresponding segment (in miliseconds) yielding a time series of again length $L = length(query)$, where each sample is a label. The resulting time series is then downscaled by a factor $T = 100$, as 100-ms subsegments provide sufficient precision for the 3-second threshold of the evaluation procedure.

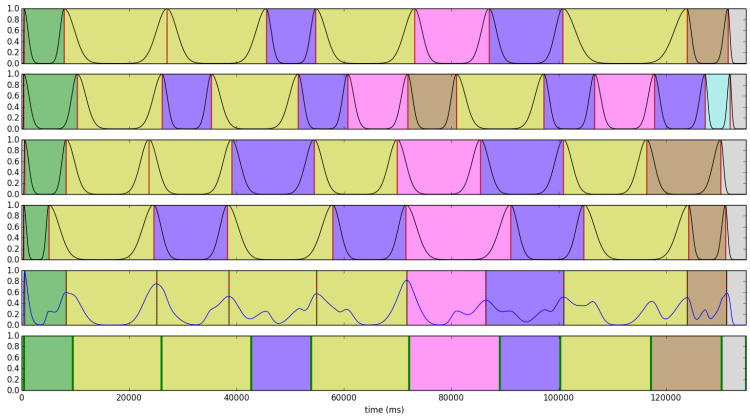


Figure 3.3: Example result for a query of the song *Can't Buy Me Love* by The Beatles with Method I. From top to bottom: neighbors' $k = 1$ to 4, result (blue line indicates the sum of all the Gaussians and vertical red lines the location of the calculated boundaries), and ground truth. Each colored rectangular corresponds to a different label.

The resulting time series (S_i^k, where $i = 1, \ldots, L_S$) will be then of length $L_S = \frac{length(query)}{100}$

9. For each of the computed subsegments S_i^k of each neighbor, the mode is calculated and assigned to a new array $R_i = Mode(S_i^k)$, so that e.g. $R_1 = Mode(S_1^1, S_1^2, S_1^3)$ for $k = 3$ (see Figure 3.4). If a mode is not found, the subsegment label of the first neighbor is considered.

10. The mode is again computed over all subsegments of the resulting array R_i between the boundaries calculated in step 7.

11. The result will be an array R of $\tilde{x}_k + 1$ labels that combined with the boundaries location from step 7 will define the final annotation.

In Figure 3.4 a simple example with $k = 3$ neighbors is shown. Figure 3.4a shows the sub-segmentation of all the neighbors' labels.

3.3.3 Method II

As opposed to Method I, where the labels are determined from the location of the boundaries, Method II uses directly the information of the labels in order to infer the locations of the boundaries. The entire process can be outlined as follows:

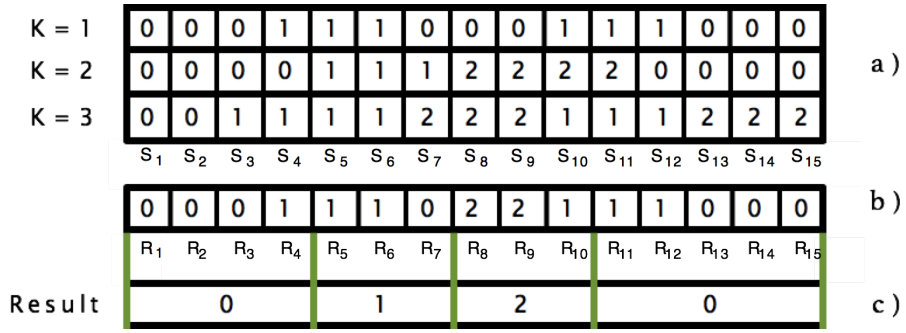


Figure 3.4: Example of the label assignment for Method I. (a) The labels are segmented for each neighbor (b) the mode for all neighbors at each subsegment is computed and (c) the mode for all sub-segments within the resulting boundaries (green) is computed.

1. Method 0 (without step 5).

2. The boundaries' locations are converted to miliseconds to maintain the resolution throughout the process, as operations with integers will be required.

3. As in Method I, every label of each neighbor is segmented into 100-ms subsegments, resulting in an array S^k of length $L_S = \frac{length(query)}{100}$.

4. The mode of all neighbors at each sub-segment S_i^k is computed and assigned to an array R (see Figure 3.7), so that e.g $R_1 = Mode(S_1^1, S_1^2, S_1^3)$ for $k = 3$.

5. The algorithm looks then through R and considers a boundary every time it finds a new label. This system as it is would create a considerable amount of oversegmentation, as there will be small sections throughout the entire length of the result.

6. To avoid oversegmentation a dilation algorithm is used. The algorithm starts from the section with the lowest amount of sub-segments and expands the value on its left if the number of same-label sub-segments is greater on the left or expands the value to its right otherwise. This process iterates until the number of boundaries of the result corresponds to the median number of boundaries of all neighbors. The result of this process can be seen in Figure 3.6.

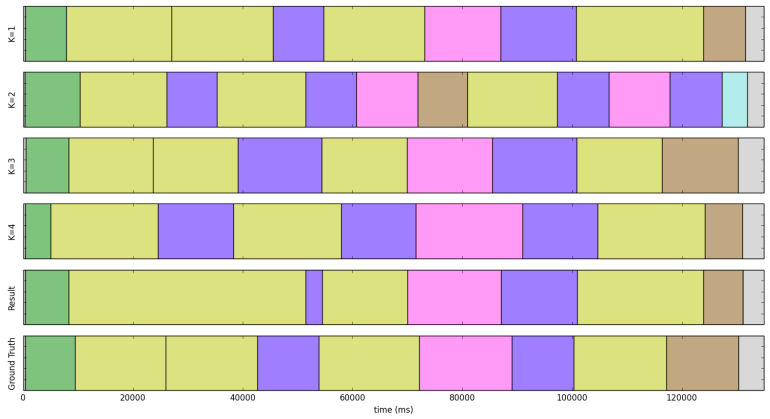


Figure 3.5: Example result for a query of the song *Can't Buy Me Love* by The Beatles with Method II. From top to bottom: neighbors $k = 1$ to 5, resulting annotations and ground truth.

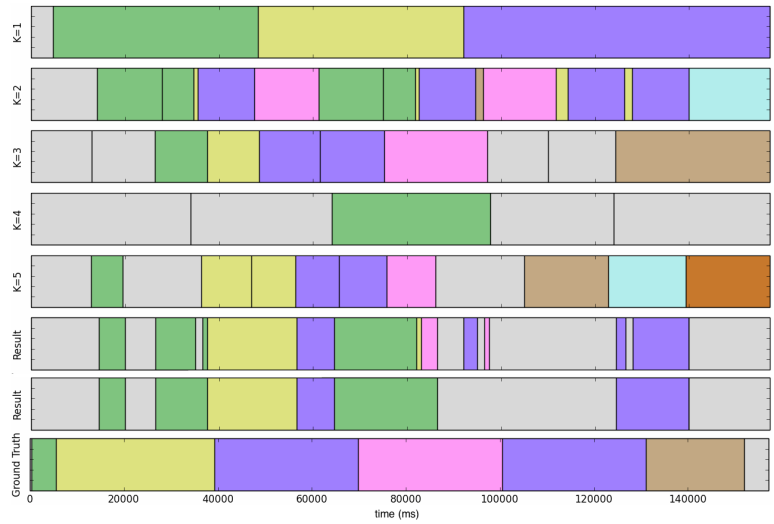


Figure 3.6: Example result for a query of the song: *Mr Moonlight* by The Beatles. From top to bottom: neighbors $k = 1$ to 5, result without oversegmentation correction, result with oversegmentation correction and ground truth.

3.3.4 Method III

The third and last method consists on a early fusion method, as opposed to methods I and II, which are late fusion. Early fusion methods perform the fusion of

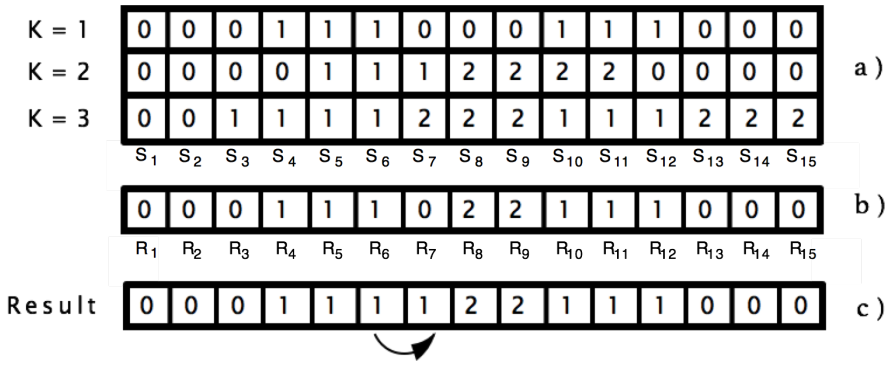


Figure 3.7: Example of the process for Method II. (a) All neighbors' labels are sub-segmented, (b) the mode for all neighbors at each sub-segment is calculated and (c) the oversegmentation is reduced by expanding from the left, since the number of same-label segments is greater than on the right.

the neighbors with respect of their features instead of the information about the boundaries locations and labels. The process can be summarized in the following steps and an example is shown in Figure 3.8:

1. From the list of the retrieved k neighbors, the arrays with the content-based information are obtained (i.e., the structure features) along with the annotations of each neighbor.

2. Each one of the arrays is then converted into the 100x100 structure feature matrix (see Figure 3.8, $k = 1$ to 5). The annotations are rescaled to match the duration of the query.

3. The mean value of all structure features columnwise is computed, yielding a new matrix where every column contains the mean of that column for every neighbor's structure features.

4. From the averaged structure features matrix, a novelty curve is determined, following the procedure detailed in [44]: the Euclidean distance between two consecutive points in the time series of structure features is computed and then normalized. The resulting novelty curve will have then a length of 100 samples (coinciding with one dimension of the structure feature matrix).

5. A simple peak detection algorithm is finally used to detect the boundaries where the changes in structure occur. A sample is considered a peak if it is above a certain threshold $\delta = 0.05$ and corresponds to the global maximum of a window of length $\lambda = 6\,s$, as defined in [44].

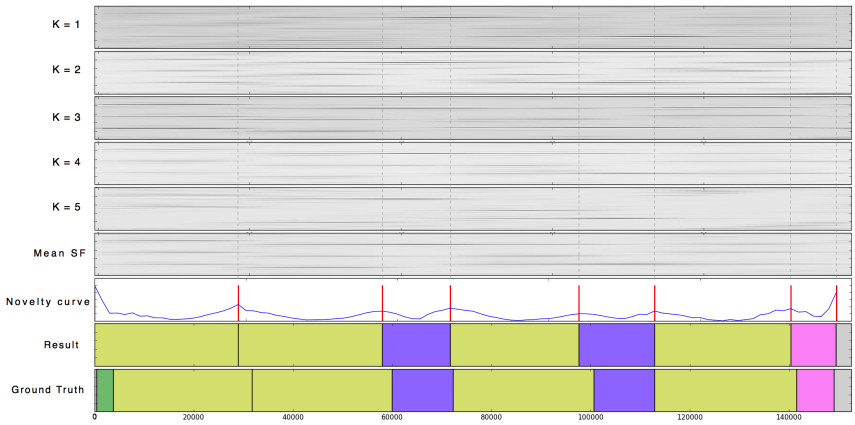


Figure 3.8: Example result for a query of the song: *Hold Me Tight* by The Beatles. From top to bottom: neighbors' $k = 1$ to 5 structure features, average structure feature matrix, novelty curve (blue) and detected location of the boundaries (red), resulting annotations and ground truth.

6. To compensate the delay introduced by the delay coordinates embedding process, the location of the peaks is offset by $\frac{(m-1)-\tau}{2}$ ($m = 3\,s$ and $\tau = 1\,s$ as in [44]).

7. The labels are subsequently assigned following the same procedure as in Method I, i.e, with the annotation information of all neighbors.

Chapter 4

Results and Discussion

In this section, the results of the evaluation are presented. As anticipated in Chapter 2, the evaluation is performed by computing the precision, recall and f-measure to evaluate the boundaries location accuracy, and the same metrics with the addition of the measure of over- and under-segmentation to evaluate the accuracy of the labels. The first part of this section 4.1 presents the overall results obtained after evaluating the algorithm and the second part 4.2 analyzes particular cases that can provide useful insights about the problems encountered. In Section 4.3 some insights about the structure of the datasets are provided.

4.1 Global Results

To evaluate the overall performance, the evaluation algorithm was run with two lists: a queries list and a candidates list. Each dataset was thus evaluated against itself and against all datasets. Table 4.1 summarizes the different configurations. Note that for the IRISA version of the RWC-P dataset, only the boundaries locations are evaluated, as the labels are not available.

To provide context as to where the baselines for the evaluation lie, five different random baseline methods are divised, which are summarized in the following:

1. System 1: Placing a boundary every 3 seconds.
2. System 2: Placing the average song boundary length averaged over entire dataset.
3. System 3: Average boundary length of entire dataset.
4. System 4: 10 boundaries randomly placed.
5. System 5: Average number of boundaries randomly placed.

QUERY-CANDIDATES	Description
MAZ-MAZ	Mazurka Project Dataset vs Mazurka Project Dataset
MAZ-ALL	Mazurka Project Dataset vs All Datasets
BTUT-BTUT	Beatles TUT Dataset vs Beatles TUT Dataset
BTUT-ALL	Beatles TUT Dataset vs All Datasets
BQMUL-BQMUL	Beatles QMUL Dataset vd Beatles QMUL Dataset
BQMUL-ALL	Beatles QMUL Dataset vs All Datasets
RWCPA-RWCPA	RWC-P AIST Dataset vs RWC-P AIST Dataset
RWCPA-ALL	RWC-P AIST Dataset vs All Datasets
RWCPI-RWCPI	RWC-P IRISA Dataset vs RWC.P IRISA
RWCPI-ALL	RWC-P IRISA Dataset vs All Datasets

Table 4.1: Summary of query and candidates lists.

Since the evaluation algorithm will consider a boundary threshold of 0.5 or 3 seconds, as mentioned in Chapter 2, placing a boundary every 3 seconds, although the precision will not be high, it will yield a recall of 100%, establishing a fairly high baseline. This baseline will then be used to compare against the results obtained. In addition, the results obtained in other state-of-the-art studies that reported the best accuracies in the last years are included. Finally, as demonstrated in [44], two different human annotators can disagree on the location and number of boundaries for a particular song. Hence we provide a measure of *human accuracy*. We refer to [44], therefore, to obtain a reference of human performance, denoted as Human in tables 4.2 to 4.5.

In tables 4.2 through 4.6, the results of the evaluation with a value of $k = 5$ neighbors are presented. The best result for each method is highlighted in bold. Figure 4.6 shows a summary for several values of k. As it can be seen, with datasets RWC-P and Beatles, the evaluation does not yield a high accuracy, with f-measure values ranging from $F_B = 0.428$ to $F_B = 0.615$ for boundary detection and from $F_L = 0.437$ to $F_L = 0.529$ for labels. These values can be considered in the limits of the baseline, which is $F_B = 0.516$ and $F_L = 0.514$ for BTUT, $F_B = 0.530$ and $F_L = 0.491$ for BQMUL, $F_B = 0.578$ and $F_L = 0.447$ for RWCPA, and $F_B = 0.577$ for RWCPI. However, with the Mazurkas dataset, f-measures for boundaries and labels increse up to $F_B = 0.848$ and $F_L = 0.843$ respectively, which, although for this paricular dataset human performance is not available, are very close to similar values of said performance in other datasets ($F_B = 0.89$ in RWC and $F_L = 0.876$ in Beatles for example).

Another aspect that can be observed is how the accuracy in general does not experience a considerable increase from the simplest method (Method 0, obtaining annotation information from only the nearest neighbor) to more complex ones, such as methods I, II and III. For example, using RWCP-A's dataset against all

others, the highest value is obtained with Method I, while using it against itself, Method 0 yields the best result, although the difference is hardly significant. This occurrence can be found as well in other datasets. Method III performs best with almost all datasets with the exception of Mazurkas where, while still obtaining a fairly high accuracy for labels, is still far below what others methods obtain.

These discrepancies in the results lead to believe that the problem does not lie in the neighbor's fusion stage, but rather in the preliminary stages of structure feature extraction or kNN. It is thus necessary to examine singular cases of particular songs in order to determine if an algorithm based on a supervised approach is indeed the most adequate for this task.

	Boundaries			Labels		
	P_B	R_B	F_B	P_L	R_L	F_L
Baseline	0.353 (0.083)	1.000 (0.000)	0.516 (0.092)	0.356 (0.1211)	1.000 (0.0000)	0.514 (0.1216)
Serrà et al. [44]	0.734	0.791	0.753	0.693	0.775	0.707
Peiszer [37]	0.515	0.824	0.617	0.611	0.623	0.597
Human	0.889	0.937	0.911	0.902	0.870	0.876
BTUT-BTUT						
M0	0.486 (0.143)	0.493 (0.161)	0.477 (0.121)	0.495 (0.1412)	0.480 (0.1383)	0.464 (0.0861)
MI	0.491 (0.138)	0.500 (0.170)	0.488 (0.138)	0.465 (0.1373)	0.556 (0.1488)	0.485 (0.0963)
MII	0.468 (0.132)	0.538 (0.157)	0.491 (0.125)	0.482 (0.1401)	0.493 (0.1197)	0.471 (0.0933)
MIII	0.544 (0.137)	0.526 (0.169)	**0.526 (0.135)**	0.448 (0.133)	0.589 (0.172)	**0.486 (0.098)**
BTUT-ALL						
M0	0.477 (0.190)	0.547 (0.226)	0.492 (0.182)	0.504 (0.1782)	0.560 (0.2018)	**0.498 (0.1375)**
MI	0.448 (0.167)	0.547 (0.194)	0.478 (0.154)	0.480 (0.1547)	0.554 (0.1821)	0.487 (0.1221)
MII	0.459 (0.180)	0.513 (0.197)	0.461 (0.156)	0.480 (0.1547)	0.524 (0.1799)	0.476 (0.1271)
MIII	0.538 (0.157)	0.522 (0.172)	**0.521 (0.146)**	0.469 (0.144)	0.584 (0.179)	0.497 (0.120)

Table 4.2: Evaluation results for all methods with the Beatles TUT's dataset (BTUT). Peiszer [37] results reported by Smith [46].

4.2 Case Study

Figures 4.1–4.3 show some particular query cases where some interesting results can be observed. Although the results in tables 4.2 and 4.3 indicate that complex methods as I, II or III do not necessarily correspond with an increase in performance compared to Method 0, in cases where the immediate nearest neighbors are not exactly a perfect match, the former methods do indeed help smooth out the results by including information from farther neighbors. This case is shown in Figure 4.1. In this case, the query of the song *Hold Me Tight* by The Beatles presents the structure $ABCCDCDCEF$, while two of the immediate neighbors ($k = 1$ and $k = 3$), although with a somewhat similar structure, do not represent an exact match. By including information from neighbors $k = 2$, 4 and 5 (*Things We Said Today*, *Tell Me What You See* and *You Won't See Me*, respectively, all

	Boundaries			Labels		
	P_B	R_B	F_B	P_L	R_L	F_L
Baseline	0.367 (0.098)	1.000 (0.000)	0.530 (0.107)	0.333 (0.1014)	1.000 (0.0000)	0.491 (0.1083)
Serrà et al. [44]	0.753	0.816	0.774	0.681	0.787	0.711
Mauch et al. [26]	†	†	†	0.610	0.770	0.660
Human	0.937	0.889	0.911	0.87	0.902	0.876
BQMUL-BQMUL						
M0	0.576 (0.193)	0.566 (0.196)	0.561 (0.177)	0.543 (0.1635)	0.540 (0.1502)	**0.525 (0.1259)**
MI	0.569 (0.169)	0.557 (0.171)	0.555 (0.155)	0.484 (0.1541)	0.606 (0.1483)	0.519 (0.1186)
MII	0.525 (0.168)	0.589 (0.158)	0.544 (0.144)	0.496 (0.1580)	0.528 (0.1356)	0.497 (0.1226)
MIII	0.649 (0.164)	0.601 (0.156)	**0.617 (0.145)**	0.479 (0.156)	0.640 (0.166)	0.529 (0.127)
BQMUL-ALL						
M0	0.521 (0.191)	0.590 (0.207)	0.535 (0.169)	0.495 (0.1715)	0.565 (0.1969)	0.495 (0.1300)
MI	0.481 (0.178)	0.582 (0.178)	0.511 (0.151)	0.475 (0.1507)	0.562 (0.1736)	0.490 (0.1164)
MII	0.501 (0.192)	0.559 (0.183)	0.503 (0.152)	0.473 (0.1531)	0.529 (0.1733)	0.477 (0.1230)
MIII	0.605 (0.152)	0.587 (0.166)	**0.587 (0.139)**	0.465 (0.142)	0.597 (0.173)	**0.502 (0.118)**

Table 4.3: Evaluation results for all methods with the Beatles QMUL's C4DM dataset (BQMUL). Mauch et al. results reported by Weiss and Bello [50]. † denotes data not reported in the original.

	Boundaries			Labels		
	P_B	R_B	F_B	P_L	R_L	F_L
Baseline	0.413 (0.098)	1.000 (0.000)	0.578 (0.094)	0.289 (0.0447)	1.000 (0.0000)	0.447 (0.0520)
Serrà et al. [44]	0.817	0.773	0.785	0.755	0.659	0.691
Kaiser et al. [19]	0.816	0.560	0.657	0.706	0.612	0.635
Human	0.921	0.891	0.899	*	*	*
RWCPA-RWCPA						
M0	0.477 (0.172)	0.519 (0.198)	**0.485 (0.159)**	0.532 (0.1245)	0.517 (0.1255)	**0.520 (0.1128)**
MI	0.470 (0.140)	0.495 (0.140)	0.476 (0.124)	0.477 (0.1082)	0.523 (0.0921)	0.493 (0.0857)
MII	0.503 (0.152)	0.449 (0.124)	0.465 (0.117)	0.481 (0.1104)	0.488 (0.0839)	0.480 (0.0861)
MIII	0.614 (0.133)	0.329 (0.119)	0.419 (0.119)	0.415 (0.099)	0.595 (0.135)	0.478 (0.083)
RWCPA-ALL						
M0	0.532 (0.158)	0.404 (0.156)	0.442 (0.129)	0.464 (0.1091)	0.464 (0.1091)	**0.470 (0.0897)**
MI	0.526 (0.138)	0.401 (0.137)	**0.445 (0.122)**	0.432 (0.0951)	0.501 (0.1414)	0.446 (0.0735)
MII	0.528 (0.165)	0.386 (0.129)	0.428 (0.111)	0.436 (0.0923)	0.471 (0.1328)	0.437 (0.0714)
MIII	0.613 (0.150)	0.355 (0.121)	0.443 (0.126)	0.399 (0.083)	0.546 (0.143)	0.447 (0.068)

Table 4.4: Evaluation results for all methods with the RWC-P AIST dataset (RCWPA). * denotes data not reported due to labels annotations not available in IRISA version. † denotes data not reported in the original. Kaiser et al. results reported in MIREX 2012.

by The Beatles) the final result obtained is a perfect match, since those neighbors do have an $ABCCDCDCEF$ structure.

Figure 4.2 shows a particular problem in Method II. Due to the design of the algorithm, successive identical repeating sections are not detected and are instead presented as part of the same segment. This leads to undersegmentation and decrease in the recall, as shown in Table 4.6 for the Mazurka dataset. Because of the generally low accuracy with other datasets, this issue is only noticeable in the

	Boundaries		
	P_B	R_B	F_B
Baseline	0.410 (0.0788)	1.000 (0.0000)	0.577 (0.0793)
Serrà et al. [44]	0.827	0.782	0.797
Rocha et al. [40]	0.700	0.664	0.673
Human	0.891	0.921	0.899
RWCPI-RWCPI			
M0	0.450 (0.157)	0.568 (0.176)	0.491 (0.144)
MI	0.441 (0.1247	0.545 (0.1096	0.480 (0.104)
MII	0.472 (0.134)	0.506 (0.111)	0.479 (0.106)
MIII	0.651 (0.140)	0.429 (0.116)	**0.509 (0.110)**
RWCPI-ALL			
M0	0.496 (0.159)	0.457 (0.136)	0.458 (0.113)
MI	0.495 (0.1294)	0.463 (0.1238)	0.468 (0.104)
MII	0.513 (0.161)	0.459 (0.120)	0.468 (0.104)
MIII	0.644 (0.144)	0.455 (0.117)	**0.528 (0.117)**

Table 4.5: Evaluation results for all methods with the RWC-P IRISA dataset (RWCPI). Label evaluation not reported due to label annotations not available in IRISA version.

	Boundaries			Labels		
	P_B	R_B	F_B	P_L	R_L	F_L
Baseline	0.441 (0.1832)	1.000 (0.0000)	0.592 (0.1623)	0.349 (0.1402)	1.000 (0.0000)	0.502 (0.1490)
Serrà et al. [44]	0.715	0.719	0.699	0.758	0.716	0.719
Human	*	*	*	*	*	*
MAZ-MAZ						
M0	0.835 (0.218)	0.837 (0.219)	0.834 (0.218)	0.840 (0.1419)	0.838 (0.1480)	0.838 (0.1480)
MI	0.850 (0.197)	0.851 (0.197)	**0.848 (0.196)**	0.840 (0.1480)	0.844 (0.1324)	0.838 (0.1386)
MII	0.859 (0.198)	0.671 (0.193)	0.745 (0.185)	0.839 (0.1414)	0.855 (0.1354)	**0.843 (0.1374)**
MIII	0.688 (0.201)	0.550 (0.209)	0.599 (0.189)	0.672 (0.168)	0.773 (0.137)	0.710 (0.143)
MAZ-ALL						
M0	0.835 (0.218)	0.837 (0.219)	0.833 (0.218)	0.839 (0.1448)	0.839 (0.1448)	0.835 (0.1463)
MI	0.847 (0.199)	0.852 (0.195)	**0.847 (0.196)**	0.841 (0.1457)	0.841 (0.1377)	0.837 (0.1403)
MII	0.858 (0.199)	0.676 (0.188)	0.748 (0.182)	0.840 (0.1398)	0.850 (0.1417)	**0.842 (0.1400)**
MIII	0.687 (0.202)	0.552 (0.209)	0.600 (0.189)	0.672 (0.168)	0.769 (0.140)	0.708 (0.144)

Table 4.6: Evaluation results for all methods with the Mazurka Project dataset (MAZ). † denotes data not reported in the original. * denotes human performance not available due to only one set of annotations being available.

Mazurka dataset.

Figure 4.3 shows the case where all retrieved neighbors are an exact match, and where Methods I and II do not stand out from Method 0, since the information retrieved are simply duplicates and therefore obtaining the nearest neighbor would be enough.

Figure 4.4 shows an example of a case where none of the three methods correctly segment and annotate the piece. While the three of them use exactly the same $k = 5$ neighbors, the different characteristics of each method segment and annotate

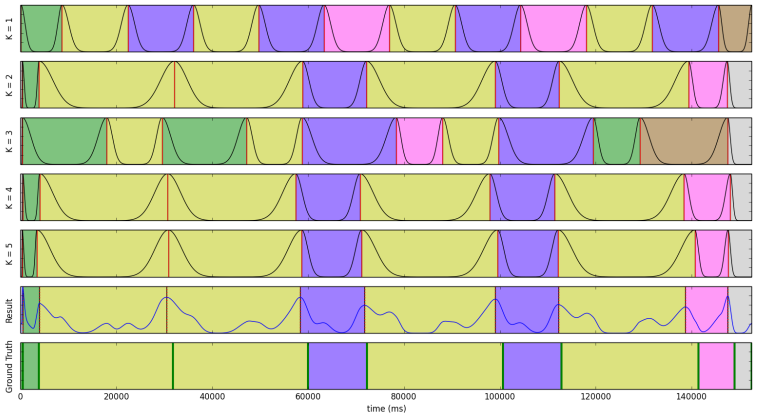


Figure 4.1: Results for the query of the song *Hold Me Tight* by The Beatles with Method I. From top to bottom: k= 1-5 nearest neighbors, final result and ground truth. Different colors indicate different labels. Red vertical lines in the result represent the resulting boundaries and green vertical lines represent the ground truth boundaries.

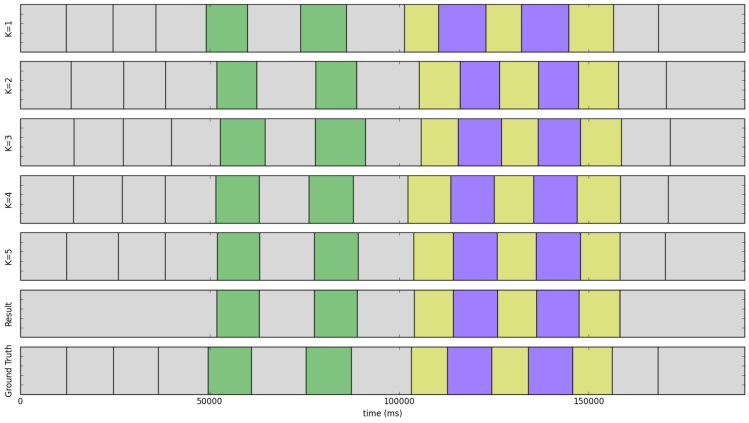


Figure 4.2: Results for the query of the Chopin's piece *Op007 No2* performed by Bacha, using Method II. From top to bottom: k= 1-5 nearest neighbors, final result and ground truth. Different colors indicate different labels. Succesive identical sections are not detected.

the piece differently. Method I fails at assessing almost all boundaries and labels. Method II clearly oversegments the piece except where there are consecutive labels of the same type, due to the issue commented above. Method III, in this case, undersegments the song while the location of the boundaries are almost all correct.

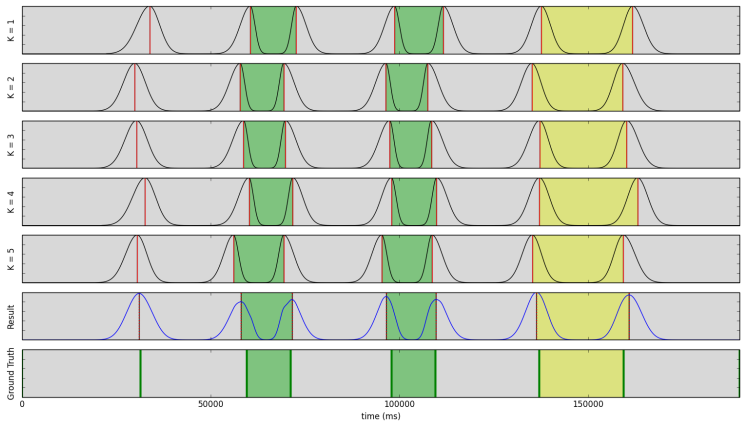


Figure 4.3: Results for the query of the Chopin's piece *Op006 No1* performed by Ashkenazy, using Method I. From top to bottom: k= 1-5 nearest neighbors, final result and ground truth. Different colors indicate different labels. Red vertical lines in the result represent the resulting boundaries and green vertical lines represent the ground truth boundaries.

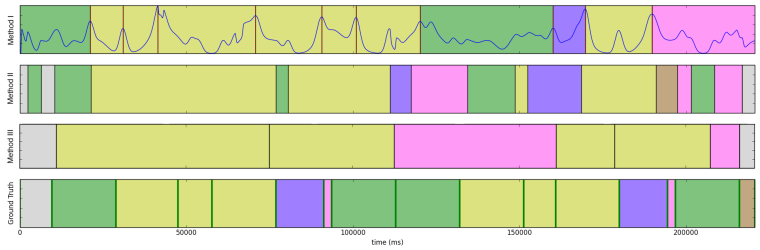


Figure 4.4: Example result of a segmentation and annotation of the piece RM-P002.wav of the RWC-P collection where none of the three methods perform as intended.

4.3 Dataset Analysis

With the aim of providing more insight into the results presented in the last sections, two tests were performed over the datasets employed. In light of the overall results obtained, where only one dataset yields high enough accuracy to be considered relevant, a study of the composition of said dataset is appropriate. The first of the two tests aims at establishing the number of songs with exactly the same, or highly similar, structure that there needs to be in a dataset to achieve an acceptable accuracy. Taking advantage of the fact that the Mazurka dataset is composed precisely of several interpretations of the same piece that are easily

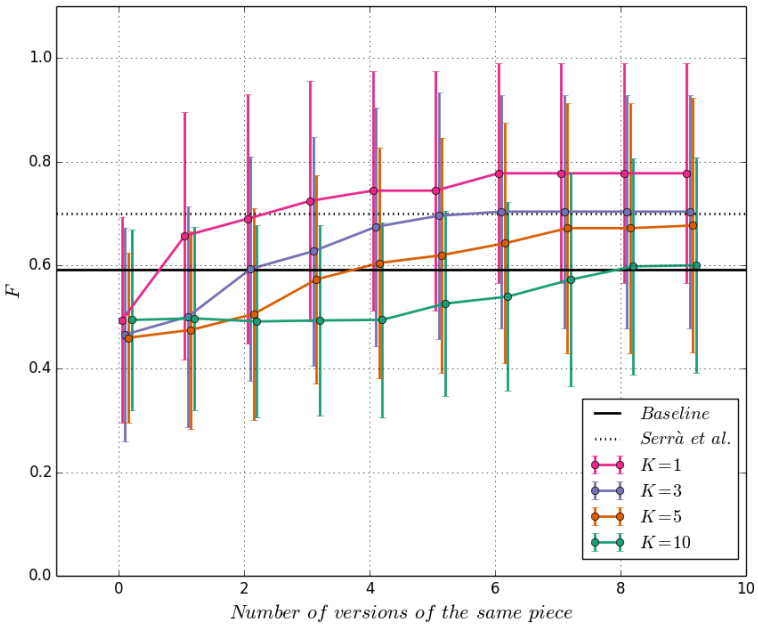
identifiable by their filename, we analyze the impact of said versions by computing the evaluation metrics over different samplings of the dataset, each one with increasing number of songs with similar structure. As the computational cost of doing the process for every piece in the dataset would be too high, the test was performed over 25 randomly chosen pieces. The results of this test are presented in Figure 4.5.

It can be observed that when there are no similar pieces of the queried song in the dataset, the f-measure is comparable for all values of k (neighbors) and below the baseline, while as the number of songs with a similar (or the same) structure increases, the lower the k, the higher the f-measure is, plateauing at a value of around 7 versions.

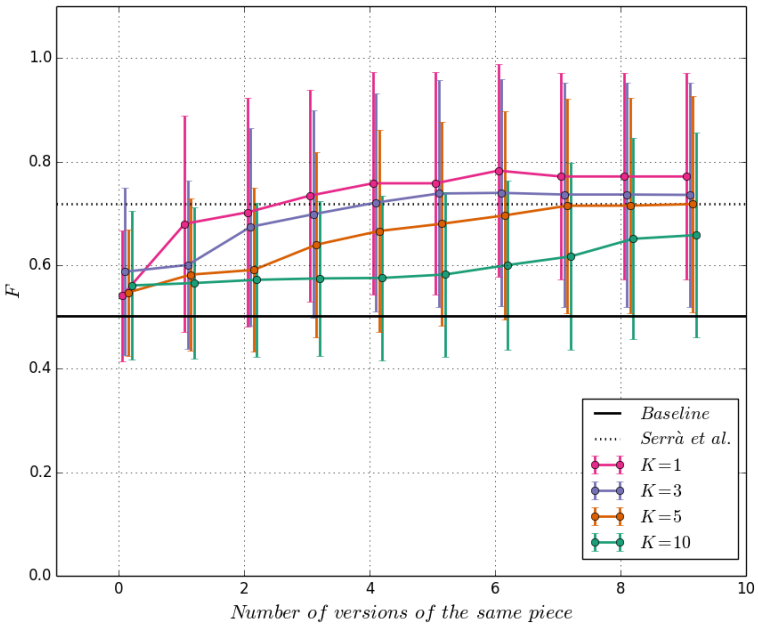
In order to further study this phenomenon, an additional test was devised to establish what the theoretical 'ceiling' could be if the retrieval system worked flawlessly. For each dataset, the evaluation metrics were computed for all songs using the rest of the same dataset as candidates and then selecting only the highest f-measure obtained for each query. The mean and standard deviation of all maximum values were then calculated to obtain what can be called our theoretical ceiling. The aim is that the song retrieved with the highest f-measure will be the most similar in terms of structure to the query, thus, selecting only those with the highest value, we can obtain a limit for our 'perfect' system. In the case of the Mazurkas' dataset, this experiment was not performed, as evaluating the algorithm with the Mazurka's dataset already yields high results. Furthermore, the composition of the Mazurka's dataset is already fairly established since, as already mentioned, it consists of several similar-structured versions of each song.

The results of this tests can be seen in Table 4.7 and in Figure 4.6 (green solid line). They show that, even if the retrieval of the most similar structures worked with a 100% accuracy, the dataset itself would limit the potential results considerably, proving that there is in fact not enough repetition in the datasets for the system to work as intented. At the same time, the f-values obtained in the tests, as can be seen are lower than other state-of-the-art approaches (see sections 4.2-4.6), but still much higher than the random baselines and the best results obtained by only retrieving $k = 1$ neighbor (Method 0). This indicates that the retrieval system is indeed a bottleneck in the considered approach. Particularly, we can hypothesize that the issue lies in the fact that the structure features do not correlate well with the manual annotations available, that is, they are not a completely accurate representation of the structure of a song.

Extrapolating from the test performed over the Mazurka dataset we can also hypothesize that there would need to be at least around 4-6 songs with very similar structure when retrieving only the nearest neighbor to achieve results comparable to the state of the art. When using more than one neighbor, the number of songs



(a)



(b)

Figure 4.5: Impact analysis of repetition of pieces in the Mazurka dataset for (a) boundaries and (b) labels f-measure.

needed is obviously on par with the number of neighbors.

One of the likely issues an algorithm such as this can find is a dataset with neighbor similarity not uniform enough. In this case, combining the Mazurkas dataset, which contains several performances of the same piece by different interpreters, with Beatles dataset results in a highly imbalanced data structure, as the Beatles dataset does not only not contain many similar-structured songs, but The Beatles' songs are well known for their sometimes irregular structures, far from common rock or pop forms. In light of these results, it is likely that the method of early fusion works better with datasets that do not in fact contain a large amount of songs with similar structure, as is the case of RWC-P and Beatles, than methods of late fusion.

The impact of the number of neighbors chosen is shown in Figure 4.6, where it can be seen how it is considerably independent from variance in k for methods I and II, while for method III the f-measure decreases inversely with k. It is also notable the difference in the datasets when observing these ocurrences, as only in the RWC-P dataset does F increase with k (method II).

	Ceiling test		Method 0		Baseline		Serrà et al.	
	F_B	F_L	F_B	F_L	F_B	F_L	F_B	F_L
BQMUL	0.763 (0.102)	0.665 (0.104)	0.561 (0.177)	0.525 (0.126	0.530 (0.107)	0.491 (0.108)	0.774	0.711
BTUT	0.784 (0.089)	0.666 (0.108)	0.477 (0.121)	0.464 (0.086)	0.516 (0.092)	0.514 (0.122)	0.753	0.707
RWCA	0.731 (0.098)	0.647 (0.094)	0.485 (0.159)	0.520 (0.113)	0.578 (0.094)	0.447 (0.052)	0.785	0.691
RWCI	0.710 (0.086)	-	0.491 (0.144)	-	0.577 (0.079)	-	0.797	-

Table 4.7: F-measures means and standard deviations (when available) of the theoretical ceiling, retrieval method, baselines and state of the art for all versions of the Beatles and RWC-P datasets.

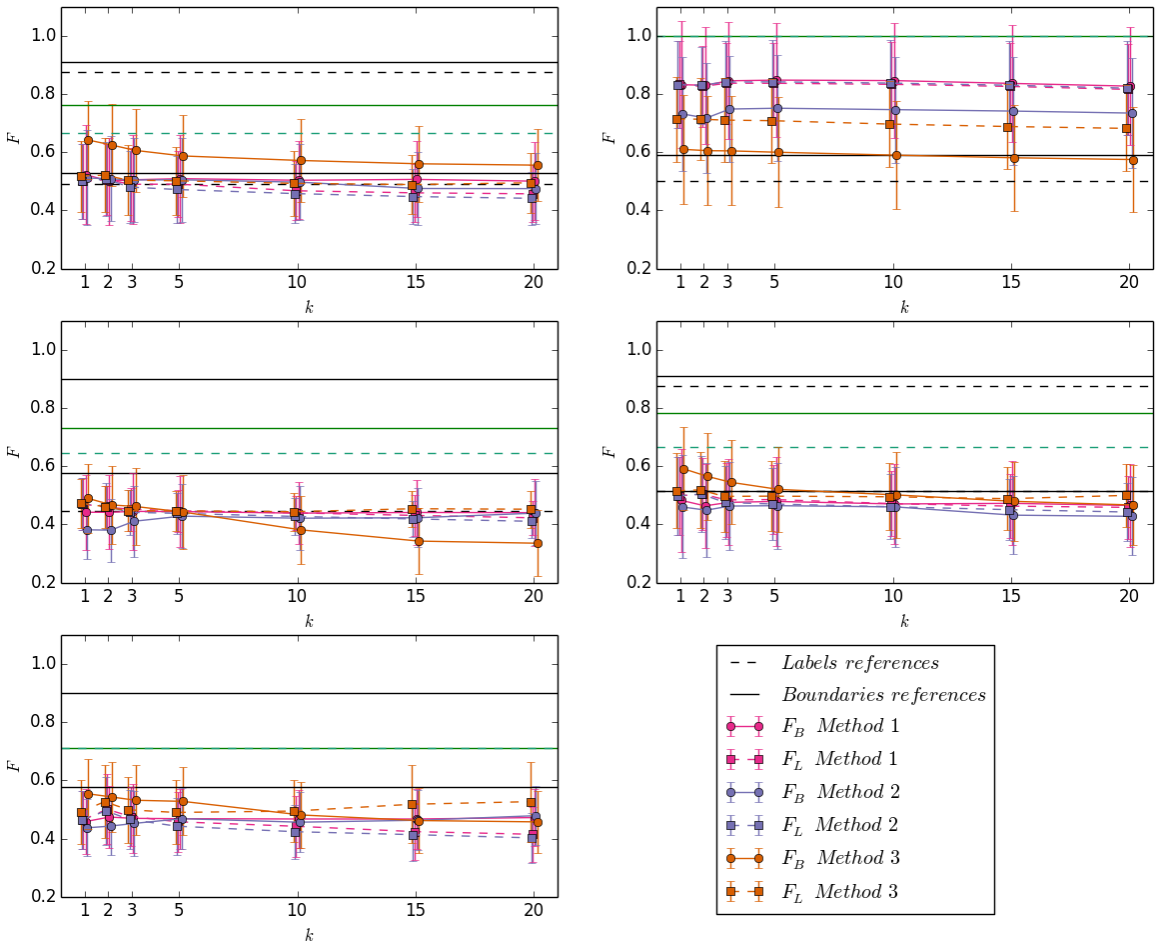


Figure 4.6: F-measures means and standard deviations for boundaries and labels in all datasets with $k = 1, 2, 3, 5, 10, 15$ and 20. From left to right and top to bottom: BQMUL, MAZ, RWCA, BTUT and RWCI. Solid (boundaries) and dashed (labels) lines represent references (baselines and human performance when available). Green solid line represents theoretical ceilings.

Chapter 5

Conclusions and Future Work

The aim of this study is to explore a novel approach to the task of automatic segmentation and annotation of musical structure by presenting a supervised case-based approach that can take advantage of the information about music structure that is already available to us, being either from dataset compilations or in the form of metadata. The first step is to extract the structural information of the different pieces by obtaining what it is called *structure features* in order to later make use of those features and integrate them into a retrieval system to obtain the most similar songs in terms of structure. Three different algorithms are then proposed and tested over three different datasets with some degree of success. The first algorithm uses late fusion of annotated data from neighbors retrieved with a kNN system to annotate new songs by using Gaussian windows to integrate annotations from all neighbors. The second algorithm uses the information about the labeled sections to infer the location of the boundaries of each section. The third algorithm uses early fusion of the structural features of the neighbors to obtain a new structure feature that represents the new song's structure. We believe the presented algorithms, although simple in nature and easy to implement, have the potential to outperform most state-of-the-art approaches assuming a large and varied enough collection of annotataed data is available. However, due to the lack of this collection, we have not been able to totally prove this claim.

5.1 Contributions

Following the goals defined in Section 1, a series of contributions have been made in the course of this study:

1. A review of state-of-the-art approaches to the task of automatic segmentation and annotation of musical structure, as well as evaluation measures and corpora used.

2. The introduction of a conceptually novel supervised approach in contrast to standard approaches so far. Three different variants to tackle this task are explored.

3. Extensive testing of the algorithms over three of the most used datasets in automatic segmentation and annotation of music structure.

4. An exhaustive report of the results found in order to provide an accessible way to make comparisons with other algorithms. In addition, a study of particular cases of interest is included, along with some considerations about the composition of the datasets used in order to better understand the results obtained.

5. Source code available to the public [1].

5.2 Future Work

Based on the results and findings gathered during this study, we provide some interesting follow-up lines of reasearch for this topic:

1. Experimentation with other datasets, that were at the time unavailable to us or that could be available in the future, with special emphasis on dataset that include repeated musical structures.

2. Experiment with alternative machine learning schemas to include in the retrieval stage of the process. Emphasis should be placed on schemas that facilitate the fusion of different annotations.

3. Incorporate new methods of neighbor fusion to the current scheme.

4. Consider other potentially improved forms of structure features (e.g. *latent structural repetition* descriptors introduced by McFee and Ellis [27]).

5. Consider other features besides chroma ones, as discussed in Chapter 2. For example rhythm or timbre.

6. Exploit existing unsupervised approaches to generate a basic annotation and then refine them by fusion in a supervised manner.

[1] http://github.com/gherrero/music-structure

Bibliography

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

[2] M.A. Bartsch and G.H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7(1):96–104, 2005.

[3] F. Bimbot, O. Le Blouch, G. Sargent, and E. Vincent. Decomposition Into Autonomous and Comparable blocks : A Structural Description of Music Pieces. In *Proceedings of International Society for Music Information Retrieval (ISMIR)*, pages 190–194, 2010.

[4] W. Chai. Semantic Segmentation and Summarization of Music. *IEEE Signal Processing Magazine*, 23(2):124–132, 2006.

[5] H. Cheng, Y. Yang, Y. Lin, and H. H. Chen. Multimodal structure segmentation and analysis of music using audio and textual information. *IEEE International Symposium on Circuits and Systems*, pages 1677–1680, 2009.

[6] R.B. Dannenberg and M. Goto. Music structure analysis from acoustic signals. In David Havelock, Sonoko Kuwano, and Michael Vorländer, editors, *Handbook of Signal Processing in Acoustics*, pages 305–331. Springer New York, 2008.

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

[8] J.P. Eckmann, S.O. Kamphorst, and D. Ruelle. Recurrence Plots of Dynamical Systems . *Europhysics Letters*, 4(9):973–977, 1987.

[9] A. Eronen. Chorus Detection with Combined use of MFCC and Chroma Features and Image Processing Filters. In *Proceedings of International Conference on Digital Audio Effects (DAFX)*, pages 1–8, 2007.

[10] J. Foote. Visualizing music and audio using self-similarity. *In Proceedings of the ACM International Conference on Multimedia (ACM-MM)*, pages 77–80, 1999.

[11] J. Foote. Automatic audio segmentation using a measure of audio novelty. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, volume 1, pages 452–455, 2000.

[12] J. Foote and M. L. Cooper. Media Segmentation using Self-Similarity Decomposition. In Minerva M. Yeung, Rainer W. Lienhart, and Chung-Sheng Li, editors, *Proceedings of Electronic Imaging Conference*, pages 167–175, 2003.

[13] T. Fujishima. Realtime chord recognition of musical sound: A system using common lisp music. In *Proceedings of International Computer Music Conference (ICMC)*, pages 464–467, 1999.

[14] E. Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, University Pompeu Fabra, Barcelona, Spain, 2006, http://www.mtg.upf.edu/node/472, Accessed August 2014.

[15] M. Goto. A chorus section detection method for musical audio signals and its application to a music listening station. *IEEE Transactions on Audio, Speech and Language Processing*, 14(5):1783–1794, 2006.

[16] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka. RWC Music Database: Popular, Classical and Jazz Music Databases. In *Proceedings of International Society for Music Information Retrieval (ISMIR)*, pages 287–288, 2002.

[17] P. Grosche, J. Serrà, M. Müller, and J. Ll. Arcos. Structure-Based Audio Fingerprinting for Music Retrieval. In *Proceedings of International Society for Music Information Retrieval Conference (ISMIR)*, pages 55–60, 2012.

[18] K. Jensen. Multiple Scale Music Segmentation Using Rhythm, Timbre, and Harmony. *EURASIP Journal on Advances in Signal Processing*, (1):11, 2007.

[19] F. Kaiser, T. Sikora, and G. Peeters. MIREX 2012 - Music Structural Segmentation Task: IRCAMSTRUCTURE SUBMISSION. Technical report, 2012, http://www.music-ir.org/mirex/abstracts/2013/KSP3.pdf, Accessed August, 2014.

[20] M. Levy and M. Sandler. Structural segmentation of musical audio by constrained clustering. *IEEE Transactions on Audio, Speech and Language Processing*, 16(2):318–326, 2008.

[21] B. Logan. Mel Frequency Cepstral Coefficients for Music Modeling. In *Proceedings of International Society for Music Information Retrieval (ISMIR)*, 2000.

[22] B. Logan and S. Chu. Music summarization using key phrases. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, pages 749–752, 2000.

[23] H.M. Lukashevich. Towards Quantitative Measures of Evaluating Song Segmentation. *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pages 375–380, 2008.

[24] C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. An Introduction to Information Retrieval. Cambridge University Press, 2008.

[25] M. Mauch, C. Cannam, and M. Davies. OMRAS2 metadata project 2009. In *Proceedings of International Society for Music Information Retrieval (ISMIR)*, 2009, Demo paper.

[26] M. Mauch, K. Noland, and S. Dixon. Using Musical Structure to Enhance Automatic Chord Transcription. *In Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pages 231–236, 2009.

[27] B. McFee and D.P.W. Ellis. Learning to Segment Songs with Ordinal Linear Discriminant Analysis. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5197–5201, 2014.

[28] M. Müller and M. Clausen. Transposition-Invariant Self-Similarity Matrices. In *Proceedings of International Society for Music Information Retrieval (ISMIR)*, pages 47–50, 2007.

[29] B.S. Ong. *Towards automatic music structural analysis: identifying characteristic within-song excerpts in popular music*. Phd thesis, Universitat Pompeu Fabra, 2005, http://www.mtg.upf.edu/node/2225, Accessed August, 2014.

[30] Elias Pampalk. *Computational models of music similarity and their application in music information retrieval*. Phd thesis, Vienna University of Technology, 2006.

[31] J. Paulus and A. Klapuri. Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. *IEEE Transactions on Audio, Speech and Language Processing*, 17(6):1159–1170, 2009.

[32] J. Paulus, M. Müller, and A. Klapuri. Audio-Based Music Structure Analysis. In *Proceedings of International Society for Music Information Retrieval (ISMIR)*, pages 625–636, 2010.

[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[34] G. Peeters. Deriving musical structures from signal analysis for music audio summary generation: "sequence" and "state" approach. In UffeKock Wiil, editor, *Computer Music Modeling and Retrieval*, volume 2771 of *Lecture Notes in Computer Science*, pages 143–166. Springer Berlin Heidelberg, 2004.

[35] G. Peeters. Sequence Representation of Music Structure Using Higher-Order Similarity Matrix and Maximum-Likelihood Approach. In *Proceedings of International Society for Music Information Retrieval (ISMIR)*, pages 35–40, 2007.

[36] G. Peeters. MIREX 2010 - Music Structure Segmentation Task: IRCAMSUMMARY submission. Technical report, 2010, http://www.music-ir.org/mirex/abstracts/2010/GP7.pdf, Accessed August 2014.

[37] E. Peiszer. *Automatic audio segmentation: Segment boundary and structure detection in popular music.* PhD thesis, Vienna University of Technology, Austria, 2007.

[38] E. Peiszer, T. Lidy, and A. Rauber. Automatic audio segmentation: Segment boundary and structure detection in popular music. In *Proceedings of Learning Semantics of Audio Signals (LSAS)*, pages 45–59, 2008.

[39] A. Rauber, E. Pampalk, and D. Merkl. Using psycho-acoustic models and self-organizing maps to create a hierarchical structuring of music by sound similarity. In *Proceedings of International Society of Music Information Retrieval (ISMIR)*, pages 71–80, 2002.

[40] B. Rocha, N. Bogaards, and A. Honingh. Segmentation and Timbre Similarity in Electronic Dance Music. In *Proceedings of the Sound and Music Computing Conference (SMC)*, number 2, pages 754–761, 2013.

[41] C.S. Sapp. Comparative Analysis of Multiple Musical Performances. In *Proceedings of International Society for Music Information Retrieval Conference (ISMIR)*, pages 2–5, 2007.

[42] J. Serrà, E. Gómez, and P. Herrera. *Audio cover song identification and similarity: background, approaches, evaluation, and beyond*, volume 274 of *Studies in Computational Intelligence*, chapter 14, pages 307–332. Springer-Verlag Berlin / Heidelberg, 2010.

[43] J. Serrà, E. Gómez, P. Herrera, and X. Serra. Chroma binary similarity and local alignment applied to cover song identification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(6):1138–1151, Aug 2008.

[44] J. Serrà, M. Müller, P. Grosche, and J. Arcos. Unsupervised music structure annotation by time series structure features and segment similarity. *IEEE Transactions on Multimedia*, 16(5):1229 – 1240, 2014.

[45] J. Serrà, X. Serra, and R.G. Andrzejak. Cross recurrence quantification for cover song identification. *New Journal of Physics*, 11(9):093017, 2009.

[46] J.B.L. Smith. *A comparison and evaluation of approaches to the automatic formal analysis of musical audio*. Master thesis, McGill University, 2010.

[47] J.B.L. Smith, J.A. Burgoyne, and I. Fujinaga. Design and creation of a large-scale database of structural annotations. In *Proceedings of International Society for Music Information Retrieval Conference (ISMIR)*, pages 555–560, 2011.

[48] D. Turnbull, G.R.G. Lanckriet, E. Pampalk, and M. Goto. A Supervised Approach for Detecting Boundaries in Music Using Difference Features and Boosting. In *Proceedings of International Society for Music Information Retrieval (ISMIR)*, pages 2–5, 2007.

[49] K. Ullrich, J. Schlüter, and T. Grill. Boundary Detection in Music Structure Analysis using Convolutional Neural Networks. In *Proceedings of International Society for Music Information Retrieval (ISMIR)*, 2014, In Press.

[50] R.J. Weiss and J.P. Bello. Unsupervised Discovery of Temporal Structure in Music. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1240–1251, 2011.

[51] R. Xu and D.C. Wunsch. *Clustering*. IEEE Series on Computational Intelligence. Wiley, 2009.