PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE

ESCUELA DE INGENIERIA

COMPUTER SYSTEM FOR HARMONIC TRANSCRIPTION OF JAZZ MUSIC

GABRIEL E. DURÁN

Thesis submitted to the Office of Research and Graduate Studies in partial fulfillment of the requirements for the Degree of Master of Science in Engineering

Advisor:

PATRICIO DE LA CUADRA

Santiago de Chile, (August, 2020)

PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA

COMPUTER SYSTEM FOR HARMONIC TRANSCRIPTION OF JAZZ MUSIC

GABRIEL E. DURÁN

Members of the Committee:

PATRICIO DE LA CUADRA

DOMINGO MERY

SEBASTIÁN FINGERHUTH

JOSÉ LUIS ALMAZÁN

Thesis submitted to the Office of Research and Graduate Studies in partial fulfillment of the requirements for the Degree of Master of Science in Engineering

Santiago de Chile, (August, 2020)

A mi Puchunca, mi compañera de vida.

Agradecimientos

En primer lugar, quisiera agradecer a mi querido profesor Patricio de la Cuadra no solamente por su valioso aporte como guía de mi investigación, sino que por despertar y potenciar mi interés por la búsqueda de conocimiento. Ha sido un largo camino, desde que el año 2013 fui su alumno en el ramo "Acústica Musical" y me maravillé ante la posibilidad combinar la ciencia con la música, hasta ahora en que dedico mi magíster al estudio de estas dos disciplinas.

También quiero agradecer al profesor Domingo Mery, quién me orientó y apoyó en el desarrollo de una gran parte de mi investigación, además de entregar una gran calidez como persona. Estoy enormemente agracedido con el profesor Rodrigo Cádiz por brindarme acceso a su computador con GPU, fue fundamental para desarrollar mis etapas experimentales. Quisiera también agradecer al profesor Sebastián Fingerhuth por su dedicación y sus aportes para mejorar este documento de tesis.

Agradezco enormemente a Christian, querido amigo y colega de ruta en este magíster. También a la gente de la rama estudiantil de la IEEE y a todos mis amigos del departamento de ingeniería eléctrica por su calidez y sentido de comunidad, hacen que todo sea más ameno.

Particularmente quiero dar la gracias a mis padres y hermanos por su apoyo y su enorme cariño, no solamente durante el desarrollo de mi investigación, sino que particularmente durante toda esta larga etapa universitaria.

Por útlimo, quiero agradacer a mi Javi querida por estar siempre a mi lado, apoyarme incondicionalmente y escuchar mis interminables divagaciones sobre este magíster. Sin ti este trabajo no sería posible, ni yo estaría donde estoy. Agradezco infinitamente la familia que hemos formado, junto con nuestra Pecory.

Contents

List of Tables

List of Figures

RESUMEN

La transcripción automática de acordes es un tema ampliamente estudiado por la comunidad científica, que presenta un problema complejo puesto que es un concepto musical estrechamente relacionado a la percepción humana. En los últimos años se han logrado sustanciales avances utilizando métodos de aprendizaje profundo, sin embargo la mayoría de éstos estudios se han desarrollado sobre géneros musicales como el pop o el rock, no siendo generalizables a la música de jazz, cuya naturaleza fuertemente improvisada causa que no se cumplan muchos de los supuestos sobre cómo se percibe la armonía.

En esta investigación se presenta un sistema computacional para realizar transcripción de acordes en canciones de jazz, abordando los desafíos específicos que éstas presentan e incorporando al análisis elementos musicales propios del género. Esto se logra realizando un estudio detallado de subproblemas, que nos permite tener un mejor entendimiento de cómo se debe adaptar los métodos estándar sobre este género musical, para así integrar diferentes elementos musicales en el análisis. Este sistema es capaz de obtener una representación de acordes del tipo "hoja guía" a partir del audio.

El primer subproblema abordado es la detección de cambios de notas del bajo, puesto que éstos están estrechamente relacionadas con cambios de acordes. Luego, se estudia la sincronización musical para lograr alinear los diferentes segmentos repetidos de una canción y analizar el contenido musical de cada uno de ellos. Por último se implementa el sistema de transcripción de acordes utilizando una red neuronal convolucional recurrente, que logra obtener en conjunto la secuencia de acordes y el pulso.

Esta metodología de transcripción de acordes logra un 71% de precisión bajo la métrica de desempeño MIREX, que es mayor que los sistemas estado del arte para la base de datos JAAH. Adicionalmente, tiene la ventaja que logra transcribir acordes y detectar el pulso en conjunto.

ABSTRACT

Automatic chord transcription is a subject widely studied by the scientific community, that is complex because chords are a musical concept closely related to human perception. In recent years, significant improvements have been achieved using deep learning methods. Nevertheless, most of these works were developed on genres like pop and rock, and cannot be extended to jazz music, whose nature is strongly improvised and most of the assumptions about how harmony is understood are not complied.

This research presents a computational system that transcribes chords from jazz songs, addressing the specific challenges they present and takes into consideration musical elements inherent to this genre. This is achieved by a comprehensive analysis of some sub-tasks, obtaining insights on how standard analysis methods should be adapted to this genre, to integrate different musical elements to the analysis. The system is able to obtain a "lead sheet" chord representation from the audio.

The first sub-task addressed is bass line onsets detection, because they are closely related with chord transitions. Later, music synchronization is studied to align repeated segments and analyze the musical content of each. Finally, the chord transcription system is implemented using a convolutional recurrent neural network, that obtains the chord sequence and the beat positions.

This chord transcription system obtains 71% of correct predictions under the MIREX evaluation metric, which is higher that state-of-the-art systems for the JAAH database. Additionally, it has the benefit that chord transcription and beat detection are performed jointly.

1 INTRODUCTION TO THE THESIS

1.1 Music Information Retrieval

Mechanical vibrations that propagates on a medium as acoustic waves are known as sound, which is represented as a one dimensional signal. It can be perceived by humans through the auditory system and processed by the brain, retrieving a large amount of information which is mainly extracted from the sound's intensity and frequency content. This information is encoded on a single dimension signal, but can be highly sophisticated, like spoken language which can be effortlessly interpreted by humans.

Music is a special class of audio signal, which is an art form expressed through sound, that can be understood and decoded by humans. It is closely related to subjective notions, like emotions and cultural traditions, but also to perceptual phenomena. Therefore, music is a high-level concept that contains information with different levels of complexity, which can be interpreted as layers of abstraction. The lowest level is a single dimension signal composed of quasi-periodic signals that are perceived as temporal events, pitches, noise, etc. On a higher level, these sounds are interpreted as musical notes or rhythmic events, that on a musical context form melodies, chords and rhythmic patterns. On the highest level, music can be associated with emotions, moods, cultural identity, etc. Music, at some level, can be understood naturally by humans, but deeper is a complex domain that is only understandable by experts or trained people.

Musical Information Retrieval (MIR) is an interdisciplinary field that aims to extract musically meaningful information mostly from audio, involving domains like psychoacoustics, signal processing, statistics, machine learning, musicology, etc. This analysis takes a scientific approach and it is normally performed with the aid of computational systems. Expecting that a computer can understand music as people do might be too ambitious, but addressing sub-tasks is reasonable. Some of the common MIR sub-tasks are onset detection, beat tracking, chord transcription, music structural segmentation, pitch detection, among others.

This research is inherently interdisciplinary and is mostly based on domains like electrical engineering, computer science and music. In particular, it uses signal processing and machine learning methods to analyze jazz music, and retrieve high-level information from the audio signal. It is required that the reader posses knowledge on signal processing and machine learning to achieve a complete understanding of this work, nevertheless only basic musical knowledge is required. Some key musical concepts that will be employed are notes, pitch, chord, measure, beat, tonality and harmony. If the reader is not familiarized with these concepts, an enlightening

explanation is presented by Schmidt-Jones (2018). Advanced and specific concepts related to jazz music will be explained in Sections 1.3-1.4 and in the Appendix A.

1.2 Understanding Automatic Chord Transcription

Pitch related tasks are not trivial because pitch is closely related to perception and psychoacoustics, while studies on musical harmony, which can be understood as a subset of pitch related tasks, are very complex because they involve a high degree of musical knowledge. Chords occur when multiple notes are played at the same time, but harmony is more abstract because is closely related to chords, but is not necessarily manifested explicitly. It can be understood as an abstract concept that is manifested through chord progressions.

Chord transcription is an ambiguous task, in which transcriptions performed by different trained musicians might not be consistent (Humphrey and Bello, 2015), because there is a subjective interpretation process involved. Nevertheless, this challenging task has been widely addressed in the MIR community and many Automatic Chord Transcription (ACT) systems have been proposed in the last two decades.

Some variations of chord sequence prediction systems can be found in the literature with different names: recognition, identification, detection, transcription or estimation. The general objective is similar, but there are substantial differences as stated by Humphrey and Bello (2015):

> The analysis here motivates the notion that this is not merely a matter of semantics, but actually a subtle distinction indicative of two slightly different problems being addressed. Chord transcription is an abstract task related to functional analysis, taking into consideration high-level concepts such as long term musical structure, repetition, segmentation or key. Chord recognition, on the other hand, is quite literal, and is closely related to polyphonic pitch detection.

This highlights that transcription is a more complex concept closely related to higher level analysis. One of the practical aspects of ACT is that the analysis cannot be static, because the transcription of a particular chord depends strongly on its own musical content, but also depends on the observation of previous and posterior chords. Thus, the analysis is not only performed on a single temporal frame, but over a sequence, which is consistent with the way humans perceive chords and harmony.

Harmony and chords can be interpreted at different levels of depth, from a simple description that only considers their essential components, to an analysis that describes the role of each note being played. At its simplest form, a chord can be classified as major/minor depending on the interval between the root note an

its third grade. Incorporating the interval to the fifth grade, the diminished and augmented classes are added. All notes being played on a musical segment relate somehow to the root and might be added to a chord label as tensions, leading to a more detailed description. ACT systems are trained over limited chord dictionaries, or chord vocabularies, which should be defined according to a specific musical task, because they partially describe the system's complexity.

1.3 Jazz and Improvised Music

Most of western tonal music consists of different sections that play a structural role in a musical piece. These sections can be all different between them, or some can be repeated along the song and others might be variations of previous segments. In either way, these sequences are structured, containing repetitive patterns which can be expressed as rhythmic repetitions, chord sequences, musical motives, etc.

In classical music and most of pop music, a song is defined on a very specific way, most of time written, and should be performed as it was conceived, leaving very little or no room for improvisation and spontaneous variations. For example, when an orchestra performs Beethoven's Fifth Symphony, every musician should play exactly what is written by the composer guided by the director's instructions and there is little freedom in terms of how the melody and the harmony should be expressed. Other genres, like pop music, behave similarly: Hey Jude by The Beatles was written with specific arrangements so every musician should perform their specific lines, the singer should not change excessively the melody or the lyrics, and the piano player must not play other chords that the ones written. Even if the musician can introduce some decorations to their lines, the global structure is fixed and every time that the verse is played, it is expected to sound unequivocally similar the other ones.

On the other hand, jazz includes elements of improvisation and performers can modify their musical lines freely. This spontaneity is reflected on the extensive instrumental solos, or the fact that the exact way that a repetition of a section should be performed is not predefined rigidly. The way musicians play is highly correlated to what is being played by the others, generating a spontaneous performance that cannot be played identically twice. It can be understood as a dialog where the musicians can respond, follow or complement what is being played by the soloist.

Nevertheless, this kind of music is not completely free and amorphous because there is an structure that is usually followed, which is defined by a fixed number of measures and a general harmonic structure that acts as the "backbone" of the song, known as *chorus*. The term chorus is not formally established, but is colloquially used by jazz musicians and will be used in this thesis, because it is a key concept of the research. This fundamental structural segment is repeated multiple times, but

at each iteration is different from the others, as can be seen in Fig.1, creating an organized musical piece that incorporates improvisation elements.

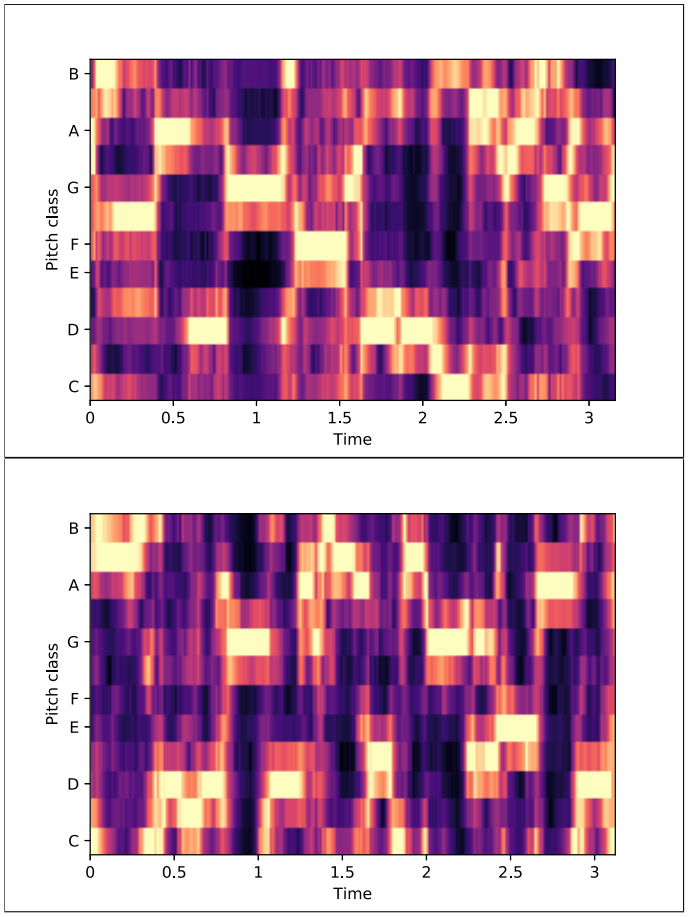


Figure 1: Chromagram of two sections of John Coltrane's Giant Steps, they do not contain the same notes. The first four measures of the "head", or chorus containing the melody, are shown in (top). Figure in (bottom) shows the same four measures of the chorus, but on the sax solo.

The chorus does not define the chords as a rigid group of notes, because they can be modified by performers, instead it defines the chords as a basic form. Pachet et al. (2013) states that these chords are rarely played as they are written. In jazz music the chorus of a song is usually represented in a *lead sheet* which contains the "essence" of a tune, as stated by Pachet et al. (2013), namely the melody and chord progression aligned to each measure, as can be seen in Fig.2 obtained from "The Real Book". A more detailed description of how jazz music works is provided in the Appendix A.

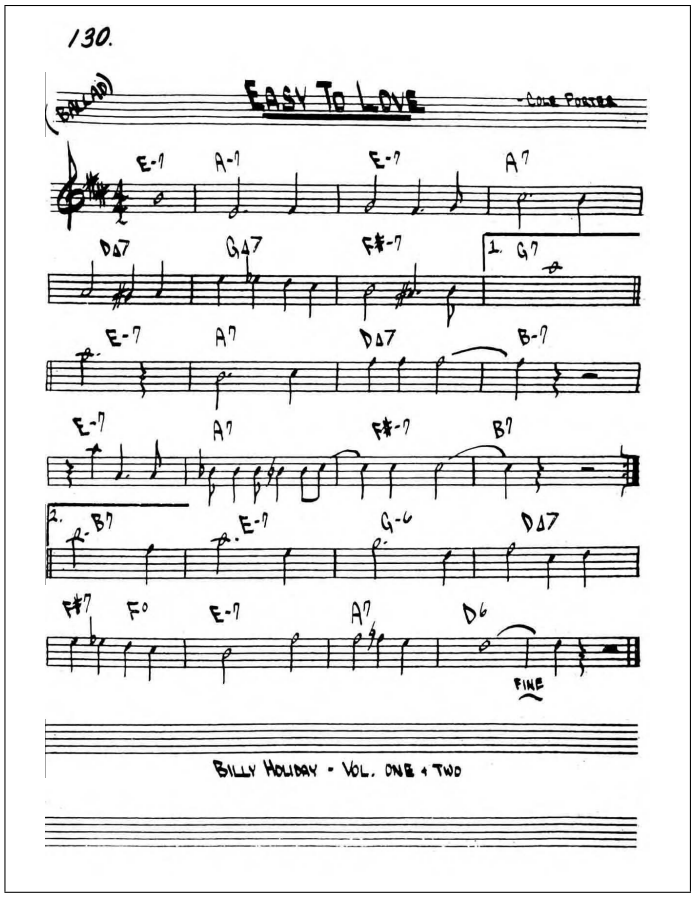


Figure 2: Lead sheet of Cole Porter's Easy To Love from "The Real Book".

1.4 Jazz Music and ACT

Jazz music present elements inherent to its nature that are not shared by other genres, suggesting that some MIR tasks should be addressed differently. There is little research addressing topics related to this musical genre (Duan and Pardo, 2011; Dittmar et al., 2018; Pachet et al., 2013). In the case of ACT, there is no research focused on jazz music to our knowledge, besides the work presented by Eremenko (2018) where is demonstrated that state-of-the art systems achieve dramatically lower results than on pop-rock songs.

From our point of view, current methods for chord transcription do not take into consideration the improvised nature of jazz and therefore cannot retrieve chords correctly. There are theoretical and conceptual musical aspects that should be resolved before performing chord transcription. What chord dictionary should be considered? When can be considered that a chord has changed? If chords are not equal between all choruses, which of them should be transcribed?

In this work, we propose a chord transcription system focused on jazz music, taking into account the way these songs are structured and how harmony is expressed. Therefore, this approach is only pertinent for songs with some constrains, that are explained later in this thesis.

Our hypothesis is that jazz choruses should not be considered as independent musical segments, but all of them should be taken into account to achieve a correct chord transcription. The chords played on a repetition will not be the same as the ones from another repetition, but all of them express some elements of the global (or essential) harmonic progression. Thus, retrieving the most important harmonic elements presented on each chorus will lead to a better chord transcription. This approach can be understood as a global analysis, that ignores the details in the harmonic and melodic content that are not shared between choruses.

This research aims to create a computational system able to transcribe the harmonic sequence of jazz songs, performing a global analysis that results on a lead sheet-like output. There are three sub-tasks that are essential before performing the chord transcription:

- Finding musically meaningful temporal events that reveal changes in the harmony. In this musical genre, the bass line carries important harmonic information and generally chord changes occur along with bass note onsets.
- Structural analysis should be performed to find boundaries between choruses.
- Music alignment, also called music synchronization, must be done between segmented choruses, because tempo fluctuates and different choruses do not

have the same duration, even if they have the same number of measures and beats.

The first and third sub-task are addressed in this thesis, each one being studied on a chapter. The second sub-task is particularly complex for this genre and a proper analysis can be, by itself, subject of analysis of a master's thesis. In order to remain focused on the chord transcription task, the structural analysis was set aside and the chorus boundaries are considered to be an input to this ACT system.

From a practical point of view, the proposed system is an aid to any person, with some musical knowledge, who wants to retrieve the chord sequence from a jazz song. It is required that the person can find the temporal boundaries between choruses, count the total measures of the song and know the musical metre, which most of amateur musicians and jazz enthusiasts can do. Even experienced musicians may benefit of this system, allowing them to automatize the process.

2 LITERATURE REVIEW

Automatic chord transcription is a highly active research topic that has been addressed since 1999, when the first system was proposed by Fujishima (1999). In the past years, it has been included among the Music Information Retrieval Exchange (MIREX[1]) challenges, which constantly encourages the development of new systems and draws interest from researchers. The topic is non-trivial and although state-of-the-art systems achieve better results since deep learning approaches were introduced (Pauwels et al., 2019; McFee and Bello, 2017), there are many issues that are far from being resolved.

It is commonly handled as a classification problem and essentially follows the standard pipeline: first, musically meaningful features are extracted and then each temporal frame is assigned to a single chord-class among a predefined chord dictionary (McVicar et al., 2014). As the inputs present temporal relations between frames, some preprocessing and postprocessing are included to provide temporal continuity and coherence. This is a very simplistic way to describe the problem, nevertheless most of the research follows this structure presenting new alternatives for one or more of these stages.

The following sections present a literature review exclusively related to chord transcription, nevertheless a focused revision of previous research related to onset detection and music synchronization can be found in Chapters 4 and 5 respectively.

2.1 Traditional Approaches

For many years the chromagram was the most used harmonic descriptor for ACT, starting by the innovative work presented by Fujishima (1999) where the chroma vectors were compared against binary chord templates and then pattern matching was performed. This approach is simple and effective on idealized cases, but does not consider the sequential nature of music into the analysis. This issue was later compensated by the work presented in Sheh and Ellis (2003), where a chromagram was used as input features to a sequence analysis model based on hidden markov models (HMM). Their method was inspired by the speech recognition research of that time and presents a probabilistic framework to model chords continuity as well as chord transitions and the parameters were estimated from the data using the expectation maximization algorithm. Later, some chromagram enhancements were presented by Bello and Pickens (2005), adding a pitch tuning stage and calculating beat synchronous features by averaging frames within beats. Also, HMM is used, but

[1] `https://www.music-ir.org/mirex/wiki/MIREX\_HOME`

expert knowledge was incorporated on parameters initialization, respecting musical meaningful relations between chords. In Harte et al. (2006) a harmonic detection system was proposed to find chord transitions, which was later complemented by Harte (2009) with a tuned chromagram to perform chord estimation.

New perspectives were addressed by other researchers, by either analyzing more specific features or trying to reach a wider retrieval of musical elements beyond chords on a single system. There has been a particular focus on the low frequency pitch content (Sumi et al., 2008; Ryynänen and Klapuri, 2008; Ni et al., 2012), which encompass the bass line notes and carries crucial information regarding chords. In particular, Sumi et al. (2008) incorporate bass pitch estimation on a probabilistic framework to recognize chords. Instead of using HMM, they use a hypothesis-search-based, which is not a usual approach in automatic chord recognition. Extending the analysis of bass line, Ryynänen and Klapuri (2008) integrate melody, bass and chord transcription using the same features, which not only tackles three problems together, but incorporates musicological knowledge into the features estimation and analysis. For the chord transcription, they input a novel pitch salience feature which describes the contribution of each musical note, and HMM for the sequence analysis. As an attempt to extract a global harmonic information, Ni et al. (2012), propose a system that jointly estimates chord, keys and bass notes on a fully data driven manner. This work incorporates many preprocessing stages to obtain an enhanced chromagram, including harmonic/percussive sound separation (and retaining the harmonic part), beat synchronization and loudness based normalization. The chromagram is split in two mutually exclusive bands and predictions are performed by a complex HMM architecture which considers hierarchical relations between key-chords-bass notes in the probabilistic modeling.

The system proposed by Mauch (2010) is one of the most ambitious and complex, as it incorporates a large amount of musical context to perform chord transcription and lay the groundwork to the well-known system Chordino[2]. It integrates chords, key, bass note, metric position and audio features on a single system, which inspired posterior works like Ni et al. (2012). The chroma features were enhanced and beat synchronized and the sequence modeling was performed by a dynamic bayesian network, which can be understood as a complex HMM model. They take advantage of repeating musical structural elements to enhance chord predictions, as explained also in Mauch et al. (2009), introducing a novel methodology for this task. Musical segments that have the same chord progression and the same number of beats were averaged into a single beat synchronous chroma feature, showing accuracy improvement and better consistency along chord sequences.

[2] `http://isophonics.org/nnls-chroma`

2.2 Deep Learning Based Methods

The term deep learning originally refers to methods based on artificial neural networks that stack a large number of hidden layers, but is usually applied to convolutional (CNN) and recurrent (RNN) neural networks regardless their number of layers. These methods have been incorporated to chord transcription on the feature extraction step, on the sequence analysis and more recently on both together.

On 2012, Humphrey and Bello (2012) stated that traditional methods were not showing much improvement on ACT over the years, thus new methods should be explored. In particular, they abandon the chroma-like features and input a CQT directly to a CNN mapping directly to a chord class probability, going through some 2D convolutional and fully connected layers. As these models have not being applied to ACT before, they explore variations on the convolutional kernel sizes and number of filter, showing that simpler architectures achieve state-of-the-art results without performing posterior sequential analysis.

From another perspective, Boulanger-Lewandowski et al. (2013) explore the use of RNN for ACT. The feature extraction stage inputs the spectrogram and performs dimensionality reduction using principal component analysis (PCA) whitening, to later feed a deep belief network that learns musical features. They use intermediate targets in the feature extraction stage to guide the low-level factors the network learns. The sequence analysis is performed by RNN that can learn temporal continuity, harmonic relations and temporal dynamics.

Some researches have focused in obtaining chroma-like representations learned directly from a STFT or CQT input. For example, in Humphrey et al. (2012) a data driven system based on CNN is used to obtain a 7-dimension audio feature that aims to represent the 6-dimensions tonnetz feature space plus a 1-dimensional element for no chord class. Similarly, Korzeniowski and Widmer (2016) propose a deep learning network based on idealized chroma extractor using artificial neural networks, under the hypothesis that normal chroma features are noisy. The idealized chromagram was compared, using a simple linear classifier to predict chords, to three "traditional" chroma configurations and have shown to perform better.

Recent works unify the feature extraction and sequence analysis stages on a single model, for example convolutional recurrent neural networks. Jiang et al. (2019) try to transcribe chords from a large vocabulary by performing a several layers CNN stage that is later fed to a bi-directional long short-term memory layer (BLSTM). The output chord decoding is obtained using conditional random fields to ensure temporal continuity between predictions. The large chord vocabulary is obtained by learning musically meaningful components of the chord, instead of independent chord classes. Under the same perspective, McFee and Bello (2017) proposed a CRNN architecture that uses an encoder-decoder architecture and incorporates intermediate targets to

guide the kind of information learned by the encoder. The convolutional stage is composed of a single 2D filter followed by 36 single-frame filters that span the whole frequency range and is then fed to a bidirectional gated recurrent unit (BGRU). Intermediate predictions are obtained from the encoded states, which consists on a 13D vector representing the chord root (includes no-chord), a 12D chroma-like representation of the active pitch classes for that chord and a 13D vector representing the chord bass note. These predictions are concatenated with the intermediate vector to produce a final chord label prediction using BGRU as decoder.

2.3 Evaluation of ACT systems

Automatic chord transcription is a complex task, which from a perception point of view is also subjective. There have been research efforts that do not propose new systems, but provide insights on how the problem is being addressed and the performance is being evaluated. These works review what was considered state-of-the-art to that date and some of the pointed issues might have been addressed in more recent researches. They provide an historical context of how ACT systems have evolved and what problems have been solved.

Cho et al. (2010) provide an exploration of how each stage affects the performance on ACT. They quantify the effect of filtering before and after performing pattern matching using varying parameters, showing that these steps have a significant impact on accuracy. With a correct choice of prefiltering and postfiltering parameters, they prove that complex pattern matching methods (in particular HMM) show only 5% increase compared to binary template matching. An extensive review of all the different methods used for ACT is provided by McVicar et al. (2014), which can be considered as a summary of all non-deep learning based methods. They list the feature extraction and sequence modelling methods, evaluation metrics and currently used datasets. They conclude that despite the steady increase on performance over the years, looking at annual MIREX submissions, most of the systems are evaluated on the same datasets and they tend to overfit. The most recent comprehensive review of ACT issues is made by Pauwels et al. (2019), where they point and comment the principal challenges to this task after 20 years of research. Most of them involve how musical properties are interpreted and transferred to computational systems, and the way that subjectivity and perceptual elements are evaluated. It is pointed out that there has been a transition from knowledge-based to data-driven approaches and that early modularized systems have been replaced by a single system since the arrival of deep learning which achieve better results, but can be difficult to interpret and to compare to other deep learning based systems.

The problem of how ACT is evaluated and how subjectivity associated to chords is tackled has also been addressed. The work presented by Humphrey and Bello

(2015) provides a deeper qualitative analysis on how ACT problems are evaluated. For example, they conclude that manual annotations, even if performed by expert musicians, should not be treated as absolute truth because they might differ from annotations made by other person. Also, they point out that comparison between annotated and predicted chords should be handled with care because most of the times different chord vocabularies are used and they have to be mapped o reduced to equivalent chords, leading to negative consequences if evaluation metrics are not chosen correctly. Similarly, an understanding on the effect of subjectivity is provided by Ni et al. (2013), where they quantify this effect using six different reference annotations on 20 songs. They show that the annotations made by an expert differ around 10% from a consensus annotation and that around 50% of the error made by state-of-the-art systems can be attributed to subjective differences. Focusing on how ACT systems should be evaluated, Pauwels and Peeters (2013) present a study on commonly used metrics and how they should be applied to express the capabilities of a given algorithm. Insights on evaluation are given regarding what chords should be compared, where the predicted sequences can be evaluated and how a score should be obtained, helping to standardize the way ACT systems are evaluated.

3 THESIS OVERVIEW

In this chapter it is explained how this thesis is structured and an overview of each chapter is presented. As explained in Section 1.4, there are two sub-task previous to chord transcription that are studied in detail in Chapters 4 and 5. The proposed chord transcription method is presented in Chapter 6, which did not achieved the same level of deepness in the analysis than the other two sub-tasks.

The sub-tasks are combined into a single system that transcribes chords of jazz songs from an audio input and a file with basic musical annotations, whose general work flow is shown in Fig.3. The output of the system is a lead sheet like chord representation, where the transcribed chords are aligned to beats and measures.

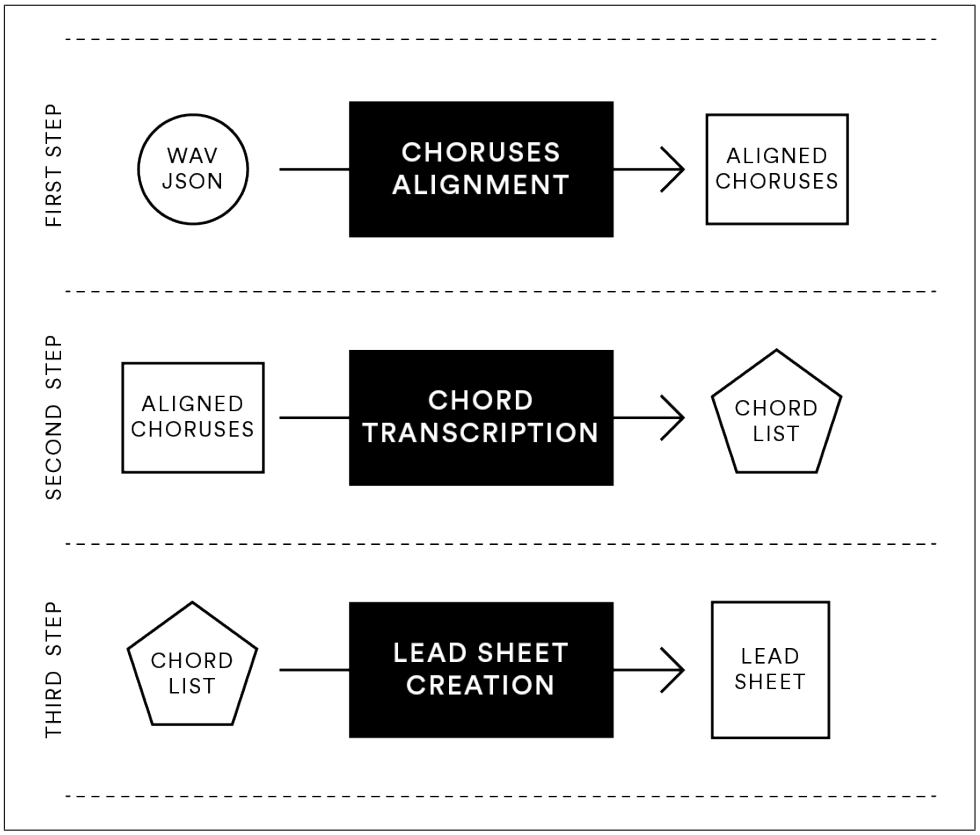


Figure 3: Work flow of the final ACT system, it consists of three sequential steps that maps an audio to a musical lead sheet representation.

3.1 Bass Line Onset Detection

In jazz music, usually chord transitions are likely to occur along with bass note onsets, which are closely related to beat onsets. A computer system able to find these temporal events and transcribe chords, can retrieve music information related to harmony similarly to humans, integrating two relevant elements to the chord transcription: What chord is currently being played and when does it changes?

Thus, finding the bass onsets prior to chord transcription might lead to a more accurate ACT system.

In Chapter 4, four bass line onset detection methods are compared and insights are obtained regarding how this sub-task should be addressed. There is little research in onset detection focused on bass lines, so the methods presented are adaptations of currently existing onset detection methods, two are signal processing based and two are deep learning based. This chapter has a conference paper structure because it will be published in the 2020 International Computer Music Conference (ICMC[3]). The version included in this thesis adds an introduction to the chapter, helping the reader to understand and contextualize the onset detection problem.

The final implementation of the ACT system merges onset detection and chord transcription into a single system, which is detailed in Chapter 6.

3.2 Choruses Alignment

Each jazz song contains two or more repetitions of the chorus and the aim of this research is to consider all of them to perform chord transcription. Tempo fluctuates differently inside each chorus, some segments were played slower or faster compared to the same segment in other choruses.

State-of-the-art music synchronization methods make assumptions about the musical content of the sequences that will be aligned, that are not shared to jazz music. Therefore, these methods should be adapted to jazz music choruses alignment.

In Chapter 5 is presented an exhaustive analysis of the standard methods for music synchronization, obtaining insights on how they should be applied to jazz music. This chapter has a conference paper structure because it was submitted to the 2020 International Society for Music Information Retrieval conference (ISMIR[4]) and is on hold to be accepted. To this chapter is added a introduction to music synchronization and its challenges, and a concise introduction to dynamic time warping (DTW), which is the standard algorithm for audio synchronization.

[3] `https://icmc2020.org/`

[4] `https://ismir.github.io/ISMIR2020/`

This stage is responsible of aligning all choruses according to their musical content, which are warped temporally resulting in equal length choruses. The method that reported best alignment results was used on the overall ACT system.

3.3 Chord Transcription

The chord transcription can be performed once all choruses are aligned, converting the predicted chord sequence into a lead sheet representation.

In Chapter 6 is explained the proposed methodology used for jazz songs ACT. The system's architecture, which is deep learning based, and all details of implementation are provided.

One of the most important elements in our proposed methodology, is the combination of all choruses of a song to the chord transcription system. To do so, different strategies are compared, obtaining insights on which is the more adequate for this genre. The lead sheet creation process is also performed, creating a friendly-user output that allows a better qualitative analysis.

4 BASS LINE ONSET DETECTION

An onset can be defined as the audio event generated by a fast rise on the signal's energy on a short amount of time. In computer music analysis that corresponds to the very beginning of a note produced by any instrument or singer. The definition is wide, if an instrument plays a note as *pianissimo* or *fortissimo* both are considered onsets regardless the amount of energy associated to each of them. Also notes produced by different instruments are considered onsets even if the *attack* produced by them is considerably different, for example a hi-hat and a flute.

Onset detection is a subject of MIR research that has been widely addressed, nevertheless there is little work on specific onset detection. Most state-of-the-art systems focus on every onset present on the audio, acting as universal onset detectors, so they are not able to discriminate the different type of onsets. Even if the theory behind onset detection systems may be similar between universal and specific onset detectors, the implementation and training (in the case of trainable methods) could be quite different and a detailed analysis is needed to comprehend the challenges presented by a specific kind of onset.

4.1 Theoretical Background

Many musical analysis tasks aim to extract high level information, such as chord estimation and musical structure segmentation. It has been shown that performing feature analysis based on musically relevant events leads to more accurate results, compared to analyses over fixed sized windows (Bello and Pickens, 2005; Harte et al., 2006; Hawthorne et al., 2017). In polyphonic music the bass line, or lower pitch notes, tend to include relevant information about a chord and the change of a bass note is often correlated to harmonic changes. Thus, it is reasonable to choose the occurrence of a bass note as a musically meaningful event of interest, for a mid-level step to a posterior higher level analysis. These are represented as a particular kind of note change, i.e the onsets produced by a specific instrument on a limited frequency range.

In this chapter, a comparison between four methods for bass note onset detection is presented. Two of them are signal processing based and the other two are deep learning based, thus trained methods. The first method performs a multipitch detection and detects when the lower pitch changes. The second one is based on a difference function over the low frequency range. The third uses a Convolutional Neural Network (CNN) as a detection function and the fourth a Recurrent Neural Network (RNN), also as a detection function.

Bass line onset detection systems usually analyze the low frequencies, in Hainsworth and Macleod (2001) the frequency range is limited to 200Hz and then energy peaks

are searched over the short time Fourier transform (STFT). A more sophisticated onset detector is used by Ryynänen and Klapuri (2007), where a multipitch algorithm is performed and only fundamental frequencies (F0) falling into a frequency range limited to 36-254Hz are considered and later an accent signal is used as onset detection function.

Deep learning based onset detectors, which are not limited to bass notes, generally train detection models and then perform a postprocessing stage by peak-picking or by analyzing the output sequence using Hidden Markov Models (HMM). In Eyben et al. (2010) a recurrent neural network (RNN) model is proposed, using a bidirectional long short-term memory (BLSTM) cell that inputs two STFT at different time resolutions filtered by a Mel scale based filterbank and their respective differences. Later a peak-picking stage is performed using an adaptative threshold. In Böck et al. (2012) a real-time RNN based approach is presented, using unidirectional LSTM cells, with an input similar to the one presented in Eyben et al. (2010) but using three different time resolutions and a Bark scale based filterbank. A peak-picking stage is used with a fixed threshold determined during the training stage. A different approach is used in Schlüter and Böck (2014), using a CNN model, also using as input three STFT at different time resolution filtered by a Mel scale filterbank. The output is smoothed and a peak- picking stage is performed with a fixed threshold.

4.2 Adapted methods

The bass line onset detection systems that will be presented are adaptations of existing methods that are not necessarily developed for onset detection.

4.2.1 Multipitch approach

A multipitch algorithm is performed following the method presented on Karjalainen and Tolonen (1999), where a time domain analysis is proposed over a modified autocorrelation function on two frequency bands. Multiple F0 candidates are found, but many correspond to multiples of the fundamental period, so a periodicity analysis is performed to remove false F0 candidates and up to 3 sources can be reliably chosen according to the authors.

According to Karjalainen and Tolonen (1999) a high pass filter should be included to reduce the effect of the high values of the autocorrelation function (ACF) coming from zero time lag. Unlike the base method, the signal is low-pass filtered with a cutoff frequency of 261Hz, corresponding to middle-C (60 on MIDI note) because we are only interested on a particular frequency range.

The ACF is obtained using a variation of the Wiener-Khinchin theorem expressed in Eq. 1, as proposed in Karjalainen and Tolonen (1999), because it is much faster due to the FFT computing efficiency. A Hamming window of 46.3ms is used with a hop of 11.6ms, because it was found to be appropriate based on experimentation.

$$ACF(\tau) = IDFT\{|DFT\{x(\tau)\}|^{\frac{2}{3}}\} \tag{1}$$

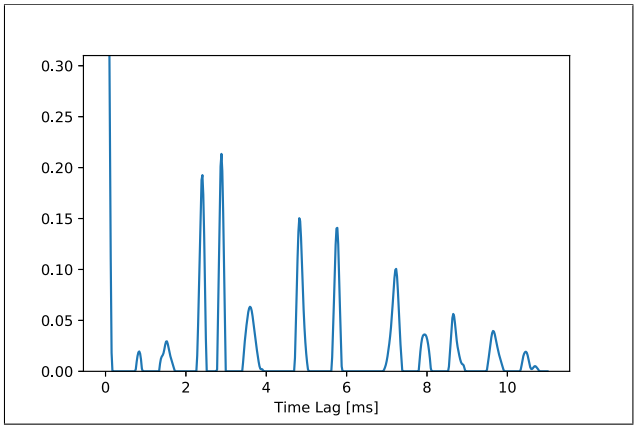
All the time lags with high periodicity are revealed, including the F0 candidates as well as all their multiples. To remove undesired candidates, the signal is iteratively expanded in time, by factors from 2 to 8, then subtracted to the original signal and the negative components are clipped to zero, as shown in Fig.4. Improving the peak picking stage proposed on the original method, the ACF peaks are chosen by quadratic interpolation of the local maximums and the ones higher than a fixed threshold t_1, are considered as fundamental frequencies.

The proposed method for onset detection first constructs a chromagram from the found peaks and their respective amplitudes. The effect of local incorrect F0 detection is reduced by convolving each row of the chromagram with a Gaussian kernel of σ=3 . The distance function $d(n)$ shown in Eq 2 is applied to reveal note change, with n representing the index of each frame and i each note index in the chromagram. As $d(n)$ is affected by onsets and offsets, it is clipped to zero and every peak higher than a fixed threshold is considered as an onset due to bass line change.

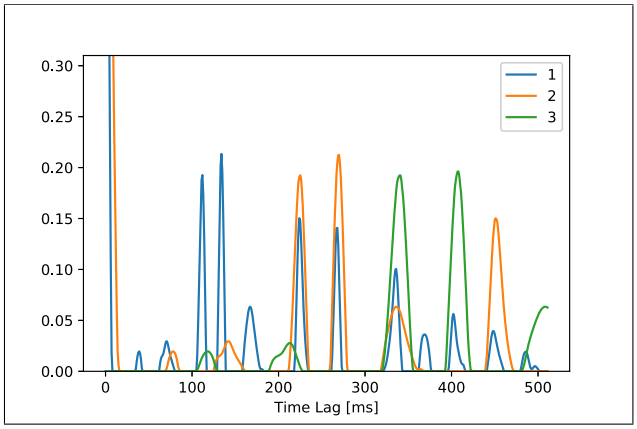
$$d(n) = \sum_{i=0}^{11} [Chroma_i(n+1) - Chroma_i(n-1)]^2 \tag{2}$$

4.2.2 Spectral Differences

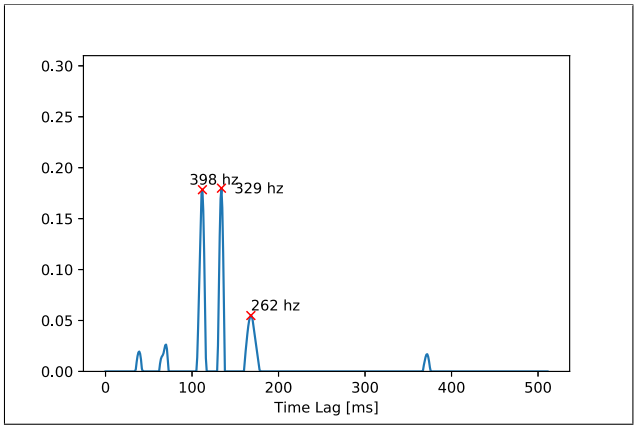
This strategy focuses on the amount of spectral change between adjacent frames. The audio is preprocessed as in Section 4.2.1 and then a Constant Q Transform (CQT) Brown (1991) is obtained. A hamming window is used with a hop of 11.6ms and the CQT spectrogram is smoothed convolving with a Gaussian kernel of σ=4. Then a *spectral difference* function is used, based on rectified differences over the spectrogram as using L-2 norm



(a) ACF.



(b) ACF superposed with itself expanded in time by a factor of 1, 2 and 3.



(c) Enhanced version of ACF.

Figure 4: ACF of a C major chord played by three trumpets. (a) ACF, (b) ACF and itself expanded in time two and three times, (c) Final version of ACF where the three peak correspond to the three fundamental frequencies.

$$SD(n) = \sum_{k=0}^{\frac{N}{2}} H([CQT_k(n+2) - CQT_k(n-2)])^2 \tag{3}$$

where $H(x)$ is a zero-clipping function defined as $H(x) = (x + |x|)/2$, k the frequency index and n the frame index. The Eq. 3 is similar to the *spectral difference* stated in Bello et al. (2005), but the main difference is that for each frame index the difference is obtained at a bigger time span ($\pm$ 2 frames) because bass note onsets may have a smoother transient that percussive sounds.

The peaks in this signal correspond to a big change of spectral content in the low frequency range, corresponding to a bass note onset. A peak peaking is performed used a non-fixed threshold defined as $thr = max(SD) \cdot 0.01$, because it was found to be somehow scale invariant.

4.2.3 Convolutional Neural Network

CNN have its origins in the computer vision community, but have been applied to many other fields in recent years, including musical information retrieval (MIR) and onset detection. They are a particular configuration of feedforward neural network that includes convolutional layers acting as features extractors, whose parameters are learned during training. Generally, 2D inputs are used for audio, probably inspired by the substantial improvements achieved in the computer vision community since the popularization of CNNs, where time-frequency representations of music, such as STFT, are preferred. In the convolutional layers, 2D filters extract local information that fit both temporal and frequency relations according to the filter size. In Pons et al. (2016) an insight is given on what kind of musical information is a network learning depending on the filters shapes.

The implementation follows a pipeline similar to the one presented in Schlüter and Böck (2014), but adapted to perform bass note onset detection and using a different network architecture. The main changes are in the input data shape, filters shapes and number of neurons in the fully connected layer.

a) Input Data

   Working with CNN allows to feed multiple channels of 2D arrays as a single input. When using time frequency representations of audio an important trade-off between frequency resolution and temporal resolutions is presented, but different resolutions can be presented to the network in different channels, leaving the possibility to learn from all of them and making the choice of the

appropriates resolutions less critical. The frequency range is limited to 40-2kHz and three spectrograms are computed using window sizes of 92ms, 46ms and 23ms, and hop size is 5.8ms. Then a frequency reduction over each channel is performed using a Mel filterbank of 39 bands, as can be seen in Fig.5.

One of the main advantages of working with CNN models for onset detection is that they can learn the temporal context of an onset in a non-causal way (both context information previous and posterior to the onset). Long context window may include more information, but lacks temporal resolution and vice versa. Then, a 58ms (10 frames) context window was found to be appropriate.

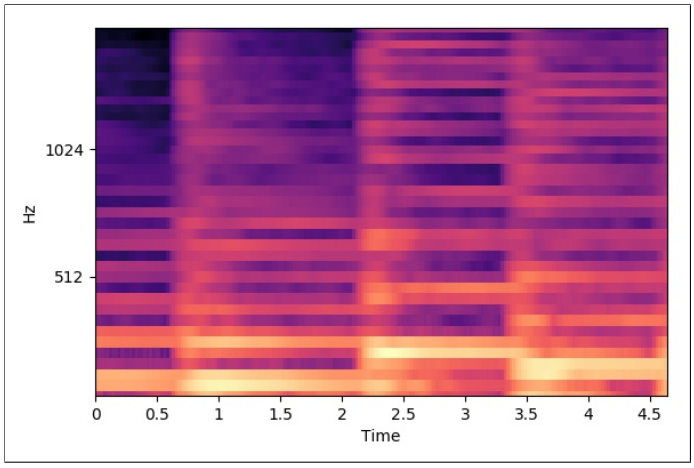
b) Model Architecture

The full model, depicted in Table 1, input shape is (10, 39, 3), corresponding to 10 frames of temporal context, 39 frequency bins and 3 channels. Then a convolutional layer of 64 filters with kernel size 5x5 (frames x frequency bands) is used with a rectified linear unit (Relu), applying batch normalization, followed by a 2x2 max-pooling layer without overlap. Another convolutional layer of 32 filters of kernel size 3x3 is use, again using Relu activation, batch normalization and followed by a 2x2 max-pooling layer. At last a fully connected layer of 200 layers is applied using dropout (with a keep probability of 0.75) and Softmax activation after the two class output.

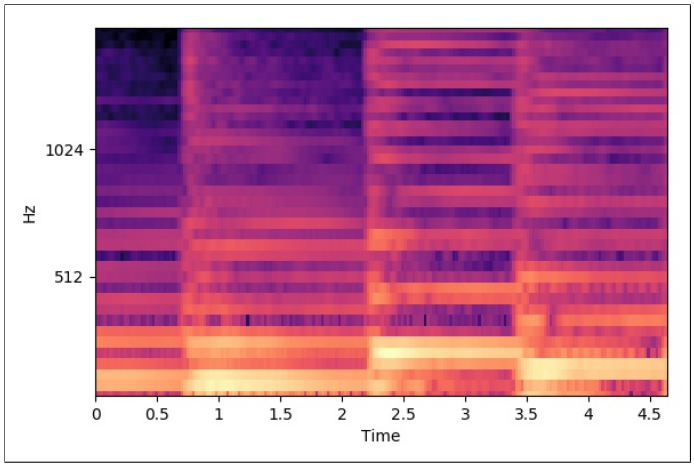
The model was trained in batches of 128 samples with binary cross-entropy loss function and using the Adam optimizer algorithm to adapt some hyper-parameters values during training.

Layer	**# Parameters**
Conv: 64x 5x5	4864
Batch Normalization	156
Max-Pooling: 2x2	-
Conv: 32x 3x3	18464
Batch Normalization	76
Max-Pooling: 2x2	-
Fully Connected: 200	115400
Batch Normalization	800
Dropout	-
Fully Connected: 2	402

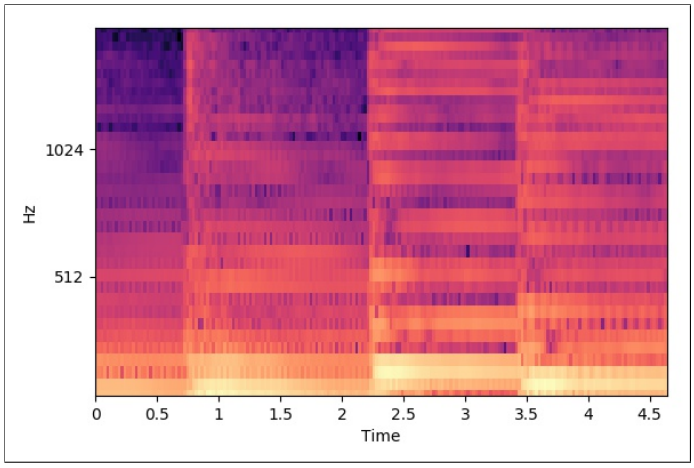
Table 1: CNN layers and number of parameters.



(a) 92ms window size.



(b) 46ms window size.



(c) 23ms window size.

Figure 5: Filtered spectrograms at different temporal-frequency resolutions.

c) Postprocessing

The CNN output is an array of 2D vectors which represents the probability of belonging to the bass onset class or the background class. In this case only the onset class probability is relevant and can be considered a onset detection function, whose peaks represent the detected onsets as can be seen in Fig.6. No minimum threshold value is used, but peaks should be separated at least by 116ms (20 frames) to prevent multiple detection caused by a same onset.

4.2.4 Recurrent Neural Network

When working with time series, RNN are a good option because they can process sequences of data on a directional way, learning temporal relations and making predictions/classifications conditioned to previous data. The temporal data is kept through hidden units (one or more layers of them) that act as a memory cells. Normally they cannot retain long term information, but some variations of RNN cells allows the efficient propagation of such information, like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU).

Basic RNN works in a causal way, they are not able to process posterior data, which might relevant in this case because onsets corresponding to different instruments may decay differently. To overcome this issue, a sophisticated configuration called Bidirectional Recurrent Neural Networks (BRNN) is used. It corresponds to two different RNNs connected to the same input, one of them analyzing the sequence data forwards and the other backwards. Their outputs are combined into a single output generating a single prediction/classification.

As with CNN, not only the onset is presented, but its temporal context, allowing recurrent networks to capture valuable information surrounding the event of interest. As onsets are manifested over a short time span they are independent to what may have occurred some seconds before, and they only depend on the occurrence of its neighbor onsets and current information.

This configuration is also similar to the one presented in Eyben et al. (2010), but the temporal differences where not used as input data, it uses GRU memory cells and more hidden units at each layer.

a) Input Data

The input data is similar to the one used in Section 4.2.3, but limited on the high range to 1kHz and the matrices are concatenated on the features axis instead of being presented in different channels.

The model is trained on the many-to-many fashion, meaning that for each input step of the sequence, one output step is obtained using the same dense

layer. Theoretically there is no restriction on the sequence length used for training, but working with batches is more efficient and requires fixed length sequences and sequences of 2320ms where found to be adequate (400 frames).

b) Model Architecture

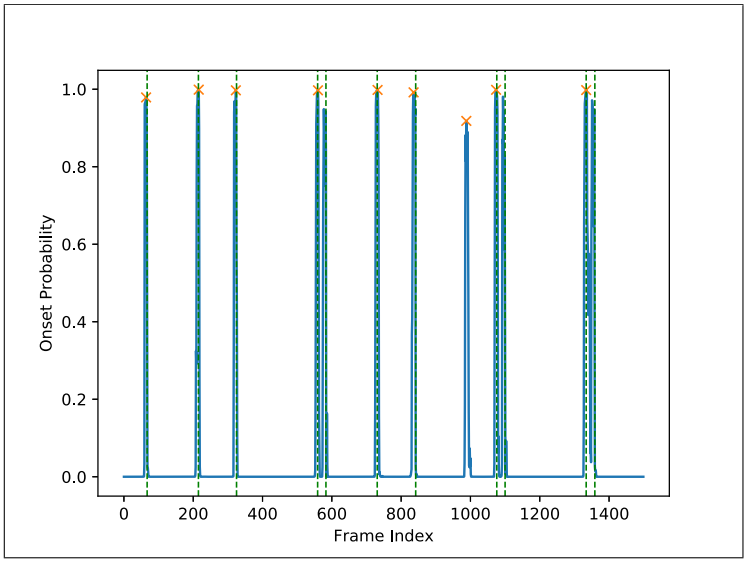
The model is composed of three layers with 60 hidden units each, using bidirectional gated recurrent units as memory cells. In total, there are 6 layers and 360 neurons. At the end a fully connected layer is applied followed by Softmax activation. The input shape for training is (400, 98) corresponding to 400 temporal steps and 98 frequency features, although for inference there is no restriction in the sequence length, so a new song can be processed as a single long sequence. The model was trained using batches of 128 samples, root mean square propagation optimization algorithm and focal loss function, originally proposed in Lin et al. (2017) for object detection. It addresses the issue that in object detection classes tend to be highly imbalanced by focusing on hard negative samples. This onset detection problem has a ratio of 1:67 onsets vs no-onset and focal loss reported better results compared to the same model trained using binary cross-entropy.

Layer	**# Parameters**
BGRU: 120 hidden units	57240
BGRU: 120 hidden units	65160
BGRU: 120 hidden units	65160
Fully Connected: 2	242

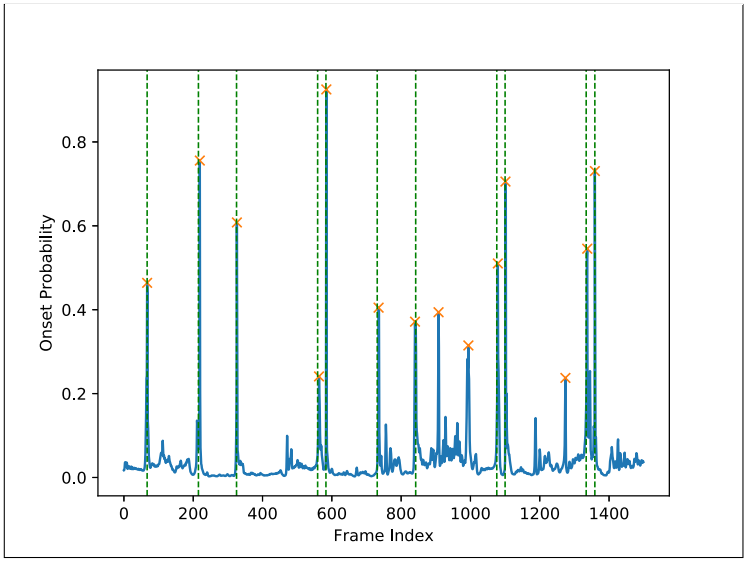
Table 2: RNN layers and number of parameters.

c) Postprocessing

Similar to Section 4.2.3, the output is the probability of that time step of belonging to each class and only the onset class is used. A fixed threshold of 0.2 was found to appropriate over training data and the minimum distance between peaks is 116ms.



(a) CNN output.



(b) RNN output.

Figure 6: Output of each deep learning method for an audio excerpt, ground truth bass onsets in green.

4.3 Experiment

4.3.1 Database

The experiments where performed over 24 audios rendered from MIDI files, corresponding to 70 minutes of music. The main advantage of working with MIDI is that the ground truth can be obtained without any ambiguity, but a drawback is that rendered audio will be exactly as it was written in the MIDI file and lacks subtleties and spontaneity, unlike audio that comes from performing musicians. As an attempt to compensate that issue, the MIDI files chosen correspond to *Bossa Nova* and different types of *Jazz* songs, featuring rich instrumentation, complex harmonic progressions and instrumental solos, making the audio less predictable, more realistic and challenging to onset detection.

20 song where randomly selected for training and the other 4 where used for testing, which are not used on the training process. As methods in Section 4.2.1 and Section 4.2.1 are non-trained methods, they were only used on the testing set. For the trained methods, the frames of all songs where mixed and then divided into 95% train and 5% validation. In Table 3 is summarized the data split used for the experiment.

Train	**Validation**	**Test**	**Cross-Validation?**
81% (57 minutes)	4% (3 minutes)	15% (10 minutes)	No

Table 3: Database split used for trained methods in percentage and minuts of music. **Train** is the data used for training, **Validation** is a set used to validate performance after each training iteration and **Test**, also known as hold-out, is the data used only to test the trained models and is not included on the training process.

The interest is to detect bass onsets, so only the onsets generated by a note whose pitch is in a low frequency range are considered. In Table 4 the chosen frequency range for bass notes used by Hainsworth and Macleod (2001), Ryynänen and Klapuri (2007) and **?** are presented. In this work, the range was low limited in 40Hz and high limited to 196Hz (corresponding to a G3 note, 55 in MIDI), similar to Hainsworth and Macleod (2001). The low range limitation is considered because pitches below this limit are very unlikely and could be considered as false detection.

Author	Frequency range (Hz)	MIDI note range
HAINSWORTH	0 - 200	0 - 55
RYYNANEN	36 - 245	26 - 59
ABEßER	41 - 392	28 - 67

Table 4: Low frequency range considered for bass notes in previous works.

4.3.2 Evaluation Metrics

The standard evaluation metrics for onset detection are recall, precision and F-measure, as in Böck et al. (2012), Schlüter and Böck (2014) and Eyben et al. (2010). They are defined from True Positives (TP), False Negatives (FN) and False Positives (FP) as follows:

$$recall = \frac{TP}{TP + FN} \tag{4}$$

$$precision = \frac{TP}{TP + FP} \tag{5}$$

$$F\text{-}measure = 2 \cdot \frac{recall \cdot precision}{recall + precision}, \tag{6}$$

where a detected onset is considered a TP when its distance to an annotated target is smaller than a predefined small temporal range, otherwise is considered a FP. Each annotated target that does not match a detected onset is considered a FN.

In onset detection the temporal accuracy of the system is also relevant, which can be measured by analyzing the time differences between the target onset and the detected onset, as follows:

$$timeDiff = targetTime - detectedTime, \tag{7}$$

only if *detectedTime* is a TP, otherwise it is not considered. Those differences are computed across the validation set for each method and then their mean and standard deviation are obtained.

4.4 Results and Discussion

The four described methods are compared with onset detection implementations available in open source Python libraries. The first one is the *onset_detect* method from Librosa McFee et al. (2015) which computes the *Spectral Flux* (SF). The others are available in Madmom Böck et al. (2016), corresponding to another implementation of the *Spectral Flux* (SF2) , the the *Superflux* (SPF) and the *Complex Flux* (CF). The input for all these methods is the band-pass filtered audio, where the lower limit is 20Hz and the higher limit is 196Hz (G3).

According to Eyben et al. (2010) some systems consider a ±50ms temporal tolerance for onset detection, but other focused on percussive onsets only ±25ms. Both alternatives where tested and reported separately in Table 5, indicated by a sub index following each method.

	Precision	**Recall**	**F-measure**
$\text{Multipitch}_{.50ms}$	0.57	0.49	0.52
$\text{Spectral Analysis}_{50ms}$	0.88	0.80	0.84
CNN_{50ms}	**0.89**	**0.97**	**0.93**
RNN_{50ms}	0.84	0.96	0.90
$\text{SF(McFee et al., 2015)}_{50ms}$	0.20	0.33	0.25
$\text{SF2(Böck et al., 2016)}_{50ms}$	0.63	0.55	0.59
$\text{SPF(Böck et al., 2016)}_{50ms}$	0.69	0.49	0.58
$\text{CF(Böck et al., 2016)}_{50ms}$	0.65	0.74	0.69
Multipitch_{25ms}	0.35	0.49	0.40
$\text{Spectral Analysis}_{25ms}$	0.70	0.63	0.66
CNN_{25ms}	**0.85**	0.93	**0.89**
RNN_{25ms}	0.83	**0.94**	0.88
$\text{SF(McFee et al., 2015)}_{25ms}$	0.03	0.04	0.03
$\text{SF2(Böck et al., 2016)}_{25ms}$	0.34	0.29	0.31
$\text{SPF(Böck et al., 2016)}_{25ms}$	0.40	0.28	0.33
$\text{CF(Böck et al., 2016)}_{25ms}$	0.39	0.44	0.41

Table 5: Performance comparison of tested methods.

The statistics of the temporal accuracy for each method are presented in Table 6.

	Mean [ms]	**STD**
Multipitch	-13 .8	0.021
Spectral Analysis	-13 .5	0.014
CNN	10.1	**0.008**
RNN	**-4.7**	**0.008**
SF	-27.0	0.022
SF2	-21.8	0.013
SPF	-21.5	0.012
CF	-20.6	0.013

Table 6: Statistics of time differences between all target and detected onsets.

The Multipitch method tend to perform poorly compared to the others because the pitch detection is not as stable as expected and that leads to many false positives, additionally around half of the onsets of interest do not have enough prominence to be considered valid peaks. The Spectral Analysis is mostly affected by other onsets that has strong presence in the frequency range of interest, such as the ones caused by some percussive instruments. Both deep learning methods have higher recall than precision and CNN methods has higher F-measure.

When a $\pm$25ms tolerance is considered, all the methods show a lower performance, where the signal processing methods drop around 0.2 in F-measure and the deep learning methods are the most stable. As can be seen in Table 6, the Multipitch and Spectral Analysis methods have negative time difference mean, implying that in average the detections fall after the targets, probably caused by the Gaussian filter that tends to smooth the detection function and delay the onsets. RNN method is the most accurate in time, which is reasonable being the only method than learns longer temporal relations between features.

4.5 Conclusions

Both deep learning achieved better results and CNN model obtained the highest F-measure. Multipitch method does not perform well compared to the others because it depends heavily on the pitch estimation stage, which is prone to false estimations under more than three simultaneous pitches, as stated by the authors. Signal processing methods used different strategies to obtain a detection function whose detected peaks turned to be very threshold sensitive, where low thresholds achieved considerably better recall but very poor precision and lower F-measure. On the other hand, deep learning methods outputs detection functions with narrower peaks and low values for non-onset frames.

A larger database used for training may help improving the deep learning methods, in particular allowing them to better generalize timbral features, be able to distinguish bass notes from other instruments and thus reducing the number of false positives.

This experiment was performed using MIDI rendered audios and thus the trained and adjusted methods, as they are, will not perform correctly on real music. In absence of databases that include hand annotated jazz songs bass notes, which is the case, it is not possible to expand this work for practical uses other that these MIDI generated audios.

5 MUSIC SYNCHRONIZATION

In music is common to find at least two time series corresponding to the same musical segment but coming from different sources. For example, a song can be performed on studio or live and in both cases a verse will correspond to the same musical part, but will not sound the same. It is to be expected that these segments were performed at different speed (different tempo), the overall tuning of the song could be the different, maybe the musicians made mistakes or made changes to their musical lines, causing that the pitch content will not be the same. Nevertheless, these audios still correspond to the same musical piece and a human can easily match patterns, rhythms or melodies between them.

The main goal of music synchronization systems is to find the optimal temporal alignment between two musical segments. This process matches frame-by-frame according to their similarities and strongly relies on the resemblance between frames in a given feature space. It is not usual to perform musical synchronization over the raw waveform because is impractical due to the high number of samples and lack of robustness against changes in timbre, intensity, etc. In general, the raw audio is transformed to a feature space containing higher level content that unveils similarities between the audios, for example the chromagram (also known as Pitch Class Profile).

In the case of jazz music, which is mainly composed by the concatenation of different repetitions of a same musical segment, it is possible to synchronize them to match musically equivalent frames. For this analysis, it is required that these choruses have been previously segmented, although there are some techniques to find the best alignment between sub-sequences.

The main motivation to synchronize these musical segments in semi improvised music lies on the fact that there is a high variance on the harmonic content between choruses, meaning that the musical content, namely notes contained on each frame, differ considerably and it could be beneficial to observe all the harmonic content along every repetition of each segment prior to perform chord transcription.

In this chapter, the Dynamic Time Warping (DTW) algorithm is explained following closely the explanation available in Müller (2015), because it is the standard method for music synchronization. A condensed explanation is presented to keep this chapter self-contained, so further details are not included. Later, in sub-sections and some key aspects of music synchronizations are defined and discussed. The performed experiments on jazz music synchronization are explained, as well as the content of all data bases used. Finally, the results are presented and discussed.

5.1 Context

In general, past research on music synchronization have not focused on the alignment algorithm itself, like DTW variants, but on the correct features used to extract meaningful information from the audio (Bello and Pickens, 2005). For example, on Izmirli and Dannenberg (2010) a deeper understanding on the effect of features is provided and is shown that the chromagram might not be the best representation for this task. In particular, they train an artificial neural network to extract chroma-like features learnt from the RWC classical collection database, for symbolic representation to audio alignment. Searching for robustness to changes in timbre, a novel variant of the chroma features is presented on Müller et al. (2009), where the relation between the first mel-frequency cepstral coefficients (MFCC) to timbre is exploited. At each audio frame these lower coefficients are discarded and the remaining are projected to the twelve chroma bins, achieving features able to retain timbre independent frequency information. One of the experiments consist on matching excerpts of two different recordings of classical and pop musical pieces. On Ewert et al. (2009) a novel feature is introduced mixing note onsets and pitches obtaining better synchronization results for music with prominent onsets, while not deteriorating the results for music with smoother onsets. The audio is analyzed on subbands, each one corresponding to a musical note, and the onsets are taken from sudden changes on each note energy and then the content of each pitch class is added, obtaining chroma-like features. The experiments are carried out on MIDI rendered audios that are randomly stretched and compressed, and then aligned with their original version. All these works use either Euclidean or cosine distance metrics to compare the feature sequences.

These previous works implicitly assume that the audios to be aligned share the same melody and chords, but differ on the instrumentation, tempo and recording conditions. When the audios to be aligned correspond to a same musical part because have the same harmonic progression, but they do not share the same melody or bass lines being played, the similarities take place on a higher musical level and alignment becomes a more difficult task, like in jazz music.

5.2 Theoretical Background

5.2.1 Dynamic Time Warping

The standard method for music synchronization is Dynamic Time Warping (DTW) (Müller, 2015), which is a efficient Dynamic Programming based algorithm that finds the optimal alignment between two time series in a given feature space. It is not required that the two sequences have the same length, but their components

should be in the same feature space, for example let's consider the time series $X = (x_1, x_2, x_3, ..., x_N)$ and $Y = (y_1, y_2, y_3, ..., y_M)$ with $N, M \in \mathbb{N}$ and $N \neq M$. x_n and y_m can be scalars or vectors, but their dimensions should be the same. The result of DTW algorithm is a list pairing samples of each sequence called the optimal path, incorporating the key idea that this alignment is not a linear mapping between them but a matching that can be warped in time assigning a single sample of one sequence to more than one sample in the other sequence.

The general algorithm is composed by three steps:

- Generate a Cost Matrix
- Obtain an Accumulated Cost Matrix
- Find the Optimal Path

and two key elements that impact directly on the overall performance of the algorithm:

- Features Space chosen to compare the sequences
- Cost function that measures the samples dissimilarity.

a) Cost Matrix:

Following the example of sequence X and Y, let's suppose they correspond to sequences of chroma vectors corresponding to two different versions of a same part of a song. So, X is a matrix of size [12, N] and Y has size [12, M]. A cost matrix C is constructed applying some distance measure c between all elements of the two sequences, obtaining a $N \times M$ cost matrix. A popular cost metric is euclidean distance, defined as:

$$c_e(x, y) = \|x - y\|, \tag{8}$$

where each element of the cost matrix is defined as

$$C(n, m) = c_{eucl}(x_n, y_m). \tag{9}$$

As we are comparing one sample to each other samples in the other sequence, low costs mean implies similarity and high cost dissimilarity. It is to be expected that some patterns would be revealed from the cost matrix, showing low distance values between similar subsections of the sequences.

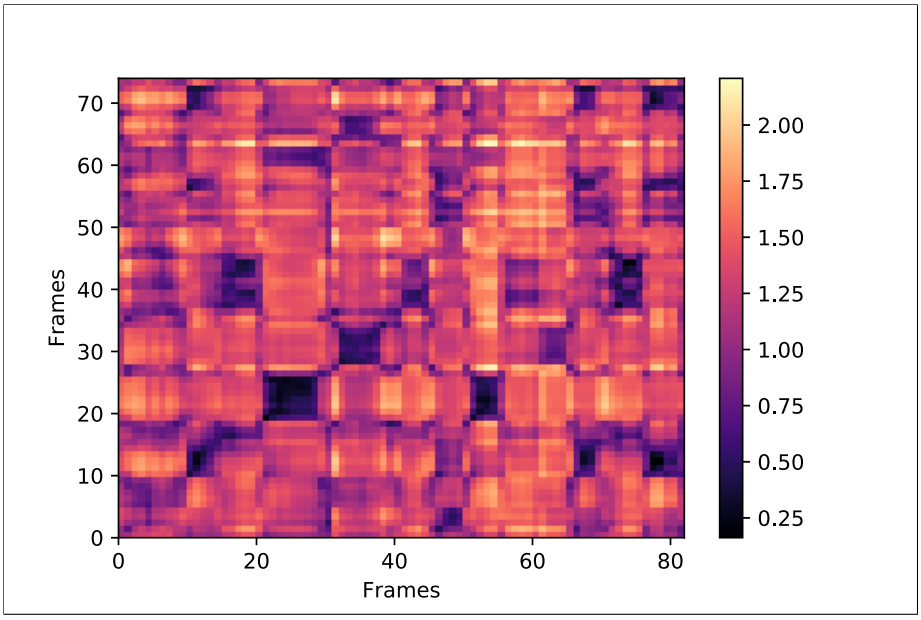


Figure 7: Cost Matrix using euclidean distance function. Darker elements show higher similarity.

b) Accumulated Cost Matrix:

The simplest choice to find the optimal path would be try all the possible combinations and choose the one with the total lowest cost. Clearly this approach is not efficient and its complexity grows exponentially with longer sequences. To optimize this process an approach is used under dynamic programming's main idea which consist in finding the main problem's optimal solution from simpler sub problems solutions. In this case, it consists in finding the optimal sub sequences within the complete sequences.

The accumulated cost matrix C_{acc} is constructed from the cost matrix, each point indicates the minimum accumulated cost at which it can be reached and does not explicit the path of pairing points that led to that point, but only that path's total cost. It can be computed recursively starting from C left and bottom borders:

$$C_{acc}(n,1) = \sum_{k=1}^{n} C(k,1) \tag{10}$$

$$C_{acc}(1,m) = \sum_{k=1}^{m} C(1,k), \tag{11}$$

for $n \in [1, N]$ and $m \in [1, M]$ and then filling in the rest left to right and bottom to top following the next equation:

$$C_{acc}(n, m) = C(n, m) + min \begin{cases} C_{acc}(n-1, m)) \\ C_{acc}(n, m-1) \\ C_{acc}(n-1, m-1) \end{cases}, \tag{12}$$

for $n \in [2, N]$ and $m \in [2, M]$.

c) Optimal Path:

The final step is finding the optimal path from the accumulated cost matrix, which is done using backtracking. It is required that both the beginning and end of the sequences match. This part of the algorithm starts from the end point and iteratively selects the next pair on the path using a specific criterion. Following an inverse process that the one described on Eq12, the point with lowest accumulated cost is chosen between the three points located at $C_{\text{acc}}(n-1, m)$, $C_{\text{acc}}(n, m-1)$ and $C_{\text{acc}}(n-1, m-1)$. The iteration stops when the begin index has been reached.

5.2.2 Features

Music is a mixture of different audio streams that form a waveform which is not very "human-readable", because the information that can be retrieved directly from it is not sophisticated and generally audio analysis is not performed in the raw time domain. DTW will work properly only if two sequences are somehow similar in their feature space, so the choice of adequate features is vital. It is expected that similar sequences have similar notes, despite the tempo at which they were performed, making the chromagram the favourite option for music synchronization because it captures melodic and harmonic information. Among the advantages of using the chromagram is its simplicity, low dimensionality and ease interpretation as musical notes are represented explicitly. There are variations and enhanced versions of the classic chromagram, but in general its key properties remain the same. On the other hand, variants like the one in Korzeniowski and Widmer (2016) propose chroma like features extracted by pre-trained systems and obtain a cleaner representation, but experiments using these features for music synchronization have not been reported.

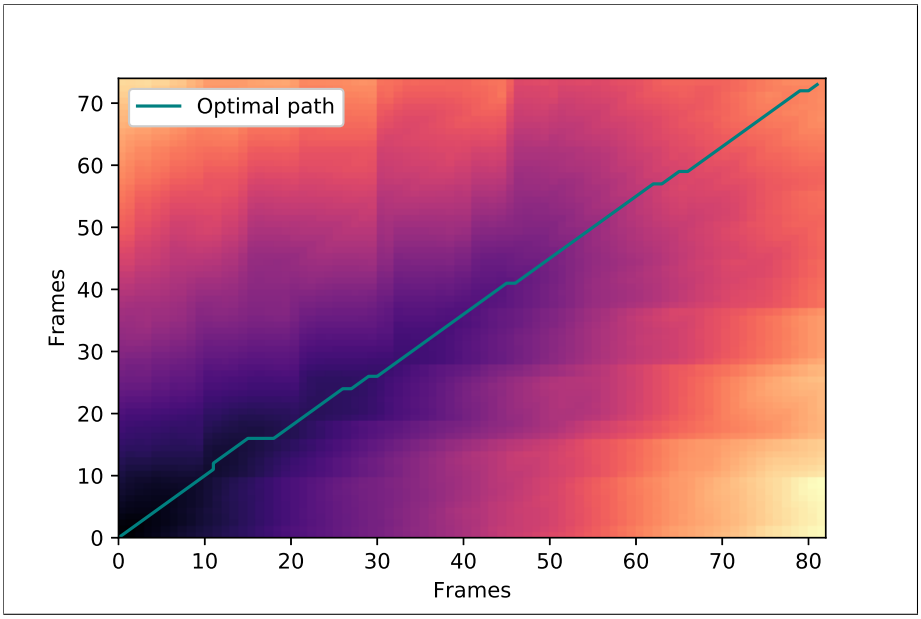
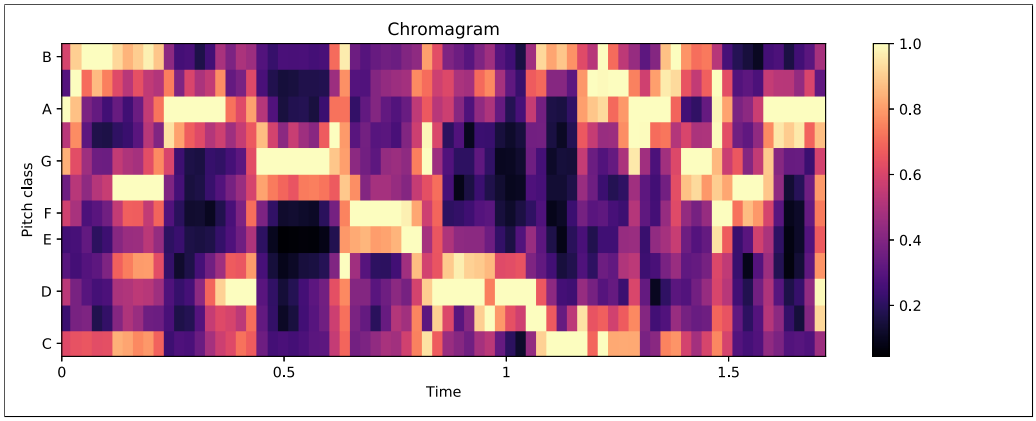


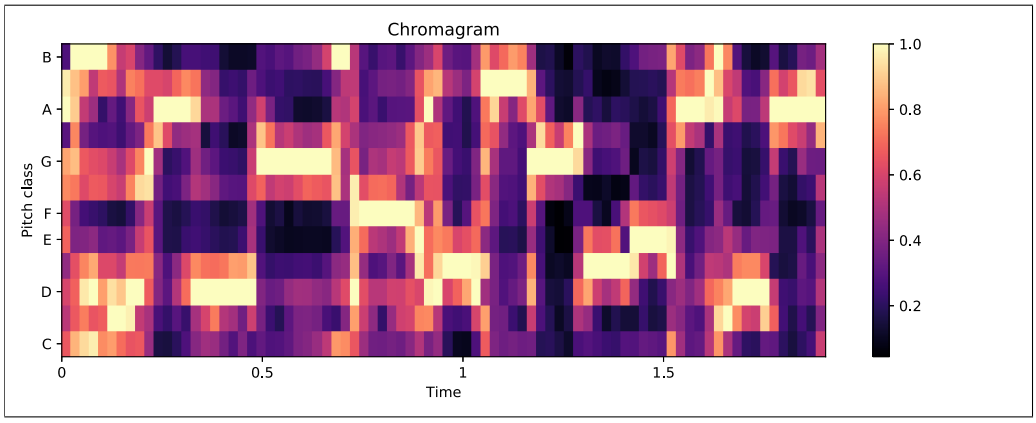
Figure 8: Accumulated Cost Matrix with the optimal path.

In the case of improvised music synchronization, it may not be a good idea to fully rely on the chroma features, because matching sequences do not have the same notes, but the same chord progression. In this feature space the musical relations between notes are not considered, meaning that the similarity between one note and all the others is constant. For instance, a triad composed by the notes C-F#-G# will be equally distanced from Cmaj (C-E-G) than Fmaj (F-A-C) because both have only one note in common. In improvised music, it is expected that chords will be enriched with non-triadic notes, while others might be discarded, causing them to look very different to the basic triad in a chromagram, as seen on Fig.9. The tonal centroid features proposed in Harte et al. (2006), sometimes called tonnetz features, are a non-linear transformation of the chromagram that reduces it to a 6-D features space that represents musically meaningful relations between notes. For example, the Euclidean distance between fifths and thirds is small and between tritones is bigger, as shown in Fig.10. It was originally proposed to detect harmonic changes in music, but seems to be well suited for this application of music synchronization.

Both chroma and tonnetz features are lossy because they are not able to retain the frequency range where each pitch came from, but making all the octaves equally relevant is not consistent with how the chords are usually played (it is unusual to play chords on a high pitch range). Having a mostly steady rhythmic pattern and predominantly playing notes belonging to the corresponding chord basic triad, the bass line seems to carry a considerable amount of meaningful musical information,



(a) Original song chromagram.



(b) Alternate take song chromagram.

Figure 9: Comparative chromagrams showing the similarity between the first four bars of two versions of John Coltrane's "Giant Steps". Although they are similar, the notes played are not exactly the same.

as explained in Section 4, especially when the alignment is focused on harmonic content. On the other hand, melody and chords are mostly located on the mid frequency range. Usually high frequency pitches are not as common as in the mid-range, but many harmonics of lower frequency pitches may be present. Chroma and tonnetz features discard all this information, so three overlapping frequency ranges are used and features on each band is computed. By doing this the pitch content is more specific as it is linked to a frequency range, with the main drawback that now there are 36 features instead of 12 (in the case of the chromagram). The frequency limits of each range are shown in Table 7 and they span almost the whole piano range (from C1 to C8) and that adjacent ranges overlap on one octave, adding redundancy, which is usually desirable because may lead to robustness.

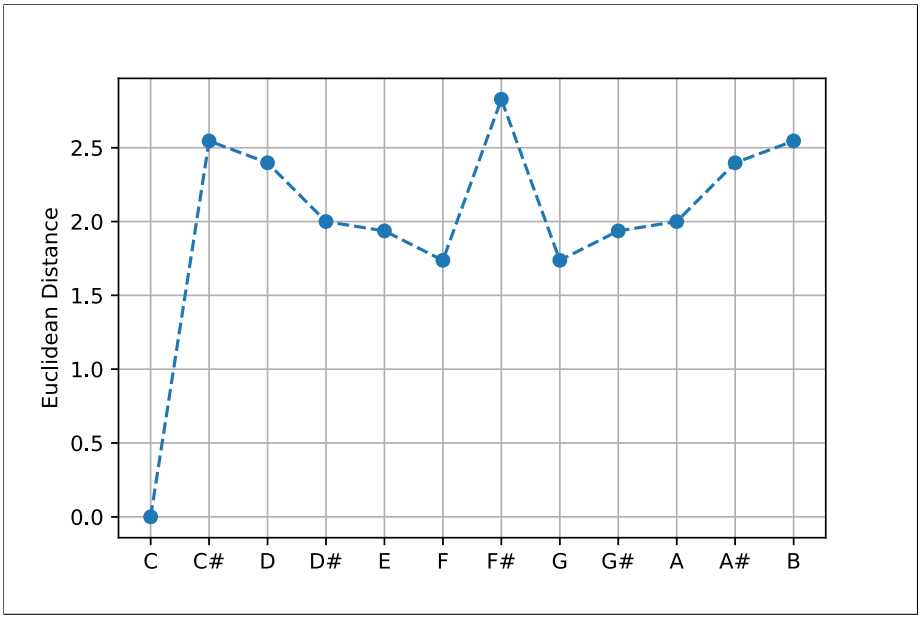


Figure 10: Euclidean distance of each note compared to C in the tonal centroid feature space.

Frequency Range	**Pitch Limits [Hz]**
Low	[32 - 261] (C1 - C4)
Mid	[130 - 1046] (C3 - C6)
Treble	[523 - 4186] (C5 - C8)

Table 7: Frequency ranges used for feature extraction.

Timbre is a musical dimension that can be also taken in to account, as it could help to improve, at finer level, the synchronization of some specific musical events with similar timbral features, like bass notes or drum beats. Even if on a jazz song each chorus repetition may contain other instrumentation and have different timbre, it is a complex musical quality closely related to human perception and might carry information useful to improve synchronization. The standard descriptors used to model timbre are the lower coefficients of the MFCC, so the first 20 were included to the feature space, either concatenating them to the feature pool or as single features.

To compare the effect of each different feature (or variations in the case of the chroma and tonnetz split in three ranges), each kind of feature was tested alone as well as with many different combinations of them. The combination of features is just the concatenation of them, increasing the dimensionality.

Features	**Dimensions**
chroma	12
tonnetz	6
MFCC	20
chroma + MFCC	32
chroma + tonnetz	18
chroma + tonnetz + MFCC	38
tonnetz + MFCC	26
chroma[ranges]	36
chroma[ranges] + MFCC	56
chroma[ranges] + tonnetz	42
chroma[ranges] + tonnetz + MFCC	62
chroma[bass-mid]	24
chroma[bass-mid] + MFCC	44
chroma[bass-mid] + tonnetz	30
chroma[bass-mid] + tonnetz + MFCC	50
chroma[bass]	12
chroma[bass] + MFCC	32
chroma[bass] + tonnetz	18
chroma[bass] + tonnetz + MFCC	38
chroma[mid]	12
chroma[mid] + MFCC	32
chroma[mid] + tonnetz	18
chroma[mid] + tonnetz + MFCC	38
tonnetz[ranges]	18
tonnetz[ranges] + MFCC	38
tonnetz[ranges] + chroma	30
tonnetz[ranges] + chroma + MFCC	50
tonnetz[bass-mid]	12
tonnetz[bass-mid] + MFCC	32
tonnetz[bass-mid] + chroma	24
tonnetz[bass-mid] + chroma + MFCC	44
tonnetz[bass]	6
tonnetz[bass] + MFCC	26
tonnetz[bass] + chroma	18
tonnetz[bass] + chroma + MFCC	38
tonnetz[mid]	6
tonnetz[mid] + MFCC	26
tonnetz[mid] + chroma	18
tonnetz[mid] + chroma + MFCC	38

Table 8: List of single features and feature combinations considered in this experiment.

5.2.3 Distance Functions

The choice of an appropriate distance function is crucial, because according to the data type and to the chosen features, some distance functions may reveal similarities more accurately than others. As the feature set contains elements of different nature in terms of the information being described and the spreading of each descriptor, it is common that features with wider ranges, and thus higher distance, dominate over features with smaller variance. Some distance function take that issue into account as others simply omit the magnitude of the values. Especially when different kind of features are being concatenated or mixed somehow, it is important to find a distance metric that adjusts correctly.

a) Euclidean distance:

   The formula is shown in Eq. 8. Is the most intuitive distance metric and it is equivalent to the distance between two points that can be measure with a ruler. It does not consider any kind of correlation between data and is very sensitive to scaling. If a 2D feature set, over one of its dimensions has a restricted range between -1 and 1, and the other dimension is spread between -1000 and 1000, the first dimension will be irrelevant in the total computed distance. So, this metric, despite being simple and intuitive, relies strongly in the data spread and in most cases will not be the most suitable choice for music synchronization, especially when combining features coming from different distributions.

b) Mahalanobis Distace:

$$c_m(x, y) = \sqrt{(x - y)^T C^{-1} (x - y)}, \tag{13}$$

   where C^{-1} is the inverse of the covariance matrix between the vectors.

   It can be interpreted as a variation of the euclidean distance that takes into account the correlation between the data. It represents the distance between a point and a distribution, alike other distance metrics that only compute isolated distances between points. For ideally uncorrelated data, the covariance matrix would just be an identity matrix (if variances are unitary) and is equivalent to the euclidean distance. In other cases, the inverse of the covariance matrix acts as a factor that uncorrelates data, making the distance computation more robust and reliable. One drawback is that it is more computationally expensive that the others because of the inverse covariance matrix, which can be slow to obtain for high dimensional data. If a big amount is data is available,

the covariance matrix can be computed once and its inverse re-used at every distance calculation, making computation faster. It should be noted that this distance needs either a pre-computed inverse correlation matrix, for which it needs data in advance, or obtain it from the two whole sequences being synchronized.

5.3 Experiment

The main goal of this experiment is to compare the performance of different combinations between feature sets and distance functions for synchronization of semi improvised music segments. This kind of music has the particularity that each repeated segment does not contain the same harmonic-melodic content than others, which represent a challenge to traditional synchronization methods. Additionally, one of the main issues of music synchronization is tempo variation, motivating the search of robust methods that can manage the speed differences between sequences, as DTW. As an attempt to systematize and organize the experimental setup, three different MIDI databases were made from a basic one, resulting in four databases that will be explained in Section 5.3.1.

As MIDI rendered audio files tend to be unrealistic and too clean, these methods were also tested on a real music database to check their robustness. All the theoretical analysis discussed in this section only applies to a very specific type of music where improvisation plays a major role, so most of the commonly used Classical and Pop music databases are not necessarily pertinent in this case. In Eremenko et al. (2018) the Jazz Audio-Aligned Harmony dataset is presented (JAAH), consisting in 113 tracks with data transcribed from the audios. The complete set of feature combinations and distance functions where also tested over this database, following same methodology that for the MIDI databases.

As the time-frequency feature representation have the well-known resolution trade-off explained in Chapter 4, for each combination of features three different temporal steps were tested: 46 ms, 92 ms and 185 ms. These temporal hops are quite large compared to other kind of music data analysis where the length of the frame sequence is not relevant and a finer temporal analysis is needed. The choice of those hop lengths was motivated by two ideas: DTW algorithm complexity increases exponentially with longer sequences so they have to be as short as possible and the idea of adding little redundancy to the DTW algorithm in terms of how similar is a frame to its neighbors. As the time at which musical notes and phrases occur is much slower that the standard temporal resolution used in MIR (normally between 23 ms and 185 ms), frames tend to be very similar one to each other and thus there is a high degree of redundancy between them. The sequence matching algorithm is

Database	Tempo	Chord Content	Bass Line Content	Copy-Pasted Between Sequences	
				Chords	Bass Line
DB0	Fixed	Triads	Triad Notes	True	True
DB1	Variable	Triads	Triad Notes	True	True
DB2	Fixed	Triads+Tensions	Triad Notes	False	False
DB3	Variable	Triads+Tensions	Triad Notes	False	False

Table 9: Summary of every MIDI database used in the synchronization experiment.

based on segments similarity, there is no need for so much redundancy and coarser temporal resolutions may be used.

All the sequences have a sample rate of 44100 Hz. Chroma features were obtained from the CQT because it achieves better frequency resolution at low frequencies and tonnetz features were computed from the chroma. All the features were extracted using the Librosa Python library (McFee et al., 2015).

As explained in Section 5.2.3, the Mahalanobis distance takes an inverse covariance matrix as input. Two different modalities of obtaining this matrix were tested: The first computes the covariance only from the features of two sequences being aligned, meaning that a new matrix is obtained for each combination of pairs. The second modality computes the inverse covariance matrix from the features of all the sequences of a same song, considering all the relations between features along the whole song. This is more efficient in terms of computation as the matrix is only obtained once per song. The first option will be referenced as Mahalanobis_2 and the second as $\text{Mahalanobis}_{\text{all}}$.

5.3.1 Databases

1) MIDI:

   The key idea of having different MIDI databases is to increase the degree of difficulty in a controlled environment. As explained, the two main challenges are tempo variation and the dissimilitude between harmonic-melodic content, so these elements were included as explained below and summarized in Table 9:

   - DB0: This database is the simplest one and represents an idealized case composed by MIDI jazz songs, where all of them include bass, harmonic instrument (piano or guitar) and instrumental solos. All of them follows a general structure: the first repetition consists on the melody, following a certain number of repetitions containing instrumental solos and the last

one is the melody again. The melody, the bass line, the rhythmic pattern and the chords have been copy-pasted, so the only melodic difference between sequences are the instrumental solos. The bass line only plays notes on the chord triad and the chords played are plain triads to keep the harmonic content as simple as possible. The tempo is fixed among sequences.

- DB1: This database inherits all the features from DB0, but incorporate tempo variations. In real music, tempo changes slowly and gradually in a unpredictable way. To emulate this effect, the tempo was modified each four measures by adding values sampled from a uniform distribution with integer values between [-3, 3] beats per minute (BPM). As the harmonic-melodic content remains simple and unchanged, the only complexity of this database compared to DB0 is the tempo variation.
- DB2: Inherits all the features of DB0, but adds complexity on the harmonic content. The chords are no longer plain triads but incorporate seventh, ninth ans thirteenth grades on a unpredictable way, there were not added following a structured random sampling, but a musically motivated criteria. The bass line and the chords are not copy-pasted, but rather re written at each repetition, so the harmonic content mutates and becomes more complex. The tempo remains constant and the only difference with DB0 are the notes included on the chords and bass lines.
- DB3: This database is a mix of DB1 and DB2: it includes random tempo stretching and compressing, as well as variations on the harmonic content included on the chords and the bass line. It is the most realistic database as it includes unpredictable note sequences, speed dynamics and every repetition of a sequence within a same song is somehow different to others.

It is important to highlight the fact that these MIDI rendered audio do not contain any percussive instrument, which keep the experiment focused on melodic-harmonic content and reduce the factors that can mislead the conclusions.

2) JAAH:

The Jazz Audio-Aligned Harmony dataset consists on 113 tracks where the following data has been transcribed from the audio:

- Complete set of beats annotations, indicating its temporal location in seconds.
- beat-aligned chord sequence.
- Labels for each structural segment. For example, each segment is named as *intro*, *head*, etc.

Other information is also included in the database, but are not relevant for this experiment. Although this dataset is originally proposed for chord transcription, it includes all the information needed to perform music synchronization experiments, because the matching beats in two sequence can be used as ground truth.

The JAAH considers a wide range of jazz sub-genres, spanning from its origins up to the beginning of modal and free jazz, nevertheless the tracks have been selected homogeneously and there is no bias on a period or sub genre. Most of the tracks follow the standard structure where the Chorus, the template fundamental structure, is repeated multiple times but performed in a different way. Some tracks do not follow this structure and only have non-matching segments, so were discarded. From the remaining ones, only the Chorus sections were used for experimentation because are the ones repeated multiple times. In total 82 tracks are used with a total of 536 Choruses.

5.3.2 Evaluation

The optimal path found by the DTW algorithm can be displayed as a trajectory on the accumulated cost matrix, as can be seen in Fig.8. On the other hand, the true alignment lives on a continuous time domain and can be approximated on a discrete time domain, resulting on the best possible alignment (ground truth) that can be plotted on the cost matrix as well.

The evaluation of music synchronization systems can be understood as the simple geometric problem of comparing the distance of the found path to the ground truth path on a 2D plane. In the case of fixed tempo databases, namely DB0 and DB2, the sequences have equal length and the ground truth corresponds to a perfect diagonal line. In other cases, the tempo varies only each four measures, so once having the measure to measure alignment it is simple to obtain the frame to frame ground truth alignment. In these cases, the ground truth path will be composed by segments intersecting their neighbors and thus creating a path without discontinuities.

To measure the error between the found path and the ground truth segments the Mean Absolute Error (MAE) was used, which is defined in Eq. 14 as follows:

$$MAE = \frac{\sum_{i=1}^{N} |y_i - x_i|}{N}, \tag{14}$$

where y_i is a point on the estimated path, x_i is its closest point from the ground truth path and N is the length of the optimal path. This expression was chosen

mainly because it is measured in temporal units making the interpretation more intuitive.

In the case of DB1 and DB3, which include tempo variations, the ground true path is a piecewise function composed by many segments, so the error is computed considering the minimum distance point and the ground truth. This is illustrated in Fig.11, where the point is close to both segments, but the red distance is smaller and will be considered as the error.

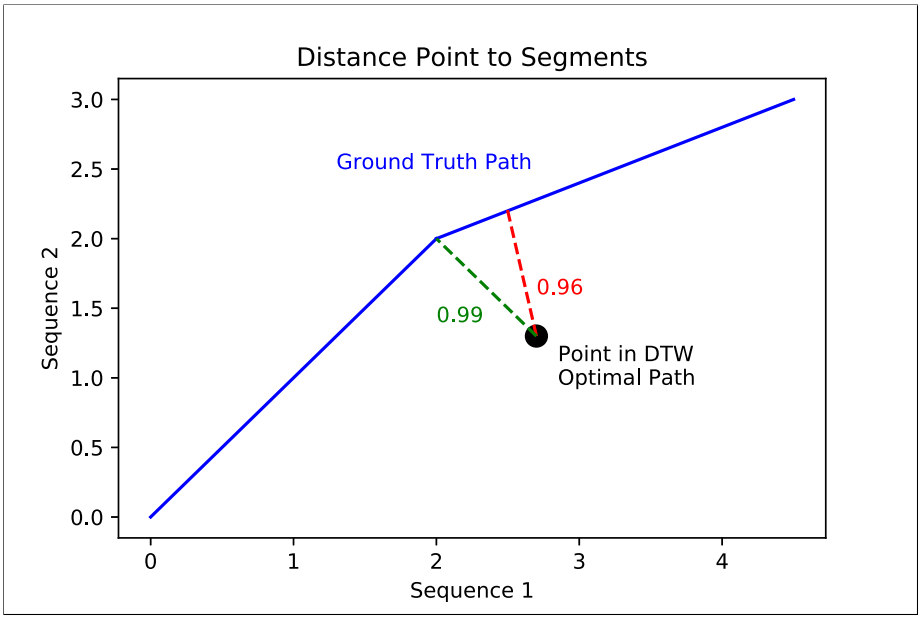
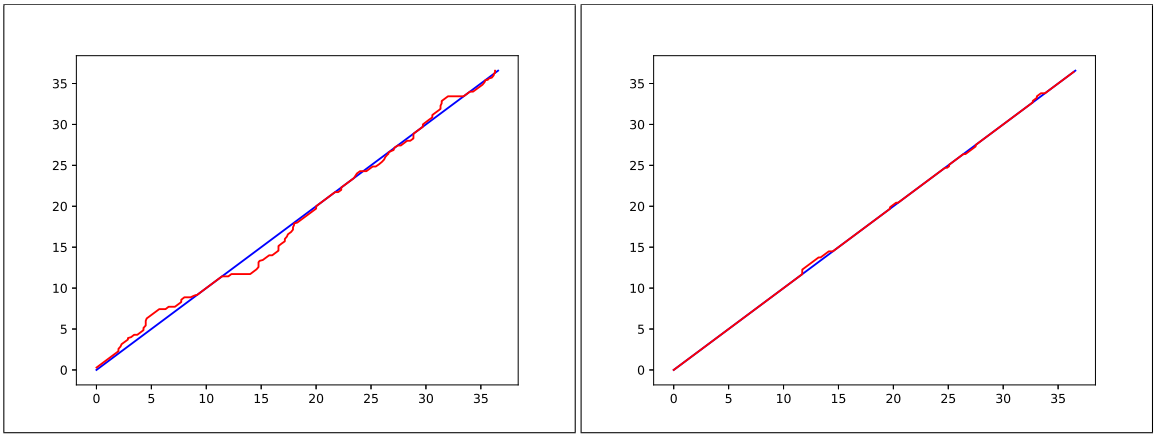


Figure 11: Illustration of point-to-segments distance calculation, in this case the red distance will be considered.

Each song is composed by at least three repetitions of a same Chorus and the synchronization experiment is performed over every possible combination of two sequences within a song. Each pair of sequences gives a MAE, so a single song will have as many results as possible pair sequences combinations and the mean of these errors is computed resulting on a song average error. At the end, all the song errors are averaged and a total error is obtained for that experiment's setting.

As error is measured on continuous time, there will be small error induced by discretization, meaning that a perfect alignment will not have zero error. All the audios are sampled at 44100Hz, so there is an inferior threshold on time resolution equal to 2.26^{-5} seconds and everything below that does not have a real meaning. Errors below that value are rounded to zero.

Also, a visual comparative plot was obtained for every pair of sequences synchronized, showing the optimal path found and the ground truth path. As can be



(a) “Bad” alignment (b) “Good” alignment

Figure 12: Comparison of synchronization performances over sequences with fixed tempo. (a) shows a bad alignment and (b) shows a better alignment, but neither of them achieved ideal results (completely superposed paths).

seen in Fig.12, a good alignment shows the optimal path close to the ground truth path (ideally superposed) and in a bad alignment they are separated. As further the optimal path is from the ground truth path, worse is the system’s performance.

5.4 Results

For sake of simplicity, only the top 5 results on each database are displayed on the following tables for each distance function. The chroma and tonnetz features obtained on each frequency range are indicated by squared brackets and when the three ranges are used, the notation will be "[ranges]" instead of "[bass-mid-treble]" for practical reasons. All results are in seconds.

5.4.1 MIDI

A summary table is presented on Table 10 where the average error over the whole feature set is presented for each distance function and for each MIDI database.

	DB0	**DB1**	**DB2**	**DB3**	**JAAH**
Euclidean	2.21e-2	1.61e-1	4.05e-2	1.84e-1	3.31e-1
Mahalanobis_2	5.70e-3	7.95e-2	1.21e-2	9.66e-2	1.83e-1
$\text{Mahalanobis}_{\text{all}}$	5.80e-3	8.10e-2	1.25e-2	9.98e-2	1.89e-1

Table 10: Average error results for MIDI and JAAH databases.

a) Euclidean Distance Results:

Database	**Features**	**Hop Length[ms]**	**MAE[s]**
DB0	chroma[ranges]+tonnetz+MFCC	185	1.15 e-3
	chroma[ranges]+tonnetz	185	1.15 e-3
	chroma[ranges]+MFCC	185	1.55 e-3
	chroma[ranges]	185	1.67 e-3
	chroma[ranges]+tonnetz+MFCC	92	2.20 e-3
DB1	tonnetz[ranges]+MFCC	46	6.31 e-2
	chroma[ranges]+tonnetz+MFCC	46	6.41 e-2
	tonnetz[bass-mid]+MFCC	46	6.42 e-2
	chroma[ranges]+tonnetz	46	6.59 e-2
	chroma[ranges]+MFCC	46	6.65 e-2
DB2	chroma[ranges]+tonnetz+MFCC	46	8.56 e-3
	chroma[ranges]+tonnetz	46	9.61 e-3
	chroma[ranges]+MFCC	46	1.16 e-3
	chroma[ranges]+MFCC	185	1.18 e-3
	chroma[ranges]+tonnetz+MFCC	185	1.32 e-3
DB3	tonnetz[ranges]+MFCC	46	6.47 e-2
	chroma[ranges]+tonnetz+MFCC	46	6.55 e-2
	chroma[ranges]+MFCC	46	6.55 e-2
	chroma[ranges]+tonnetz	46	6.69 e-2
	chroma[ranges]	46	6.70 e-2

Table 11: Top 5 results using Euclidean Distance over each database.

b) Mahalanobis$_2$ Distance Results:

Database	**Features**	**Hop Length[ms]**	**MAE[s]**
DB0	chroma+tonnetz[ranges]+MFCC	185	0
	chroma[bass]+MFCC	185	0
	chroma+tonnetz[mid]+MFCC	185	0
	chroma[bass-mid]+tonnetz+MFCC	185	0
	chroma[ranges]+MFCC	185	0
DB1	tonnetz+MFCC	46	2.58 e-2
	tonnetz[mid]+MFCC	46	2.59 e-2
	chroma+[mid]tonnetz+MFCC	46	2.76 e-2
	tonnetz[bass]+MFCC	46	2.76 e-2
	tonnetz[ranges]+MFCC	46	2.78 e-2
DB2	tonnetz[ranges]+MFCC	185	0
	chroma+tonnetz[ranges]+MFCC	185	0
	chroma+tonnetz[mid]+MFCC	185	1.81 e-4
	chroma[ranges]+tonnetz+MFCC	92	2.56 e-4
	chroma[ranges]+MFCC	92	3.40 e-4
DB3	tonnetz[ranges]+MFCC	46	2.61 e-2
	chroma[bass]+tonnetz+MFCC	46	2.67 e-2
	chroma[bass]+MFCC	46	2.70 e-2
	tonnetz[bass-mid]+tonnetz+MFCC	46	2.70 e-2
	chroma[ranges]+MFCC	46	2.75 e-2

Table 12: Top 5 results using Mahalanobis$_2$ Distance over each database.

c) Mahalanobis$_{\text{all}}$ Distance Results:

Database	**Features**	**Hop Length[ms]**	**MAE[s]**
DB0	chroma[bass]+MFCC	185	0
	chroma[bass-mid]+tonnetz+MFCC	185	0
	chroma+tonnetz[mid]+MFCC	185	0
	chroma[bass-mid]+MFCC	185	0
	chroma[ranges]+tonnetz+MFCC	185	0
DB1	tonnetz[mid]+MFCC	46	2.67 e-2
	tonnetz+MFCC	46	2.74 e-2
	tonnetz[bass]+MFCC	46	2.83 e-2
	tonnetz[ranges]+MFCC	46	2.88 e-2
	chroma[mid]+tonnetz+MFCC	46	2.91 e-2
DB2	chroma+tonnetz[ranges]+MFCC	185	3.63 e-4
	chroma[ranges]+tonnetz+MFCC	46	4.50 e-4
	chroma[ranges]+tonnetz+MFCC	185	4.58 e-4
	chroma[ranges]+MFCC	92	4.63 e-4
	chroma[ranges]+MFCC	185	4.97 e-4
DB3	tonnetz[bass-mid]+MFCC	46	2.80 e-2
	chroma[ranges]+MFCC	46	2.89 e-2
	chroma[ranges]+tonnetz+MFCC	46	2.94 e-2
	tonnetz[ranges]+MFCC	46	2.95 e-2
	chroma+tonnetz[mid]+MFCC	46	2.97 e-2

Table 13: Top 5 results using Mahalanobis$_{\text{all}}$ Distance over each database.

5.4.2 JAAH

Distance Function	Features	Hop Length[ms]	MAE[s]
Euclidean	chroma[ranges]+tonnetz+MFCC	46	1.29 e-1
	chroma[ranges]+tonnetz	46	1.32 e-1
	chroma[ranges]+MFCC	46	1.34 e-1
	chroma[ranges]	46	1.41 e-1
	chroma[bass-mid]+tonnetz+MFCC	46	1.52 e-1
Mahalanobis_2	chroma[ranges]+tonnetz+MFCC	46	4.92 e-2
	chroma[ranges]+MFCC	46	5.07 e-2
	chroma+tonnetz[ranges]+MFCC	46	5.29 e-2
	chroma[bass-mid]+MFCC	46	5.38 e-2
	chroma[bass-mid]+tonnetz+MFCC	46	5.56 e-2
$\text{Mahalanobis}_{\text{all}}$	chroma[ranges]+MFCC	46	5.18 e-2
	chroma[ranges]+tonnetz+MFCC	46	5.22 e-2
	chroma+tonnetz[ranges]+MFCC	46	5.62 e-2
	chroma[bass-mid]+tonnetz+MFCC	46	5.72 e-2
	chroma[bass-mid]+MFCC	46	5.74 e-2

Table 14: Results for every distance function on JAAH database.

5.5 Analysis

As many options were tested only some general conclusions can be easily obtained, but other are not obvious and some deeper analysis of the results is needed. A further understanding of how each feature, and all of its variations, affects the the performance is vital. Each feature is tested alone and in feature sets, so it is possible to compare the average performance of two mutually exclusive subsets: one with all the possibilities including that specific feature and another that does not contain it. Sometimes these subsets can be complementary, for example when comparing all the results including MFCC vs the ones that does not include it, but not in other cases, like when comparing the results using a hop length of 46 ms versus 92 ms (because the 185 ms option is not considered).

For each pair of compared subsets the average performance is computed, they are compared in terms of their ratio and the percentage of improvement is presented. The pertinent pair of feature pairs compared are:

- Every hop length compared to each other. The three combinations are presented independently (46 vs 92, 46 vs 185 and 92 vs 185).
- With MFCC vs Without MFCC.

- Using single full frequency range chromagram vs limited frequency ranges chromagram. The "normal" option is compared independently to each combination of limited range options.
- Using single full frequency range tonnetz vs limited frequency ranges Tonnez reported similar tendencies that the ones with chromagrams.
- chromagram vs tonnetz. Compares all options including any kind of chromagram vs the alternatives with tonnetz.

For simplicity, only the results for the JAAH database are displayed (Table 15), but MIDI ones can be found on the Appendix section. However, the trends revealed of the JAAH database are also manifested on the MIDI one, so there are no inconsistencies on the conclusions obtained. There is one exception were a pattern on the results is not valid for both cases, and will be highlighted later. Table 16 shows the improvements of one subset over another, understanding that smaller results are better. The comparison should be understood as how better is the first subset than the second one. For example for Mahalanobis_2 in the "MFCC With/Without" row the correct interpretation is that the average results including MFCC are 67.1% better that without MFCC. These results were arranged so the improvements are always positive, meaning that the first subset always has lower average that the second. One immediate conclusion is that, although the average's magnitudes vary, the tendencies are the same for all three distance functions and some parameters lead to better alignment all the tested cases.

It should be noted that these averages are at least twice as the best results shown in Table 14, showing that there is a considerable variance among the different feature configurations and that a correct choice of parameters is imperative.

5.6 Discussion

The MIDI rendered audio are not as realistic as the performed by musicians and lacks many elements that are difficult to reproduce: spontaneity, ambient noise (on live recordings), mistakes made by musicians, variations in timbre, etc. As is to be expected, these factors make music alignment easier and even in the most realistic case (DB3, where the tempo varies and chord are enriched) the best results are around half of the best results over the JAAH database. Anyways the results are on the same scale of magnitude and the reported differences are not unreasonable for the comparison of simplified laboratory data against a real-world case. When comparing MIDI and JAAH databases it is preferable to consider only the DB3 results, as it is more complex and similar to real music. It is also relevant to highlight that the same tendencies can be found on both MIDI (DB3) and JAAH databases, regarding how

Subset Type	Specific Subset	Average Over Distance Functions [s]		
		Euclidean	**Mahalanobis$_2$**	**Mahalanobis$_{all}$**
Hop Length	46 ms	2.34e-1	1.30e-1	1.34e-1
	92 ms	3.26e-1	1.78e-1	1.83e-1
	185 ms	4.33e-1	2.40e-1	2.51e-1
MFCC	With	3.00e-1	0.92-e1	1.00e-1
	Without	3.63e-1	2.79e-1	2.83e-1
Chromagrams	"Normal"	3.37e-1	1.61e-1	1.67e-1
	[bass]	3.52e-1	1.83e-1	1.88e-1
	[mid]	3.85e-1	1.86e-1	1.93e-1
	[bass-mid]	2.25e-1	1.18e-1	1.22e-1
	[bass-mid-treble]	1.79e-1	1.00e-1	1.03e-1
Tonnetz	"Normal"	3.17e-1	1.75e-1	1.81e-1
	[bass]	3.95e-1	2.43e-1	2.51e-1
	[mid]	4.00e-1	2.43e-1	2.50e-1
	[bass-mid]	3.12e-1	1.70e-1	1.76e-1
	[bass-mid-treble]	2.71e-1	1.48e-1	1.54e-1
Musical feature	Chromagram	3.22e-1	1.69e-1	1.74e-1
	Tonnetz	3.92e-1	2.72e-1	2.81e-1

Table 15: Average results for each mutually exclusive subset on JAAH database

Subset Type	Specific Subset Comparation	% of Improvement Over Distance Functions		
		Euclidean	**Mahalanobis$_2$**	**Mahalanobis$_{all}$**
Hop Length	46/92	28.2	26.8	27.0
	46/185	45.8	45.8	46.6
	92/185	24.6	26.0	26.9
MFCC	With/Without	17.3	67.1	64.7
Chromagrams	[bass-mid-treble]/"Normal"	46.9	38.0	38.0
	[bass-mid-treble]/[bass]	49.2	45.3	45.1
	[bass-mid-treble]/[mid]	53.5	46.3	46.4
	[bass-mid-treble]/[bass-mid]	20.7	15.3	15.6
Tonnetz	[bass-mid-treble]/"Normal"	14.4	15.4	14.9
	[bass-mid-treble]/[bass]	31.3	39.0	38.4
	[bass-mid-treble]/[mid]	32.3	39.1	38.2
	[bass-mid-treble]/[bass-mid]	13.0	12.9	12.3
Musical feature	Chromagram/Tonnetz	17.9	37.3	38.1

Table 16: Percentage of improvement of one sub set over another on JAAH database. It can be understood as how much improved the results of the first subset over the second one.

different distance functions behave or how each feature affects the overall alignment. This concordance confirms that MIDI based experiments are compelling and that the conclusions obtained from it are pertinent.

As can be seen in the results for DB0 and DB2, it is possible to achieve perfect alignment when tempo is fixed using Mahalanobis. Even the non-perfect results are clearly on a smaller scale magnitude that the cases where tempo varies. Sequences with fixed tempo will have equal length and the optimal path on the accumulated cost matrix is the diagonal and no time warping is needed. Additionally, if the sequences are very similar in a certain feature space under a distance metric it is not surprising that perfect alignment can be achieved, taking into account that fixed tempo alone is not enough to achieve this ideal result. On Table 11 can be seen that Euclidean distance metric is not capable to reveal the underlying similarities between two sequences, even if they are almost identical like in DB0.

Another essential aspect shown on the MIDI experiments is that variations on the harmonic-melodic content are not as challenging as tempo variations, in fact the shows results of DB2 and DB3 are very similar. Analyzing the whole set of averages from Table 10 can be seen that results from DB2 are around 2 times the ones from DB0. This ratio increases drastically when comparing DB0 to DB1, because under the Euclidean metric the latter is 7 times bigger and for both Mahalanobis options is around 14 times bigger.

The performance of the Mahalanobis distance functions is homogeneously better than using Euclidean distance. The only difference between these metrics is the inverse covariance matrix, that gathers relations between feature, showing data driven approaches, despite adding some computation time, are needed when comparing sequences whose features are complex and heterogeneous. As can be seen on Table 14 the best result achieved using Euclidean and Mahalanobis_2 have the same features and yet the errors are on another scale magnitude, confirming that the incidence of the covariances "flattening" caused by the inverse covariance matrix plays a major role. On the other hand, there is no considerable differences on Mahalanobis_2 and $\text{Mahalanobis}_{\text{all}}$ performances, but the latter achieves consistently higher error (around 3% higher). If there are no limitations on the computation time, it is preferable to compute the inverse covariance matrix for each pair of sequences.

One result that was not expected is the major role that plays the MFCC features on music synchronization, because it was added being considered a non-crucial feature that could lead to small improvements. Nevertheless, it is present among the best feature sets on almost every experiment and when using Malahanobis distance functions its impact is prominent, as can be seen on Table 16 the results including MFCC are 67.1% better results that the ones without these features. This improvement is considerably smaller using Euclidean distance, only 17.3%, revealing that MFCC is not only relevant when comparing to itself (like in Euclidean metric), but

really highlights when its relations with melodic-harmonic features can be considered. The MFCC are difficult to interpret and is mostly used as a descriptor of timbre and quality of sound (often used on speech recognition) and it does not reveal directly any pitch related musical information. Anyhow, it still a frequency descriptor that proves to be related with features like chroma or tonnetz and under a suitable metric these correlations can be revealed and used to improve music synchronization.

The only aspect that does not behaves coherently between the MIDI and JAAH databases is the effect of the hop length, which is rather opposite. Specifically, this unique behaviour is only found on the fixed tempo MIDI databases and lies on the fact that the 185ms hop length achieved better results that the others, which is different to the tendency shown on other databases. When analyzing these fixed tempo experiments globally the average results does not exhibit that tendency, actually the 46ms hop length results are slightly smaller, as can be seen on the Appendix. Anyhow, when there is certainty that the tempo does not change, the hop length is not a crucial parameter and bigger steps can be chosen.

On the non-fixed tempo cases a tendency is observed uniformly across all distance functions: smaller hop lengths produce better results. In real music tempo is constantly expanding and compressing itself on small amounts on a unpredictable way, making synchronization challenging as seen on the MIDI databases results. Higher time resolution will capture more specific features and it is to be expected that will lead to more accurate alignments, despite increasing the length of the sequences and drastically augmenting DTW algorithm complexity. Nevertheless, the complexity does not affect negatively the results and as can be seen on Table 16 46ms hop length results are around 45% smaller that 185ms and 27% that 92ms steps. A similar effect is seen on MIDI's DB1 and DB3.

The last elements to consider regarding the feature sets are the tonal-harmonic features. As discussed on Sec 5.2.2, every frequency range carry a different amount of musically relevant information, so the traditional chromagram losses some valuable information. By splitting in three overlapping frequency ranges and adding different combinations of them to the feature sets, is possible to know which ranges carry valuable information. The first observation is that the concatenation of all three chromagram reported consistently lower error that the "normal" chromagram and the highest improvement was achieved for Euclidean metric (46.9% improvement and for both Mahalanobis 38.0%). When comparing the three ranges features with single (bass or mid) or double (bass and mid) ranges, some musically interesting aspects are revealed regarding where is located the most important content. Comparing the full range (bass, mid and treble) features against only bass and only mid, lead to similar improvement but the former carrying slightly more information. In jazz and many other music genres most of the pitches are located on the mid frequency range (usually the melodies, piano or guitar chords, etc) and only some notes on the

bass range, nevertheless in this experiment both proved to carry a similar amount of meaningful content to perform music synchronization, although these ranges overlap on the octave comprised between C3 and C4. The treble frequency range is the one that reports smaller improvement when added to bass and mid ranges, but in average the lowest error is achieved considering the three ranges together. In the case of the tonnetz features the improvements achieved with Euclidean metric are similar or lower that for the other metrics. As for chroma features, both bass and mid ranges contribute a similar amount of information and treble a smaller amount.

The tonnetz features were initially proposed an alternative to the chromagram as it allows modelling musical relations between notes. Despite this particularity, the chroma reports better results for all three databases, but the gap is more significant for Mahalanobis metrics. An important insight about chroma features is that, from a psychoacoustic point of view, similitude between notes relies not only on the fundamental frequency, but on the number of harmonics they have in common, causing that two notes sound similar. This similarity is also expressed on the chromagram as the fundamental frequency and all the harmonics can be considered and included, and as the tonnetz is computed from the chromagram these harmonics are also included.

5.7 Conclusions

An extensive music synchronization experiment is presented using the DTW algorithm, including four variations of a MIDI jazz songs database and a real music jazz database (JAAH). The goal was to obtain insights on semi improvised music synchronization, which includes stylistic elements that difficult this problem, and have a deeper understanding on how different features and distance functions behave. A large feature set, that includes standard and modified frequency based features, was tested and compared, as well as three different Distance functions: Euclidean and Mahalanobis with two options for computing the inverse covariance matrix.

The simpler MIDI database contains songs that have fixed tempo and the only difference between choruses is the melody and instrumental solos (bass line and chords are copied between them). The second database introduce random small variations of tempo, the thirds is same as the first one but the chords are enriched with tensions and the base line is different for each chorus. The fourth, and more realistic, database includes both tempo variations and modifications on the melodic-harmonic content. The JAAH database is a collection jazz songs, including different subgenres, with written annotations of beats and chords and 82 songs were used.

The results over MIDI rendered audios showed that variations in tempo are considerably more challenging, for this musical genre synchronization, than variations on the harmonic and melodic content. When the tempo is fixed, the problem is not

so complex and perfect (or almost perfect) alignment can be achieved with the right feature-distance function combination. In these cases, the choose of a hop length is not vital because results are similar and higher steps (like 185ms) can be used to have shorter sequences and decrease the DTW complexity. On all other cases where tempo varies, including the JAAH database, smaller hop lengths are preferable as a higher time resolution have proven to obtain smaller error.

The Mahalanobis metric has proven to produce results with around 45% smaller error that using Euclidean metric, and there is no significant impact, on neither computation time nor performance, if the inverse covariance matrix is computed from the whole song or only from the two sequences being analyzed. This reveals that there are noteworthy correlations between data and that taking them into account is crucial, specifically when combining heterogeneous feature spaces.

The MFCC are not usually used for music synchronization, but combined with other melodic-harmonic features leads to important reductions on the alignment error, especially when using Mahalanobis distance function (67.1% lower in average). On the same line, the tonal centroid features (tonnetz) were added to the feature set, but consistently performed worse that the chromagram. Splitting the frequency range, spanning from C1 to C8, on three overlapping ranges allows to obtain range focused chromas, which obtained a improvement of 38%, using Mahalanobis, compared to its traditional version.

In this experiment, the DTW algorithm was not modified, and the focus were the distance functions and features. Taking as a baseline the Euclidean metric with the "normal" chromagram, and comparing it with the best result on the JAAH database using Mahalanobis_2 metric, depicted in Table 14, the former has an error 5.6 times higher (0.279ms vs 0.0492ms). This proves that a correct choice of distance function and feature set is vital when performing alignment between different choruses of a jazz song, as they can lead to significantly better results.

6 AUTOMATIC CHORD TRANSCRIPTION

In this chapter the chord transcription stage is described, which analyzes jazz song whose choruses have been previously segmented and aligned. We propose a dual-objective chord transcriber system that can jointly predict beat positions and chords of a jazz song, considering the contribution of all choruses. The beat detector embedded into the system is a replacement of the beat onset detector stage, which cannot be implemented because publicly available jazz databases do not include bass line transcription. Nevertheless, it does include beat onset times, which also represent temporal events closely related to chord transitions and most of the obtained from Chapter 4 are valid.

6.1 Context

Modern approaches to automatic chord transcription tend to be deep learning based, which can combine effectively the extraction of musically relevant features and a sequence analysis that provide temporal coherence to the chord predictions (McFee and Bello, 2017; Jiang et al., 2019; Wu et al., 2019). In comparison with non-integrated systems, that perform each step as an independent process, this kind of architecture allows to input data with little preprocessing and output directly class probability for each chord (or each chord component), that can be postprocessed, but chord labels can already be obtained. One of the advantages of this workflow is that context, in both time and frequency, can be learned on a data-driven manner, allowing the system to incorporate a wide range of musical features.

Most of ACT systems focus on predicting a list of chord labels from a musical piece, considering it as a single sequence. Generally, it is expected to obtain these chords according to the explicit musical content from the audio and state-of-the-art systems can achieve a very detailed description of chords constituent elements. For example, large-vocabulary systems can determine the bass note and each musical tensions beyond the chord triad (seventh, ninth, etc), working with more that 200 classes, like in (Jiang et al., 2019).

6.2 Proposed Approach to Jazz Songs ACT

Jazz music has some properties that are not shared with other genres, causing some assumptions considered by past approaches to no longer be valid regarding the way chords are manifested and how harmony is expressed. The effects and limitations of ACT on jazz music has only been addressed, to our knowledge, by Eremenko

(2018), where a large jazz database is presented and analyzed. In their work, state-of-the-art systems are tested, showing around half of the accuracy than on pop-rock music databases and additionally the author proposed an ACT system that achieved similar performance to the state-of-the-art.

Every chorus of a jazz song contains variations of a same harmonic progression and the global, or "essential", chords can only be retrieved analyzing more than a single chorus. Our main goal is to obtain a lead sheet-like chord transcription, inspired on the chord sequence representation used in the "Real Book", which is a compendium of well-known jazz songs.

The content of all choruses is obtained by averaging them frame by frame and we explore three different approaches to do so: The first one is inspired by the work presented by Mauch et al. (2009) and averages the time-frequency representation of the audios prior to perform automatic chord transcription. The second approach averages the intermediate representations, consisting on the latent features and the intermediate supervised predictions obtained from it. The third one performs chord transcription on each chorus independently and averages the output class probabilities before choosing the most probable chord.

We use a deep learning architecture based on convolutional recurrent neural networks (CRNN) and test the proposed methodology on a jazz music database. Details of the network's implementation, database manipulation and chord transcription evaluation are presented in the following sections. We conclude presenting and discussing the results, including quantitative and qualitative analysis.

6.3 Deep Learning Network Architecture

6.3.1 Referent: CREMA

A drawback of deep learning based methods is the difficulty to understand what the model is learning and it is non-trivial to obtain insights that can lead to further improvements on ACT systems. The model proposed by McFee and Bello (2017), that became publicly available under the name of CREMA[5], achieves flexibility to learn high level frequency content, while providing a good understanding on the network's behavior.

The system they propose is inspired on a encoder-decoder architecture, that inputs a spectrogram-like audio representation and outputs predicted chord classes. Although the encoder and decoder part are integrated on a single system, they can be described separately. The former is responsible for the feature extraction from

[5] `https://crema.readthedocs.io/`

the input, while the latter analyzes this intermediate representation, called latent features, performing sequence analysis and predicting chord classes.

The network's training include intermediate supervision, that the authors call structured training, consisting on training targets that should be learnt by the encoder part. These targets correspond to elements that put together constitute a chord: root note, pitch classes and bass note. This allows the network to learn elements that are shared between similar chords, instead of only learning mutually exclusive chord classes. These intermediate outputs force the encoder to retrieve latent features that describe musically meaningful elements, allowing an easier understanding of the network's behaviour.

Intermediate training targets are presented as follows: The first is a 13-length vector that represents root note, the second has 12 elements and represents the pitch classes active on a chord (as binary template) and the third, similarly to the first, has 13 elements but represents the bass note. Root and bass consider the 12 musical notes and a symbol for no chord (N).

The model presented by McFee and Bello (2017) and the latter implementation made in CREMA are slightly different, the most significant difference is that the latter includes a postprocessing stage consisting of a HMM that models chord transitions probabilities and obtain the final chord sequence using virterbi decoding. This postprocessing smooths chord transitions and corrects some bad predictions, although its impact is not reported.

6.3.2 Dual-Objective Architecture

The deep learning architecture implemented in this research is strongly based on CREMA, but introduces some variations. First, the number of convolutional layers and their number of filters is increased and second, the structured training targets are not the same.

It is desirable that the network can retrieve information not only explicitly from the audio, but from higher level musical concepts. For example, the bass line (usually found as a *walking bass*) will not play only the root of the chord, but also other notes related to that chord, like its third and fifth grades. In this case, we cannot rely on the transcription of a particular bass note, but on the relation between all the notes played on the scope of a chord. Thus, the bass note cannot be used as an intermediate target, because it is not possible to find a single bass note corresponding to a chord.

Instead, we added a beat detector to the intermediate supervision acting as a binary classifier. It is common that chord transitions are aligned to beats, thus beat onsets should be considered as it encourages the system to only transition from one chord to another on beat times. This element takes advantage of the close relation existing between bass note onsets and beat onsets due to the *walking bass* technique,

which facilitates the beat detection task. Like is presented by McFee and Bello (2017), the intermediate outputs are concatenated with the latent features and fed to a bidirectional gated recurrent unit (BGRU) network that act as a decoder and generates class predictions on a one-to-one manner.

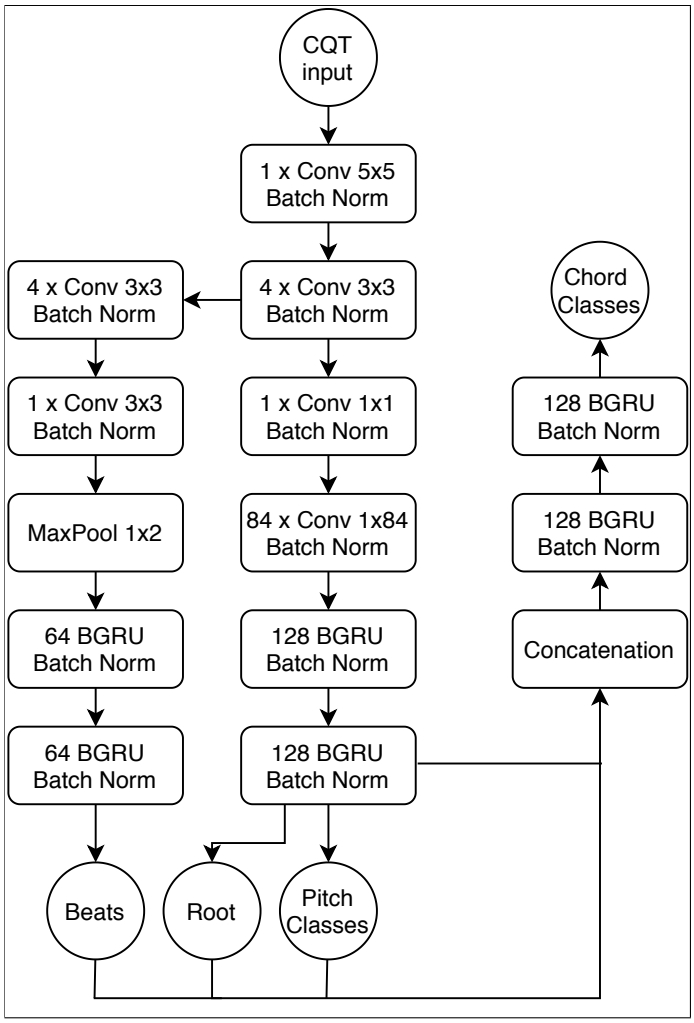


Figure 13: Architecture of the CRNN

The full architecture, depicted in Fig.13, has 1,087,194 parameters and the details of implementation, regarding input data and training are shown below.

a) Input Data:

From the raw audio, the CQT is computed with one bin per semitone spanning from 32.7 Hz to 3951.1 Hz (from C1 to B7), using a hop length of 46 ms. The result has 84 frequency bins and the only preprocessing performed is a tuning correction, ensuring that every bin represent the same frequency content for all songs.

b) Training:

As this model includes multiple outputs, the total loss (objective function for the parameters estimation) is the sum of four different components: The root loss, the chord pitches loss, the beat onset loss and the chord loss. All of them are considered as classification problems with mutually exclusive classes, except for the chord pitches, that is addressed as a multi-label task. Consequently, for roots, pitch classes and chords the categorical cross-entropy loss is used (CE) and for beats a binary cross-entropy loss (BCE). Each loss function is shown in Eqs (15,16), where t_c is the true label for class c and p_c represents the prediction score for that class. Obviously, BCE is just CE with two classes, but is shown independently to emphasize the binary nature of the beat detection problem. The total loss is described in Eq. 17, representing the sum of every individual component scaled by factors α, β, γ and δ. These coefficients were hand-tuned according to the error ranges of each component, aiming to ensure the predominance of the chord loss over the intermediate outputs.

$$BCE = -t \log(p) - (1 - t) \log(1 - p) \tag{15}$$

$$CE = \sum_{c}^{C} -t_c \log(p_c) \tag{16}$$

$$Total_{loss} = \alpha(CE_{roots}) + \beta(CE_{pitch}) + \gamma(BCE_{beats}) + \delta(BC_{chords}) \tag{17}$$

The training was performed in patches of 5.52 seconds chosen randomly at each iteration over the database (epoch) from a pool of all training patches. The batch size was 64, using the ADAM optimizer.

The validation, that acts as a performance indicator during training that is used to prevent overfitting, was not performed on random patches, but on randomly selected songs, that were put aside at the beginning of training stage. We found out that validating on patches from the same songs used for training does not reveal overfit, because of the high similarity between these segments.

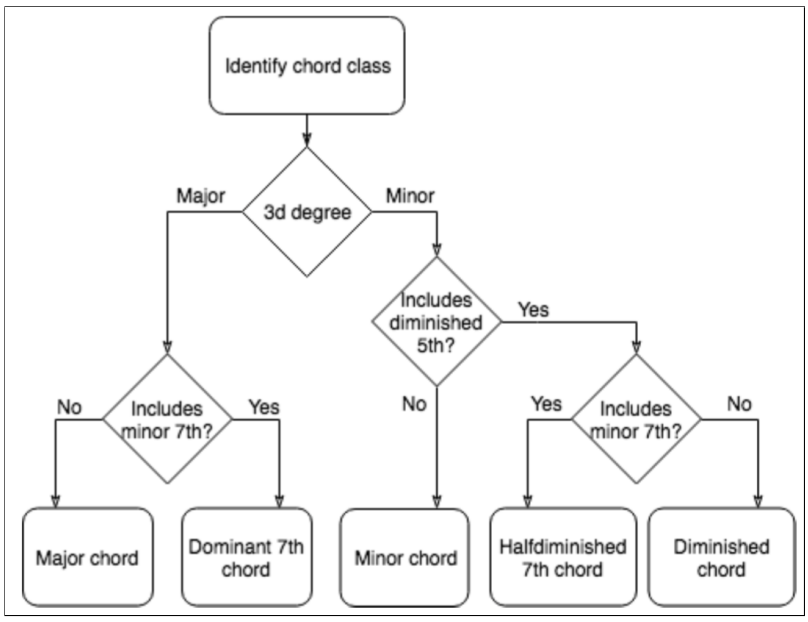


Figure 14: Flowchart explaining how chords are classified in this work. Extracted from Eremenko et al. (2018).

6.4 Experiment

6.4.1 Chord vocabulary

In this work, the chord dictionary proposed by Eremenko et al. (2018) will be used, where insights are provided regarding how chords should be classified in jazz music, resulting in five classes that play a different harmonic function. These chord classes are major (maj), minor (min), major with dominant seventh (7), half-diminished seventh (hdim7) and diminished (dim), and are assigned following the instructions that can be seen on the diagram depicted on Fig.14, which is extracted directly from Eremenko et al. (2018). Additionally a no-chord symbol is considered (N), resulting in 61 classes.

6.4.2 Database

The JAAH database presented by Eremenko et al. (2018) suits perfectly to the purpose of this work, including manually annotated chords for jazz songs, chord boundaries and beat onsets. As we are interested on exploring the impact of chorus repetitions on ACT, songs that do not have two or more choruses should not be

used. The database is filtered and from the 113 songs, only 80 are used for training and 6 for testing[6], corresponding to almost 4 hours for training and 15 minutes for testing. The train-test split is not performed randomly, because that does not ensure that all chord types are present in the testing subset. Anyhow, these songs were not chosen following any particular subjective criterion. At training, 2 song are randomly selected to perform validation, the whole data split is summarized in Table 17.

Train	**Validation**	**Test**	**Cross-Validation?**
89.7% (229 minutes)	4.5% (11 minutes)	5.8% (15 minutes)	No

Table 17: Database split used to train the CRNN model. **Train** is the data used for training, **Validation** is a set used to validate performance after each training iteration and **Test**, also known as hold-out, is the data used only to test the trained models and is not included on the training process.

The annotated chords were mapped to the used chord vocabulary following these steps: First, inversions are omitted and all chord tensions other than seventh are suppressed (9, 13, etc). Second, the chord class is assigned according to Fig.14.

The annotated chords are not equal between choruses, because the manual transcription include variations that might have been played. For example, the chorus where the melody is performed may contain some special arrangements that are not included on the solos, and the annotated harmonic sequences are slightly different. These detail elements are undesired in our general harmonic analysis, even if they are useful for other harmony analysis tasks. To solve this issue, we compare the chords of each measure across all choruses and choose the statistical mode. This process introduces losses, but aims to mimic what a musician would do if asked to transcribe the chord sequence of a jazz song on a single chorus lead sheet.

The chord distributions are highly biased to the most frequent keys in jazz song, that are the ones easier to play in instruments like saxophone or trumpet (Eb, Bb, C, etc). To prevent the system to bias towards those keys, we include data augmentation consisting of shifting each song on -5 and +6 semitones. This process can be noisy because it involves varying the pitch without changing te song speed or length. It is critical to perform a good quality pitch shifting, to prevent artifacts and too much "phasiness", so the Rubberband[7] library was used, which is specialized in changing the speed and pitch of audio.

[6]Test songs can be heard at `https://open.spotify.com/playlist/5zMErSoqMBcmIiunMmF4JC?si=RM7G_-kMQJ-csblIwN5F2A`

[7]`https://rubberbandaudio.com/`

6.4.3 Chorus Averaging

Besides the work presented by Mauch et al. (2009) there is no precedent, to our knowledge, on how structural segmentation can be integrated to ACT, so we explore three alternatives, depicted in Fig.15, and compare the results:

a) CQT averaging: Originally presented by Mauch et al. (2009), they mix the audio content before being analyzed. The resulting averaged chorus can be messy, especially if the song contains many choruses, as all notes played on a measure across choruses will be considered.

b) Latent Features Averaging: The latent features, from which the intermediate outputs are obtained, carry most of the melodic/harmonic musical content extracted by the network. This content is already refined from the CQT form and probably prioritize notes that preponderate on the chord. As the decoder part of the system inputs the concatenation of the latent features and the intermediate outputs, the latter are also averaged, and only the resulting mixed "latent chorus" is decoded.

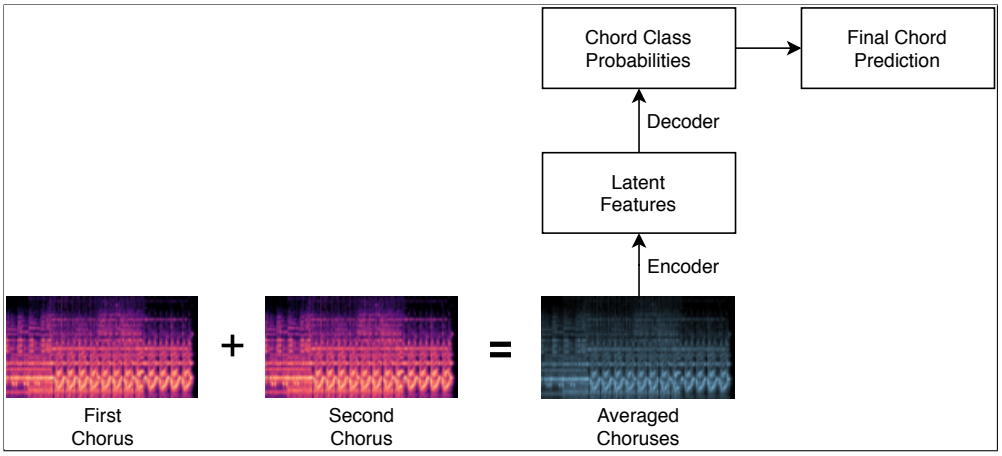
c) Predictions Averaging: The final chord probabilities present the most polished content and the transcription from a single chorus is, ideally, correct or not far, because it predicted similar chords. Thus, averaging these probabilities between all repetitions, rather that introduce noise, sum up the prediction of all choruses.

6.4.4 Evaluation Metrics

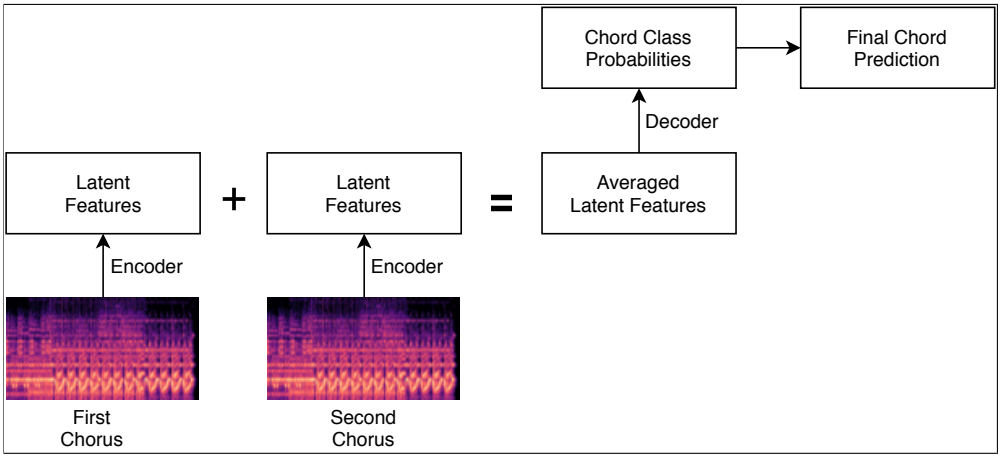
The problem of measuring the performance of ACT systems is complex, because there are many musically meaningful elements that must be considered. Chord classes cannot be considered as independent, because some of them share most of the pitch content, for example C and C7, or Dmin and F.

Instead of computing a single evaluation metric, a set of them is used, where each express different aspects of the chords, providing a deeper understanding of how harmony is being transcribed. In our case, chords are evaluated element-wise because both ground truth and predicted sequences have the same length.

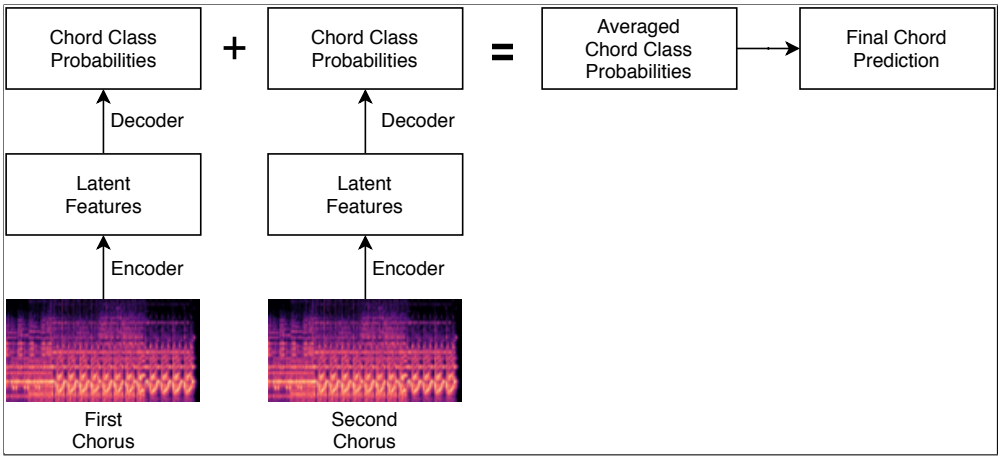
There are standard evaluations implemented by Raffel et al. (2014) on the Python library MIR_EVAL, but not all of them are appropriate to the chord classes used in this work. In particular, we use the standard accuracy and the following subset of evaluation metrics:



(a)



(b)



(c)

Figure 15: Three alternatives for choruses averaging. (a) correspond to CQT averaging, (b) latent features averaging and (c) chord class probabilities averaging.

	Acc	Root	Maj/Min	Third	Triad	MIREX
no-averaged	0.52	0.59	0.59	0.58	0.58	0.60
CREMA	0.36	0.49	0.47	0.46	0.46	0.48
cqt-av	0.60	0.66	0.67	0.64	0.64	0.67
lat-av	0.62	0.69	0.68	0.67	0.66	0.68
pred-av	**0.64**	**0.70**	**0.71**	**0.68**	**0.68**	**0.71**

Table 18: Chord evaluation metrics. The ones above show performances for all choruses. Results below show methods that combine predictions and thus obtain chords for only one chorus.

a) root: Only compares if chords have the same root.

b) maj/min: Compares if chords share the root and the chord type, if it is major or minor. Other chord classes are not considered.

c) third: Compares if chords share the root and third. For example, chords Emin and Edim match.

d) triad: Compares if chords have the same basic triad. For example, chords E7 and E match, but Emin and Edim do not.

e) MIREX: Chords match if they share at least three pitch classes. For example F7 and Adim match. This evaluation metric is used on the Music Information Retrieval Exchange (MIREX) competitions on chord transcription.

The beat detection problem is very similar to the bass onset detection presented in Chapter 4 and the same evaluation metrics are used, i.e precision, recall and F-measure. The hop length used in this stage is 46ms (8 times larger than for bass onset stage), which implies that a tolerance of $\pm$25ms forces the frames to match exactly, which highly penalizes performance. On the other hand, $\pm$50ms correspond to $\pm$1 frame tolerance, that allows more flexibility. Both results will be considered, to show the sensibility of the detection evaluation metrics in this problem.

6.5 Results

Averaging choruses on the CQT is abbreviated cqt-av, on the latent features lat-av and on chord predictions pred-av. The results over all choruses, without averaging, is also shown as no-averaged.

To compare our chord transcription results with a state-of-the-art system, we choose CREMA and compute only the evaluation metric pertinent to our study

	Precision	Recall	F-measure
no-averaged$_{50ms}$	0.89	0.87	0.88
cqt-av$_{50ms}$	0.96	0.94	0.95
lat-av$_{50ms}$	0.95	0.94	0.95
no-averaged$_{25ms}$	0.51	0.50	0.50
cqt-av$_{25ms}$	0.58	0.57	0.58
lat-av$_{25ms}$	0.63	0.62	0.63

Table 19: Beat detection metrics.

(Section 6.4.4). In Eremenko et al. (2018) is presented the performance of CREMA over the whole database, but is not compatible with our version because we only test on six songs and they only provide total accuracy (scored 40.26%). Chord transcription results are shown in Table 18.

We also computed a confusion matrix for the roots and for the chord type (maj, 7, min, etc), shown in Fig.16-17. The rows represent the annotated data and the columns the predicted. These matrices are normalized over the annotated data, so the sum of each row is 100, meaning that these values are not absolute. For example, in Fig.16 all the "N" symbols were predicted as Eb, which does not mean that our system will always confuse them. In fact, just a few "N" symbols are available on the test set and all of them were classified as Eb.

The beat detection results are shown in Table 19. It is important to notice that, except for no-averaged, these beats correspond to the beat sequence of a single chorus and not of the whole song.

The predicted chord sequence is converted to a lead sheet representation. An example is depicted in Fig.18, but all the obtained lead sheets can be found on the Appendix.

The JAAH annotations provide a general description of the harmonic content, but do not provide the details of all notes being played. From that perspective, is hard to obtain insights from the mistakes that our system makes. When a chord is misclassified it will be beneficial to know if the notes of the ´´wrong chord" were played. To obtain a deeper qualitative understanding, the same MIDI database described in Chapter 5.3.1 was analyzed by our system. We are interested on the chords that are misclassified under the MIREX evaluation metric (is the most flexible), because they correspond to the most critical errors.

To analyze the MIDI content, beat-synchronous chromagrams are computed from the MIDI, so they only include the fundamental frequencies. This a easy to read strategy to obtain insights. For example, for the song "Afternoon in Paris" the measures 13 to 16 represent an example of bad chord transcription, as seen on Fig.19. These results are extended in the Appendix.

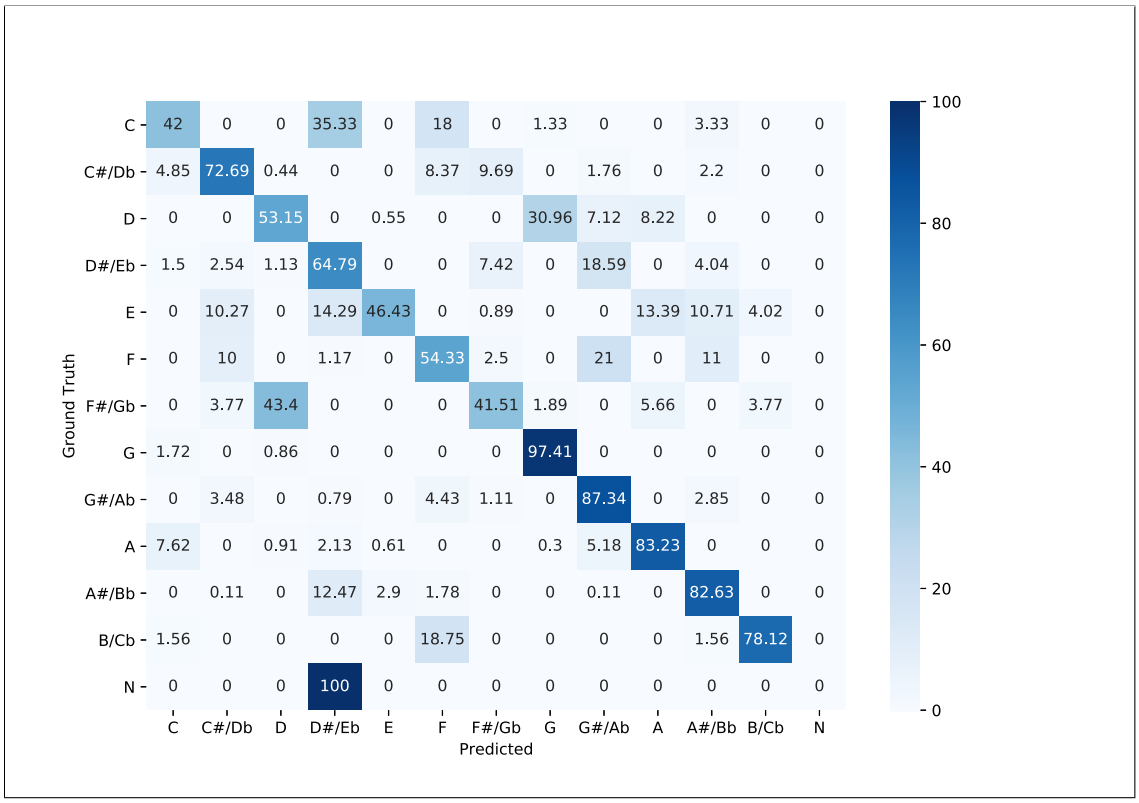


Figure 16: Roots confusion matrix.

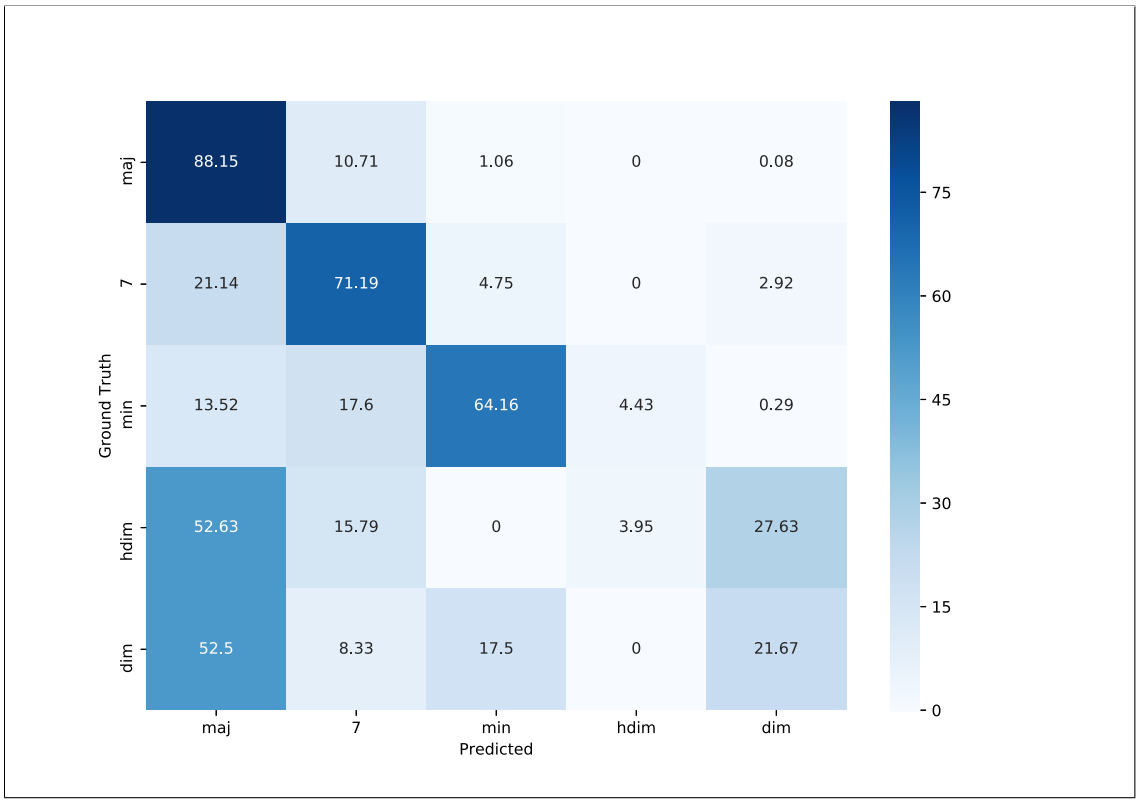


Figure 17: Chord type confusion matrix.

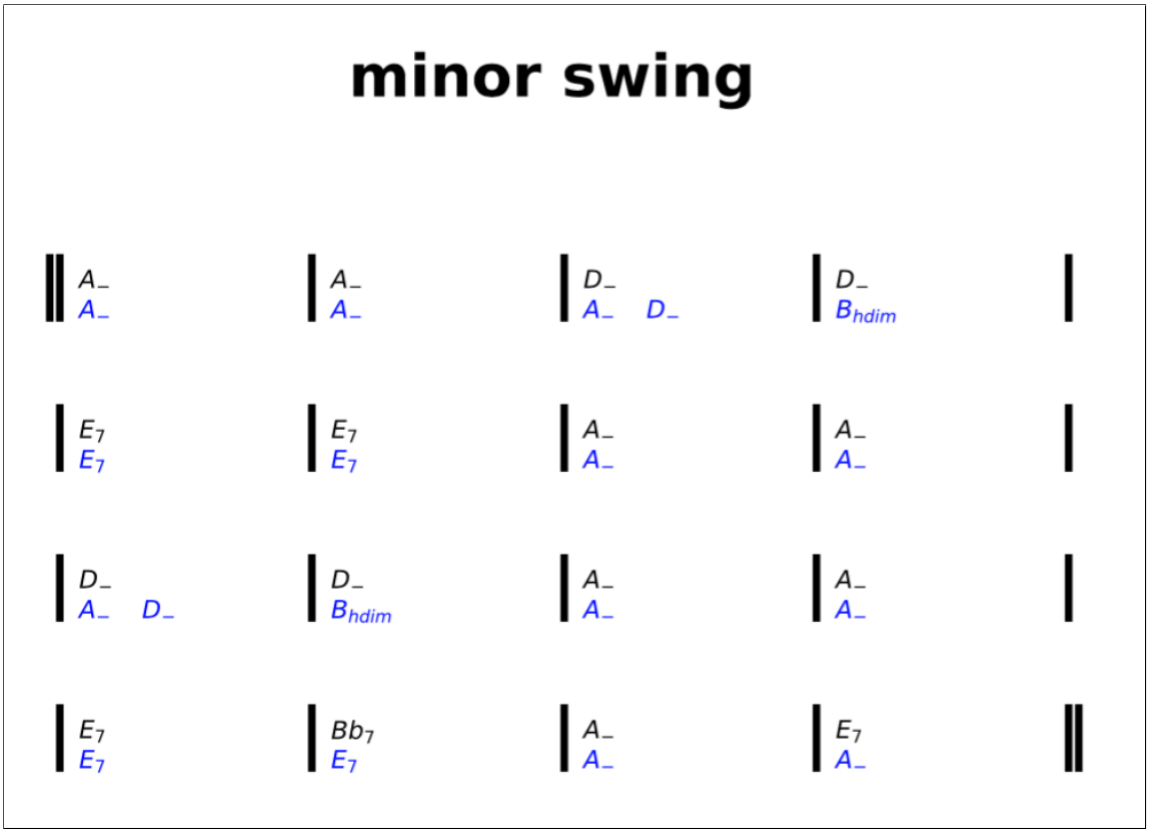
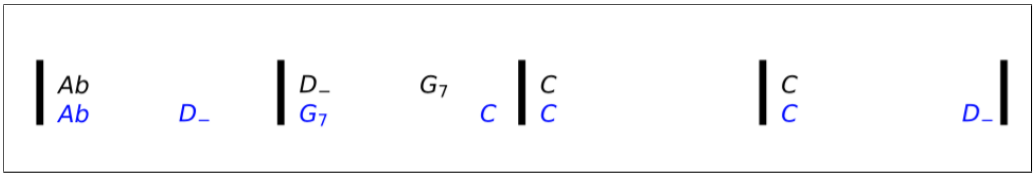


Figure 18: Obtained lead sheet-like representation for song "Minor Swing". Annotated chords are in black and predicted in blue.

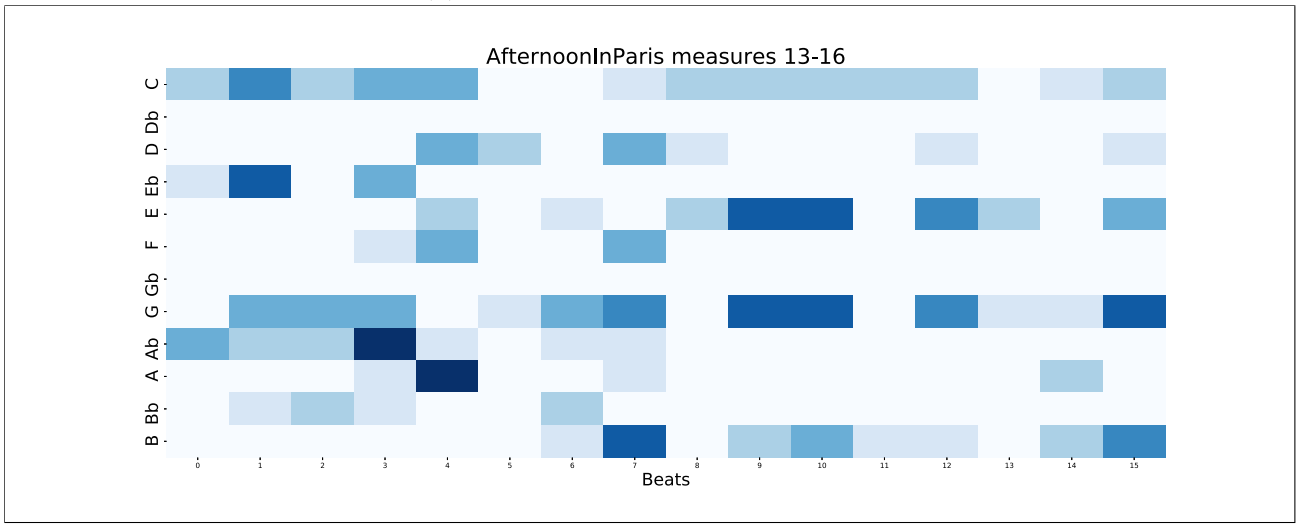
6.6 Discussion

As can be seen on Table 18, our system gets considerably higher performance than CREMA (at least 10% for each metric), when choruses are not averaged. Our model architecture is similar to the one used by CREMA, but the latter implement, as postprocessing, hidden Markov models to enhance the chords transitions. The performance improvement can be explained by the fact that our model is trained and tested on a genre specific database and probably will not perform properly on other genres. At the same time, it confirms that the reference system, trained on a larger dataset than ours, does not generalizes correctly to jazz music and if retrained considering this genre, will probably be more robust.

In general, the system detects correctly the chord changes, especially on coarse time quantization like beats. Nevertheless, around 29% of the chords are misclassified (using the MIREX evaluation metric) and it is systematically found that these error are mostly caused by chord transitions that are advanced or delayed in more than



(a) Lead sheet-like chord sequence.



(b) Beat-aligned chromagram.

Figure 19: MIDI version of song "Afternoon In Paris", measures 13 to 16. (a) shows the annotated chord sequence in black and the predicted in blue, on a lead sheet style. (b) presents the pitch content for the same measures shown in (a), on an idealized chromagram.

one beat. As depicted in Fig.19, during the first two measures the predicted chords change sooner than they should, even if the pitch content is not related to the prediction. This effect is found even when the two chords involved are not similar, implying that the error does not depend directly on the musical content of that temporal frame. Instead, it is caused by the decoder, which is capable to capture musical relationships between frames, but can introduce delays. It is worth noting that the system uses bidirectional gated recurrent units, so the concept of delay can be applied to both temporal directions.

This delay introduced by the system in some cases might be the result of a smoothing effect, where the chord is not predicted only from the current frame, but from a wider context, and temporal precision might be reduced during the process. An option to correct this problem is to implement, like in CREMA, a postprocessing stage using HMM and some efficient decoding to model chord transition probabilities. Anyway, it is desirable for jazz music chord transcription that the output does not

depend excessively on each frame's pitch content, but on a wider context. From that point of view, even if this temporal delay decreases the evaluation metrics, the loss in temporal precision is not that critical because the harmonic sequence is being modeled correctly.

Even if evaluation metrics were carefully chosen and the chord dictionary is well suited for jazz music, there are aspects regarding how information is notated and how performance is measured, that can be improved for this musical genre. As can be seen in Fig.18, on the 14th measure the annotated chord is Bb7 and the predicted chord is E7. Under all current evaluation metrics the predicted chord is wrong, but the two chords are closely related because they are tritone substitutions (a technique widely used in jazz) and there should be at least one metric that takes this aspect into account. Also, the chord dictionary imposes some limitations to evaluation metrics because they are oversimplified. For example, an annotated C that is predicted as Emin will be considered a misclassification even if the chords are similar.

7 GENERAL CONCLUSIONS

7.1 Conclusions

We presented a computer system for automatic chord transcription on jazz music, that performs an exhaustive analysis of the audio incorporating the musical attributes of the genre. The research focused on specific aspects of jazz music that were exploited, in particular the repetitive nature of a fundamental segment called chorus and the stylistic bass playing technique, called *walking bass*, which contains essential harmonic content.

A detailed study on two sub-tasks, that were considered as part of our system, was performed and presented on individual chapters. The first correspond to bass line onset detection, which is vital because these onsets are temporal events closely related to chord transitions. The second is music synchronization, that in the case of jazz music is crucial for aligning different choruses. These two sub-researches concluded on conference papers, one has already been accepted and the other is on hold to be accepted. The last stage is chord transcription, which has not been studied with the same depth as bass onset detection and music synchronization.

The system inputs the raw audio and the choruses boundaries. The first steps consist on aligning the choruses, incorporating the insights obtained from the study on music synchronization. The second step inputs the CQT of each chorus and transcribes the chords and the beat positions on a single system. The third step converts the chord list into a lead sheet representation, that is user friendly and is the standard format to represent the chord progression of jazz songs.

From the bass onset detection sub-task, we conclude that is a problem that can be satisfactorily solved using deep learning methods, even without a very large training database. Other signal processing methods also achieve good results, but are clearly outperformed by the deep learning based methods, in particular using convolutional neural networks. In jazz music bass onset and beats are related because of the *walking bass* technique, and we extended the bass onset detector into a beat detector embedded in the chord transcription system.

The music synchronization study focused on the features inputted to the dynamic time warping algorithm, and how distance metrics exploited their similarities. We found that using large feature sets, that describe musical aspects like pitch and timbre, are considerably better at unveiling similarities between musical segments that single features like chomagram. We also identified the Mahalanobis distance metric as the most suitable for this task.

Regarding the chord transcription stage, we merged a beat detector and a chord predictor into a single system. We explored three options to combine the harmonic

content of each chorus and we concluded that averaging the chord probabilities provides the best results. This means that combining the sequences after being processed by a trained system leads to higher performance that mixing the raw inputs or the intermediate latent features.

We believe that this research provides useful insights regarding how MIR can be applied to jazz music, showing that previously developed methods, that were designed for other genres, can not be extended to jazz. Useful insights were presented regarding how bass onset detection can be performed. On the synchronization subtask is presented a features-distance metric combination that had not been used before and achieves improved results for choruses alignment. The exploration of three methods to average each chorus's content proves that averaging over chord predictions leads to higher scores. ACT and beat detector systems were combined, allowing to have better beat-synchronized chord sequences which improves readability and discourages out-of-beat chord transitions. The last contribution is that we made public our code and trained models, allowing music enthusiast, jazz amateurs and even professional musicians to obtain the transcribed lead sheet-like chord progression from a song using our system.

7.2 Future Work

The presented chord transcription system performs all the needed stages except for the music structure analysis, because chorus boundaries are currently given as input. We decided to exclude this sub-task because it is difficult enough to be a new research topic by itself, going beyond the scope of this master's thesis. Some exploratory tests were performed using the standard structure segmentation methods, obtaining poor results and suggesting that for this musical genre new methodologies should be examined. In particular, deep learning methods have been used recently and some researchers are currently working on improved approaches, which could be easily included to our work due to its modular workflow.

The results we presented on jazz chord transcription are considerably lower than for other genres, like pop or rock. That shows a research opportunity towards jazz music analysis, there is room for improvement not only on better deep learning models, but on the way musical knowledge is transferred into computer systems.

The JAAH database, which is used train the ACT system, is moderately large and sufficient for many musical research tasks. Nevertheless, it is not large enough to train a robust chord transcription system without overfitting. In our case, the system was trained a few iterations over the training set before validation loss started to raise. We believe that a larger jazz songs database could significantly improve the results especially for chord classes that have little presence, like diminished or half diminished.

Regarding the genre focused analysis we presented for jazz music, the musicological perspective was included superficially and the musical analysis was not performed with the adequate depth. These aspects were not further explored because they are beyond the scope of this research. The next steps of this work should include an exhaustive literature review on jazz harmony and probably needs supervision from a jazz musician, musicologist and/or music historian. A better understanding from the musical perspective may lead to changes on the way an ACT system is designed and implemented.

Focused MIR in jazz music is relatively unexplored, we believe that there are many challenges that are still unresolved, especially regarding how music is annotated and evaluated. We strongly believe that on this genre the harmonic related tasks should focus on a general perspective, rather than on a detailed analysis. Nevertheless, at the same time the details and stylistic elements should be incorporated as well, so at training time the system should learn to discriminate the relevant content from the others. From that perspective, not all standard chord evaluation metrics are compatible with chord dictionaries that suits jazz music, so new metrics can be proposed for this task.

References

Bello, J. P., Daudet, L., Abdallah, S. A., Duxbury, C., Davies, M. J., and Sandler, M. B. (2005). A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13:1035–1047.

Bello, J. P. and Pickens, J. (2005). A robust mid-level representation for harmonic content in music signals. In *ISMIR*, pages 304–311.

Böck, S., Arzt, A., Krebs, F., and Schedl, M. (2012). Real-time onset detection with recurrent neural networks.

Böck, S., Korzeniowski, F., Schlüter, J., Krebs, F., and Widmer, G. (2016). madmom: a new Python Audio and Music Signal Processing Library. In *Proceedings of the 24th ACM International Conference on Multimedia*, pages 1174–1178, Amsterdam, The Netherlands.

Boulanger-Lewandowski, N., Bengio, Y., and Vincent, P. (2013). Audio chord recognition with recurrent neural networks. In *ISMIR*.

Brown, J. C. (1991). Calculation of a constant Q spectral transform. *The Journal of the Acoustical Society of America*, 89(1):425–434.

Cho, T., Weiss, R. J., and Bello, J. P. (2010). Exploring common variations in state of the art chord recognition systems.

Dittmar, C., Pfleiderer, M., Balke, S., and Müller, M. (2018). A swingogram representation for tracking micro-rhythmic variation in jazz performances. *Journal of New Music Research*, 47:113 – 97.

Duan, Z. and Pardo, B. (2011). Aligning semi-improvised music audio with its lead sheet. In *ISMIR*.

Eremenko, V. (2018). Automatic harmony analysis of jazz audio recordings. Master's thesis.

Eremenko, V., Demirel, E., Bozkurt, B., and Serra, X. (2018). Audio-aligned jazz harmony dataset for automatic chord transcription and corpus-based research. In *ISMIR*.

Ewert, S., Müller, M., and Grosche, P. (2009). High resolution audio synchronization using chroma onset features. *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1869–1872.

Eyben, F., Böck, S., Schuller, B. W., and Graves, A. (2010). Universal onset detection with bidirectional long short-term memory neural networks. In *ISMIR*.

Fujishima, T. (1999). Realtime chord recognition of musical sound: a system using common lisp music. In *ICMC*.

Hainsworth, S. W. and Macleod, M. D. (2001). Automatic bass line transcription from polyphonic music. In *ICMC*.

Harte, C. (2009). Automatic chord recognition using quantised chroma and harmonic change segmentation.

Harte, C., Sandler, M., and Gasser, M. (2006). Detecting harmonic change in musical audio. In *AMCMM '06*.

Hawthorne, C., Elsen, E., Song, J., Roberts, A. M., Simon, I., Raffel, C., Engel, J., Oore, S., and Eck, D. (2017). Onsets and frames: Dual-objective piano transcription. In *ISMIR*.

Humphrey, E. J. and Bello, J. P. (2012). Rethinking automatic chord recognition with convolutional neural networks. *2012 11th International Conference on Machine Learning and Applications*, 2:357–362.

Humphrey, E. J. and Bello, J. P. (2015). Four timely insights on automatic chord estimation. In *ISMIR*.

Humphrey, E. J., Cho, T., and Bello, J. P. (2012). Learning a robust tonnetz-space transform for automatic chord recognition. *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 453–456.

Izmirli, Ö. and Dannenberg, R. B. (2010). Understanding features and distance functions for music sequence alignment. In *ISMIR*.

Jiang, J., Chen, K., Li, W., and Xia, G. (2019). Large-vocabulary chord transcription via chord structure decomposition. In *ISMIR*.

Karjalainen, M. and Tolonen, T. (1999). Multi-pitch and periodicity analysis model for sound separation and auditory scene analysis. *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, pages 929–932 vol.2.

Korzeniowski, F. and Widmer, G. (2016). Feature learning for chord recognition: The deep chroma extractor. In *ISMIR*.

Lin, T.-Y., Goyal, P., Girshick, R. B., He, K., and Dollár, P. (2017). Focal loss for dense object detection. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007.

Mauch, M. (2010). Automatic chord transcription from audio using computational models of musical context.

Mauch, M., Noland, K. C., and Dixon, S. (2009). Using musical structure to enhance automatic chord transcription. In *ISMIR*.

McFee, B. and Bello, J. P. (2017). Structured training for large-vocabulary chord recognition. In *ISMIR*.

McFee, B., Raffel, C., Liang, D., Ellis, D. P. W., McVicar, M., Battenberg, E., and Nieto, O. (2015). librosa: Audio and music signal analysis in python.

McVicar, M., Santos-Rodríguez, R., Ni, Y., and Bie, T. D. (2014). Automatic chord estimation from audio: A review of the state of the art. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22:556–575.

Müller, M. (2015). Fundamentals of music processing: Audio, analysis, algorithms, applications.

Müller, M., Ewert, S., and Kreuzer, S. (2009). Making chroma features more robust to timbre changes. *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1877–1880.

Ni, Y., McVicar, M., Santos-Rodriguez, R., and Bie, T. D. (2012). An end-to-end machine learning system for harmonic analysis of music. *IEEE Transactions on Audio, Speech, and Language Processing*, 20:1771–1783.

Ni, Y., McVicar, M., Santos-Rodriguez, R., and Bie, T. D. (2013). Understanding effects of subjectivity in measuring chord estimation accuracy. *IEEE Transactions on Audio, Speech, and Language Processing*, 21:2607–2615.

Pachet, F., Suzda, J., and Martínez, D. (2013). A comprehensive online database of machine-readable lead-sheets for jazz standards. In *ISMIR*.

Pauwels, J., O'Hanlon, K., Gómez, E., and Sandler, M. B. (2019). 20 years of automatic chord recognition from audio. In *ISMIR*.

Pauwels, J. and Peeters, G. (2013). Evaluating automatically estimated chord sequences. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 749–753.

Pons, J., Lidy, T., and Serra, X. (2016). Experimenting with musically motivated convolutional neural networks. *2016 14th International Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 1–6.

Raffel, C., McFee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., and Ellis, D. P. W. (2014). Mir eval: A transparent implementation of common mir metrics. In *ISMIR*.

Ryynänen, M. and Klapuri, A. (2007). Automatic bass line transcription from streaming polyphonic audio. *Signal Processing*, pages 1437–1440.

Ryynänen, M. P. and Klapuri, A. P. (2008). Automatic Transcription of Melody, Bass Line, and Chords in Polyphonic Music. *Computer Music Journal*, 32(3):72–86.

Schlüter, J. and Böck, S. (2014). Improved musical onset detection with convolutional neural networks. *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6979–6983.

Schmidt-Jones, C. (2018). *Understanding Basic Music Theory*. 12Th Media Services.

Sheh, A. and Ellis, D. P. W. (2003). Chord segmentation and recognition using em-trained hidden markov models. In *ISMIR*.

Sumi, K., Itoyama, K., Yoshii, K., Komatani, K., Ogata, T., and Okuno, H. G. (2008). Automatic chord recognition based on probabilistic integration of chord transition and bass pitch estimation. In *ISMIR*.

Wu, Y., Carsault, T., and Yoshii, K. (2019). Automatic chord estimation based on a frame-wise convolutional recurrent neural network with non-aligned annotations. *2019 27th European Signal Processing Conference (EUSIPCO)*, pages 1–5.

Appendices

A How is Jazz Music Structured?

Jazz music is strongly based on improvisation and spontaneity, but there are rules and some basic structure on a song. It has to be noted that there are periods in jazz music associated with sub-genres that can be very different from each other. For example, the jazz from the early 20th century consisted predominantly on large ensembles known as big bands and the songs had a rigid structure. Later, small ensembles became popular, giving more preponderance to improvisation and the song's structure was flexible. Modern approaches, like free jazz, do not follow the traditional rules, but create new ones and sometimes the boundaries of jazz music definition are not clear.

Despite the differences between jazz sub-genres, there are conventions that are usually followed by jazz musicians. One of the most important, is that there is a fundamental musical structural segment that is repeated multiple times and is colloquially called *chorus*, which has a certain number of measures and a fixed harmonic sequence. This harmonic sequence is understood as a reference that usually is not performed as it is written, because each musician will ornament the chords at each repetition of the chorus. These variations can be expressed as notes that are added or subtracted to a chord or by anticipating/delaying the chord in time.

For example on a "G7 - C" chord progression a musician will understand the dominant tonic relationship and might add as many ornaments and tensions as desired as long as they do not change the functional relation (although the functional relations are ignored sometimes). It can be performed as a "G7b9 (G-B-D-F-Ab) - Cmaj7 (C-E-G-B)" one time and in next repetition performed as "G13 (G-B-D-F-A-E) - C9 (C-E-G-C-B-D)", without changing the hierarchical relations, but making them to sound different and transforming them into slightly different chords. It should be mentioned that the way that chords are ornamented and substituted has changed drastically with the evolution of jazz music, which at its origins was simpler and mostly diatonic, to later become more complex, like in bebop, using enriched and sophisticated chords. A good explanation of how harmony can be interpreted in jazz music can be found in Eremenko et al. (2018).

In this kind of songs, the convention is that at the first chorus the melody is played, followed by other repetitions featuring instrumental solos. Each solo is performed over the chorus a certain number of times and at the end of all instrumental solos, the melody is played one last time. Optionally it can also contain an intro, outro and less commonly, a bridge.

The instrumental solos generally comprise most of the song and are improvised, meaning that they are created at the moment and not learned in advance. The notes played by the soloist are strongly influenced by the harmonic progression defined in the chorus, but there are no rules determining what notes can (or cannot) be played.

Thus, the melodic line of the solo depends on the musician's personal style and is influenced by the musical accompaniment provided by the other musicians.

A jazz song is never performed the same twice, even if two different versions can be similar, because the musical solos will not be the equal. It is usual in jazz albums that alternate takes of a song are included.

B Synchronization Results

DB	Metric	Min	Max	Mean
cqt[ranges]_tonnetz_MFCC_hop2048	euclidean	4.14e-02	5.48e-01	1.29e-01
cqt[ranges]_tonnetz_hop2048	euclidean	4.08e-02	5.53e-01	1.32e-01
cqt[ranges]_MFCC_hop2048	euclidean	4.08e-02	5.54e-01	1.34e-01
cqt[ranges]_hop2048	euclidean	4.26e-02	5.54e-01	1.41e-01
cqt[bass]_cqt[mid]_tonnetz_MFCC_hop2048	euclidean	6.30e-02	4.92e-01	1.52e-01

(a) Top 5 results for JAAH database using euclidean distance metric.

DB	Metric	Min	Max	Mean
cqt[ranges]_tonnetz_MFCC_hop2048	mahalanobis	1.38e-02	1.45e-01	4.92e-02
cqt[ranges]_MFCC_hop2048	mahalanobis	1.36e-02	1.34e-01	5.07e-02
cqt_tonnetz[ranges]_MFCC_hop2048	mahalanobis	1.31e-02	1.70e-01	5.29e-02
cqt[bass]_cqt[mid]_MFCC_hop2048	mahalanobis	1.31e-02	2.18e-01	5.38e-02
cqt[bass]_cqt[mid]_tonnetz_MFCC_hop2048	mahalanobis	1.39e-02	2.23e-01	5.56e-02

(b) Top 5 results for JAAH database using Mahalanobis_2 distance metric.

DB	Metric	Min	Max	Mean
cqt[ranges]_MFCC_hop2048	mahalanobis_cov	1.40e-02	1.71e-01	5.18e-02
cqt[ranges]_tonnetz_MFCC_hop2048	mahalanobis_cov	1.41e-02	1.96e-01	5.22e-02
cqt_tonnetz[ranges]_MFCC_hop2048	mahalanobis_cov	1.45e-02	1.73e-01	5.62e-02
cqt[bass]_cqt[mid]_tonnetz_MFCC_hop2048	mahalanobis_cov	1.44e-02	2.00e-01	5.72e-02
cqt[bass]_cqt[mid]_MFCC_hop2048	mahalanobis_cov	1.35e-02	2.19e-01	5.74e-02

(c) Top 5 results for JAAH database using $\text{Mahalanobis}_{\text{all}}$ distance metric.

Figure 20: Synchronization results for JAAH database.

```
Overall Mean: 0.33150257352043594

mean 2048: 0.2346, mean 4096: 0.3268, mean 8192: 0.4331
    2048/4096: 0.718, difference of 28.2%
    2048/8192: 0.542, difference of 45.8%
    4096/8192: 0.754, difference of 24.6%

mean pure_cqt: 0.3374, mean cqt[bass]: 0.3525, mean cqt[mid]: 0.3851 mean cqt[bass+mid]: 0.2259, mean cqt[ranges]: 0.1791
    ranges/pure cqt: 0.531, difference of 46.9%
    ranges/bass cqt: 0.508, difference of 49.2%
    ranges/mid cqt: 0.465, difference of 53.5%
    ranges/bass_mid cqt: 0.793, difference of 20.7%

mean pure_tonnetz: 0.3170, mean tonnetz[bass]: 0.3950, mean tonnetz[mid]: 0.4009 mean tonnetz[bass+mid]: 0.3120, mean tonnetz[ranges]: 0.2715
    ranges/pure tonnetz: 0.856, difference of 14.4%
    ranges/bass tonnetz: 0.687, difference of 31.3%
    ranges/mid tonnetz: 0.677, difference of 32.3%
    ranges/bass_mid tonnetz: 0.870, difference of 13.0%

mean with MFCC 0.3009, mean without MFCC: 0.3637
    with/without MFCC: 0.827, difference of 17.3%

mean only with cqt 0.3222, mean only with tonnetz: 0.3924
    cqt/tonnetz : 0.821, difference of 17.9%
```

(a) Comparative results for JAAH database using euclidean distance metric.

```
Overall Mean: 0.1832115610784922

mean 2048: 0.1305, mean 4096: 0.1782, mean 8192: 0.2409
    2048/4096: 0.732, difference of 26.8%
    2048/8192: 0.542, difference of 45.8%
    4096/8192: 0.740, difference of 26.0%

mean pure_cqt: 0.1617, mean cqt[bass]: 0.1832, mean cqt[mid]: 0.1866 mean cqt[bass+mid]: 0.1183, mean cqt[ranges]: 0.1002
    ranges/pure cqt: 0.620, difference of 38.0%
    ranges/bass cqt: 0.547, difference of 45.3%
    ranges/mid cqt: 0.537, difference of 46.3%
    ranges/bass_mid cqt: 0.847, difference of 15.3%

mean pure_tonnetz: 0.1753, mean tonnetz[bass]: 0.2432, mean tonnetz[mid]: 0.2433 mean tonnetz[bass+mid]: 0.1701, mean tonnetz[ranges]: 0.1482
    ranges/pure tonnetz: 0.846, difference of 15.4%
    ranges/bass tonnetz: 0.610, difference of 39.0%
    ranges/mid tonnetz: 0.609, difference of 39.1%
    ranges/bass_mid tonnetz: 0.871, difference of 12.9%

mean with MFCC 0.0920, mean without MFCC: 0.2792
    with/without MFCC: 0.329, difference of 67.1%

mean only with cqt 0.1698, mean only with tonnetz: 0.2727
    cqt/tonnetz : 0.623, difference of 37.7%
```

(b) Comparative results for JAAH database using Mahalanobis_2 distance metric.

```
Overall Mean: 0.1895685195531913

mean 2048: 0.1340, mean 4096: 0.1836, mean 8192: 0.2511
    2048/4096: 0.730, difference of 27.0%
    2048/8192: 0.534, difference of 46.6%
    4096/8192: 0.731, difference of 26.9%

mean pure_cqt: 0.1672, mean cqt[bass]: 0.1886, mean cqt[mid]: 0.1935 mean cqt[bass+mid]: 0.1228, mean cqt[ranges]: 0.1036
    ranges/pure cqt: 0.620, difference of 38.0%
    ranges/bass cqt: 0.549, difference of 45.1%
    ranges/mid cqt: 0.536, difference of 46.4%
    ranges/bass_mid cqt: 0.844, difference of 15.6%

mean pure_tonnetz: 0.1818, mean tonnetz[bass]: 0.2512, mean tonnetz[mid]: 0.2502 mean tonnetz[bass+mid]: 0.1764, mean tonnetz[ranges]: 0.1547
    ranges/pure tonnetz: 0.851, difference of 14.9%
    ranges/bass tonnetz: 0.616, difference of 38.4%
    ranges/mid tonnetz: 0.618, difference of 38.2%
    ranges/bass_mid tonnetz: 0.877, difference of 12.3%

mean with MFCC 0.1001, mean without MFCC: 0.2838
    with/without MFCC: 0.353, difference of 64.7%

mean only with cqt 0.1743, mean only with tonnetz: 0.2814
    cqt/tonnetz : 0.619, difference of 38.1%
```

(c) Comparative results for JAAH database using $\text{Mahalanobis}_{\text{all}}$ distance metric.

Figure 21: Synchronization comparative results for JAAH database.

	DB	Metric	Min	Max	Mean
tonnetz[ranges]_MFCC_hop2048	3DB	euclidean	3.90e-02	9.78e-02	6.47e-02
cqt[ranges]_tonnetz_MFCC_hop2048	3DB	euclidean	4.09e-02	1.04e-01	6.55e-02
cqt[ranges]_MFCC_hop2048	3DB	euclidean	4.05e-02	1.05e-01	6.55e-02
cqt[ranges]_tonnetz_hop2048	3DB	euclidean	4.19e-02	1.05e-01	6.69e-02
cqt[ranges]_hop2048	3DB	euclidean	4.06e-02	1.07e-01	6.70e-02

(a) Top 5 results for MIDI 3 database using euclidean distance metric.

	DB	Metric	Min	Max	Mean
tonnetz[ranges]_MFCC_hop2048	3DB	mahalanobis	1.15e-02	5.19e-02	2.41e-02
cqt[bass]_tonnetz_MFCC_hop2048	3DB	mahalanobis	1.17e-02	5.19e-02	2.67e-02
cqt[bass]_MFCC_hop2048	3DB	mahalanobis	1.19e-02	5.02e-02	2.70e-02
tonnetz[bass]_tonnetz[mid]_MFCC_hop2048	3DB	mahalanobis	1.13e-02	7.16e-02	2.70e-02
cqt[ranges]_MFCC_hop2048	3DB	mahalanobis	1.36e-02	5.38e-02	2.75e-02

(b) Top 5 results for MIDI 3 database using Mahalanobis_2 distance metric.

	DB	Metric	Min	Max	Mean
tonnetz[bass]_tonnetz[mid]_MFCC_hop2048	3DB	mahalanobis_cov	1.12e-02	5.43e-02	2.80e-02
cqt[ranges]_MFCC_hop2048	3DB	mahalanobis_cov	1.44e-02	5.47e-02	2.89e-02
cqt[ranges]_tonnetz_MFCC_hop2048	3DB	mahalanobis_cov	1.42e-02	5.63e-02	2.94e-02
tonnetz[ranges]_MFCC_hop2048	3DB	mahalanobis_cov	1.14e-02	5.36e-02	2.95e-02
cqt_tonnetz[mid]_MFCC_hop2048	3DB	mahalanobis_cov	1.24e-02	5.60e-02	2.97e-02

(c) Top 5 results for MIDI 3 database using $\text{Mahalanobis}_{\text{all}}$ distance metric.

Figure 22: Synchronization results for MIDI database.

```
Overall Mean: 0.1844959431808691

mean 2048: 0.1031, mean 4096: 0.1693, mean 8192: 0.2811
    2048/4096: 0.609, difference of 39.1%
    2048/8192: 0.367, difference of 63.3%
    4096/8192: 0.602, difference of 39.8%

mean pure_cqt: 0.1911, mean cqt[bass]: 0.1959, mean cqt[mid]: 0.1810 mean cqt[bass+mid]: 0.1154, mean cqt[ranges]: 0.1065
    ranges/pure cqt: 0.557, difference of 44.3%
    ranges/bass cqt: 0.544, difference of 45.6%
    ranges/mid cqt: 0.589, difference of 41.1%
    ranges/bass_mid cqt: 0.923, difference of 7.7%

mean pure_tonnetz: 0.1778, mean tonnetz[bass]: 0.2015, mean tonnetz[mid]: 0.2146 mean tonnetz[bass+mid]: 0.1552, mean tonnetz[ranges]: 0.1502
    ranges/pure tonnetz: 0.845, difference of 15.5%
    ranges/bass tonnetz: 0.746, difference of 25.4%
    ranges/mid tonnetz: 0.700, difference of 30.0%
    ranges/bass_mid tonnetz: 0.968, difference of 3.2%

mean with MFCC 0.1745, mean without MFCC: 0.1950
    with/without MFCC: 0.895, difference of 10.5%

mean only with cqt 0.1746, mean only with tonnetz: 0.2078
    cqt/tonnetz : 0.841, difference of 15.9%
```

(a) Comparative results for MIDI 3 database using euclidean distance metric.

```
Overall Mean: 0.09669605324355991

mean 2048: 0.0582, mean 4096: 0.0862, mean 8192: 0.1457
    2048/4096: 0.674, difference of 32.6%
    2048/8192: 0.399, difference of 60.1%
    4096/8192: 0.592, difference of 40.8%

mean pure_cqt: 0.0919, mean cqt[bass]: 0.0990, mean cqt[mid]: 0.0915 mean cqt[bass+mid]: 0.0668, mean cqt[ranges]: 0.0592
    ranges/pure cqt: 0.644, difference of 35.6%
    ranges/bass cqt: 0.598, difference of 40.2%
    ranges/mid cqt: 0.647, difference of 35.3%
    ranges/bass_mid cqt: 0.886, difference of 11.4%

mean pure_tonnetz: 0.0983, mean tonnetz[bass]: 0.1202, mean tonnetz[mid]: 0.1206 mean tonnetz[bass+mid]: 0.0868, mean tonnetz[ranges]: 0.0762
    ranges/pure tonnetz: 0.775, difference of 22.5%
    ranges/bass tonnetz: 0.634, difference of 36.6%
    ranges/mid tonnetz: 0.632, difference of 36.8%
    ranges/bass_mid tonnetz: 0.878, difference of 12.2%

mean with MFCC 0.0478, mean without MFCC: 0.1481
    with/without MFCC: 0.323, difference of 67.7%

mean only with cqt 0.0908, mean only with tonnetz: 0.1333
    cqt/tonnetz : 0.681, difference of 31.9%
```

(b) Comparative results for MIDI 3 database using Mahalanobis_2 distance metric.

```
Overall Mean: 0.09986102202652325

mean 2048: 0.0591, mean 4096: 0.0882, mean 8192: 0.1524
    2048/4096: 0.670, difference of 33.0%
    2048/8192: 0.388, difference of 61.2%
    4096/8192: 0.579, difference of 42.1%

mean pure_cqt: 0.0948, mean cqt[bass]: 0.1009, mean cqt[mid]: 0.0938 mean cqt[bass+mid]: 0.0726, mean cqt[ranges]: 0.0623
    ranges/pure cqt: 0.658, difference of 34.2%
    ranges/bass cqt: 0.618, difference of 38.2%
    ranges/mid cqt: 0.664, difference of 33.6%
    ranges/bass_mid cqt: 0.859, difference of 14.1%

mean pure_tonnetz: 0.1009, mean tonnetz[bass]: 0.1278, mean tonnetz[mid]: 0.1226 mean tonnetz[bass+mid]: 0.0905, mean tonnetz[ranges]: 0.0782
    ranges/pure tonnetz: 0.775, difference of 22.5%
    ranges/bass tonnetz: 0.612, difference of 38.8%
    ranges/mid tonnetz: 0.638, difference of 36.2%
    ranges/bass_mid tonnetz: 0.864, difference of 13.6%

mean with MFCC 0.0517, mean without MFCC: 0.1505
    with/without MFCC: 0.344, difference of 65.6%

mean only with cqt 0.0934, mean only with tonnetz: 0.1365
    cqt/tonnetz : 0.684, difference of 31.6%
```

(c) Comparative results for MIDI 3 database using $\text{Mahalanobis}_{\text{all}}$ distance metric.

Figure 23: Synchronization comparative results for MIDI database.

C Chord Transcription Results

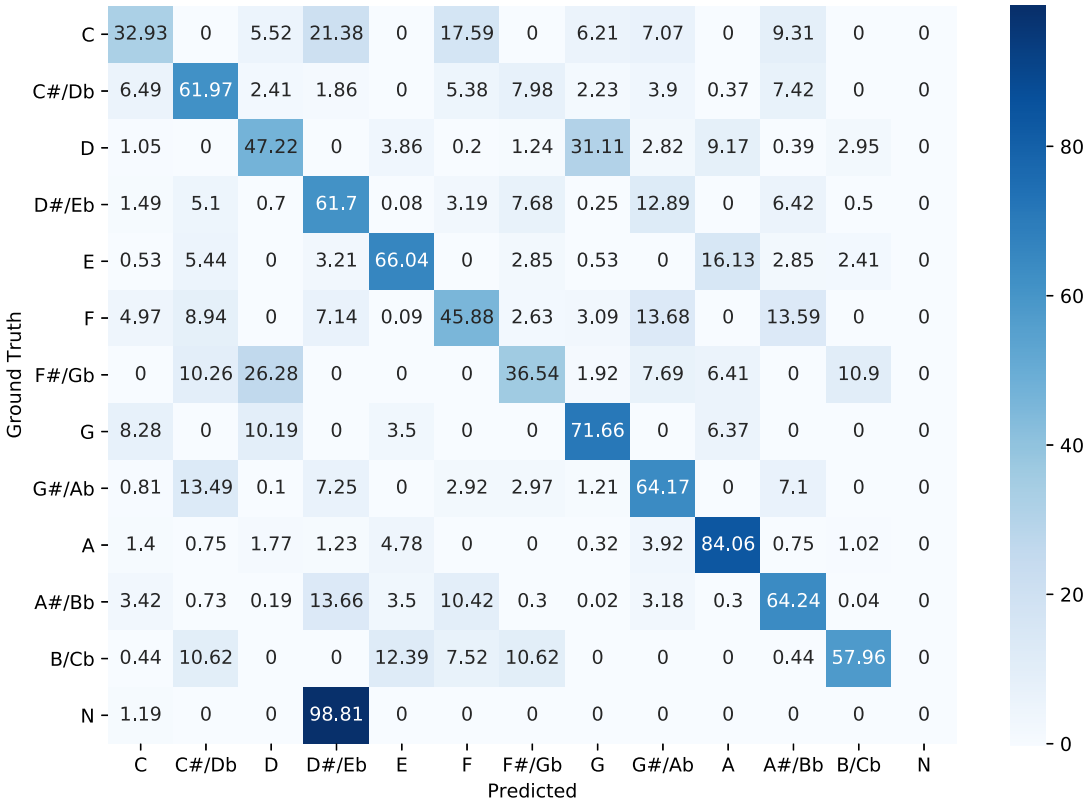


Figure 24: Roots confusion matrix for the non-averaged version.

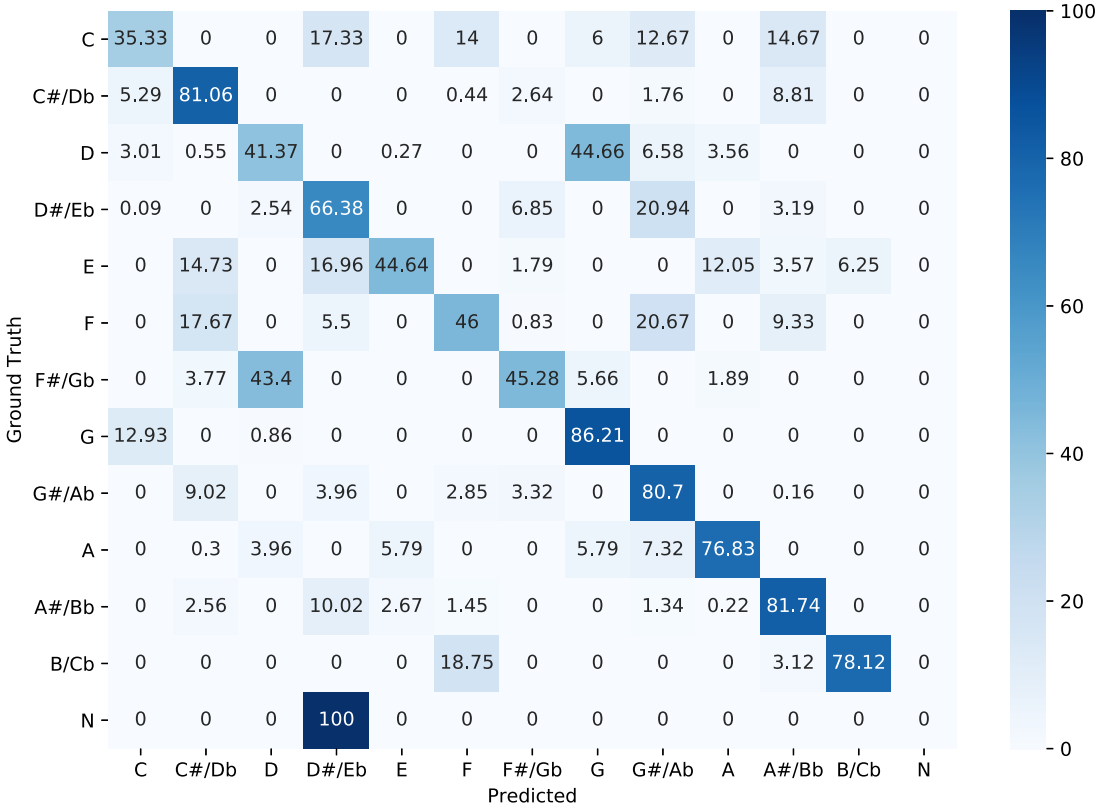


Figure 25: Roots confusion matrix for the averaged CQT version.

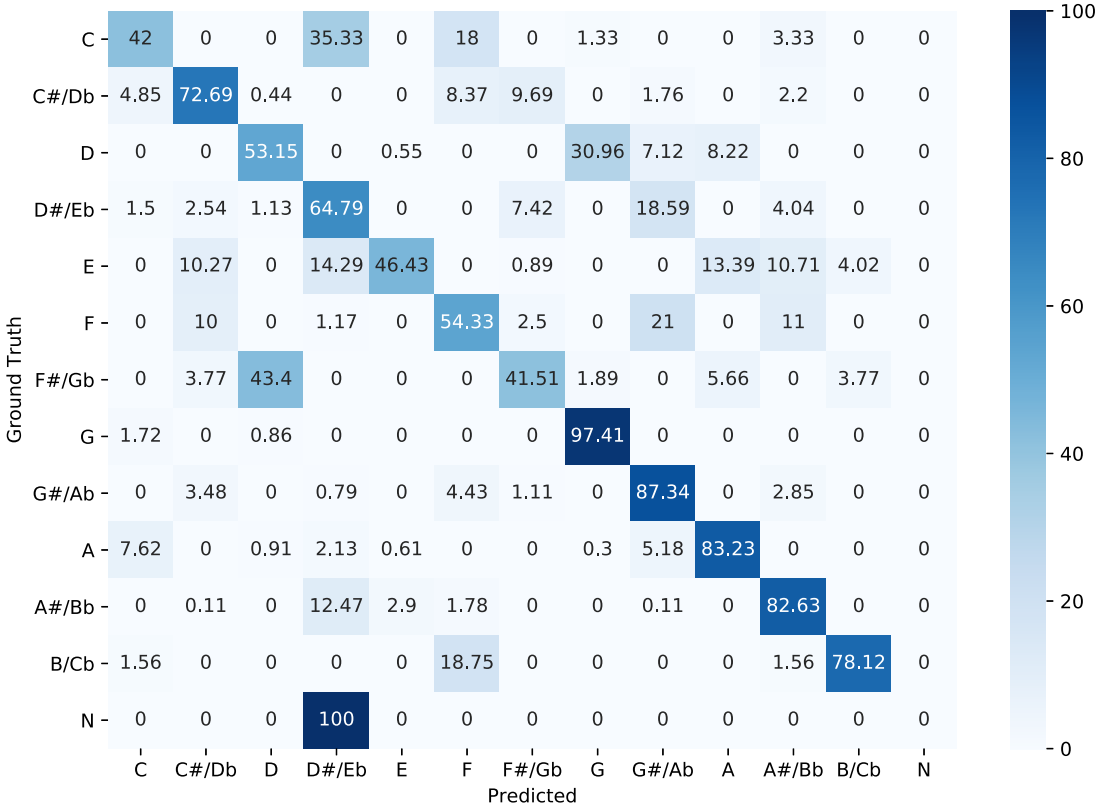


Figure 26: Roots confusion matrix for the averaged latent features version.

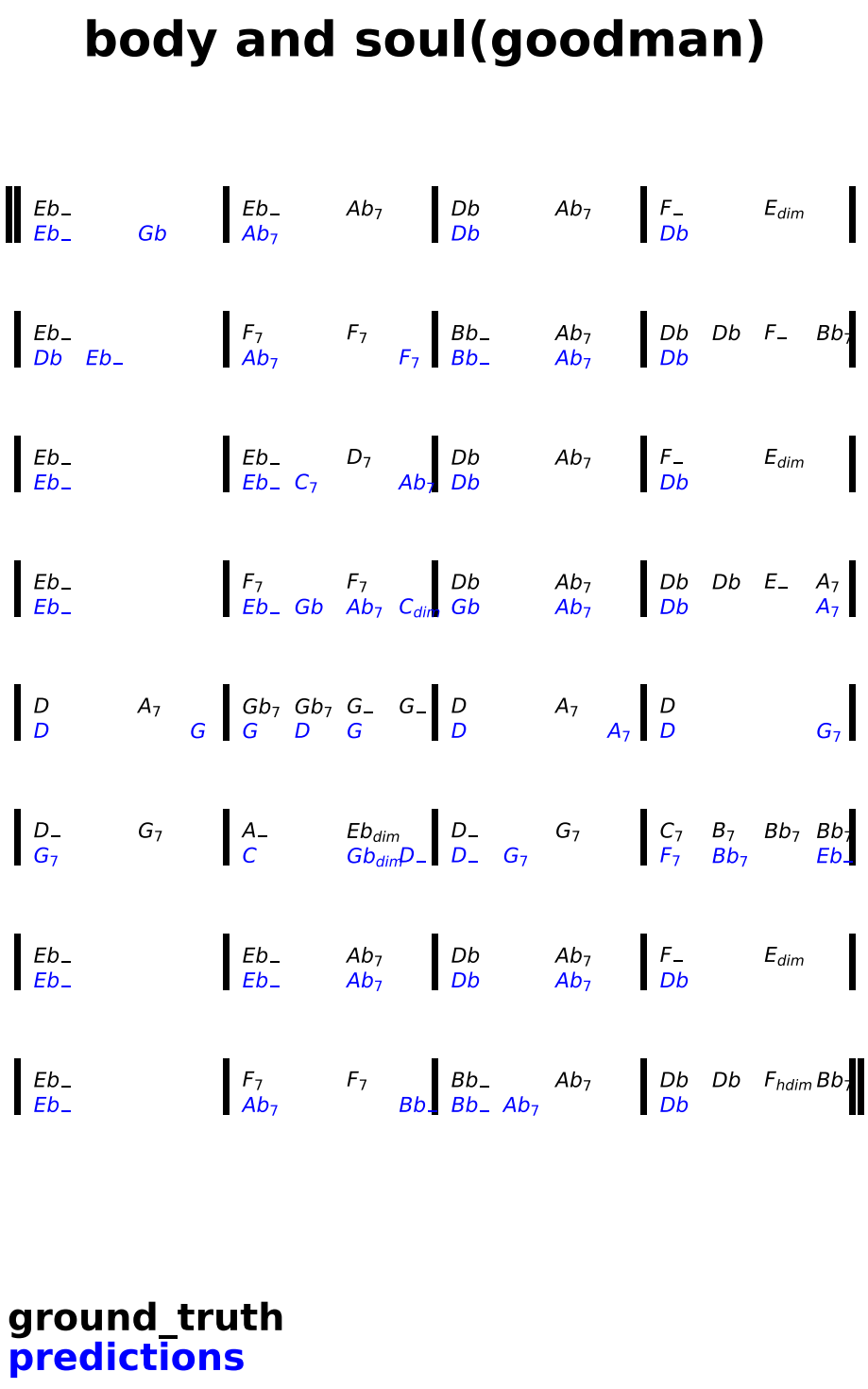


Figure 27: Obtained lead sheet-like representation of test song “Body and Soul”.

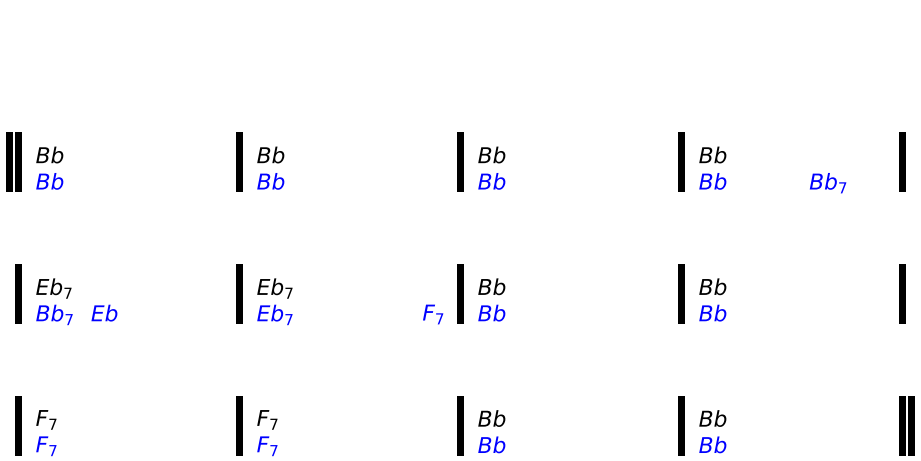


Figure 28: Obtained lead sheet-like representation of test song “Breakfast Feud”.

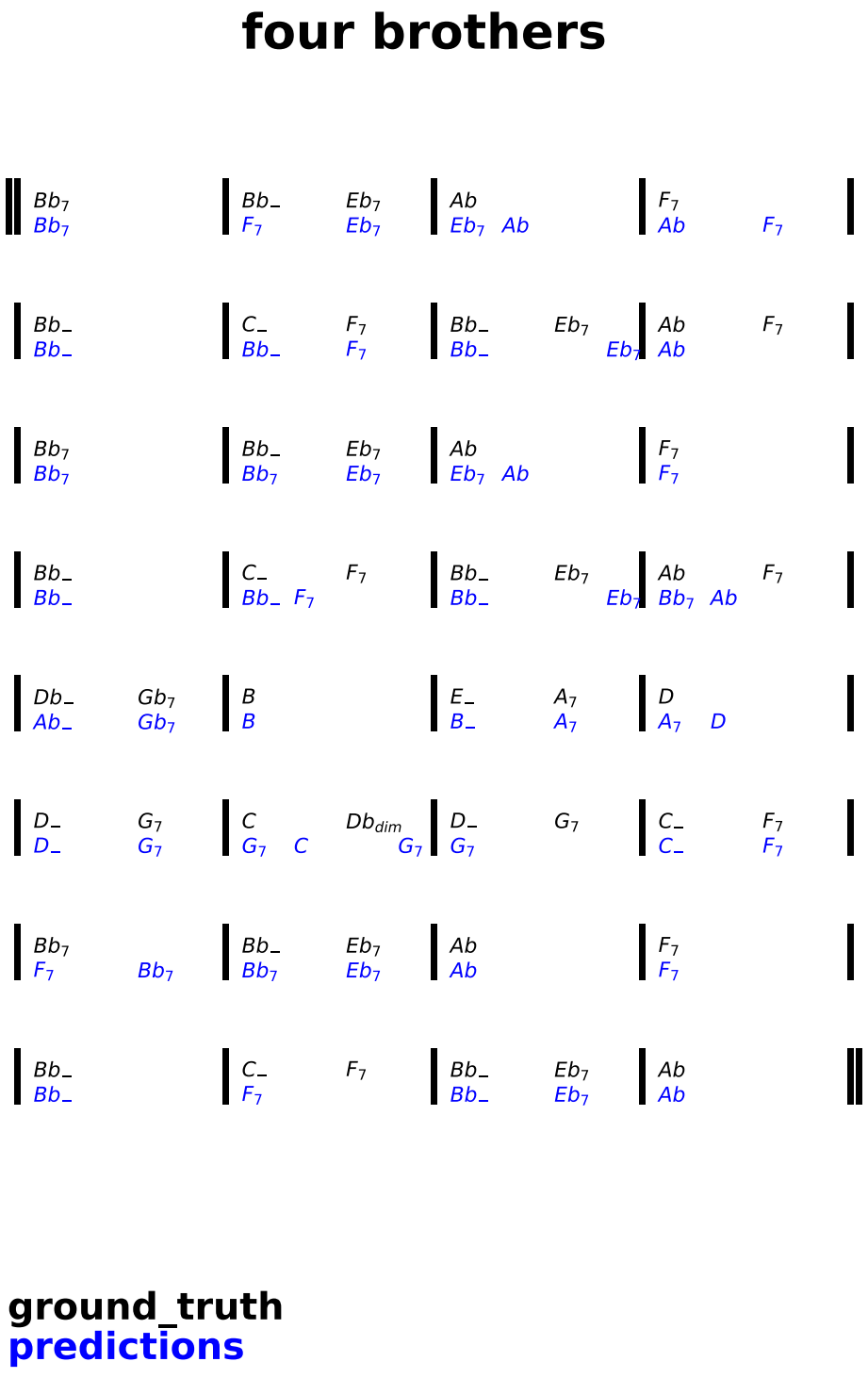


Figure 29: Obtained lead sheet-like representation of test song “Four Brothers”.

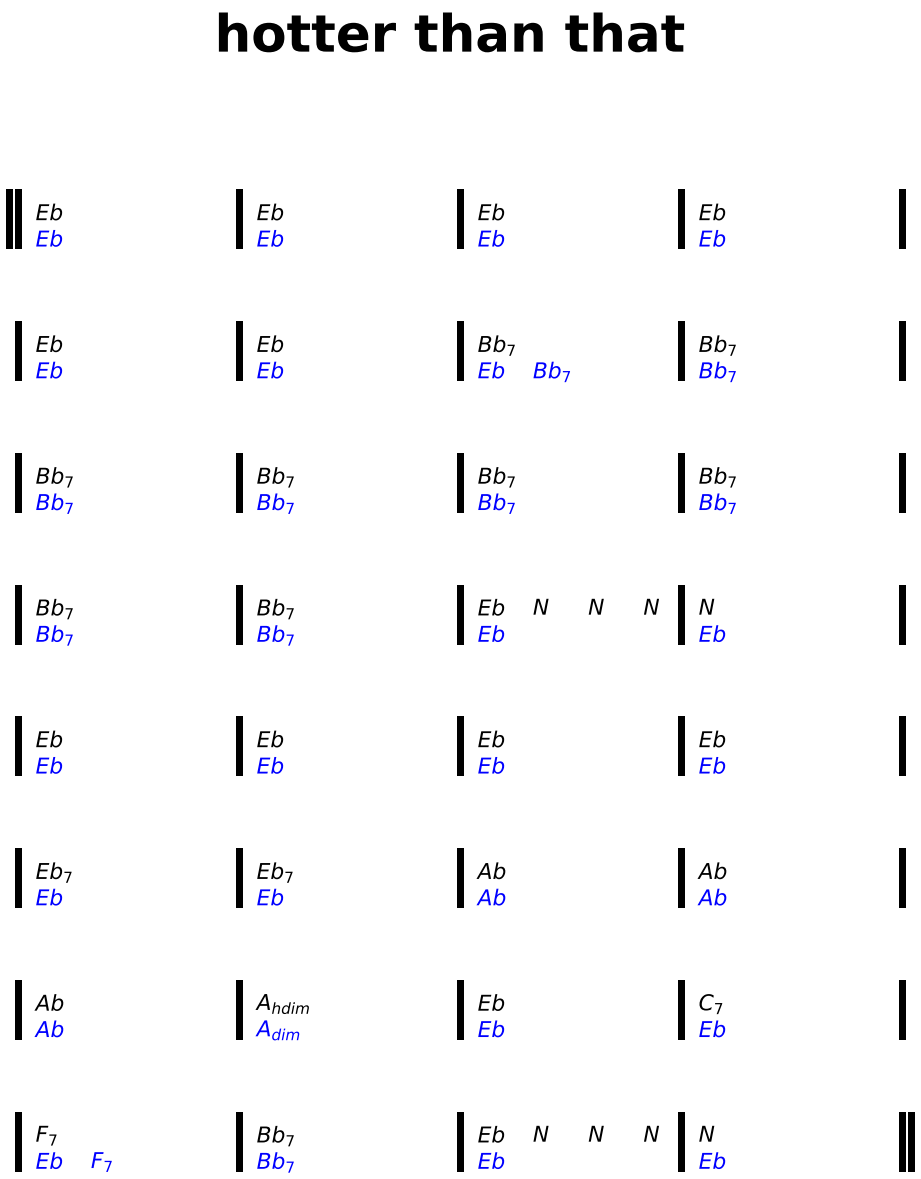


Figure 30: Obtained lead sheet-like representation of test song “Hotter Than That”.

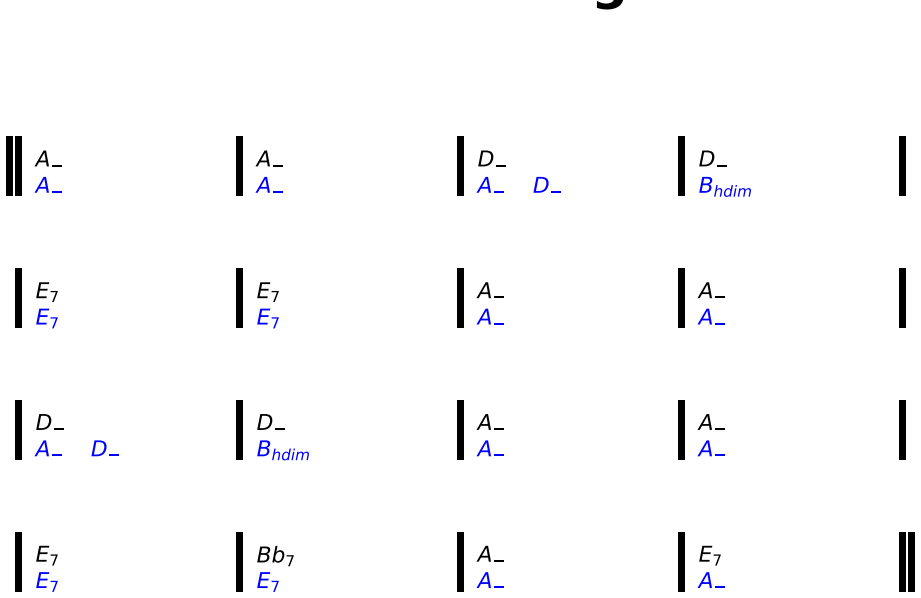


Figure 31: Obtained lead sheet-like representation of test song “Minor Swing”.

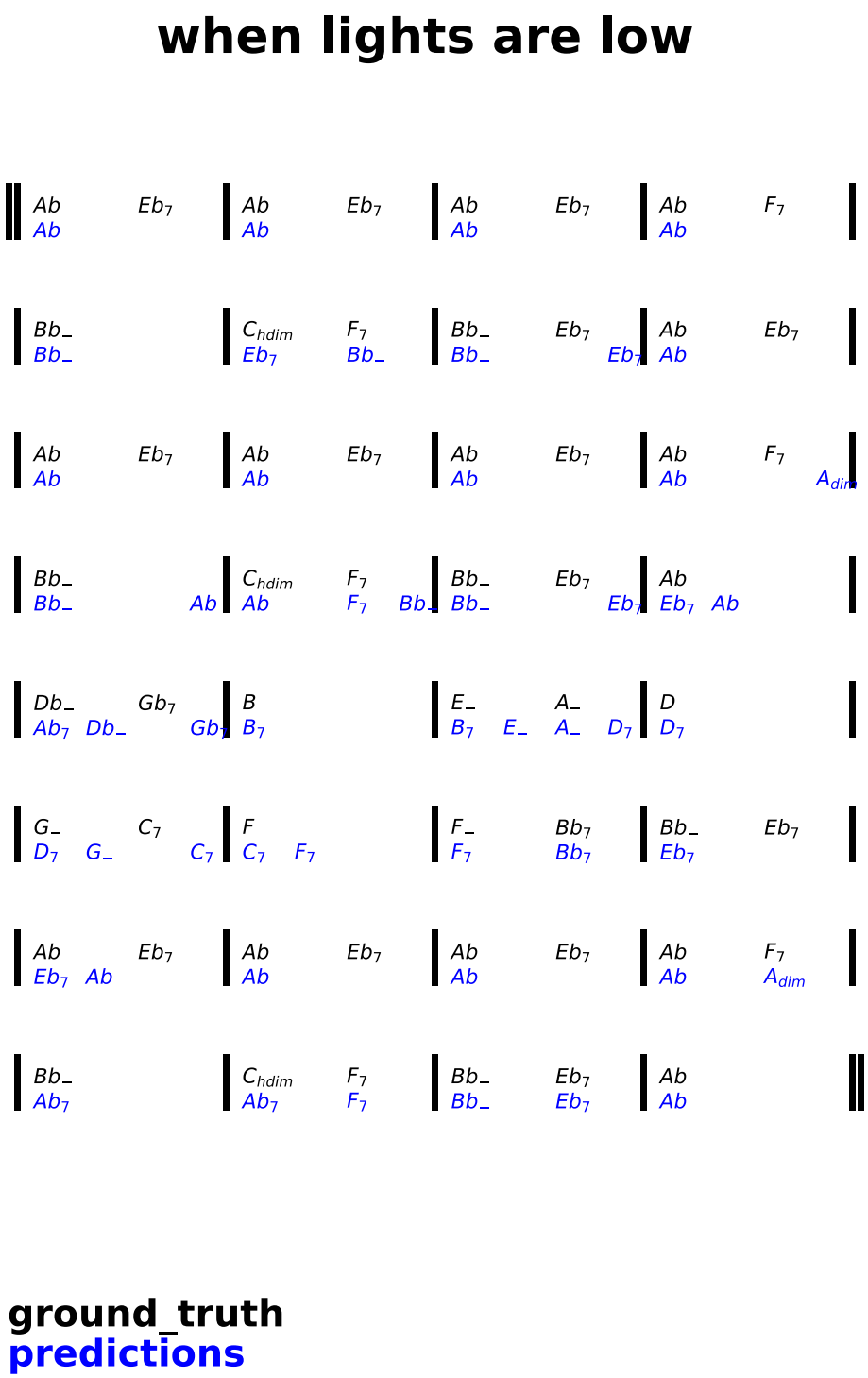


Figure 32: Obtained lead sheet-like representation of test song “When Lights Are Low”.

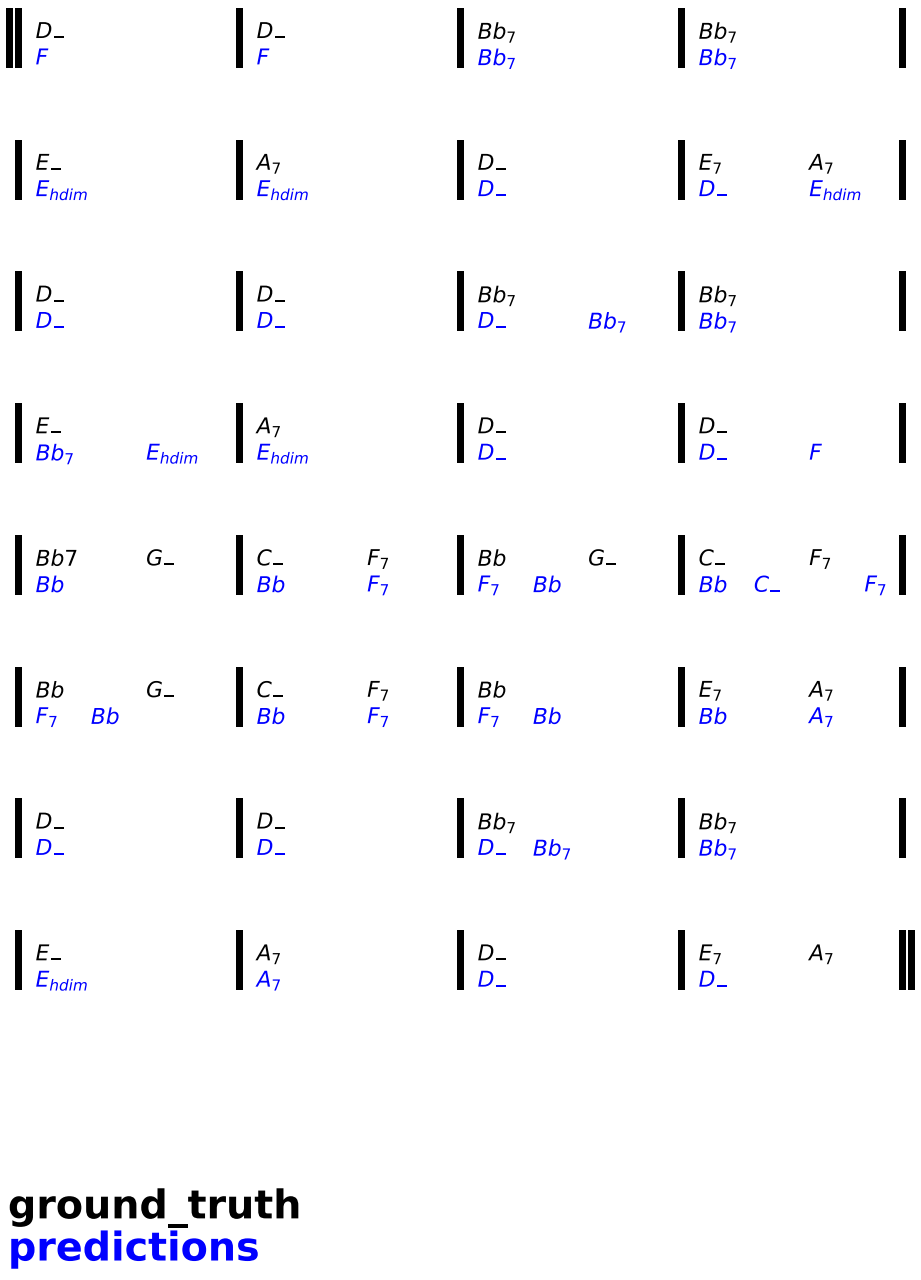


Figure 33: Obtained lead sheet-like representation of MIDI rendered song “Bernie’s Tune”.

Figure 34: Beat-synchronous idealized chromagram from MIDI rendered song "Bernie's Tune".

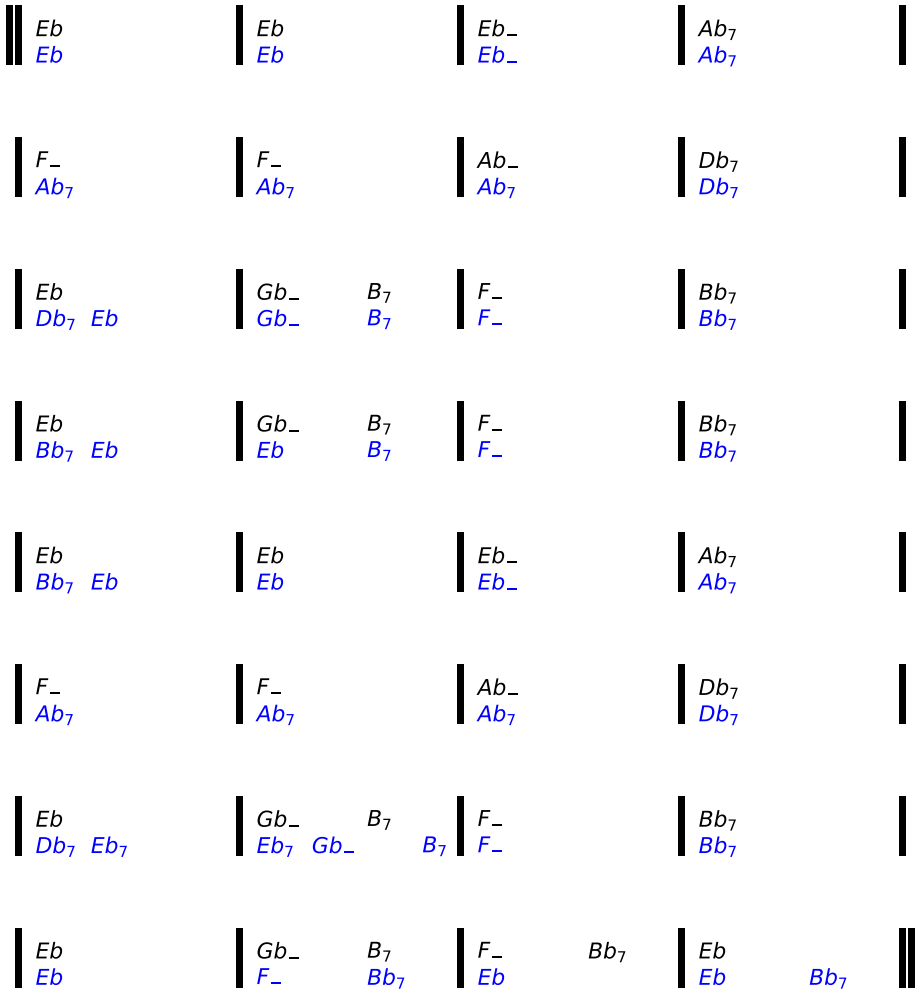


Figure 35: Obtained lead sheet-like representation of MIDI rendered song “Four”.

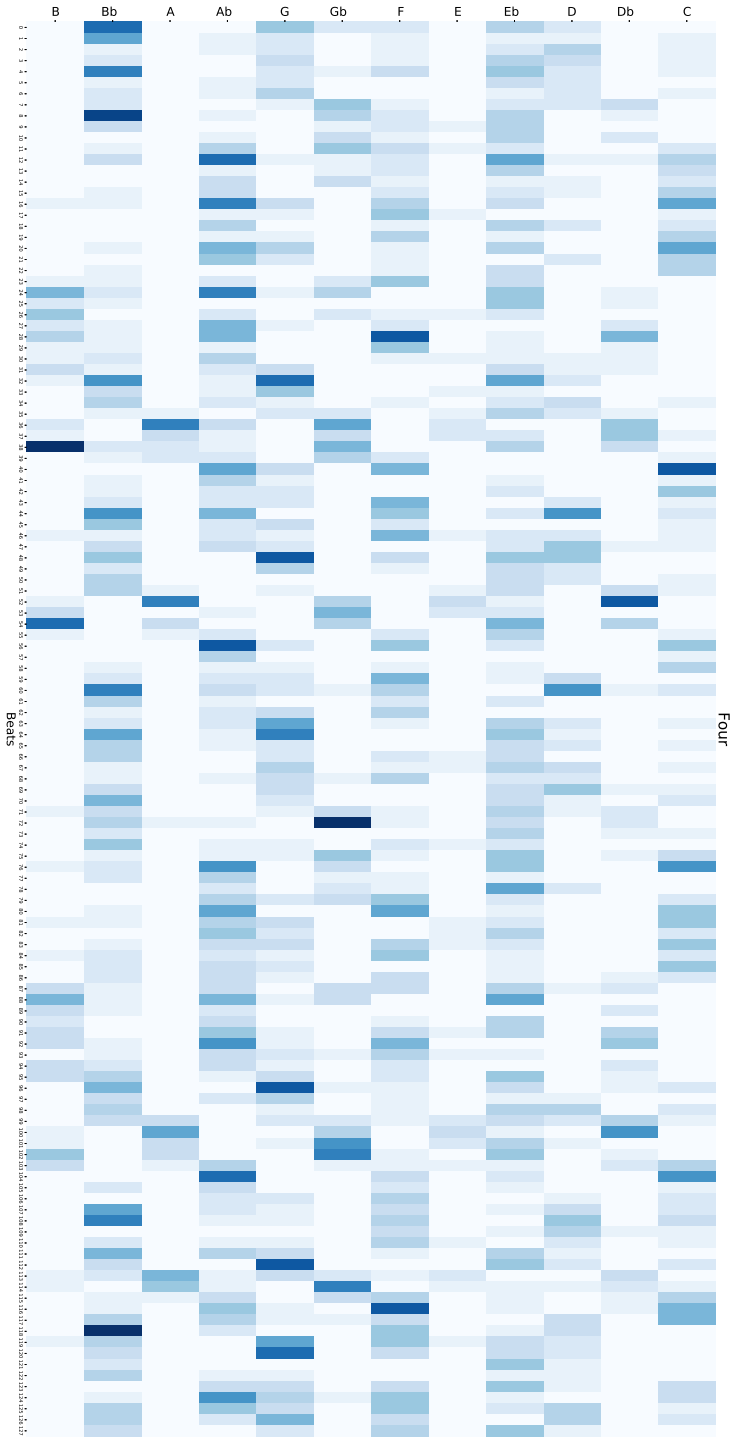


Figure 36: Beat-synchronous idealized chromagram from MIDI rendered song "Four".