ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ARTIFICIAL INTELLIGENCE

MASTER THESIS

in

Knowledge Engineering

KNOWLEDGE-BASED CHORD EMBEDDINGS

CANDIDATE
Nicolas Lazzari

SUPERVISOR
Prof. Valentina Presutti

CO-SUPERVISOR
Dott. Andrea Poltronieri

Academic year 2021-2022

Session 4th

Abstract

This thesis develops Artificial Intelligence methods to contribute to the advancement of computational musicology, an interdisciplinary field that uses computers to study music and to automate tasks such as music classification, music similarity, music generation, music transcription, music recommendation, etc. In systematic musicology a musical composition is defined as being the combination of harmony, melody and rhythm. According to de La Borde, harmony is the component showing most complexity and alone "merits the name of composition´´. Harmony is arguably the foundation of a musical composition, therefore this thesis focuses on analysing the harmonic aspect from a computational perspective. There are three main ways of representing music through a machine: sheet music data (e.g. digitisation of scores as images), symbolic data (e.g. musical notation in digital form such as MusicXML), audio data (e.g. representation of acoustic waves such as MP3). We concentrate on symbolic music representation and address the problem of encoding and formally representing harmonic chord progressions in musical compositions, in the context of western music. Informally, chords are sets of pitches played simultaneously, and chord progressions constitute the *harmony* of a musical composition. Our approach combines machine learning techniques with knowledge-based techniques. We design and implement the Modal Harmony ontology (MHO), using OWL (the standard web ontology language). It formalises one of the most important theories used to interpret western music: the Modal Harmony Theory. We propose and experiment with different types of embedding methods to encode chords mainly inspired

by Natural Language Processing and adapted to exploit the peculiarities of the musical domain. We use both statistical (extensional) knowledge by relying on a huge dataset of chord annotations (ChoCo) and intensional knowledge by relying on MHO. To compare and evaluate the different solutions, including embeddings resulting from the combination of the two approaches, we apply them to two different musicologically relevant tasks: chord classification and music structure segmentation. Chord classification is verified by comparing the result of the Odd One Out algorithm to the result of the deductive classification based on MHO. The results show good performance (accuracy: 0.86). As for music structure segmentation, we feed a recurrent neural network with our proposed embeddings to classify music segments and compare the results with a gold standard dataset. Results show that the best performance (F1: 0.6) is achieved with embeddings that combine intensional and extensional knowledge. This result outpeforms the best existing method (F1 = 0.42) for music structure segmentation based on symbolic music representation. It is worth noticing that embeddings based only on MHO allow to achieve a performance very close to the best one (F1 = 0.58). In this case, we remark that creating the embeddings only requires the ontology as an input as opposed to statistical approaches that rely on large amount of data. All data and software produced are publicly released under open source license [1].

[1] `https://github.com/n28div/chord-embeddings`, `https://github.com/n28div/modalharmonyontology`

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Music is among the most widely acknowledged form of artistic expression in modern society. Whether consciously or unconsciously, it is a fundamental part of everyone's life. It comes with little surprise that music has permeated into the Artificial Intelligence field well, branching into its own prolific research area. The history of combining music with Artificial Intelligence is, indeed, as long as the Artificial Intelligence field itself.

> [The Analytical Engine] might act upon other things besides number, were objects found whose mutual fundamental relations could be expressed by those of the abstract science of operations, and which should be also susceptible of adaptations to the action of the operating notation and mechanism of the engine...Supposing, for instance, that the fundamental relations of pitched sounds in the science of harmony and of musical composition were susceptible of such expression and adaptations, the engine might compose elaborate and scientific pieces of music of any degree of complexity or extent.
>
> Ada Lovelace [100]

1.1 AI as a tool for Music enthusiasts

Computer science in general has been applied to music in countless ways, from artistic processes that generate musical composition [20] to information retrieval systems that allow digital catalogs to be enhanced by content-based approaches [87]. One of the most important and straightforward applications of AI to the music field is to provide a set of low-level tools to complement the activity of a music expert [118]. Successful approaches have been proposed in this direction [70, 73, 107] following a *reductionalist* approach: complex problems have been divided into smaller sub-problems addressed with increasingly complex solutions [117].

Technologies that obtained impressive state-of-the-art results in complex fields, such as Natural Language Processing, have been applied to the musical domain, with promising results [23, 41, 62, 154]. Their lack of explainability, however, is a particularly concerning issue in the musical domain.

The analysis of music, and in general the theoretical examination of musical compositions, has a rich legacy of theories and approaches that have been criticised, disproved, abandoned, and rediscovered [55]. The minds of musicians from the present and the past have been forged, consciously or unconsciously, by such theories. Analysing past compositions is a fundamental element in the growth of an artist. It is by means of the analysis process that an artist becomes intimate with a particular work and is able to distill such intimacy on its own personal art [1].

An explicit relation between the concepts learned by an Artificial Intelligence system and existing musical theories needs to be drawn, to maintain a common thread with such a legacy and enable artists to empower AI as part of their own creative instinct.

The focus of this thesis is to study the impact of systems based on Semantic Web technologies (see Section 2.3) on the automatic analysis of musical compositions when those systems are combined with the most recent Artificial

Intelligence techniques. Indeed, while the usages and customs of the musical language [32] have evolved in time, the musical vocabulary through which a composition is realized remained stable, at least in the context of Western culture [9]. As such, a plethora of valuable theories can be reorganized, reshaped, and transformed to obtain a new set of tools that can enable music specialists, artists, and in general all music enthusiasts to effortlessly "stand on the shoulder of the giants".

1.2 Open challenges

Addressing open problems in the music field with the use of techonology and AI inevitably involves the analysis of musical compositions. This is pursued through the use of two main approaches based, respectively, on symbolical representations and audio representations [146] of music.

With the advent of modern technologies and the ability to process a large amount of data in an efficient way, the Music Information Retrieval field (described in Chapter 2) has seen a new offspring. Audio-based approaches transitioned from an appealing research area to a concrete and valuable set of tools, that allow processing music in an automated yet accurate way [89, 114]. The same happened with symbolic representations, for instance as a way to enhance the audio-based systems [23, 24, 75, 122, 123] by exploiting the content-aware nature of a symbolic representation [146].

A knowledge-based approach acting as a bridge between musicologically grounded theories and state-of-the-art Artificial Intelligence techniques would be beneficial to many music-related tasks, such as style characterization [146], hierarchical music structure analysis [114] and music information retrieval tasks in general [43]. Nonetheless, it represents a big challenge for both the musicological field, since it requires a unification effort between different music theories, and the Artificial Intelligence field where a significant effort in the formalisation of concepts that are usually loosely structured is required.

A popular example is represented by the Gene Ontology (GO) [3]. With the intent of formalising and unifying biological concepts, GO represents one of the most valuable tools in the biological research field. It allows researchers to efficiently browse and search an immense amount of interconnected resources that were previously scattered between different systems, preserve biological knowledge in a unified system, and most importantly, enables researchers to automatically transfer known properties of experimentally tractable organisms to other organisms that have similar genes but are less tractable in an experimental setting [3].

The value of the Gene Ontology goes beyond its academic success. It represents a virtuous example of how combined effort and free knowledge lead to advances that are not only relevant for the field itself, but for humankind in general [86].

The development of a musical correspondence to the Gene Ontology is an ambitious project that requires the combined effort of Artificial Intelligence experts and Music Theory experts.

1.3 A knowledge-based approach to chord representation

This work represents a step toward the creation of novel tools that exploit musicological-grounded knowledge through modern Artificial Intelligence methods, by addressing the problem of automatic analysis of harmonic sequences.

Three aspects of musical compositions are traditionally analysed by musicologists: melody, rhythm, and harmony [119]. Even though these aspects are deeply related one to another, it has been argued that harmony alone merits the name of musical composition [52]. The focus on harmonic sequences is hence to be taken in virtue of its characterisation role in a musical composition. The

results obtained throughout this work can be seen as an explorative set of experiments that can eventually be adapted and generalized to the analysis of all the elements that contribute to a musical composition.

> Since musical analysis became a pursuit in its own right [...] various analytical methods have been devised [...]. These methods differ in the nature of the musical material under study, and in the form of their output, but they are similar in their goal and operating mode: they consist in chopping the musical material into pieces, comparing these pieces and classifying them, in order to eventually reformulate the original material with the terms of an established corpus of concepts. Through such a reconstruction, a successful analysis may eventually provide a sense of possession, an intimate feeling of appropriation of the analyzed material which is comparable to the feeling the composer has for his own creation.
>
> François Pachet [118]

The analysis of a harmonic sequence requires a profound knowledge of music theory and of the context in which the analysis is being performed. Analysing a Rock composition [38] is radically different than analysing a Classical composition [95]. Those differences, however, are not laid on how an analysis is performed: all the fundamental music laws that are applied to Rock compositions can be applied to Classical compositions or Jazz compositions [9]. It is the objective of the analysis, the ultimate classification of the progression, that differs. For instance, the analysis of a Jazz composition has a precise and practical outcome for a musician: that of guiding the choice of the melodies that are to be played in some specific part of the composition [118]. On the other hand, analysing a Classical composition is directed at contextualising the composer's choices within past composers, to identify its influences and understand the evolution of music throughout historical epochs [110].

Regardless of the analysis objective, the building block of a harmonic sequence, the chord, is common to all modern western music. An effective representation of chords allows its classification with respect to its most primitive and fundamental aspects, common to all modern music. It is by means of specialising such representation that different types of automatic harmonic analysis can be achieved. We argue that the combination of different type of analysis, which is objectively difficult since it involves a profound knowledge of many different musical theories, might uncover interesting interpretations of composition that were previously ignored. As an example, the popular jazz composition *Solar* by *Miles Davis* has been interpreted as a blues composition in [118]. While an oracle able to judge such interpretation as correct (or incorrect) does not exist and never will [55], a discussion between musicologists can take place on the basis of a solid incipit: *the tune Solar by Miles Davis can be objectively seen as a Blues composition by means of a sound argumentation*.

Finding an effective chord representation is a long-researched problem. Different representations have been proposed, such as textual representation based on the syntactical aspects of a chord [64], geometrical representation [40] and digital score representation [113, 60]. These are all tailored to their respective target application and provide little to no degree of adaptation to the many challenges of automatic music analysis tasks.

This work focuses on proposing a new chord representation method that can accurately encode fundamental chord properties and be contextualised in an efficient way to tackle any task that involves the use of chords. In the spirit of reusing and exploiting the legacy of musical theory, we investigate how a knowledge-based approach can be combined with modern Artificial Intelligence techniques to obtain accurate, flexible, and efficient chord representations.

1.3.1 Proposed approach

An ontology that models the theory of Modal Harmony [9] building upon previous methods [48, 126] is formalised at first using technologies from the Semantic Web [10]. The resulting system allows one to effortlessly classify chords on the basis of well-known music theory concepts.

This provides us with a semantically valid analytical tool to investigate whether state-of-the-art techniques, inspired by the Natural Language Processing (NLP) field, can be adapted to the music field. The generalization ability of a learned representation is measured as a function of the concepts encoded in the knowledge graph. Optimal representations must reflect the musicological knowledge that is formalized in the knowledge graph. Moreover, it represents a valid starting point for the development of methods that provide efficient and exact solutions to Music Information Retrieval tasks, such as functional harmony analysis [72] and roman notation inference [103].

Learned word embedding representations have revolutionized the field of NLP [30] and been proven to be effective in modeling harmonic sequences as well [2, 93]. Nonetheless, they have been shown to encode the semantic meaning of a word only to a limited extent [131]. We address this issue by means of two different lines of attacks. In the former we investigate if such a lack of semantic understanding can be complemented by a more careful encoding method based on musical theory. We present two novel chord embedding methods in this direction, intervals2vec and pitchclass2vec, that outperform other embedding methods. In the latter, we investigate the possibility of obtaining accurate and efficient chord representations directly from an inherently semantically consistent source: the proposed knowledge graph. Such a method outperforms both intervals2vec and pitchclass2vec. We combine both lines of attack as well to retain the best of the two worlds and obtain state-of-the-art results.

Finally, to assess its pragmatic applicability, we apply our novel chord

representation method to the task of musical structure segmentation. Musical structure segmentation is the task that involves the identification of hierarchical structures in musical compositions, such as the identification of the *verse* or the *chorus* of a tune, and is traditionally performed using audio-based representations. Approaches relying only on symbolic annotations received little attention and have been shown to underperform when compared to audio-based approaches.

With the use of a straightforward model, we achieve new state-of-the-art results, comparable to the one obtained by means of an audio-based representation. The chord representation method entirely founded on the knowledge graph achieves near-state-of-the-results when compared to the other proposed methods. This is particularly interesting given that, differently from the other methods, it does not have any dependency on the amount of data available but rather depends on the quality of the knowledge graph.

A correctly formalised knowledge graph is hence not only fundamental in accessing theoretical musical knowledge in a unified and accessible form, but it can also be used to obtain accurate and efficient chord representation methods that achieve state-of-the-art results on relevant pragmatic tasks.

We identify the following research questions, that are addressed throughout the rest of this work:

RQ1 Is it possible to formalize and encode music theories in an accurate form?

RQ2 How can we assess the quality of chord representation methods?

RQ3 Is it possible to identify a set of requirements for accurate chord representations?

RQ4 Is it possible to use chord representations to obtain results comparable to audio-based representations?

1.4 Contribution

The contribution of this work can be summarised as follows:

- formalisation and development of a novel Knowledge Graph and a relative ontology, the Modal Harmony Ontology, that models the theory of Modal Harmony;
- development of a novel chord embedding method that achieves state-of-the-art results in modeling the theory of Tonal and Modal Harmony;
- development of a structure segmentation algorithm, based on chord embedding methods, that outperforms related work and obtains new state-of-the-art results.

All the developed resources are publicly under open source license. The code to reproduce the experiments of Chapter 5 is available at `https://github.com/n28div/chord-embeddings`; the ontology presented in Chapter 4 is available at `https://github.com/n28div/modalharmonyontology`; the implemented extension of the Music Theory Ontology, described in Section 4.2, is available at `https://github.com/n28div/MusicTheoryOntology`.

1.5 Structure

The thesis is organized as follows:

- Chapter 2 introduces the required concept and terminology on music theory (Section 2.1 and Section 2.2), knowledge graph and ontologies (Section 2.3), and embedding methods (Section 2.4);
- Chapter 3 provides an overview of related works;
- Chapter 4 describes the development of the ontology on modal theory (Section 4.2) and the corresponding Knowledge Graph (Section 4.3);

- Chapter 5 presents the embedding methods that have been developed and the experiments that have been performed;
- Chapter 6 describes how the presented embedding methods are used to achieve state-of-the-art result on the musical structure segmentation task;
- Chapter 7 provides an overview of the most important results of previous chapters;
- Chapter 8 summarizes the thesis work and highlights some future works.

Chapter 2

Background

This chapter introduces the background knowledge required to make this thesis self-contained. Basic concepts of music theory are introduced in Section 2.1. Section 2.2 discusses the main tasks and challenges in Music Information Retrieval. Section 2.3 introduces Knowledge Graphs and more generally the Semantic Web. Section 2.4 gives an overview of the relevant techniques for word embedding.

2.1 On music theory and chords

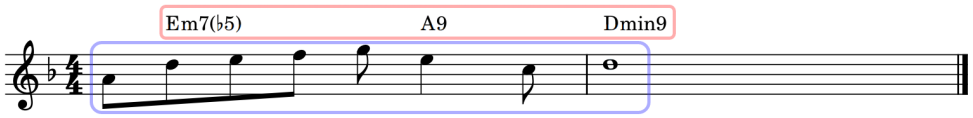


Figure 2.1: A famous theme [1] represented on a music score. The harmony of the theme is highlighted in red while the melody is highlighted in blue. The rhythm of the melody and of the harmony is described by the $\frac{4}{4}$ at the beginning and by the shape of each note.

[1] `https://www.youtube.com/watch?v=krDxhnaKD7Q`

Music theory is a vast and complex field that encompasses a wide range of topics, from the study of musical notation and harmony to the analysis of

musical structures and forms. The study of music theory requires a deep understanding of musical elements such as melody, rhythm, harmony, and form, as well as an appreciation for the historical and cultural context in which music is created. Musicology is the academic field that studies and researches musical theory and is divided into two main branches: historical and systematic musicology [110]. Historical musicology investigates the evolution of music throughout time and how music can be organized according to epochs [110]. Systematic musicology aims at defining the foundational *laws of music* [119] according to three main aspects: rhythm, harmony, and melody. See Figure 2.1 for a visual example of rhythm, harmony, and melody. This thesis focuses on analyzing the harmonic aspect from a computational perspective.

Before diving into notes, chords, and intervals, however, an important point needs to be addressed. Throughout history, many different musical theories have been proposed, but a single, comprehensive and general *Theory of Music* is yet to be made: music is not a physical object and as such a single source of truth can not be identified. Even the most acclaimed theories are subject to harsh criticism, there will always some component of subjectivity that can not be separated in a definite way from musical enjoyment [49, 81, 124].

When analysing music, the cultural setting is a fundamental aspect that cannot be neglected. In this thesis, we always refer to traditional western musical theory. This is not a limitation nor a restriction, but a conscious application of concepts that are not universally true and should always be contextualised in the cultural setting in which they will be applied.

The aim of this work is to provide a novel method for the representation of chords expressed in symbolic notation, i.e. the form in which they are represented in the musicological field. A chord is the main building block in a harmonic progression, which can be seen as a sequence of related chords [9]. The accurate representation of musical chords is arguably the most important

step that needs to be addressed when studying and analysing musical compositions from a harmonic point of view.

> Composition consists in two things only. The first is the ordering and disposing of several sounds [...] in such a manner that their succession pleases the ear. This is what the Ancients called melody. The second is the rendering audible of two or more simultaneous sounds in such a manner that their combination is pleasant. This is what we call harmony and it alone merits the name of composition.
>
> Jean-Benjamin de La Borde [52]

In the following sections, we provide a simple formalisation of the musical concepts that we use in the rest of the thesis. We start by explaining what chords are and their form. We then provide the required information to understand why and how chord sequences are analysed. Finally, we briefly show the current ways of representing chords in symbolic notations.

2.1.1 Chords are sets of related notes

Before introducing what a chord is, we need to address the concept of *note* and how it relates to the auditory tones that are perceived by listeners.

> The basic materials of music are sound and time. [...] Sounds are used to structure time in music [...]. It is the sensation perceived by the organs of hearing when vibrations (sound waves) reach the ear. [...] The frequency of a vibration refers to the number of increased and decreased pressure cycles that occur per unit of time, usually one second. [...] Sound has four identifiable characteristics or properties: pitch, intensity, duration, and timbre. [...] Pitch is the highness or lowness of a sound. Variations in frequency are what we hear as variations in pitch: the greater the number of

> sound waves produced per second of an elastic body, the higher the sound we hear; the fewer sound waves per second, the lower the sound.
>
> Bruce Benward and Marilyn Saker [9]

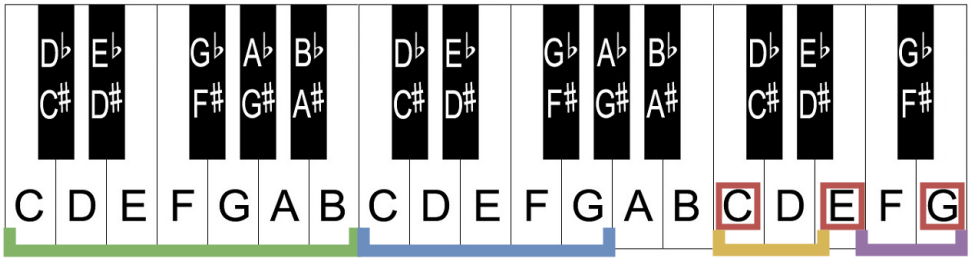


Figure 2.2: Diagram of a piano keyboard. Each key is labeled with its respective note name(s). An octave between two C is highlighted in green. A perfect fifth is highlighted in blue. A major third is highlighted in yellow while a minor third is highlighted in purple. A *C major* triad is highlighted in red.

In the context of western music theory, the whole spectrum of human audible pitches is divided into *octaves*. Octaves are identified as multiples of a particular reference frequency. For instance the pitch $A4$ - a A note in the $4th$ octave of a piano keyboard - has frequency $440Hz$. In order to obtain an A in a different octave it is sufficient to multiply the $A4$'s frequency by an integer $n \in \mathbb{N}$ (i.e. $A5 = A4 \cdot 2 = 880Hz$). The whole spectrum of an octave is divided into a set of 12 elements named pitches. Each multiple of a single pitch is classified as a note. This system takes the name of 12 Notes Equal Temperament system [9] (ET). See Figure 2.2 for a visual reference on an octave (highlighted in green).

In this thesis, we refer to notes using their English naming convention, which is the range of alphabet letters from A to G (see Table 2.1 for a reference of the English and Italian naming conventions compared), to which one or more optional modifiers, namely ♭ (flat) or ♯ (sharp) can be applied. Each combination of a letter and modifiers corresponds to a specific pitch. When a pitch is associated with more than one note name, those notes are classified as enharmonic notes.

A	A♯ B♭	B	C	C♯ D♭	D	D♯ E♭	E	F	F♯ G♭	G	G♯ A♭
La	La♯ Si♭	Si	Do	Do♯ Re♭	Re	Re♯ Mi♭	Mi	Fa	Fa♯ Sol♭	Sol	Sol♯ La♭

Table 2.1: English (top) and Italian (bottom) note naming convention. The most common enharmonic notes are listed.

The relationship between two notes is defined as an interval [9]. The smallest interval is the one between two strictly adjacent notes. Intervals are classified after their distance in number of notes or following a specific naming convention. For instance, the relationship between a C and a G, which are divided by 7 semitones (i.e. 7 consecutive pitches), is called a *Perfect Fifth*. See Figure 2.2 for visual reference on a perfect fifth (highlighted in blue). The etymology of the term is strictly related to the ET system. The term *Fifth* derives from the fact that, if one takes as reference the major scale of the starting note, the fifth note of the scale is 7 pitches distant from the starting note. This can be observed from Figure 2.2: the note C has a distance of 5 white keys from the note G. All the white notes are in the same key, *C major*. The term *Perfect* comes from purely historical reasons: when the ET system has been defined, the frequencies have been arranged in such a way that a *Fifth* interval has an exact ratio of $3 : 2 = 1.5$ with the frequency of the starting note. For instance if we take as starting note an A, whose frequency is $440Hz$, its *Perfect Fifth* has frequency $\frac{3}{2} \cdot 440Hz = 660Hz$. The other frequencies are arranged in order to obtain the best approximation to their correct ratio [6].

The combination of three or more notes is called a chord [9] and is the basic unit of a harmonic progression. The order in which notes appear in a chord is an important aspect for its classification. The first note of a chord is called the root note and names the chord. The relationship between the root note and the remaining notes determines its quality. For instance, a chord is *major* if the interval between its root note and the second note of the chord is

a *major third* (i.e. the notes in the interval are 4 pitches distant, see the interval highlighted in yellow on Figure 2.2 for a visual reference) and the interval between its root note and its third note is a *perfect fifth*. An alternative definition can be provided by taking the relationship between adjacent notes instead of the relationship between the root note and the other components. For instance, a *major* chord can also be defined as a set of notes whose relationship between the first two notes is a *major third* and the relationship between the last two notes is a *minor third* (i.e. notes are 3 pitches distant, see the interval highlighted in purple on Figure 2.2) [9].

Chord qualities are not to be taken as a necessary condition for a set of notes to be considered a chord. Compositions might contain chords that do not fall into any chord quality classification, but a subset of their note does. In that case, we say that the chord is classified as the quality of such subset with the addition of a specific degree, such as in the case of an *E minor seventh with a flat five*, notated as *Em7(♭5)* in Figure 2.1.

Along with its quality, a chord can be classified by the number of notes it is composed as well. For instance, a chord composed of 3 notes is called a *triad* [9]. In Figure 2.2 a visual reference of a *C major* chord is highlighted in red.

2.1.2 Chord notation

From written notation on music scores [9], to digital music scores [60, 113], chords can be represented in many different ways with different levels of granularity. In this thesis we always represent chords using their full name, e.g. *C major*, or by using Harte notation [64] e.g. *C:maj*.

Harte notation was proposed as a simple and intuitive textual notation for musically trained individuals [64]: a chord is always in the form `<root>:<shorthand>(<degrees>)/<bass>` where:

- `<root>` refers to the root note of the chord;

- `<shorthand>` is used to represent the quality of a chord or in general a specific set of intervals defined by experts;
- `<degress>` are optional notes than can be added (or subtracted) from a chord;
- `<alternative bass>` is a note that is played below the root for stylistic reasons, which should not be considered as the chord's root.

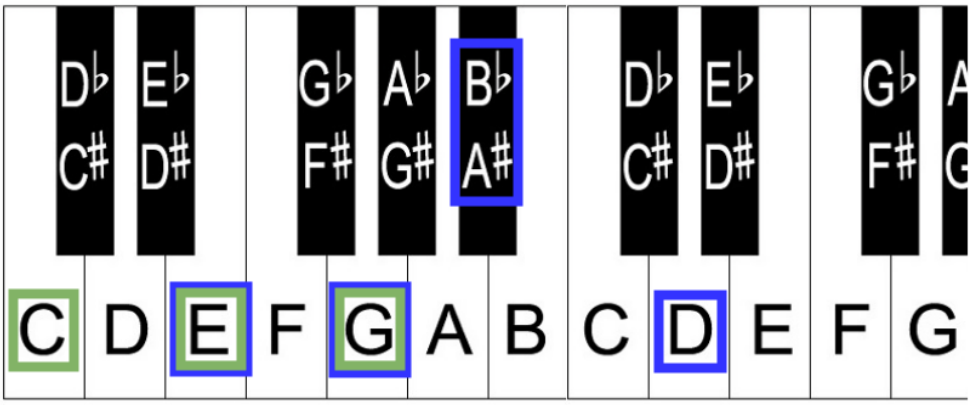


Figure 2.3: Example of *C:maj* (in green) and *E:min7(b5)* (in blue)

For example, a *C major* major chord is represented as a *C:maj* while a *E minor seventh with a flat five* can be represented as *E:min7(b5)*. See Figure 2.3 for a visual reference.

2.1.3 Chords are set of related notes because scales are

In Section 2.1.1 we define a chord as a set of notes classified by the intervals that relates such notes. When an artist writes a musical composition, its attention is usually focused on a specific subset of the whole set of notes. Those subsets take the name of *musical scales* and are of great interest in the musicological field.

In this section a brief description of the concept of scales and tonality is provided. In Chapter 4.2 a more comprehensive description is provided.

A scale is a collection of pitches in ascending or descending order. Scales have developed and changed over the span of various historical periods [9]. The most important ones in western music are the *Major* and *Minor*, which are at the basis of the Tonal theory. The Tonal theory dates back to the Baroque era (1600) [9] and is by far the most important theory that has been used to analyse western music. Other theories have been proposed in order to analyse music that make use of different scales, such as Schenkerian theory [133], Atonal theory [49] or Modal theory [9].

Scales and chords might seem unrelated at first, however, they are different angles from which the same concept can be seen: groups of coherent notes. When a trained musician thinks of a chord he or she does not specifically think of the notes that it is composed of, but rather the scale that contains the notes of that chord [84]. We explore this aspect in Chapter 4, where we model the theory of Modal Harmony by building a knowledge graph that can be used to automatically analyse chords (Section 4.3) and evaluate their encoding with our novel representation method (Chapter 5).

2.2 Music Information Retrieval

Music in general has always been related to technology, see for instance the usage of digital sound storage devices or the use of synthesizer in the music making process. The evolution of technology has been closely followed by music through a continuous redefinition of concepts and terminology [44]. Musicology has evolved and blended with computer science in what has been called Computational Musicology, which is the analysis of music through the use of technology and computers. Even though it has been a problematic transition [66, 83], computational musicology is today a prolific interdisciplinary research area [107].

An active sub-field of computational musicology is Music Information Retrieval (MIR). MIR is a multi-disciplinary field focused on the research and

development of tools to support better usage, understanding, and searching of music catalogs and music knowledge in general [43]. First introduced in the mid-60s, MIR grew consistently with the advent of large available datasets of musical resources. A great number of conferences has established since then, such as the annual International Society of Music Information Retrieval [1] (ISMIR). With the explosion of the usage of Deep Neural Networks, MIR has seen an ever-growing interest, combining state-of-the-art techniques with musical grounded knowledge. This can be partially linked to the advent of music streaming platforms and social media: the necessity of accurate recommending systems, recognising plagiarism for copyrighting purposes, and providing artists with tools that allow them to produce higher quality compositions quicker are among the main driving forces that fuels MIR research [33].

The MIR field can be divided into two main branches based on the representation level on which music is modeled [87]: symbolic-based approaches and audio-based approaches. Symbolic-based approaches, also defined as content-aware techniques, represent musical entities by means of music-theory-inspired methods, such as Harte notation defined in Section 2.1.1 [146]. Audio-based approaches model musical entities through the use of their signal representation [146] - also called audio representation. A multi-modal approach that combines both aspects has been shown to be effective in obtaining accurate MIR systems [91, 98, 99].

In this work, we focus on symbolic approaches.

Music is perceived as a language on its own and is used to convey emotions and ideas through a precise vocabulary [32]. While it might be tempting to draw a direct correspondence with Natural Language Processing, working with symbolic representations of musical entities involves a number of

[1] `http://ismir.net`

features, such as harmonic, melodic, and rhythmic information, that are completely absent in natural language. Nonetheless, both fields share many scientific and technical challenges. Applying techniques that have been proven successful in NLP is a promising approach in MIR [2, 89, 93]. We argue that such techniques, however, need to be contextualized in the musical domain, in order to avoid a shallow application that would result in an over-simplification of the domain and would disregard centuries of ideas and intuitions in musicological research.

2.3 Knowledge graphs

The Semantic Web is a concept developed by Tim Berners-Lee [10] as an evolution of the World Wide Web. The idea is to enable machines to access the semantic meaning of a web page in a structured way.

This is performed by means of a standardised set of technologies and knowledge representation techniques, such as the use of the Resource Description Framework (RDF) [37] and OWL Ontologies [97]. RDF is used to describe the content expressed by a web page while OWL is used to express the semantics of the relationships between concepts and entities declared by the former. Through the identification of each entity using a unique identifier (URI) it is possible to connect information from different sources in a reliable way with respect to the semantics of the data.

One of the key technologies associated with the Semantic Web are Knowledge Graphs. They are based on RDF, OWL and other semantic web standards to represent and organize information in a structured and interconnected way.

RDF is a standard defined by the W3C. It provides a data model to address the representation of complex knowledge bases [37] expressed as triples of the form $(subject, predicate, object)$. RDF is usually serialized in XML, which allows easy interoperability and integration with existing web technologies.

The concept of Knowledge Graph originated first in the 80s, as a way to

define a system that integrates knowledge from different sources. It became a very popular technology in 2012, when Google announced a semantic enhancement of its search engine [45] based on Knowledge Graphs. Different definitions have been proposed for knowledge graphs, some defining a minimum set of requirements [120] while others defining a specific type of data structure [46].

In this thesis we refer to the definition of KG provided by Färber et al [46]:

> We define a Knowledge Graph as an RDF graph. An RDF graph consists of a set of RDF triples where each RDF triple (s, p, o) is an ordered set of the following RDF terms: a subject $s \in U \cup B$, a predicate $p \in U$, and an object $U \cup B \cup L$. An RDF term is either an URI $u \in U$, a blank node $b \in B$, or a literal $l \in L$. U, B, and L are pairwise disjoint.
>
> Färber et al [46]

One key aspect of Knowledge Graphs is the integration of rule-based systems and reasoning techniques directly into the data model, as a way to automatically infer classes and relations based on the semantics of the graph's nodes. This is achieved through the use of ontologies.

> An ontology is as a formal, explicit specification of a shared conceptualization that is characterized by high semantic expressiveness required for increased complexity. [...] The difference between a knowledge graph and an ontology could be interpreted either as a matter of quantity (e.g., a large ontology), or of extended requirements (e.g., a built-in reasoner that allows new knowledge to be derived).
>
> Ehrlinger et Wöß [45]

Ontologies are usually expressed using the OWL standard [97] and, through the expression of a taxonomy and a set of axioms, allow the validation and inference of new data on the basis of the entities that are part of the KG.

Ontologies and in general Knowledge Graphs have been applied to several domains, as a mean to unify the available knowledge in a unique, interconnected resource. The most successful application is arguably the Gene Ontology Resource [3], which has been in continuous development for over 20 years and represents the most important resource for biological research.

2.4 Word embeddings

Representing words as vectors in a geometrical space is a prominent research field in Artificial Intelligence. The core idea is that of finding a representation of words that can be easily handled and transformed with the use of computers [149]. In 2000 Bengio et al. introduced the concept of dense word embeddings [8]. Such representations are composed of real-valued vectors that are usually learned through a corpus of data. The training principle is based on the Distributional Hypothesis: words that are used in a similar context, neighboring words in a corpus, tend to have a similar semantic meaning [132].

With the advent of Deep Learning [82] and the ability to exploit huge corpus of data in an efficient yet accurate form, the computation of word embeddings has seen a new offspring and several different methods have emerged [149]. One of the most influential approaches has been proposed as part of the *word2vec* method: the CBOW (Continous Bag-of-word) and Skipgram models.

Both models are intended as ways of exploiting the Distributional Hypothesis [132] from two similar points of view: predicting the context of a word. In particular, the CBOW method learns a representation by predicting a word given its context while the skipgram model, given an arbitrary word, learns to discriminate against words that never appear in its context. In Figure 2.4 a comparison between the two methods is represented.

Learning a word embedding can hence be seen as a classification task, in which the objective is to discriminate between a word and its neighboring

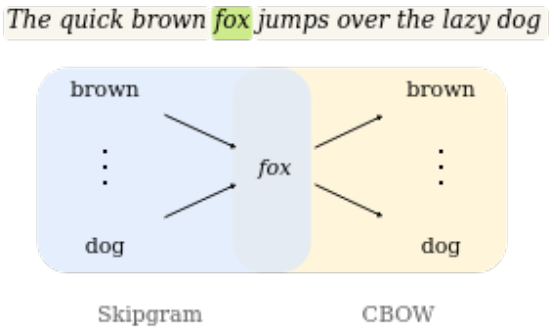


Figure 2.4: CBOW (blue) and skipgram (yellow) methods compared. The word to be embedded is *fox* (green) and the context words are *The, quick, brown, jumps, over, the, lazy, dog*.

context. In order accurately perform such a task, negative examples need to be taken into account, i.e. words that are not in the neighboring context of a word. The simplest and most effective solution is to use as negative samples all those words that, indeed, are not in the neighboring context of a word. When a corpus composed of a large number of words is taken into account, such as in most of the tasks in the natural language processing field, this procedure quickly becomes computationally intractable. In order to mitigate such complexity, negative examples are sampled from the pool of words that are not part of the neighboring context of a word, in a process named *negative sampling*. The number of negative examples to sample is a parameter of the model and, alongside the context window size, has been shown to be the most influential parameter that needs to be considered when training embeddings [21].

This kind of word embedding has been shown to encode interesting compositionality properties as well: for instance, summing together the words *Czech* and *currency* results in an embedding that is close to the embeddings of *koruna*, the Czech republic currency.

Chapter 3

Related work

Chapter 3 overviews the related works and is divided into two main sections: Section 3.1 describes existing ontologies to model symbolic music knowledge, while in Section 3.2 we describe the methods used to represent chords as multidimensional vectors.

3.1 Music ontologies

Many different approaches have been presented to model musical elements using ontologies in the last decade. The MIDI Linked Data Cloud [102] models the interconnection of symbolic resources represented by MIDI resources files, the CHARM Ontology [63] models musical structure based on the CHARM specification: a representation system that allows modeling abstract concepts that are implicitly represented by musical compositions. MusicOWL [71] is an ontology designed specifically to model music sheets, much like the Music Notation Ontology [25]. The Chords Ontology [48] and the Music Theory Ontology [126] have been proposed to model the fundamental rudiments that compose music and are analyzed in detail in Section 3.1.1 and Section 3.1.2, respectively. The Functional Harmony Ontology [72] has been recently proposed as a way to automatically perform functional analysis on harmonic progressions and is analyzed in Section 3.1.3.

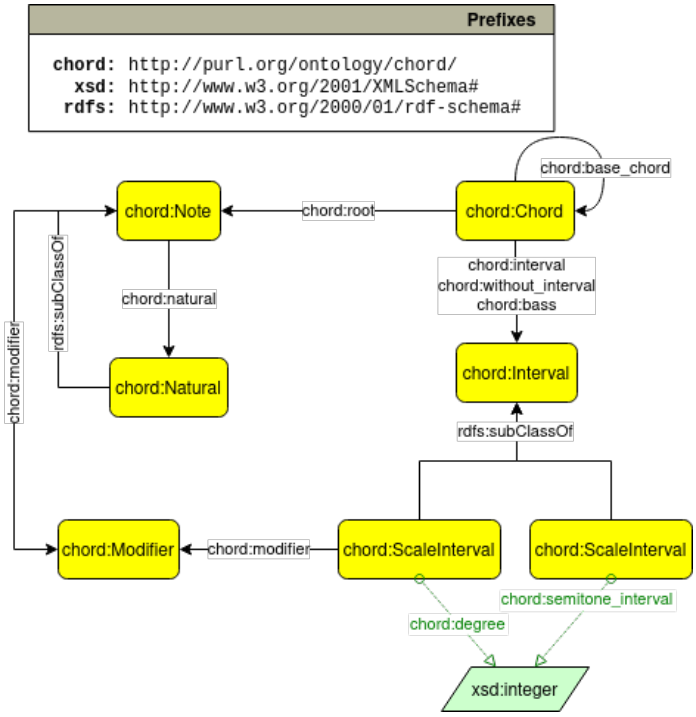


Figure 3.1: Chord ontology using Graffoo diagrams

3.1.1 OMRAS2 Chord Ontology

The OMRAS2 project, namely *Ontology-driven Music Retrieval & Annotation Sharing service*, is a collection of tools, ontological framework and software to share music data on the Semantic Web [48]. Among the results of such project, the Chord Ontology has been produced [1]. The ontology model is based on the formalization of chords defined by Harte [64].

In Figure 3.1 a visual representation of the Chord ontology is provided. A chord is identified by its root note (as explained in Section 2.1.1), its component intervals (or missing intervals) and an alternate bass, if present. The conversion between a chord in Harte format to an entity of type *chord:Chord* is trivial. In Figure 3.2 a *C:maj* chord is represented using the Chord ontology.

The ontology implemented in Chapter 4.3 reuses the Chord ontology to represent chord individuals. This allows an easy yet expressive way of formalizing chords, since all the chords that are part of the knowledge graph are

[1] `http://purl.org/ontology/chord/`

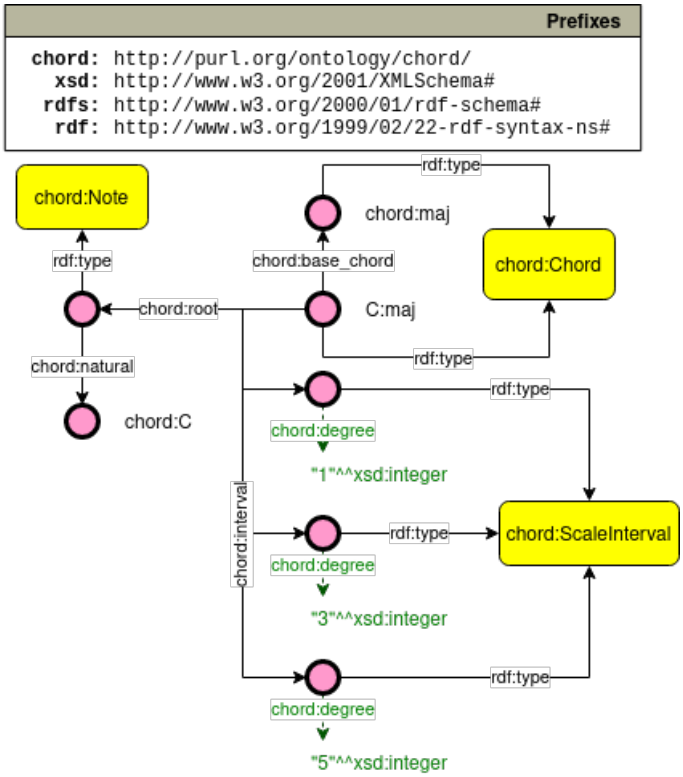


Figure 3.2: *C:maj* chord represented using the Chord ontology

expressed in Harte notation.

3.1.2 Music Theory Ontology

The Music Theory Ontology (MTO) [126] is an ontology developed to represent the main rudiments that are used to express symbolic music and allow the analysis of musical composition.

It reuses the Chord ontology described in Section 3.1.1 and extends its scope to take into account other musical concepts. For instance, one of the main differences lies in the way chords are represented. MTO represents chord qualities as classes, whereas the Chord Ontology represents chord qualities as individuals. This particular choice is fundamental in the ontology developed in Chapter 4, since it allows us to abstract the formalisation from the actual chord qualities and implement axioms that make use of general chord qualities. The same is performed with intervals, where the interval quality is represented

as a class.

In MTO chord qualities are modeled as a subclass of the class `Interval`, called `HarmonicInterval`. Different levels of granularity are taken into account which involves both the number of notes that composes the chord and its quality, as described in Section 2.1.1. This is a key aspect at the core of the ontology developed in Chapter 4. We extend it to cover a bigger set of chord qualities that are then used to automatically classify chords through the inference mechanism.

MTO addresses the concept of progressions as well. Chord progressions are modeled as subclasses of the `HarmonicProgression` class, which is a subclass of the `Progression` class. The latter specializes the class of `Progression` to scales as well (defined in Section 2.1), modeled by the `MelodicProgression` class. Such class is specialized in an extensive taxonomy of scale types, among which the `MajorScale` and `NaturalMinorScale` are defined. The usage of such classes allows the definition of the Tonal Harmony theory (Section 2.1) in terms of classes and properties defined by MTO.

Few general object properties are defined, mainly to model the relationship between chords and scales and their components in terms of notes (`hasNote`, `hasRootNote`, `hasTonic`, `hasDiatonicDegree`). The property `hasDiatonicDegree` is extended in Section 4.2 to allow the assertion of the relationship between a scale and the chords that can be interpreted as part of such scale.

3.1.3 Functional Harmony Ontology

Functional Harmony Ontology (FHO) [72] is an ontology developed to perform functional analysis of chord sequences in the context of the theory of Modal Harmony.

The functional analysis of a musical composition is the task of classifying its harmonic progression (or respectively its melodic progression) using the rules defined by a musical theory. In particular, by assigning a role to each

component of the progression, any subsection of the whole sequence can be classified on the basis of the rules proposed by the reference theory. This can be seen as a way to *"explain"* why a chord is part of a sequence and how it can be interpreted in the context of the whole progression or composition. Such a task is particularly useful to musicians in the composition of a musical piece or when improvising a melody over a chord progression, since it gives a conceptualization of which melody is better suited for a harmonic progression or vice-verse which harmony is better suited for a melody [84].

This is performed through extensive use of Description Logic (DL) axioms.

The ontology reuses MTO (see Section 3.1.2), and extends it by formalizing the musical concepts of tonalities, chords and chord's function. Chords, differently to MTO and Chord ontology, are represented as classes instead of instances. This allows the usage of DL axioms to model chord sequences. The ontology can be used to analyze chord sequences and it achieves results comparable with similar tools [72] such as HarmTrace, which allows the derivation of a tree-shaped functional analysis of a composition [35] through the use of a rule-based system. Even though FHO is not able to infer the hierarchical structure that can be obtained by HarmTrace, the authors shows that the classification obtained is as informative as the one provided by HarmTrace.

There is a significant content overlap between FHO and the Modal Harmony Ontology of Chapter 4. In fact, FHO has been of major inspiration in the implementation of our ontology. The formalization proposed in Chapter 4, however, is fundamentally different to FHO in the conceptualization and the modeling choices of the Modal Harmony theory.

In FHO the formalization of the Modal Harmony theory is bounded to the chord sequence analysis task. Each chord is represented as a class and a set of axiom are specified to classify its function.

Extending such an ontology to classify additional chords is difficult, as the chords modeling process has not been documented. Even though the ontology

proposed in Chapter 4 addresses the same musical theory as FHO, the problem is modeled from a more general point of view, in which an higher degree of flexibility is maintained when defining the classification rules for each chord. Instead of modeling the whole Functional Modal Harmony theory only the Modal Theory is address. The concept modeled by FHO can be implemented reusing the ontology proposed in Section 2.1. We argue that such an approach would results in an higher interoperability, easier ontology update and, most importantly, to a more general approach that can be extended to other theories as well.

Indeed, as explained in Section 2.1, there is not a single source of truth for a musical theory. The functional definition of a chord often depends on the context of application, the source of the musical composition or even the personal preferences of the analyst.

3.2 Chord embeddings

In this section, we provide an overview of the methods that have been used to compute chord embeddings and the results that have been obtained.

In chord2vec [93] the authors implement a chord embedding method based on the skipgram model adapted to the musical domain. Chords are encoded using their constituent notes and are then embedded using the skipgram model presented in [104]. Three different embedding flavors are tested:

- *bilinear*, which assumes each note in the chord to be independent;
- *autoregressive*, which loosen the independence assumption;
- *sequence2sequence*, which implements an encoder-decoder model and implements a language model.

The described chord2vec flavors are listed in increasing complexity and expressivity order. This order reflects their performances as well. In particular,

the *sequence2sequence* model outperforms all of the other models in the average negative log-likelihood per chord, which means that the model is able to accurately act as a language model. Our proposed embedding method is similar to chord2vec and can be seen as a generalization of the *autoregressive* model. We re-implemented a variation of the *bilinear* model based on the original word2vec skipgram model, as suggested on an additional report [2] by the authors of chord2vec, since it allows us to perform a direct comparison with other types of embeddings.

In [80] the authors encode chords using their textual label and compute embeddings using word2vec. The resulting embeddings are analyzed visually through dimensionality-reduction techniques and evaluated in two different tasks: artist attribute prediction and language modeling. From visual inspection, the chords are observed to be embedded in a position that directly resembles the circle of fifth, which is a well known construct to measure the distance between chords [9]. The trained embeddings are also able to accurately approximate a language model, implemented using a Recurrent Neural Network, when compared to other encoding techniques based on bag-of-pitches i.e. chords are represented as a one-hot-encoding of the note components, and a bag-of-word method in which each chord is represented using TF-IDF from the corpus statistics. Good results are obtained in the artist attribute prediction as well. A Convolutional Neural Network is used to predict the attributes of each artist, such as gender and country. The model that uses embeddings outperforms all the other models in the classification of every attribute. A similar approach is performed in [2] in which chord sequences trained using word2vec are evaluated in a language modeling task and a clustering task in the context of historical chord sequence modeling. The results from the language modeling task confirms the fact that such embeddings are adequate to model chord progression. Furthermore the authors shows that historical differences are encoded as well: composers that are notoriously hard to analyse,

[2] `https://github.com/Sephora-M/chord2vec/tree/master/report`

such as Debussy, have a much lower accuracy when compared to better understood composers. In the clustering task the authors are able to show how such embeddings are able to correctly classify functionally equal chords, which are different chords that can be interpreted with the same functional role [2]. We compare our novel embedding method to embeddings obtained with the same methodology of the related works.

Chapter 4

Symbolic Music and Knowledge Graph

Chapter 4 describes the creation of an ontology that models the theory of Modal Harmony and a corresponding Knowledge Graph. Section 4.1 provides a description of the Modal Theory that we model. In Section 4.2 we provide a description of the modeling phase of the ontology and finally, in Section 4.3 we describe the Knowledge Graph that is built upon such ontology.

4.1 Modal harmonic theory

In this Section, we briefly introduce the concepts that we model in the proposed ontology. They are based on The jazz theory book by Mark Levine [84] and on Music in Theory and Practice Volume 1 by Bruce Benward [9].

As already explained in Section 2.1, chords are sets of coherent notes and, as such, they are subsets of scales. One could even think of a chord as being a scale on its own and, more generally, part of another scale. This type of relationship is at the basis of the theory of Modal Harmony, which we refer to as TMH from here on. TMH is based on scales composed of seven notes. Each scale, together with its root, is classified by its tonality. The tonality of a scale is yet again a note. For instance, one might have a scale S which has

root note C and tonality E. Seven scales are defined by TMH: Ionian, Dorian, Phrygian, Lydian, Mixolydian, Aeolian, and Locrian. The Ionian scale is the equivalent of a major scale while the Aeolian is the equivalent of a minor scale, hence the theory of Tonal Harmony, based only on Major and Minor scales, is a proper subset of the theory of Modal Harmony. Each of these scales is built starting from a specific grade (i.e. position in the scale) of the major scale. For instance, the Ionian scale is a major scale built from the first grade (the first note) of the major scale, while the Phrygian scale is built from the third grade (the third note) of a major scale.

Take for example the scale of *C major*, we have that its root is C and its tonality is C as well. If we want an Ionian scale in the tonality of C then we should build a major scale starting from the first note of the *C major* scale, which is the major scale itself. If we want to build the Phrygian scale in the tonality of C then we need to build a major scale starting from the third note of the *C major* scale, which is E. The resulting scale has tonality C and is composed of the notes, in order, $[E, F, G, A, B, C]$. The root note of the scale is hence E, since it is the first note of the scale, and is named *E Phrygian*, since it has been composed using the definition of Phrygian scale. The tonality of the scale is still C, the note that we used as the foundation of the scale in the first place. Indeed, if we take a look at this scale, we can see how it has the same set of notes of the *C major* scale, in a different ordering.

When we make use of scales composed of seven notes, each position in the scale is associated with a name, to avoid naming degrees in a numeric form. These positions are, in order: tonic, supertonic, mediant, subdominant, dominant, submediant and leading-tone. Upon each of these roles, using the remaining set of notes that compose the scale, a number of chords can be constructed, which has as root note the corresponding note of that role. For instance, in a *C major* scale, the first note (C) is the tonic note and the major chords that are built upon that note are called tonic chords. Given the scope of this thesis, we do not provide a detailed description of which chord quality

corresponds to a specific role. It is sufficient to know that each role has one or more corresponding chord qualities.

If we take again the example of the *E Phrygian* scale analyzed before, we have seen that it contains the same set of notes of the *C major* scale, hence the set of chords that are part of that scale are exactly the same. Indeed, the first chord of *E Phrygian*, the tonic chord, has root note E. Using the set of available notes, such chords turn out to be a minor chord, which is the third chord, the mediant chord, of the scale that originated *E Phrygian*, *C major*.

This kind of relationship between chords and scales is a key aspect of TMH: each chord can be classified with a role based on the context it is interpreted in. This allows the classification of a chord based on its function (i.e. its role) within a scale and is the basis of Functional Harmonic Analysis [127].

4.2 An ontology of Modal (and Tonal) harmonic theory

Section 4.2 describes the formalization of the theory of Modal Harmony described in Section 4.1 in an ontology, to which we refer to as Modal Harmony Ontology.

We use Graffoo diagrams [1] to describe the implementation and modeling choices. Graffoo diagrams represent classes as yellow rectangles. Whenever the rectangle border is dotted it represents a class restriction, usually expressed using Manchester syntax [67]. Pink circles are used to represent individuals of the ontology. Classes and individuals are connected by arrows whose label is the name of the predicate that describes such a relation. We represent inferred properties as dotted red arrows.

The competency questions answered by this ontology are the following:

- Which are the notes in a mode?

[1] `https://essepuntato.it/graffoo/`

- What is the role of a note in a mode?
- In which role can a note be classified?
- Which chords are in a mode?
- Which are the roles of a chord?
- Which are the chords that absolve a role?

We formalize some examples of how these questions are answered using SPARQL [65] queries in Section 4.3.

Chord classification by using reasoning

In order to model chords, the implemented ontology imports two other ontologies: the Chord Ontology [48] and the Music Theory Ontology [126]. Both ontologies are combined and extended in order to obtain an ontology that, using reasoning, can classify automatically the quality of a chord from its constituent notes.

We extended the Music Theory Ontology (mto) to include 24 missing intervals, most of which are compound intervals. Compound intervals are intervals that can be expressed as a combination of smaller intervals. For instance, a *major ninth* interval can be decomposed into a *perfect octave* interval to which an additional *major second* interval is added. We extend the ontology with additional chord qualities as well. 56 additional chord qualities are added and commented on the ontology. Each of these classes is used to automatically classify a chord instance represented using the chord ontology.

```
@prefix mto: <http://purl.org/ontology/mto/> .
@prefix mto-kb: <http://purl.org/ontology/mto/kb/> .
@prefix chord: <http://purl.org/ontology/chord/> .

<C:maj> rdf:type chord:Chord ;
        chord:root mto-kb:C ;
        chord:interval mto:MajorThirdInterval ;
        chord:interval mto:PerfectFifthInterval .
```

Listing 4.1: *C:maj* represented in turtle syntax

In Listing 4.1 a *C:maj* chord is presented in Turtle syntax [7]. It is defined as a chord composed of a *major third interval*, a *perfect fifth interval*, and a *C* root.

Note that all major chords are defined as chords whose components are a *major third interval* and a *perfect fifth interval*. We automatically infer such classification with the axiom presented in Listing 4.2 in Machester syntax [67].

```
Class: MajorTriad
  EquivalentTo: (chord:interval value mto:MajorThirdInterval)
      and (chord:interval value mto:PerfectFifthInterval)
```

Listing 4.2: Axiom to infer the classification of a major triad

Such an axiom can be expressed for any chord quality, see Figure 4.1 for a visual representation of the axiom represented as a general pattern for the classification of chords using OWL reasoning.

The described axioms can be inferred by a reasoner that implements the OWL-EL profile, which is able to perform inference in polynomial time [109]. While the proposed extension does not answer any additional competency

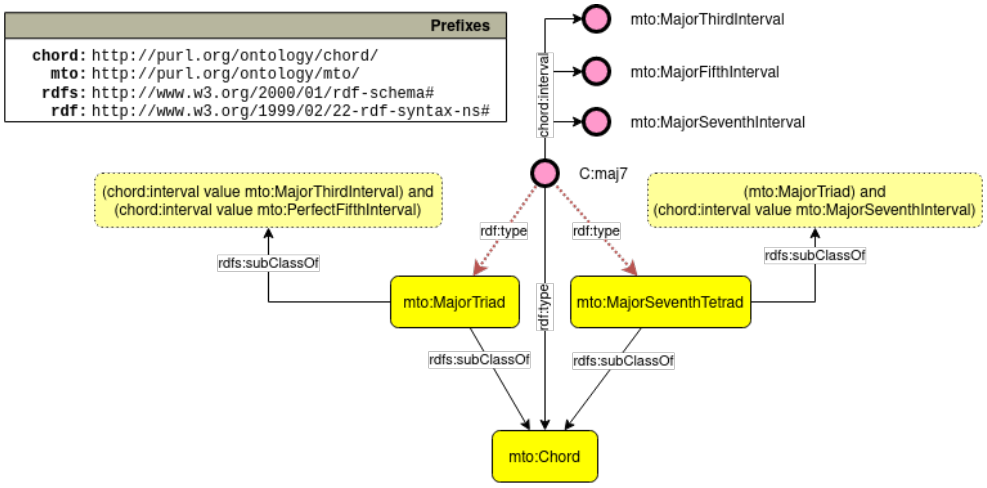


Figure 4.1: Graffoo representation of the extended MTO ontology in the classification of a *C:maj7* chord, which is a *Major Triad* and a *Major Seventh Tetrad*. The inferred predicated are depicted as red dotted arrows.

question with respect to the original MTO proposal, being able to automatically infer the classes of a chord represents a proper, valuable addition.

Moreover, when chords are expressed in Harte notation [64], the proposed extension allows the usage of general chord qualities when defining ontologies that models harmonic progressions.

For instance, in [51] Allen Forte identifies the famous Tristan chord [95] in a number of popular American ballads. Given a set of tunes annotated by an expert in Harte notation it is easy to accomplish such task by defining an axiom for the Tristan chord, as exemplified in Listing 4.3.

```
Class: TristanChord
  EquivalentTo: (chord:interval value mto:
     AugmentedFourthInterval) and (chord:interval value mto:
     AugmentedSixthInterval) and (chord:interval value mto:
     AugmentedNinthInterval)
```

Listing 4.3: Axiom to infer the classification of a Tristan chord, which is composed of an augmented fourth interval an augmented sixth interval and an augmented ninth interval.

4.2.1 Modal Harmony ontology

Section 4.2.1 describes the modeling choices in the implementation of an ontology that formalises the concepts of Modal harmony explained in Section 4.1. The ontology imports the extended version of the Music Theory Ontology and use it as a structural backbone, in order to define relations between chords as a function of their quality.

Figure 4.2 shows the main pattern used to represent modal scales in the ontology. Each modal scale, such as Ionian, Dorian and so on, is represented as subclasses of `mto:Scale`.

In the example, the Ionian scale is represented and the restriction imposed on chords that have the role of Ionian tonic chords are shown. Such quality restrictions are a key aspect of this ontology. They can be updated by domain experts and eventually refined given the domain of application of the ontology. Restrictions can be highly dependent on the music genre: the concept of tonic chord is slightly different between Rock music [38] and Jazz [84]: the former has a broader definition while the latter is usually characterized by richer chords and therefore has a more restrictive definition.

In order to automatically infer that a chord is part of a scale the root note

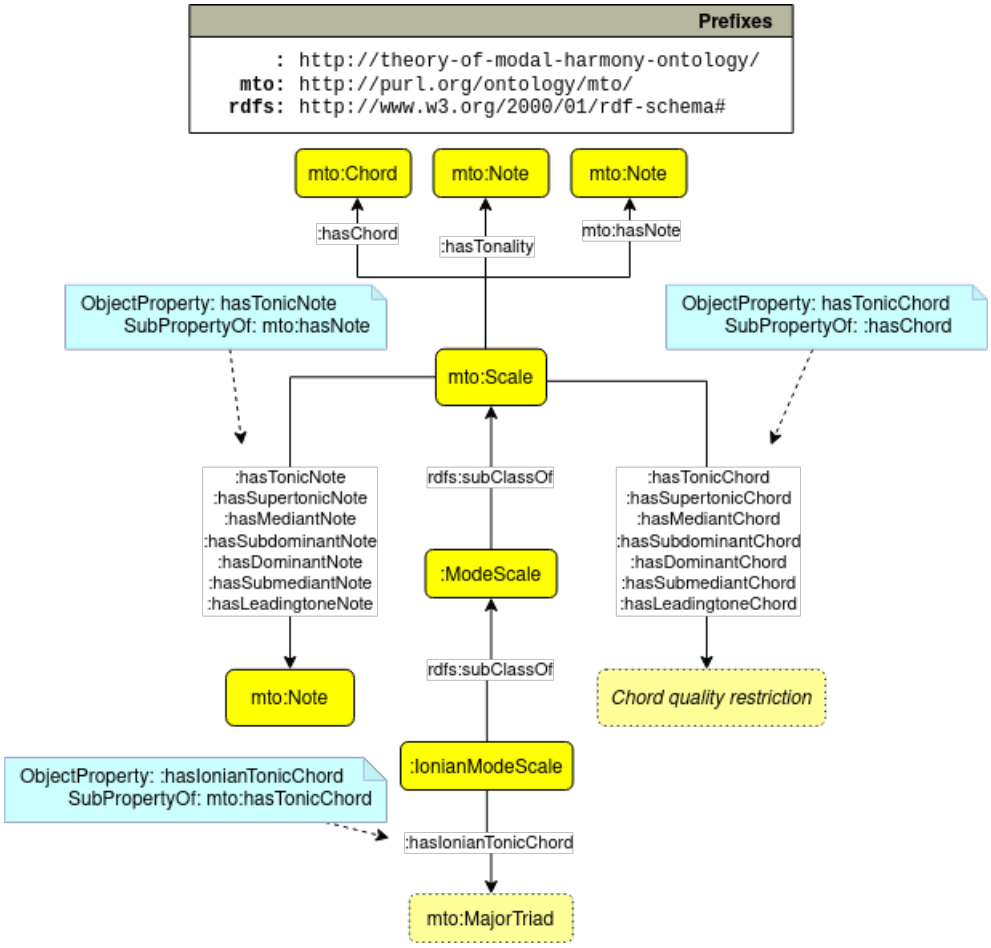


Figure 4.2: Main structure of the Modal Harmony ontology.

of a chord needs to be taken into account as well as the tonality of the scale. In the example of Figure 4.2, the major triads that should be classified as tonic chords of the Ionian scale are only those whose root note matches the root note of the scale and the chord is classified as a major triad. One possible approach is to define domain and range on such properties. Even though this results in correctly inferring the scale type and the chord quality, it fails to automatically instantiate the property between all compatible instances. For example, if a chord, e.g. *C:maj*, is part of the tonic chords of a Ionian scale - through the `:hasIonianTonicChord` property - then it is inferred to be a major triad chord. The inverse cannot be inferred automatically: given a *C:maj* chord the creation of the `:hasIonianTonicChord` need to be manually asserted in the ontology.

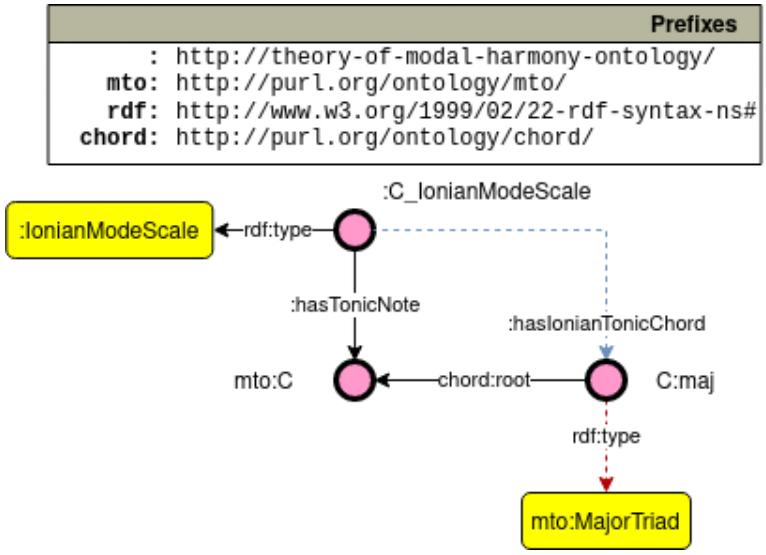


Figure 4.3: Example of required inferences to classify a *C:maj* chord as the Ionian tonic chord of a C Ionian scale. Inferred properties are represented as red dotted arrows. Properties that needs to be inferred by the proposed ontology are represented as blue dotted arrows.

In Figure 4.3 a visual example is depicted. The property `rdf:type` between the individual *C:maj* and the class *mto:MajorTriad* is automatically inferred by the reasoner using the axioms defined by the MTO extension presented in Section 4.2. The property `:hasIonianTonicChord`, represented using a dotted blue, cannot be inferred automatically through the use of domain and range axioms.

To implement such mechanism we rely on the use of the *rolification* modeling technique [76]. Such technique allows the definition of axioms that acts as rules of the form *if-then* using OWL2 [76].

We *rolify* an arbitrary class C by creating an ad-hoc property for that particular class, called R_C. The restriction $C \equiv R_C.Self$ is imposed on such class: if an individual of that class exists, then an instance of the property R_C is inferred where that same individual is both subject and object of R_C. Formally, for every entity e of type C we have that $R_C(e, e)\ \forall\ e \in C$.

A rule of the form *if-then* can be described by using property chain axioms, which are OWL axioms that define the composition of two properties.

For instance given the properties R_1, R_2 and R_3 and a property chain axiom $R_1 \circ R_2 \rightarrow R_3$ defined on such rules, we have that $\forall A, B, C\ R_1(A, B) \wedge R_2(B, C) \rightarrow R_3(A, C)$ with A, B, C individuals of the ontlogy.

Take as an example a toy ontology that models mouses and elephants. We can model the fact that mouses are always smaller than elephant using an *if-then* rule, here formalized in First Order Logic:

$$Mouse(x) \sqcap Elephant(y) \rightarrow smaller(x, y)$$

where x and y are variables, $Mouse(x)$ and $Elephant(x)$ are unary predicates that represents, respectively, that a variable is a mouse or an elephant and $smaller(x, y)$ is a binary predicate with the semantics of $x < y$.

Using the rolification technique we can define the same rule as

$$R_M \circ U \circ R_E \sqsubseteq R_s$$

where R_M and R_E are respectively the rolification of the $Mouse$ and $Elephant$ predicates, U is the universal property (*owl:topObjectProperty* in OWL2) and R_s is the predicate $smaller$.

In Listing 4.4 an Example of the rolification technique applied to Ionian major chords is provided in Turtle syntax.

```
ObjectProperty: :R_IonianModeScale
Class: :IonianModeScale
  EquivalentTo:
    :R_IonianModeScale some Self

ObjectProperty: :R_MajorTriad
Class: mto:MajorTriad
  EquivalentTo:
    :R_MajorTriad some Self

ObjectProperty: :hasIonianTonicChord
  SubPropertyChain:
    :R_IonianModeScale o :hasTonicNote o inverse(chord:root)
       o :R_MajorTriad
  Domain:
    mto:IonianModeScale
  Range:
    mto:MajorTriad
```

Listing 4.4: Example of rolification to infer Ionian tonic chords in turtle syntax.

A general algorithm can be derived to automatically implement the described rolification technique. The resulting ontology allows domain and range axioms to be used as preconditions on the element of the rule for the application of a property chain axiom by the reasoner. This allows the implementation of the model in Figure 4.3.

More formally, given a property P in which a general property chain $chain(P)$ is defined and given a sub-property $p \sqsubseteq P$ on which domain and range axiom are defined respectively as $domain(p)$ and $range(p)$, we can specialize the property chain $chain(P)$ to be automatically realised by a reasoner using Algorithm 1.

Algorithm 1 Automatic rolification algorithm. The ▶ symbol is used to represent the new Manchester syntax axioms that needs to be asserted in the ontology.

Require: P an object properties, optionally with a property chain $chain(p)$ axiom inferred
Ensure: $\forall P' \in P$, P' have domain and range restrictions defined
  for all $P' \sqsubseteq P$ **do**
    ▶ `ObjectProperty:` $\mathtt{R}_{domain(P')}$
    ▶ `Class:` $\mathtt{R}_{domain(P')}$ `EquivalentTo:` $\mathtt{R}_{domain(P')}$ `some Self`
    ▶ `ObjectProperty:` $\mathtt{R}_{range(P')}$
    ▶ `Class:` $\mathtt{R}_{range(P')}$ `EquivalentTo:` $\mathtt{R}_{range(P')}$ `some Self`
    ▶ P' `SubPropertyChain:` $domain(P')$ `o` $chain(P)$ `o` $range(P')$
  end for

For example in Figure 4.2 the property `:hasTonicChord` needs the property chain `:hasTonicNote`$\circ$`inverse(chord:root)`, which states that a tonic chord is a chord whose root is the same as the scale's root, to be inferred in order to allow the correct rolification of its sub-properties, such as `:hasIonianTonicChord`. Furthermore, `:hasIonianTonicChord` needs to define proper domain and range axioms, in this case respectively `:IonianModeScale` and `mto:MajorTriad`.

The algorithm has linear complexity $O(|P|)$, since we loop through every property only once. An exponentially complex reasoning profile is required, since we might need to infer the inverse of object properties, represented by the OWL-DL reasoning profile [109]. Some reasoners, however, allow the inference of inverse properties even when using more efficient profiles, such as OWL-EL.

The final ontology is automatically generated by using the music21 library [34] to retrieve the association between a modal scales and its notes. Algorithm 1 is used to rolify the classes that represent a mode. A total of 6344 axioms are implemented in the ontology.

4.3 Modal Harmony Knowledge Graph

Given the ontology presented in Section 4.2, we populate a Knowledge Graph (KG) by converting a corpus of chords in a notation compatible with the extended version of the Music Theory Ontology and further classify the chords

using the ontology developed in Section 4.2.

The entities of the KG are extracted from ChoCo [36]. We load all the chord entities along with the relevant ontologies in Stardog [2] to obtain the classification results of each chord. Stardog is used because of its efficient implementation of the EL reasoning profile [135], which allows a complete usage of the reasoning requirements defined in Section 4.2.

A total of 7651 chords individuals are contained by the KG, which allows to answer the competency questions formulated in Section 4.2 by using the following SPARQL queries:

```
PREFIX mto: <http://purl.org/ontology/mto/>
PREFIX : <http://theory-of-modal-harmony-ontology/>
SELECT ?note
WHERE {
    :CSharp_IonianModeScale mto:hasNote ?note .
}
```

Listing 4.5: SPARQL query for the competency question *Which are the notes in mode?* using the Ionian mode in $C\sharp$.

[2] https://www.stardog.com/

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX mto: <http://purl.org/ontology/mto/>
PREFIX : <http://theory-of-modal-harmony-ontology/>

SELECT ?note ?role
WHERE {
    :CSharp_IonianModeScale ?rolePred ?note .
    ?rolePred rdfs:subPropertyOf mto:hasNote ;
              rdfs:label ?role .
}
```

Listing 4.6: SPARQL query for the competency question *Which is the role of a note in a mode?* using the Ionian mode in $C\sharp$.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX mto: <http://purl.org/ontology/mto/>
PREFIX mto-kb: <http://purl.org/ontology/mto/kb/>
PREFIX : <http://theory-of-modal-harmony-ontology/>

SELECT DISTINCT ?scale ?role
WHERE {
    ?scale ?rolePred mto-kb:C .
    ?rolePred rdfs:subPropertyOf mto:hasNote ;
              rdfs:label ?role .
    FILTER (?rolePred != mto:hasNote) .
}
```

Listing 4.7: SPARQL query for the competency question *In which role can a note be classified in?* using the Ionian mode in $C\sharp$.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <http://theory-of-modal-harmony-ontology/>
SELECT ?chord
WHERE {
    :CSharp_IonianModeScale :hasChord [ rdfs:label ?chord ] .
}
```

Listing 4.8: SPARQL query for the competency question *Which chords are in a mode?* using the Ionian mode in $C\sharp$.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX chord: <http://purl.org/ontology/chord/>
SELECT DISTINCT ?role
WHERE {
    [] ?pred [ a chord:Chord ; rdfs:label "C:maj" ] .
    ?pred rdfs:subPropertyOf fho:hasChord ;
          rdfs:label ?role .
}
```

Listing 4.9: SPARQL query for the competency question *Which role are absolved by a chord?* using a *C:maj* chord.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX : <http://theory-of-modal-harmony-ontology/>
SELECT DISTINCT ?chord
WHERE {
    [] :hasLocrianTonicChord [ rdfs:label ?chord ] .
}
```

Listing 4.10: SPARQL query for the competency question *Which are the chords that absolve a role?* using the Locrian tonic role.

The presented SPARQL queries provide an empirical evidence that it is possible to accurately formalize and encode music theories, as addressed in RQ1 of Chapter 1 - namely *Is it possible to formalize and encode music theories in an accurate form?*.

4.3.1 Roman Notation inference using SPARQL query

A number of tasks, relevant to MIR, can be addressed using the presented knowledge graph. One of those is the inference of Roman Notation [144] from a sequence of chords. Roman chord notation is a chord notation tighly related to the functional analysis of the harmonic progression of a composition. In its most basic form, given a reference scale such as the *C major scale*, each chord is represented by the degree of its root note in the reference scale. The degree is encoded using the roman numeral representation (e.g. degree 7 is represented ad *VII*).

In Listing 4.11 the SPARQL query to perform such task is reported for the chord progression *C:maj - G:maj - A:min - F:maj*, a very common chord progression in pop music [3] usually annotated as *I - V - vi - IV*.

The query retrieves scale to which each chord belongs to and convert its

[3] See `https://en.wikipedia.org/wiki/I-V-vi-IV_progression` for an extensive list of songs using the same progression.

role in Roman Notation, which is one of the annotations provided in the ontology. It may happen that a scale contains only a subset of the notes in the progression. In that case the query might result in a partial annotation of the sequence. In Table 4.1 the result of query 4.11 is described. Partial annotations have been manually removed. The traditional *I* - *V* - *vi* - *IV* annotation is correctly retrieved by the query, alongside many other annotations. Each of this can be seen as a different way to musically interpret the chord sequence and are a useful tool for trained musicians, both in the composition and the improvisation phase [84].

```
PREFIX mto: <http://purl.org/ontology/mto/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX : <http://theory-of-modal-harmony-ontology/>

SELECT DISTINCT ?scaleLabel (GROUP_CONCAT(DISTINCT ?
    chordRoman; SEPARATOR=" ") AS ?p)
WHERE {
    { ?chord rdfs:label "C:maj" } UNION
    { ?chord rdfs:label "G:maj" } UNION
    { ?chord rdfs:label "A:min" } UNION
    { ?chord rdfs:label "F:maj" }

    ?scale ?scaleChordPred ?chord ;
         rdfs:label ?scaleLabel .

    ?scaleChordPred rdfs:subPropertyOf :hasChord .
    FILTER(?scaleChordPred != :hasChord) .
    BIND(IRI(CONCAT(STR(mto:), REPLACE(REPLACE(STRAFTER(STR(?
        scaleChordPred), STR(:)), "has", ""), "Chord", "")))
        AS ?chordRole) .
    ?chordRole skos:altLabel ?chordRoman .
}
GROUP BY ?scaleLabel
```

Listing 4.11: SPARQL query for the competency question *Which are the possible Roman Notation interpretation of a chord progression?*

The task of notating chords in Roman Notation can be achieved with the music21 library as well [34]. When using music21, however, a prior knowledge of the reference scale within which the progression should be interpreted needs to be explicitly provided. Using a query similar to the one in Listing 4.11

Scale	Roman annotation
F Lydian Mode	V ii iii I
D Dorian Mode	°vii IV V iii
A Aeolian Mode	iii °vii I vi
F♯ Minor Scale	iii °vii I vi
G Dorian Mode	IV ii °vii
G Mixolydian Mode	IV I ii °vii
E Phrygian Mode	vi iii IV ii
B Locrian Mode	ii vi °vii V
C Ionian Mode	I V vi IV
C Major Scale	I V vi IV

Table 4.1: Roman text annotation of the chord progression *C:maj - G:maj - A:min - F:maj*

does not require any additional information besides the chord progression itself. More complete methods have been proposed in literature to perform the same task [19, 103]. The recognition of complex roman notations, such as parallel chords [9], are taken into account by such methods while the query in Listing 4.11 can only handle simple notations and should be seen as a proof of concept of applying the KG to the roman annotation task. We claim, however, that by extending the presented query into a proper annotation system it would be possible to obtain a method that is directly comparable to the related works. We will investigate this option in future works.

Chapter 5

Chord Embedding

The current section describes in detail all the techniques that we used to compute chord embeddings as well as the experiments that we performed to test which one is better suited for the task. We tackled the problem of encoding a chord from three different angles, described respectively in Section 5.2, Section 5.3, and Section 5.4. In particular, in Section 5.2 we encode a chord using its syntactical level. This is an important aspect since, when music is transcribed by experts, the choice of chord naming is the result of a meticulous analysis. The subjectivity of an annotator, its musical proficiency, and instrumental preferences represents an important aspect in the annotation process [74] which needs to be taken into account when approaching musical symbolic notations. In Section 5.3 we encode a chord by taking into consideration its intensional aspect: the notes that it is composed of. Similar works, such as *chord2vec* [93] showed that more advanced representations, based on the notes that form a chord, are a prerequisite for obtaining accurate results. In Section 5.4 a chord is encoded using the ontology Knowledge Graph presented in Chapter 4. This allows the representation to encode the knowledge modeled by the graph, with the result of an embedding that directly reflects music theory. In Section 5.5 we combine the embeddings presented in the previous sections using meta-embedding techniques. This results in a representation

that is jointly able to model different aspects of a chord while retaining a simple and flexible methodology. Throughout the whole chapter, we evaluate the embeddings using the methodology described in Section 5.1.

Implementation details All the models are trained using an *NVIDIA RTX 3090* on a set of ≈ 16000 chord progressions (with a total number of over $1M$ chord instances), taken from the Chord Corpus (ChoCo) dataset [36]. ChoCo is a chord dataset consisting of more than 20000 tracks taken from 18 different professionally curated datasets. All datasets have been parsed in JAMS [69] format and converted in Harte notation [64].

5.1 Evaluation method

Section 5.1 addresses the problem of evaluating the quality of chord representation methods, as proposed in the research question 2 of Section 1 - namely *How can we assess the quality of chord representation methods?*. The evaluation phase is a delicate and crucial step in the development of embeddings, in particular in those situations in which a low-resource language is being modeled. That is when few reliable resources are available, such as in the symbolic music domain. Embeddings are traditionally evaluated following two methodologies: intrinsic evaluation and extrinsic evaluation [4]. The most common approach is the extrinsic evaluation: in order to assess their generalization capability embeddings are evaluated in a pragmatic way, such as in classification problems or in general in downstream tasks. We perform such evaluation in Chapter 6. Throughout this whole chapter, we evaluate models based on intrinsic evaluation. In particular, we use the *Odd One Out* metric proposed in [139].

Simply put, the Odd One Out metric takes as input a set of chords C and a chord $\hat{c} \notin C$. A random number of chords k is sampled from C and the average of the embeddings μ_k is computed. If the similarity between the chord

$\hat{c}$ and μ_k is smaller than the similarity of each element of C with μ_k then the embedding is able to recognize that $\hat{c}$ does not belong to the initial set of chords C. Formally we say that

$$\text{OddOneOut}(C, c) = \mathbb{1}[c = \hat{c}]$$

with

$$\hat{c} = argmin_{x \in C \cup \{c\}} \frac{x \cdot \mu}{||x||||\mu||}$$

and

$$\mu_k = \frac{1}{k+1}(c + \sum_{i=1}^{k} c_i$$

, with $c_i \in C$. By relying on multiple sets C, we are able to approximate the accuracy of the embedding method with respect to a given classification.

The classification is performed using the Knowledge Graph described in Chapter 4. In particular, we consider 10 sets C by randomly sampling scales and the corresponding chord functions from the KG. We build two different evaluating sets, one which is used to assess the accuracy of the model with respect to the Tonal theory and one with respect to the Modal theory. We evaluate all the models using the Tonal theory test set and the best models are then tested on the Modal theory set as well. We set $k = 4$ and perform 1000 runs for each set C, as suggested in [139]. The categories are listed in Table 5.1.

We can define a simple baseline score that assigns a random similarity score to each couple of chords, which results in an Odd One Out accuracy acc of $\mathbb{E}[acc] = \frac{1}{5} = 0.2$.

5.2 Syntactically based embeddings

As already mentioned, the way a chord is labeled, that is the name that an expert decides to give to a chord, is sometimes as important as the chord itself.

Tonal theory set	Modal theory set
V Aeolian in C♯♯/D/E♭♭	VII Locrian in A♯♯/B/C♭
V Aeolian in E♯/F/G♭♭	III Dorian in A♯/B♭/C♭♭
VII Aeolian in D♯♯/E/F♭	III Dorian in D♯♯/E/F♭
III Ionian in A♭/G♯	III Locrian in A♭/G♯
IV Aeolian in B♯/C/D♭♭	III Phrygian in B♯/C/D♭♭
VI Aeolian in A♭/G♯	IV Aeolian in E♯/F/G♭♭
VI Aeolian in B♯/C/D♭♭	IV Phrygian in A♯/B♭/C♭♭
VI Ionian in A♯/B♭/C♭♭	VI Aeolian in A♭/G♯
I Aeolian in A/B♭♭/G♯♯	II Dorian in A♯/B♭/C♭♭
I Aeolian in B♯♯/C♯/D♭	I Phrygian in D♯♯/E/F♭

Table 5.1: Categories used to evaluate embeddings.

The name of a chord conveys the overall idea that the composer, or the annotator, has of the entire harmonic progression. One popular example is the so-called Tristan chord from Wagner's *Tristan Prelud*, which has been widely analyzed over the course of the last century [95] and is still an open discussion in the musicological field.

More generally, all chords have a set of enharmonic chords: identical chords in terms of note content that are annotated using a different label. This is closely related to the concepts of *hyponymy* and *hypernymy* in the linguistics field. A word w_1 is the hypernym of a word w_2 (respectively w_2 is a hyponymy of w_1) if the semantic relation between the two terms can be conceptualized as an *is-a* relation [54]. In the musical field, the set of notes that compose a chord can be considered as the hypernym of the different labels that can be assigned to that chord. It is the context in which the chord appears that the most appropriate label is determined, i.e. its hyponym in the linguistic analogy. In the current section, we do not take into consideration the context in which a chord appears - all chords that are labeled in the same way act as synonyms.

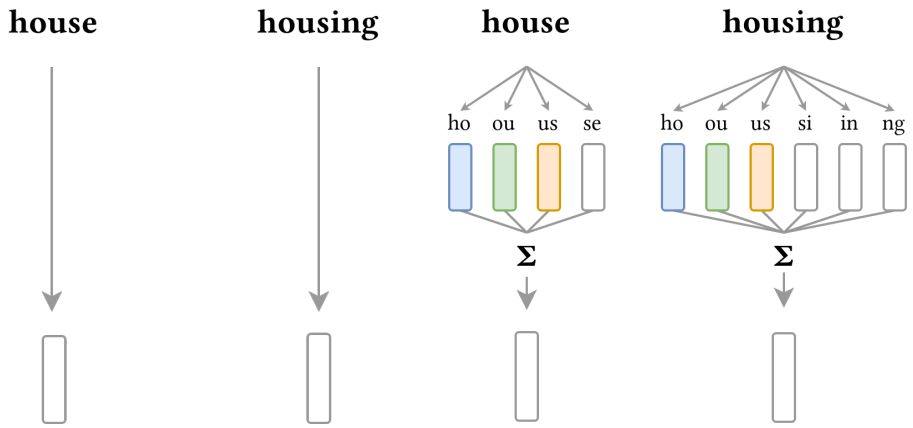
In [2] the authors show how using a method based on word2vec [104], embeddings based on the label of a chord are able to correctly encapsulate

musical knowledge. The encoding is used on two different tasks: chord clustering and log-likelihood estimation. The log-likelihood estimation task is used to investigate the historical harmonic style of different composers, which strongly correlates with current musicological knowledge. For instance, the model finds it difficult to predict chords from artists that make sporadic use of common harmonic progression. The chord clustering task, on the other hand, highlights how it's possible to observe similarities between functionally equivalent chords (chords that share notes with each other) and a well-defined difference between functionally different chords. Continuous word representations are hence adequate to encode chords in the first place, and more importantly, they are able to autonomously internalize relationships that have been previously observed by domain experts.

The main issue with such an approach, however, is that a comprehensive set of chords needs to be analyzed. The problem of out-of-vocabulary terms, i.e. terms that are not part of the training set of the embedding method, is a significant issue in the musical domain. Take for instance the case of modulation, which is the technique in which a harmonic progression changes its musical key. Modulation is generally performed through the use of the so-called pivot chords [9]. If such a chord were to be hidden from a musicologist, it would be harder to analyze the harmonic progression and the overall tune would be interpreted in a radically different way: "Modulation is the essential part of the art. Without it there is little music." [52].

For this very reason, we use fasttext [13] instead of using word2vec. When computing the representation of a word, none of its morphological components are taken into account. Let us take for instance two morphologically similar words, *house* and *housing*. Their computation does not share any common element and the final representation of the words is not influenced by their similarities. Fasttext was presented as a solution to this issue and has proven to be more effective in the representation of a word. The novel aspect is in the way representations are computed. At first, the n-grams that compose a word

are extracted. For each n-gram a continuous vector representation is computed, using the same methodology as word2vec. The representation of the original words is finally obtained as the sum of its n-gram components. Using this technique, the final representation of a word is conditioned by its morphological structure. When two words share one or more n-grams their vectors is the sum of at least one common element, which induce a similarity bias in both vectors. It is easy to see how that approach compensates for out-of-vocabulary terms: when using word2vec, out-of-vocabulary terms need to be represented as static vectors, randomly sampled from a normal distribution for example. Fasttext, instead, is able to compute the representation in a meaningful way, given that at least one of the n-grams in the out-of-vocabulary term is part of the training corpus. Figure 5.1 shows a visual comparison between word2vec (Figure 5.1a) and fasttext (Figure 5.1b).



(a) word2vec (b) fasttext

Figure 5.1: Word2vec (a) and fasttext (b) embedding methods. With word2vec each embedding is computed independently of its morphological structure. Fasttext instead compute the representation as the sum of the n-grams that compose a word. Words that share one or more n-grams have a similar representation as they are computed in a similar way. In the example, 2-grams are represented but in general n-grams up to the length of the term are commonly used.

5.2.1 Experiments

In this section, we describe the experiments performed using fasttext. As proven in [21], hyperparameters are an important aspect that needs to be taken into consideration when training embeddings. We tested a wide range of hyperparameters, summarized in Table 5.3. In Table 5.2 the best configuration found is reported.

Context window	Embedding dimension	Training epochs	Negative samples	Accuracy
2	100	5	5	0.477

Table 5.2: Fasttext best model

Parameter	Values
Context window	2, 3, 5, ∞
Negative samples	5, 20
Embedding dimension	50, 100, 150, 200
Training epochs	1, 5

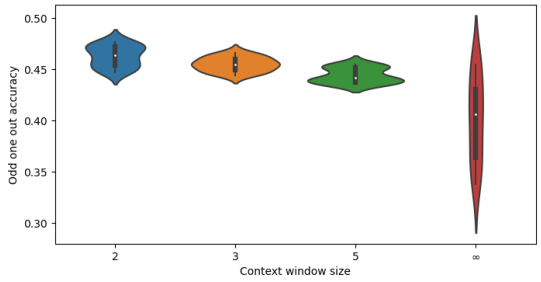
Table 5.3: Hyperparameters tested on fasttext model

Context window	**-0.718364**
Embedding dimension	**0.307366**
Training epochs	0.005743
Negative samples	-0.014698

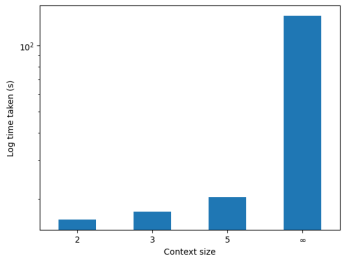
Table 5.4: Pearson correlation between the hyper-parameters of Table 5.3 and the accuracy measure.

In Table 5.4 the Pearson correlation between the hyper-parameters of Table 5.3 and the accuracy measure is described. The embedding dimension and the context window are the most important parameters. Surprisingly, the context window is negatively correlated with the accuracy measure as can also be seen in Figure 5.2a. The model is able to encapsulate the functional property of a chord by only relying on close neighbors. Indeed, each chord's function is mainly determined by its neighboring chords rather than distant chords [9].

Using bigger window sizes directly results in longer training time as well, as can be seen from Figure 5.2b. This aspect discourages the use of a high-value of such a parameter.



(a) Violin plot on the importance of context size (x axis) with respect to accuracy (y axis).



(b) Time taken in seconds (y axis, log scale) to train embeddings as function of the window size (x axis).

Figure 5.2: Impact of window size in training embeddings with fasttext

Using a bigger embedding dimension is mildly correlated with higher accuracy. The impact of such hyperparameter in the final accuracy is less pronounced, as can be seen from Figure 5.3.

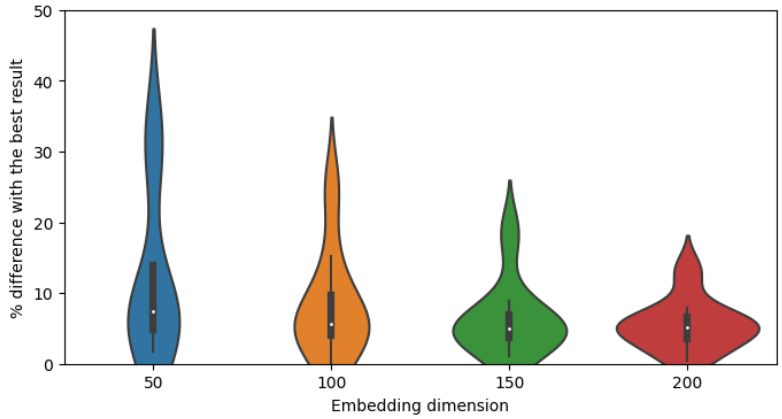


Figure 5.3: Violin plot on the importance of embedding dimension (x) with respect to accuracy (y). Accuracy is expressed as the percentage difference from the best result.

5.3 Chords are set of related notes

As shown in Section 5.2, encoding chords using their syntactical form is sufficient to obtain a model that is able to encapsulate important musical features. In Section 2.1 of Chapter 2, and in Section 5.2, we stated that chords are essentially an ordered list of notes.

This is a fundamental aspect and plays a major role in every musicological analysis of the harmonic structure of a piece. In Section 5.2 we compared the set of notes in a chord and its label to the concept of hypernym-hyponym relation in the linguistics domain. A similar approach that deals with the set of notes that compose a chord has been introduced in the mid-60s, with the name of *pitch-class set theory* [49]. Initially designed to model *atonal* music (music that does not make use of the traditional western music Tonal theory), it represents one important tool in musicological analysis and has been extensively used to analyze tonal compositions [50], derive geometric representations of chords [130] and similarity functions between chords [108]. To fully grasp such an approach a deep understanding of musical theory is required. By means of over-simplification, one can essentially consider the pitch-class set theory as a method that models similarity and relations between chords through the use of set theory. This is done by defining transformations between sets and providing an extensive classification of the most common sets that emerge in music. Such sets are composed of the common western notes, in the case of tonal music. More generally, one can consider an arbitrary number of notes, without the restriction of the traditional ET system (Section 2.1), and use numerical notation as elements of the set. For instance, take the chord *C:maj*, composed of the notes *C, E, G*. The pitch-class of such chord is defined as $pc(\textit{C:maj}) = \{\textit{C, E, G}\} = \{0, 4, 7\}$.

It is easy to see why such a method is better suited at modeling chords information when compared to using naming information, through the use of a very simple example: suppose that given two chords, *C:6* and *A:min7*, one

wants to analyse the similarity between the two chords. From a syntactical point of view, both chords are completely different. Their root note, C and A, are completely different as well as their quality, *6th* and *minor 7th*. If we take into account their pitch-class sets it is easier to draw a connection between the two: $pc(C{:}maj) = \{C, E, G, A\} = pc(A{:}min7)$.

5.3.1 chord2vec

In chord2vec [93] the notion of pitch-class is used to encode chords - each chord is represented by using the set of notes that it is composed of. Chord2vec is inspired by word2vec where chords are represented by the notes that they are composed of rather than their label. For instance, this approach allows the model to represent both *C:6* and *A:min7* as the same exact entity since they are composed of the same exact set of notes. We re-implemented such a method to test its accuracy on the evaluation methodology described in Section 5.1. A detailed description of the results is presented in Section 5.3.6.

Even though chord2vec has been shown to be able to accurately learn chord representations, we argue that its encoding method could be misleading when particular chords are represented. The characterization of a chord by using the set of notes it is composed of, i.e. its pitch-class set, is indeed a deeply criticized approach in the musicological field [124]. Among other things, one of the main problems with such an approach is that all the notes that are part of a chord are treated equally. If one does so, it is common to wrongly classify fundamentally different chords as similar chords. Take for instance the previous example, using *C:6* and *A:min7*. Even though both chords share the same set of notes, the order in which they are played is radically different.

In Figure 5.4 both chords are visually represented using musical notation. It is clear from such a representation how the order in which the notes appear changes the chord and consequently its final perceived sound. When using the approach described by chord2vec, this kind of characterization is not taken

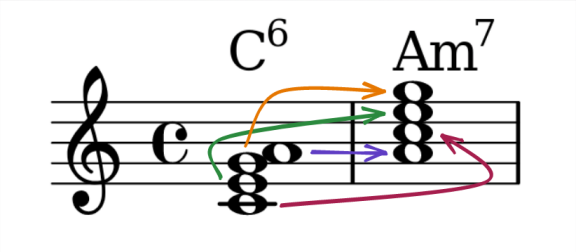


Figure 5.4: *C:6* and *A:min7* symbolic representation. Connections between the same notes are drawn (C in red, E in green, G in blue, A in yellow)

into account. In order to improve upon this issue, we propose a novel encoding method based on the intervals of a chord.

5.3.2 pitchclass2vec

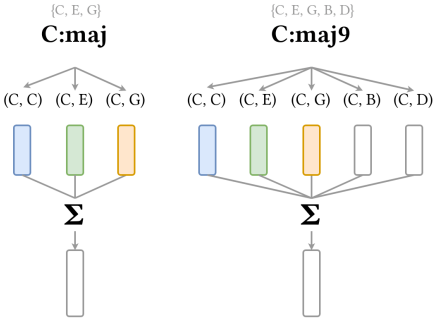
Rather than representing a chord by its pitch-class set, we use the ordered pitch-class set of a chord, which takes into account the order in which the notes are played in the chord. This approach has been explored before in the musicological domain [22] as a way to contextualize the pitch-class set theory with other relevant theories in the musical domain. We refer to this method from here on as *pitchclass2vec*.

Given a chord c composed of a set of notes $\mathcal{C} \subset \mathcal{N}$, with $\mathcal{N}$ the set of all notes and C the pitch-class of the chord, we encode c as the Cartesian product $\mathcal{I}_c = root_c \times \mathcal{C}_c$ between the root note of the chord $root_c$ and the pitch-class of the chord $\mathcal{C}_c$. The vectorial representation of the chord is obtained by using the same method as fasttext [13], given $\mathbf{u}_c$ the vector representation of the chord c, we have that

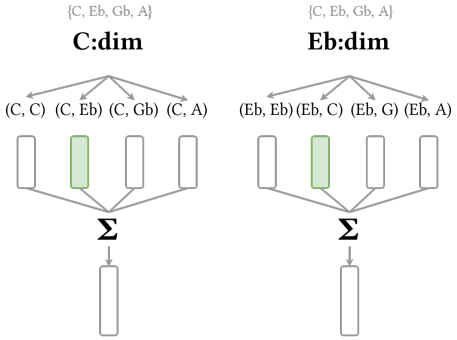
$$\mathbf{u}_c = \sum_{i \in \mathcal{I}_c} \mathbf{u}_i$$

where $\mathbf{u}_i$ is the vector representation of the tuple $x_i \in \mathcal{I}_c$. This formalization can be seen as an extension of the chord2vec [93] method, in which the chord's inner structure is taken into consideration as well.

In Figure 5.5 a visual reference on how pitchclass2vec handles the above-mentioned situation is presented, in particular in Figure 5.5a two chords that



(a) *C:maj* and *C:maj9* chord embeddings. The vectorial representation is computed from common elements and shares some aspects from both the source components.



(b) *C:dim* and *Eb:dim* chord embeddings. Both chords are composed of the same notes but use mostly different components.

Figure 5.5: Visual reference on how the pitchclass2vec embedding method handles chords with a similar set of notes (represented above the chord name).

share a similar set of intervals are computed from common components while in Figure 5.5b two chords that are composed of the same set of notes but a different set of intervals are computed starting from mostly different components.

5.3.3 intervals2vec

The encoding presented in Section 5.3.2 is theoretically sound from a musicological perspective. One might argue, however, that the restriction imposed in the modeling phase, namely the representation of a chord as the sum of the interval vectors between the root note and the other notes, might unnecessary bias the model. To further test this option we define an extension of pitchclass2vec, called intervals2vec, in which the vectorial representation of a chord is obtained through the combination of the intervals between the notes in the chord. More formally, a chord c is encoded as the Cartesian product $\mathcal{I}_c = \mathcal{C}_c \times \mathcal{C}_c$, where $\mathcal{C}_c$ is the pitch-class set of c.

In Figure 5.6 a visual comparison between the three methods is presented.

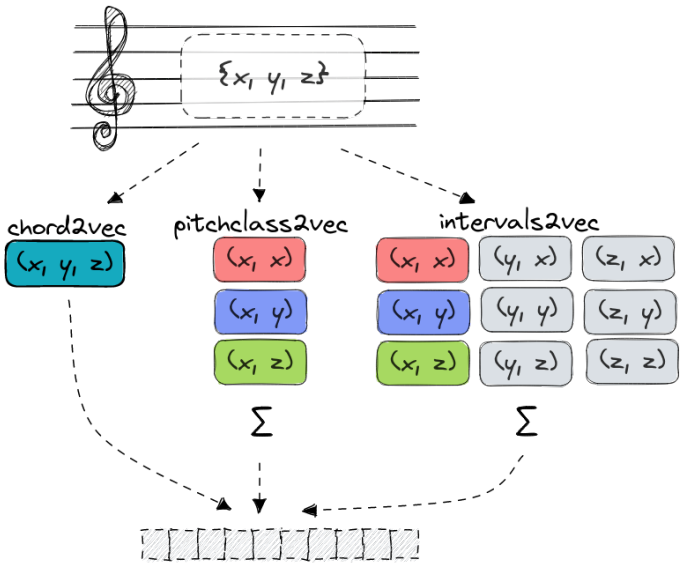


Figure 5.6: Visual comparison between the note-based embedding methods.

5.3.4 Duration-based loss scaling

An important aspect that we ignored in the previous modeling phase the duration of chords, which has been shown to be an important piece of information that, when taken into account, results in better MIR systems [122, 75, 123]. Chord embeddings can benefit from such information as well. It has been shown that harmonic cognition - the perception of hierarchical structures in a chord progression - is conditioned by the duration of a chord [79, 11]. Furthermore, given the fact that the learning process is based on the Distributional Hypothesis [132], taking into account the duration of a chord can help mitigate noise in a harmonic progression. For instance, it is common for composers to use *passage chords*, chords that are harmonically unrelated to their neighbors, as a way to add expressiveness to their composition [68, 150, 137]. It is reasonable to assume that such chords have a short duration, or else they would defy the definition of passage chord in the first place. In this section, we model the embedding training procedure to reduce the importance of such chords and treat them as *edge-cases*.

The dataset used to train the embeddings, ChoCo [36], provides the duration of each chord alongside the symbolic annotation. The encoding of the duration, however, is not consistent between data samples: music pieces whose annotation has been performed starting from an audio source express durations in seconds, while annotations extracted from music scores express durations in musical terms. Given this limitation, we use the raw duration, regardless of its source, as a measure of how important a sample should be weighted when optimizing the loss function of the model. In particular, we re-scale each duration to be in the range $[0, 1]$ composition-wise. Even though this is an approximation of the actual duration of a chord, it gives a consistent measure between different samples. The final loss function for a single sample is hence

$$loss = -d((y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})))$$

where d is the duration of the neighboring sample.

When sampling negative examples (see Section 2.4) we sample duration values from a Gaussian distribution whose mean and standard deviation is computed from the piece's durations. This approach can be considered as adding a random chord in a musical composition whose duration is similar to the duration of the other chords. Using a Gaussian distribution allows samples to be coherent with the actual distribution of the durations of each composition. See Figure 5.7 for a plot of the normalized distributions on the whole dataset.

5.3.5 Interval weighting

The underlying intuition in the proposed models is that, in order to accurately represent a chord, the relation between its constituent notes should be taken into account rather than isolated notes. In Figure 5.8 the distribution of intervals in the KG produced in Chapter 4 is shown.

The distribution of intervals follows a Zipf power-law [112]. Even though

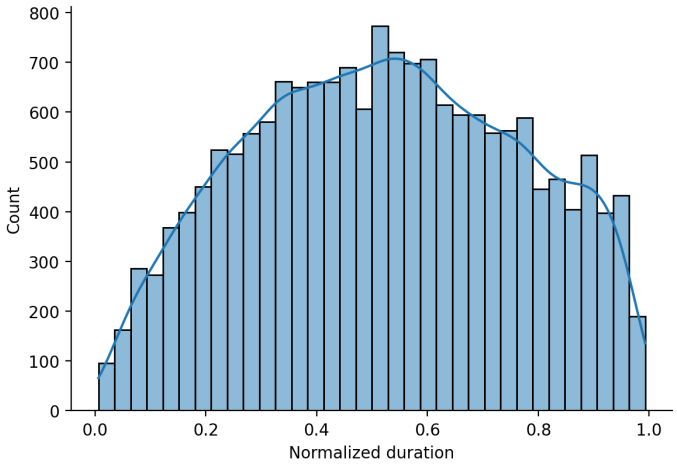


Figure 5.7: Distribution of the durations of each chord in the dataset. Each duration has been normalized to be in the range $[0, 1]$ composition-wise.

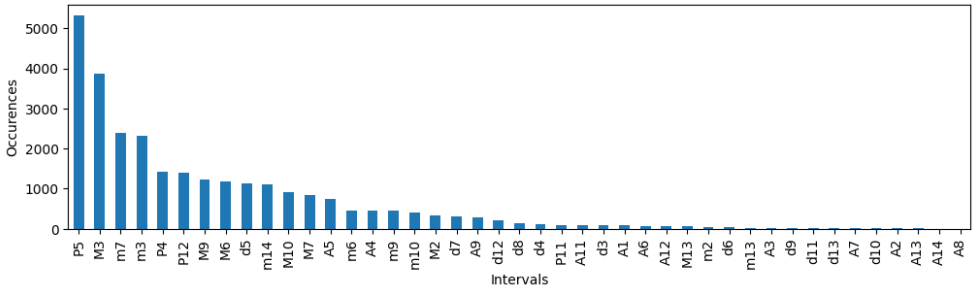


Figure 5.8: Distribution of the intervals in the KG

this does not directly imply that an interval should be more or less important than others, it is reasonable to assume that some intervals should be treated differently than others. Embedding techniques based on structural components have proven to be effective in other settings, such as modeling natural languages whose vocabulary is formed of idioms that can be composed to form new terms [149]. To better exploit the structure of a chord we extend our embedding method such that, instead of summing the representation of each interval, we perform a weighted sum. The weights of each interval are additional parameters that are going to be jointly learned with the embeddings.

We employ two different formalisations: global and contextual weighting. With global weighting, we assign to each interval a corresponding weight such

that intervals are considered coherently between different chords. With contextual weighting, we suppose that the importance of each interval depends on the presence (or absence) of other intervals as well. This follows from a simple intuition: take for instance the chords *C:maj*, *C:maj7* and *C:maj9*, whose notes are respectively $\{C, E, G\}$, $\{C, E, G, B\}$ and $\{C, E, G, B, D\}$. The interval (C, B) is crucial to be able to distinguish between a *C:maj* and a *C:maj9*, however, the same interval is less expressive when compared to a *C:maj7* or a *C:maj9*, where it is the (C, D) interval the most informative one. We obtain such a context-aware interval weighting schema by computing the weight of each interval through the use of a bidirectional LSTM layer, which outputs a value in the range $[0, 1]$.

5.3.6 Experiments

This section is divided as follows:

1. in the first part, a set of identical experiments to identify the most influential hyper-parameters is performed on each encoding method presented at the beginning of this chapter and summarised in Figure 5.6;

2. in the second part the best set of hyper-parameters are tested using the models described in Section 5.3.4 and Section 5.3.5.

All the experiments follow the same methodology used in Section 5.2.1.

Hyper-parameter tuning

We perform an extensive set of tests on each embedding method to evaluate the relevance of each hyper-parameter on the accuracy of the embeddings, similarly to what we do in Section 5.2.1.

In Table 1 5.5 the search space for each parameter is defined. We make use of a lower embedding dimension when compared to the experiment of Section 5.2.1 for a simple reason: the number of possible terms that need

Parameter	Values
Context window	2, 3, 5, ∞
Negative samples	5, 20
Embedding dimension	1, 2, 5, 10
Training epochs	1, 5
Combination method	sum, mean

Table 5.5: Hyperparameters tested on the chord2vec, pitchclass2vec and intervals2vec encoding methods

to be embedded when using a method based on chord notes is at most 2^{12} since a chord can be composed of at most 12 different notes. This is a much smaller vocabulary than the one of Section 5.2, which contains the n-grams of every chord in the dataset. We suppose that the amount of information that needs to be learned by the embeddings is smaller, hence the use of a smaller embedding dimension. Nonetheless, we test bigger embeddings, which result in more expressive models, in Section 5.3.6 using the best set of parameters identified in the current Section, to assess whether this hypothesis holds.

Parameter	chord2vec	pitchclass2vec	intervals2vec
Context window	3	∞	5
Negative samples	20	5	5
Embedding dimension	10	5	5
Epochs	5	5	1
Combination method		sum	sum
Accuracy	0.2197	**0.5386**	0.4355

Table 5.6: Best hyperparameters for chord2vec, pitchclass2vec and intervals2vec

In Table 5.6 the best hyper-parameters for each method are described. Pitchclass2vec outperforms both intervals2vec and chord2vec embeddings in the accuracy score. Using a subset of all the available intervals results in much more accurate embeddings when compared to the usage of all the notes or of all the intervals. We can confidently say that a method based on intervals, such as pitchclass2vec or intervals2vec, is far more suited than the chord2vec method when encoding musical chords. The same might not be true when

comparing pitchclass2vec and intervals2vec since, indeed, the amount of information they are encoding is different and hence the restriction imposed on the embedding dimension could be harmful. We better investigate this aspect in Section 5.3.6.

	chord2vec	intervals2vec	pitchclass2vec
Context window	0.029333	-0.008908	-0.048394
Negative samples	-0.010442	-0.007299	-0.005596
Embedding dimension	**0.985730**	**0.773704**	**0.732459**
Epochs	0.005528	-0.004935	-0.002536

Table 5.7: Pearson correlation between parameters of Table 5.5 and model accuracy

In Table 5.7 the Pearson correlation between the parameters described in Table 5.5 and the model accuracy are described. Similarly to what has been observed in Section 5.2.1, the embedding dimension ranks as the most influential parameter regardless of the analyzed model. Indeed, the correlation between accuracy and embedding dimension is ≥ 0.7 for all the models: increasing the embedding dimension does generally improve the accuracy and expressiveness of the model. Using an embedding dimension of 5, however, performs slightly better than 10 in both pitchclass2vec and intervals2vec, as can be seen in Figure 5.9.

Interestingly the context window is irrelevant when modeling chord sequences. In Figure 5.10 the distribution of the accuracy is compared between each context size, for both pitchclass2vec and intervals2vec. All the context windows within the same encoding method share a similar accuracy: the context window parameter have a low influence on the accuracy of the model. Besides, we can see from the same Figure how pitchclass2vec is, on average, more accurate than intervals2vec since the distribution of such a model is skewed to the right.

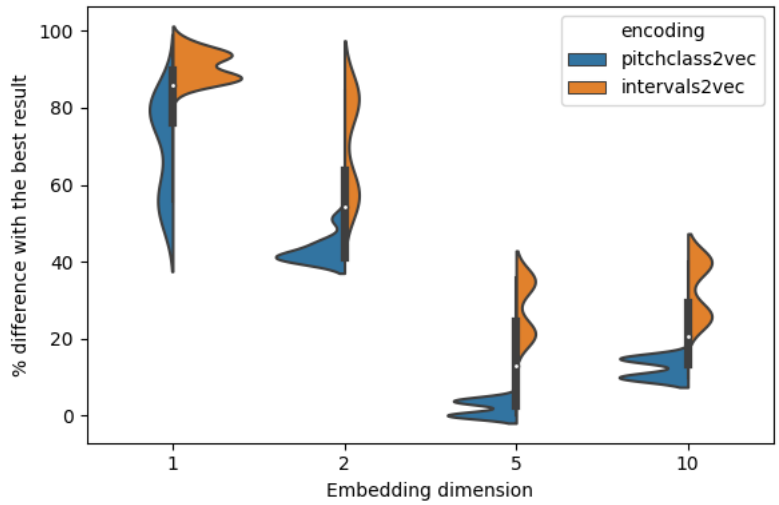


Figure 5.9: Violin plots on the importance of embedding dimension (x axis) with respect to accuracy (y axis). Accuracy is expressed as the percentage difference from the best result.

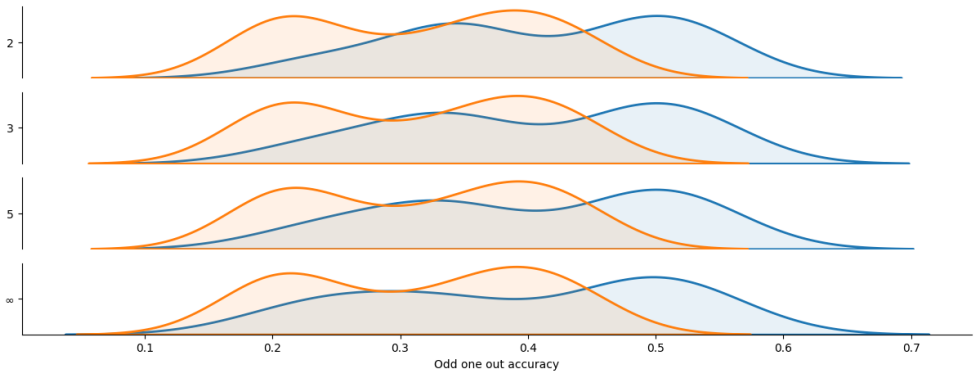


Figure 5.10: Relevance of the context window on the accuracy. Each context window is plotted separately, accuracy is on the y axis. Pitchclass2vec are the blue lines, intervals2vec the orange ones.

	intervals2vec	pitchclass2vec
Context window	0.092977	0.068643
Embedding dimension	**0.520061**	**0.606784**
Epochs	-0.003870	-0.001938

Table 5.8: Pearson correlation between larger embedding dimension and accuracy

Bigger embedding dimension

As seen in Table 5.7, the embedding dimension parameter is the one with the higher correlation with an accurate model. To better investigate this aspect we train both pitchclass2vec and intervals2vec using a larger embedding dimension: $[12, 50, 150, 200]$ dimensional vectors.

The Pearson correlation between accuracy and larger embedding dimension is reported in Table 5.8. Both models are positively correlated with the embedding dimension. A bigger embedding dimension does result in more accurate representations. On the other hand, a low correlation with the context window size confirms the fact that when chord embeddings are trained with a note-based encoding, a bigger context size can be avoided, which results in a faster training process as well.

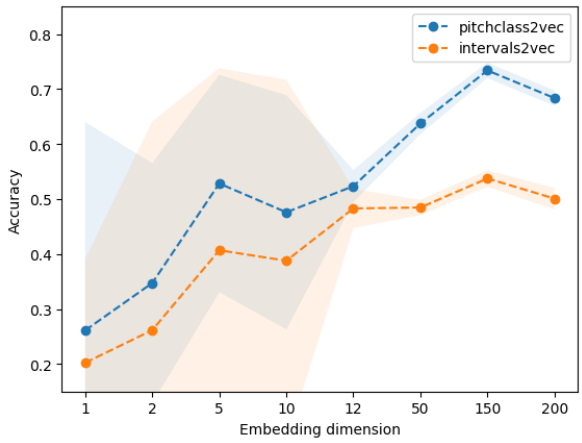


Figure 5.11: Accuracy as a function of the embedding dimension, compared between pitchclass2vec and intervals2vec.

In Figure 5.11 the best model for each embedding dimension is devised.

The accuracy is averaged over the different combinations of parameters that are tested and the 95% confidence interval is pictured as shaded color. We can see that, as long as a large embedding dimension is used, the context window dimension becomes less influential. Regardless of the choice of the parameters, pitchclass2vec always outperforms intervals2vec, with nearly 20% of difference using the best model.

Model	Context window	Dimension	Epochs	Negative samples	Accuracy
intervals2vec	∞	150.0	5	20.0	0.5374
pitchclass2vec	5.0	150.0	1	20.0	**0.7338**

Table 5.9: Best models obtained with larger embedding dimension.

In Table 5.9 the best results for each model are described. The most accurate model, pitchclass2vec with an embedding dimension of 150, only requires one training epoch and a small context window to obtain an accurate representation. Conversely, intervals2vec requires an infinite context window that, alongside less accurate embeddings, results in a less computationally efficient model during the training phase.

Scaled loss

Given the results of the previous Section, we only analyze if using chord durations does result in more accurate chord representation using the pitchclass2vec encoding method.

Training epochs	Accuracy
50	0.7324
20	0.7324
15	0.7324
1	0.7324
5	0.7323
Accuracy without loss scaling	**0.7333**

Table 5.10: Result of using a scaled loss with pitchclass2vec encoding method.

In Table 5.10 the results of scaling the loss function using chord durations do not seem to have any beneficial effect on getting a more accurate model. Even though the lack of coherent information in the used dataset does not allow a correct usage of chord durations, we argue that the consideration of Section 5.3.4 are still valid and would result in a more accurate embedding method once chord durations will be available in a coherent form.

Weighting intervals

Model	Training epochs	Hidden size	Accuracy
pitchclass2vec baseline			0.7333
pitchclass2vec fixed	50		**0.7538**
pitchclass2vec contextual	5	5	0.7424
intervals2vec baseline			0.5374
intervals2vec fixed	1		0.5372
intervals2vec contextual	5	50	**0.5393**

Table 5.11: Results obtained by weighting intervals.

In Table 5.11 the best results of the embedding using intervals weighting are reported. We test both the pitchclass2vec and intervals2vec embedding methods. Even though intervals2vec proved to be a sub-optimal representation when compared to pitchclass2vec, we argue that this difference might be leveled by using a specific interval weighting schema. Indeed, the intervals2vec method can be seen as a generalization of the pitchclass2vec method: we obtain a method equivalent to pitchclass2vec by assigning a weight of 0 to each interval whose first element is not the root note.

Learning the weighting schema allows the exploration of an interval combination that can potentially outperform pitchclass2vec.

The results in Table 5.11, however, denies that possibility. Regardless of the weighting schema, intervals2vec is not able to fill the gap with pitchclass2vec. On the other hand, an additional weighting schema on pitchclass2vec results in a more accurate model.

Surprisingly, the fixed approach outperforms the contextual approach. We will analyze such learned weights in future works, to check for correspondences with musicological knowledge, such as if the learned weights correlates with consonance and dissonance defined in other works [58].

5.4 Music Knowledge-based embeddings

This Section describes an orthogonal method compared to the ones of Section 5.2 and Section 5.3. The common ground between both approaches is based on the Distributional Hypothesis [132] and is the theoretical basis for what has been a complete revolution in the field of natural language processing [30].

Such an approach, however, has been shown to only capture a limited semantics of words [131]. Knowledge-based approaches tackle the problem of defining the representation of a term from a different point of view. The focus is on the identification of an intensional description that characterizes a term. Knowledge bases like WordNet [105] or FrameNet [5] have been proposed as models of natural language based on linguistic knowledge. Given the continuous evolution in the usage of the natural language, it is difficult to settle on a stable representation. Furthermore, deriving such knowledge bases requires a deep understanding of the linguistic framework and of the language itself.

We argue that in the music field, this is an easier task to accomplish. Take for instance the knowledge graph proposed in Section 4.3. The accompanying ontology provides a complete formalization of an important music theory, that has been critiqued, tested, and refined by expert musicologists over the span of decades, if not centuries. While music has certainly evolved since the baroque era, most of its elements are still the same as nowadays. In western music, the set of notes used to compose music are the same as 500 years ago throughout all of Europe and in general throughout all those areas that have been influenced by European culture [9].

A completely knowledge-based approach, however, is still limited by the

set of technologies available. For instance, building a more complex and constrained ontology would probably result in the use of axioms that requires a complex reasoning process, which is not solvable in an efficient way [109].

This section describes the computation of embeddings based on the knowledge graph described in Section 4.3 with the use of rdf2vec [128]. Rdf2vec is a method inspired by word2vec, which aims at the translation of a knowledge graph in a vectorial representation.

This is done by extracting walks between entities in the knowledge graph. A walk is a sequence of objects and predicates that connects two entities in a knowledge graph. The elements of this sequence are then converted into alphanumerical tokens, using their IRI (see Section 2.3) or a user-defined representation, and are finally embedded using word2vec.

This method can be seen as the embedding of a verbalized version of the knowledge graph, in which the sentences are composed of the verbalization of the nodes in the knowledge graph. Rdf2vec, when compared to similar works that obtain embeddings of KG using geometrical models [27], has proved to be able to better encode the concept of similarity and relatedness [129].

5.4.1 Experiments

This Section describes the experiments performed using rdf2vec. We train the embeddings on the whole knowledge graph implemented in Section 4.3 and evaluate each experiment using the methodology described in Section 5.1.

Many different walk extraction strategies can be implemented using rdf2vec. The extraction of walks from a graph is a complex task that can take up to days in complex settings. This work relies only on random walks since they have been shown to consistently obtain good representations [129] across different domains and applications.

In Table 5.12 the hyper-parameters tested are described. Training epochs refer to the number of epochs used to train the word2vec embeddings. Max

Parameter	Values
Training epochs	1, 10
Max walking depth	4, 10
Max number of walks	2, 5, 10, 50, 100

Table 5.12: Hyperparameters tested on the chord2vec, pitchclass2vec and intervals2vec encoding methods

walking depth is the maximum number of nodes that are spanned by a single walk while the max number of walks is the number of random walks extracted from the knowledge graph for each entity that is going to be classified.

Parameter	Pearson correlation
Max walking depth	0.058318
Training epochs	0.317552
Max number of walks	0.695166

Table 5.13: Pearson correlation between parameters of Table 5.12 and model accuracy

In Table 5.13 the Pearson correlation between the hyperparameters of Table 5.12 and the accuracy score is described. Surprisingly, the walking depth does not seem to influence the result as much as the other parameters.

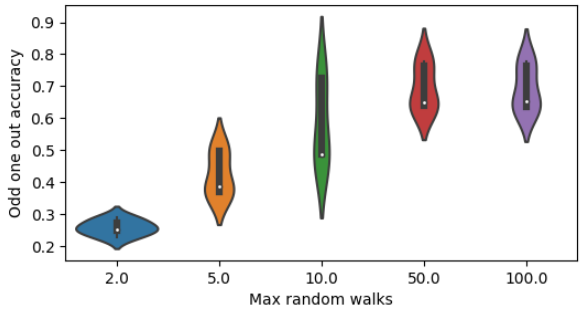


Figure 5.12: Violin plots on the importance of max random walks (x axis) with respect to accuracy (y axis).

The most important parameter, as can also be seen in Figure 5.12, is the number of random walks extracted. Unsurprisingly, the difference between 50 random walks and 100 random walks is marginal. This is a direct consequence

of the structure of the knowledge graph of Chapter 4. Based on the theoretical considerations of Section 4.1, a chord is part of 84 modes in the worst possible case, since there are at most 12 notes and 7 modes.

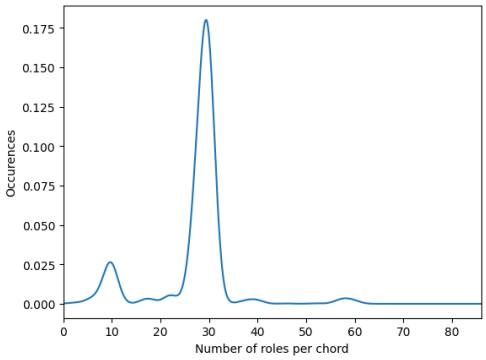


Figure 5.13: Distribution of number of roles per chord.

This number is however much lower in practice. In Figure 5.13 the number of roles for each chord is plotted. The shape of the distributions is a mixture of two Gaussian distributions: most of the chords have indeed much less than 84 roles. If we take into consideration the distribution of chords in the dataset as well, it is even more evident that the number of roles for each chord is actually lower than 10 on average.

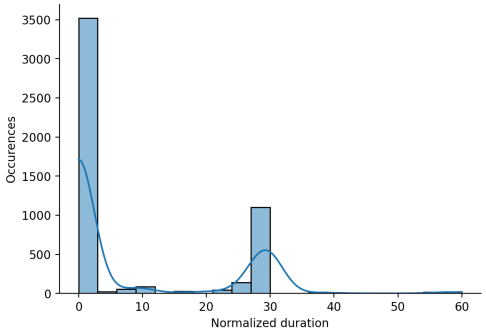


Figure 5.14: Distribution of number of roles per chord weighted by the chord distribution in the dataset.

In Table 5.14 the best set of hyperparameters are described. The evaluation methodology needs to be taken into account when the results of Table 5.14 are analyzed. It is clear that the evaluation measure proposed in Section 5.1 might turn in favor of an embedding method that is tightly based on the same KG

Epochs	Accuracy	Max walking depth	Max number of walks
10	0.778	10.0	50.0

Table 5.14: Best hyperparameters for the rdf2vec embedding model

used to generate the test set in the first place. In Figure 5.12, however, we can see that when using an inappropriate hyperparameter setting, the accuracy of the model drops drastically.

This is a clear sign that the accuracy of this embedding method is not to be reconducted to a favorable evaluation setting, but rather to an encoding method that is able to encapsulate an intensional representation of chords that is not possible under the Distributional Hypothesis.

5.5 Meta-embedding: The Theoretic, the Syntactic and the Semantic

Model	Accuracy	
	Tonal Harmony	Modal Harmony
chord2vec	0.2197	0.2517
fasttext	0.4770	0.5839
intervals2vec + RNN weight	0.5393	0.5166
pitchclass2vec + static weight	0.7538	0.6639
rdf2vec	**0.7780**	**0.8145**

Table 5.15: Overview of the best trained models.

In Table 5.15 an overview of the best embedding methods is presented from worst to best accuracy on both the Tonal Harmony evaluation test and the Modal Harmony evaluation test as well.

All the proposed models outperform the related work proposed in literature, chord2vec. In particular, the best embedding method is obtained by using the rdf2vec model, which outperforms significantly all the other embedding methods in both evaluation tasks.

Interestingly, the fasttext model performs much better on the Modal Harmony task than on the Tonal Harmony task. The opposite happens with pitch-class2vec, which is less accurate on the Modal Harmony task. As already described in previous sections, each embedding method is radically different from the others and there might be some particular pieces of information that are better represented by one model with respect to the others.

In order to exploit the best aspect of each embedding method, in this section we combine the most accurate embedding methods, using the meta-embedding [14] technique.

A meta-embedding is the combination of different pre-trained embedding methods to obtain more accurate embeddings. In principle, a meta-embedding is able to exploit the knowledge encoded in embeddings trained using different techniques or different data sources.

Many different meta-embedding techniques have been proposed, from the simple concatenation of the vectors to the usage of autoencoding methods. It is still unclear which of these techniques achieves better results and under which conditions [14]. This is partially because the intrisic evaluation of embeddings is often a difficult task to define and formalize correctly and only extrinsic evaluation is performed [4]. In such cases, supervised methods such as autoencoding models generally obtain better performances [14].

We only concatenate and average the embeddings, since they have been proven to be effective methods despite their simplicity [14]. Note that when averaging embeddings the dimensionality of the starting vectors might be different. We pad all vectors to the same size, using 0 as padding value, similarly to what has been done in [28].

In Table 5.16 the results of meta-embeddings are described. Meta-embeddings result in much more accurate representations for nearly all the combinations when compared to their respective components of Table 5.15. There is not a substantial difference between concatenating and averaging embeddings.

Even though concatenation performs slightly better, averaging is still an approach that is worth taking into consideration since the resulting embeddings have a lower memory requirement given their lower dimensionality.

Combining fasttext, pitchclass2vec and rdf2vec outperforms all the other approaches. As predicted at the beginning of the current section, each embedding is able to complement the areas where other approaches are less accurate.

In Table 5.17 the Pearson correlation between the components of a meta-embedding and the accuracy score is described. Given the result of Table 5.15 and Table 5.16 it is easy to see how the method based on rdf2vec out-performs all the other methods and has the biggest impact on the accuracy of a meta-embedding. This is an important finding of this work and is further addressed in Chapter 7.

fasttext	intervals2vec	pitchclass2vec	rdf2vec	Accuracy Modal	Accuracy Tonal
Concatenation					
•	•			0.5367	0.4365
•		•		0.7111	0.5016
•			•	0.8170	0.7798
	•	•		0.5974	0.47
	•		•	0.8224	0.7527
		•	•	0.8629	0.7929
•	•	•		0.6113	0.4795
•	•		•	0.8267	0.7564
•		•	•	**0.8648**	**0.7931**
	•	•	•	0.8431	0.7555
•	•	•	•	0.8473	0.7577
Average					
•	•			0.5367	0.4365
•		•		0.6946	0.5122
•			•	0.8170	0.7798
	•	•		0.5916	0.4746
	•		•	0.8224	0.7527
		•	•	0.8604	0.794
•	•	•		0.6049	0.4832
•	•		•	0.8267	0.7564
•		•	•	**0.8623**	**0.7941**
	•	•	•	0.8425	0.7568
•	•	•	•	0.8469	0.7591

Table 5.16: Accuracy results on Modal Harmony and Tonal Harmony evaluation settings by combining different types of embeddings. The best results are represented in bold.

Evaluation	fasttext	intervals2vec	pitchclass2vec	rdf2vec
	Concatenation			
Modal	-0.150989	-0.362462	0.049097	**0.937772**
Tonal	-0.163884	-0.289746	-0.104155	**0.990836**
	Averaging			
Modal	-0.154799	-0.343054	0.028164	**0.950195**
Tonal	-0.163676	-0.298559	-0.093986	**0.988918**

Table 5.17: Pearson correlation with accuracy between the components of a meta-embedding.

Chapter 6

One embedding to segment them all

This section presents an additional set of experiments to evaluate the embedding methods proposed in Chapter 5 on an extrinsic evaluation setting. The evaluation is performed on the music structure segmentation task, described in Section 6.1, using the embedding methods that achieved the most accurate results on the evaluation described in Section 5.1.

6.1 Music structure segmentation

Music structure is one of the tools used by composers to tell a story.

> Music-making is, to a large degree, the manipulation of structural elements through the use of repetition and change.
>
> Gary Burns [17]

The repetition of harmonic progressions (sequences of chords), in particular in the context of western tonal music, gives to artists the ability to guide listeners through a journey that creates dramatic narratives, conveying a sense of conflict that demands a solution [143].

Listeners, regardless of their level of musical knowledge and harmonic

sensitivity [142] or their cultural origins [138], use intuitive knowledge to organize their perception of musical structures [29]. Indeed, there is empirical evidence that neural activity correlates with musical structure in listeners' perception [77]. Moreover, the structuring and predictability of musical compositions is also recognised as an important factor in therapies that involves the treatment and assessment of children and adolescents with autistic spectrum disorder [153].

Figure 6.1: Structure of Helter Skelter by The Beatles. Chords are presented in Harte format [64].

Figure 6.1 shows the structure of *Helter Skelter* by The Beatleas alongside its chords. By means of the alternation between *verse* and *refrain* the artist establishes a common repetitive pattern. The addition of an instrumental section after the second *refrain* and the repetition of the *intro* reinforces the repetitive aspect of the composition. The upcoming *outro* section denies the expectation of a new *verse*, right before the song ending section. Expectation and the way it is fulfilled or denied is an essential part in musical enjoyment [143]. In fact, it has been shown empirically that the emotional response to a musical composition varies as the degree of repetition changes [90]. Understanding musical structures is hence fundamental in music analysis and composition. Artists can benefit from the feedback provided by a system able to highlight possible hierarchical structures in their compositions.

6.1.1 The form of a musical composition

Music structure segmentation is a broad term related to the study of musical form, which describes how musical pieces are structured. In particular it can be divided in two main categories: phrase-structure segmentation and global segmentation [59]. Phrase-structure segmentation consists in detecting

sections from the melodic information of a piece. While the aim of phrase-structure is not to obtain a global segmentation, the detection of sections provides valuable insights in the task of global segmentation.

From here when we refer to global music structure segmentation as music structure segmentation. Music structure segmentation consists in identifying and labelling key music segments (e.g. *chorus*, *verse*, *bridge*) of a music piece [96]. Given a musical composition, its musical segmentation is the identification of non-overlapping segments, to which we refer to as sections. Each section is characterized by a label that classifies its function such as *intro* or *verse* in figure 6.1. A correct segmentation does not necessarily assign the correct labels to each section of the composition, but rather focuses on the correct estimation of the boundaries of each section. Once boundaries has been accurately predicted, an additional labeling process can performed to obtain the final annotation [114].

6.1.2 Related works

Most of the recent methods and research approaches are based on audio analysis techniques [114]. Automatic segmentation on audio signal is a prolific research field in which many different solutions have been presented, ranging from self-similarity matrices [152, 151] to neural network based methods [134, 148, 94]. Harmonic content has been used to improve those methods both using probabilistic models [121] and transformer based models [24]. Significant research has been performed on phrase-level structural segmentation based on melodic [53, 18, 145] as well as polyphonic content [101].

We address the task of music structure segmentation by only relying on symbolical notation. Even though this might seem as an inconvenient choice, the assumption behind this approach is that the identification of harmonic subsequences (harmonic patterns) can be influential in defining the structure of a song and the sections of which it is composed. By taking a closer look at

Figure 6.1 it is easy to notice how harmonic information can provide valuable information in the structure segmentation task: all *verse*s are roughly based on the same harmonic progression (E, G, A, E) while *refrain*s are based on a different harmonic progression (A, E, A, E, E). A segmentation strongly based on those recurrent patterns is likely to be coherent with the way the composer shaped the progression in the first place.

To the best of our knowledge, the only approach proposed in literature for global music segmentation on symbolic harmonic content is FORM [39].

FORM performs structural segmentation on harmonic structures encoded as sequences of strings, where each string represent a chord, by exploiting repeated patterns. This is performed through the use of suffix-trees [61]. Suffix-trees are data structures designed to search efficiently for patterns in a string. Trees are built in a way such that when the path that connects a root node to a leaf is a unique sub-sequence of a string. It is easy to see why such data structures are best suited to extract repeated patterns: whenever an intermediary node is found on the path that connect the root node to a leaf node, then the partial path up to the intermediary note is a sub-string common to at least two sub-sequences. From this observation a number of definition are formalized in [61], one of which is the concept of *left-diverse* node. A *left-diverse* node is an intermediary and, as said before, it represents a sub-sequence that appears at least twice in the whole sequence. In particular a node is *left-diverse* is by extending the sub-sequences on the left, that is adding characters at the start of the sub-sequence, then it would lose its status of repeated sub-sequence. For instance if we take the string *XabcYabc*, the sub-sequence *abc* is represented in the tree by a *left-diverse* node. If we were to update the string to *XabcXabc* then the sub-string *abc* loses its status of *left-diverse* node, since by extending it to the left we have another repeated sub-sequence, *Xabc*. In FORM a partial segmentation is obtained by labeling all the occurences of a sub-sequence represented by a *left-diverse* node with the same label. A final segmentation is then obtained by labeling the remaining sub-sequences as their preceding

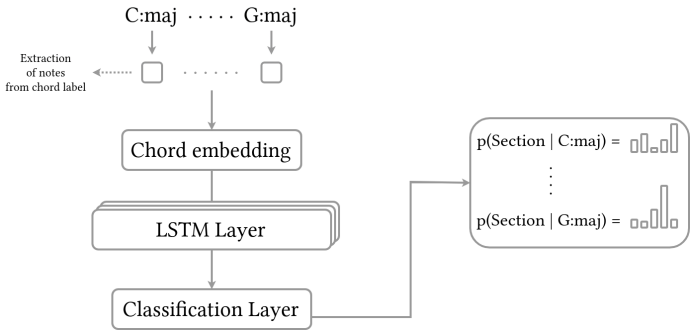


Figure 6.2: Visual depiction of the implemented LSTM model.

neighbouring section.

The key aspect, and main issue, with FORM is in way chord labels are compared. The string representation does not take into account semantic similarity between chords nor the algorithm is able to detect near-similar patterns, i.e. patterns whose difference can be ignored in the context of music structure segmentation.

To mitigate this aspect, the original work transforms all chord labels in two at most 24 classes of chords, 12 major chords and 12 minor chords. Every other chord feature is removed. The results are then compared with a random baseline that generates arbitrarily long structures and to a heuristic that assigns to each composition the typical pop song structure *ABBBBCCBBBBCCDCCE* [39], in which each different label represent a structure in the chord progression and is stretched to fit the whole sequence. We re-implemented FORM in order to compare the results of the proposed method with the current state of the art.

6.1.3 Proposed model

In order to test the effectiveness of the embeddings presented in Chapter 5 in the Structure segmentation task, we developed a baseline model using a stacked LSTM-based neural network, depicted in figure 6.2. The model represents an end-to-end solution: the probability that each chord belongs to a

particular Section is computed, and can hence be seen as the combination of a structure detection algorithm and a relabeling algorithm [114] jointly combined.

We train our model on the Billboard dataset [16] provided by Mirdata [1] library [12]. The dataset is composed of 889 expert annotated tracks, each composed of a sequence of chords in Harte format [64], and a sequence of structure labels. Labels are provided in a similar format to the one presented by SALAMI [136]. 80 unique section labels are present in the whole dataset. We preprocess each label and reduce the number of unique labels to 11 by combining all those labels that fall under the same definition given by [136]. A complete reference of the label conversion step is given in table 6.1.

Source labels	Converted label
`[verse]`	*verse*
`[prechorus, pre chorus]`	*prechorus*
`[chorus]`	*chorus*
`[fadein, fade in, intro]`	*intro*
`[outro, coda, fadeout, fade-out, ending]`	*outro*
`[applause, bass, choir, clarinet, drums, flute, harmonica, harpsichord, instrumental, instrumental break, noise, oboe, organ, piano, rap, saxophone, solo, spoken, strings, synth, synthesizer, talking, trumpet, vocal, voice, guitar, saxophone, trumpet]`	*instrumental*
`[main theme, theme, secondary theme]`	*theme*
`[transition, tran]`	*transition*
`[modulation, key change]`	*other*

Table 6.1: Label conversion reference. Each label is stripped out of numbers and symbols before the conversion.

The model of Figure 6.2 optimizes a Binary Cross Entropy loss function in which the target labels are the ones in Table 6.1. We use the same set of hyperparameters on each experiment: 5 LSTM layers with an hidden dimension of 256 and a dropout probability of 0.2 to regularize the training phase

[1] `https://github.com/mir-dataset-loaders/mirdata`

and prevent overfitting on the data.

6.1.4 Experiments

6.1.4 Section describes the results of the experiments performed with the model described in Section 6.1.3.

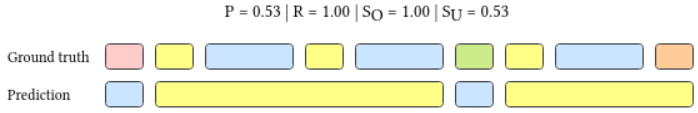
Note that the dataset used to train the model is a subset of ChoCo [36]. This might lead to label leaking [78]. Label leaking is a specific issue related to machine learning models [78]. It arises when the data on which the classifier is trained encodes some subtle information from the testing data as well. Those subtle information are then exploited by the model to obtain accurate result on the test set as well, without any generalization towards new testing data.

In our setting, by training embeddings on the Billboard dataset as well, some pieces of information on the section label might be indirectly learned by the the embeddings. In order to such issue we train the embeddings from scratch on a subset of ChoCo where the whole Billboard dataset has been removed.

Model	P	R	$F1$	S_U	S_O	S_{F1}
FORM_{raw}	**0.673**	0.337	0.42	0.673	0.337	0.42
$\text{FORM}_{\text{simple}}$	0.663	0.34	0.423	0.663	0.34	0.423
fasttext	0.616	0.604	0.596	**1**	**1**	**1**
pitchclass2vec	0.617	0.591	0.586	**1**	**1**	**1**
rdf2vec	0.619	0.584	0.581	0.962	0.926	0.944
meta-embedding	0.624	**0.608**	**0.598**	**1**	**1**	**1**

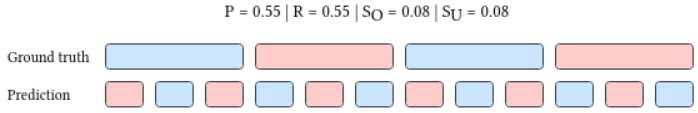
Table 6.2: Results from the segmentation algorithm

The results of the experiments are summarised in Table 6.2. We evaluate the segmentation results by computing pairwise precision, recall and F1-score (P, R and $F1$ in Table 6.2) [85] along with under-segmentation, over-segmentation and normalized cross entropy F1 (S_U, S_O and S_{F1} in table 6.2) [92]. Every metric is computed using the standard MIR evaluation library mir_eval [125].



(a) High over-segmentation example. $R = 1$ since if we take each chord in the sequence pairwise then each chord that should be in the same section is indeed in the same section. $S_O = 1$ since the accuracy of the prediction can be easily explained by the over-segmentation phenomena. Conversely, $P = 0.53$ and $S_U = 0.53$ clearly show how the prediction is not able to capture all the needed segments but rather merges ground truth segments together.

(b) High under-segmentation example. The exact opposite of Figure (a) is displayed. The prediction is not able to capture segments and rather place each chord on its own segment.



(c) P, R and S_O, S_U compared. In this edge case the main difference between the two measures is highlighted. While P and R suggests a decent segmentation S_O and S_U clearly states a completely wrong segmentation. Pairwise metrics can be misguiding in absence of S_O and S_U.

Figure 6.3: Examples of metric computation on relevant instances.

Under-segmentation and *over-segmentation* are two metrics specifically designed for the evaluation of automatic music segmentation methods. When a method has an high over-segmentation measure, the final prediction accuracy is influenced mostly by false fragmentation. Conversely an high under-segmentation measure means that the prediction's segments are the result of ground-truth segments being merged together [92].

Pairwise metrics are computed as the usual precision, recall and F1 scores on the set of identically labeled pairs in the sequence. Precision and recall can be interpreted as the accuracy influenced respectively by under-segmentation and over-segmentation. On the other hand under and over-segmentation scores are computed by taking into account the normalized conditional entropy of the segmentation.

In short, S_O gives a measure about the information is missing in the predicted segmentation, given the ground truth segmentation, while S_U gives a measure of how much noisy information are the result of the predicted segmentation [92]. A graphical explanation of these concepts is provided in Figure 6.3 (all the examples are taken from [92]).

We evaluate our models based on the $F1$ and S_{F1} scores of Table 6.2 since both metrics gives a balanced measure of over and under segmentation.

$\text{FORM}_{\text{simple}}$ detects repetitive patterns from simplified chord labels, as shown in [39]. The chord simplification process extracts the *root* note from the chord and classifies it either as *major* or *minor*. FORM_{raw} uses the same labels used by the embedding method described in Chapter 5. There is not a significant difference between the two encoding methods. This can be reconducted to the way FORM searches for repeated patterns: while it is true that many compositions fundamentally share the same exact harmonic progression in similar sections, it might happen that small subtle differences in the harmonic choices of the artist results in slightly different harmonic progressions. See for instance the example of Figure 6.1. The first two verses have a very similar harmonic progression, but they appear as different patterns if the sequence is analyzed using a string-based method.

All the models based on embeddings in Table 6.2 outperforms FORM. In particular, we can see that the methods based on the Distributional Hypothesis, fasttext and pitchclass2vec, outperforms the rdf2vec method. This is not completely unexpected due to the fact that embeddings that make use of the distributional aspect of the dataset condition their representations based on the actual usage of the chords. Such bias proves to be essential in the task of structure segmentation. Nonetheless embeddings that are based on the constituting elements of a chord, such as the rdf2vec method, obtain very competitive results when compared to the other methods.

Surprisingly, the fasttext method obtains performances that in some metrics perform better than pitchclass2vec. As already stated in Section 5.2, the

way in which chords are written carries its own semantic meaning. When using high quality datasets, such as the one provided by ChoCo, the time spent on obtaining accurate notations pays off in terms of generalization capabilities of an embedding method. The best method, as when evaluating based on the methodology of Section 5.1, results to be meta-embedding method, which outperforms the other methods in all the measured metrics. This complements the partial answer, addressed in Section 5.1, to the research question 2 presented in Chapter 1 - namely *How can we assess the quality of chord representation methods?*. The proposed intrinsic evaluation method results are in line with the extrinsic evaluation presented in the current Chapter.

Comparison between the results in Table 6.2 and state-of-the-art methods in audio-based structure segmentation is difficult, since the dimension of the available datasets is considerably different. Audio-based systems are usually trained and tested on the popular SALAMI dataset [136], which contains twice the amount of data when compared to the dataset we used, Billboard [16]. In [114] a review of audio-based music segmentation algorithm is performed. State-of-the-art results are obtained by approaches based on Convolutional Neural Networks, with a pairwise F1 scores of 58.09 ± 15.77 which is a similar result to the one obtained on Table 6.2. This provides a positive answer to the research question 4 presented in Chapter 1 - namely *Is it possible to use chord representations to obtain results comparable to audio-based representations?*.

Chapter 7

Discussion

Chapter 7 presents a discussion on the results obtained, the limitation of the presented methods, and future works.

7.1 A semantic approach is enough

The Modal Harmony Ontology and the Knowledge Graph presented in Chapter 4 represents a first step towards the definition of a set of ontologies, and in general of Semantic Web technologies, that allow the formalisation of musical theories in an efficient, sound and complete form.

A general design pattern can be extracted from the methodology presented in Section 4.2 to model many other different musical theories. Other research activities have moved in that direction but are limited by the lack of a structured framework that can be used as a backbone [1]. Design and implementation efforts need to gravitate around the composability and interconnectivity. Such foundational aspects are fundamental in order to obtain tools that can be used by any music expert and specialized to the different tasks that populate the music theory landscape.

Valuable musicological knowledge is difficult to shape in a form that can

[1] See the Modal Tonal ontology `https://github.com/polifonia-project/modal-tonal-ontology` and the Tonalities pilot `https://github.com/polifonia-project/tonalities_pilot` developed as part of the Polifonia project.

be easily modeled and understood by computers and algorithms. A future step that needs to be taken into account is the creation of a framework, in the form of a formal top-level ontology, that allows researchers to focus on content of the theory they are modeling, rather than the knowledge engineering effort required to efficiently use Semantic Web techniques. Such framework can be built upon solid foundational ontologies, which have shown to be a successful way of structuring knowledge [116]. It is the case of DOLCE [56] that allows to obtain a semantically richer interpretation of WordNet [105] by modeling human common sense, or SUMO [115], which promotes data interoperability and information search and retrieval through the formalisation of set of high-level concepts.

A set of established guidelines to develop such system can be drawn from the pragmatic success obtained by those approaches, such as the Gene Ontology in the biological field [3, 31], the Gold Ontology [47] and BabelNet [111] in the linguistics field or general knowledge bases that encodes human knowledge, such as WikiData [147] and YAGO [141].

We argue that a knowledge-based approach would be beneficial not only from a knowledge structuring and preservation point of view: it would rather enable Artificial Intelligence researchers to seamlessly collaborate with musicologists. The outcome would be the development of methods that not only statistically correlates with musicological knowledge, but are rather built upon the semantics of musicological knowledge in the first place.

7.2 Intensional representations for extensional applications

One of the most surprisingly pleasant results of this thesis lies in the results obtained in Chapter 6.1 both from a MIR and a future perspective point of view.

In the first place, we showed that the embeddings developed in Chapter 5 are able to obtain new state-of-the-art results in a relevant task to the MIR field, such as the structural segmentation task, despite the use of a straightforward model that has not been specialized to particularly fit the task. There are many experiments that can improve the presented algorithm, including the adaptation of semantic segmentation techniques from the Computer Vision field [106], the adaptation of techniques that perform syntactic or semantic text recognition from the Natural Language Processing [88, 26], or the combination of audio-based representation with symbolic-based representation from the Music Information Retrieval field in the first place [114]. All of these approaches are promising research paths that will be explored in future works.

We claim that the most interesting results of this thesis lies in the quality of the experiments of Section 5.5 and collaterally of Section 6.1.4. In Section 5.5 the accuracy obtained by embedding of the Knowledge Graph of Chapter 4 outperforms the other type of embeddings. This is particularly surprising if we take into account the amount of data that is used to train such a method when compared to the other approaches. When training chord embeddings with *traditional methods*, that is using methods that are based on the Distributional Hypothesis such as word2vec, fasttext or pitchclass2vec proposed in Section 5.3, a large amount of high-quality data is required. The accuracy of these models has been proven to be strictly related to the amount of data available [15]. In the NLP such huge set of textual resources can be obtained by the exploiting web pages [57]. The same cannot be said for accurate musical annotations where symbolic music annotations can be compared to a difficult-to-transcribe [140] low-resource language. The results presented in Section 5.5 and 6.1.4 suggest that an orthogonal approach, based on intensional representation of chords, is a promising research direction in the MIR field that need to further develop and presents an answer to the research question 3 presented in Chapter 1 - namely *Is it possible to identify a set of requirements for*

accurate chord representations?. Such requirements can be identified as the combination of NLP inspired techniques, such as the one presented in Section 5.3, and embedding computed from knowledge graph based systems modeled after well-known music theories, such as the method of Section 5.4 based on the Knowledge Graph presented in Chapter 4.

Chapter 8

Conclusion

The presented thesis proposes a new set of methods for the representation of symbolic chord annotations (in Chapter 5) that obtains new state-of-the-art results both in an intrinsic evaluation setting (in Section 5.5) and in an extrinsic evaluation task (in Section 6.1.4). The proposed methods pave the way for new approaches to MIR challenges that involve symbolic chord representations or the combination between audio-based representation and symbolic chord representations.

The combination of techniques from the Natural Language Processing field and the Semantic Web field results in a structure segmentation algorithm that achieves new state-of-the-art results and encourages the usage of the proposed embedding methods to other tasks that can be approached using symbolic representations. In future works we will explore this possibility, in particular by using the proposed chord embeddings as a way to enhance the accuracy of automatic chord transcription systems [123] and cover song detection algorithms [42].

The presented Modal Harmony Ontology and its corresponding Knowledge Graph (in Section 4) represents a valuable approach that can be extended to model other musical theories as well. We will explore this oppurtunity and extend the presented method to model the melodies of a composition as well.

Finally, other MIR tasks can be solved efficiently by means of Semantic

Web technologies, as briefly shown in Section 5.4. We will explore this opportunity in future works, by expanding on the presented method for Roman Notation inference and investigate other possible applications.

Bibliography

[1] T. W. Adorno and M. Paddison. On the problem of musical analysis. *Music Analysis*, 1(2):169–187, 1982.

[2] E. Anzuoni, S. Ayhan, F. Dutto, A. McLeod, F. C. Moss, and M. Rohrmeier. A historical analysis of harmonic progressions using chord embeddings. In *Proceedings of the 18th Sound and Music Computing Conference*, pages 284–291, 2021.

[3] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, et al. Gene ontology: tool for the unification of biology. *Nature genetics*, 25(1):25–29, 2000.

[4] A. Bakarov. A survey of word embeddings evaluation methods. *CoRR*, abs/1801.09536, 2018. arXiv: `1801.09536`. URL: `http://arxiv.org/abs/1801.09536`.

[5] C. F. Baker, C. J. Fillmore, and J. B. Lowe. The berkeley framenet project. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*, 1998.

[6] J. M. Barbour. *Tuning and temperament: A historical survey*. Courier Corporation, 2004.

[7] D. Beckett, T. Berners-Lee, E. Prud'hommeaux, and G. Carothers. Rdf 1.1 turtle. *World Wide Web Consortium*:18–31, 2014.

[8] Y. Bengio, R. Ducharme, and P. Vincent. A neural probabilistic language model. *Advances in neural information processing systems*, 13, 2000.

[9] B. Benward. *Music in Theory and Practice Volume 1*. McGraw-Hill Higher Education, 2014.

[10] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific american*, 284(5):34–43, 2001.

[11] E. Bigand, F. Madurell, B. Tillmann, and M. Pineau. Effect of global structure and temporal organization on chord processing. *Journal of Experimental Psychology: Human Perception and Performance*, 25(1):184, 1999.

[12] R. M. Bittner, M. Fuentes, D. Rubinstein, A. Jansson, K. Choi, and T. Kell. Mirdata: software for reproducible usage of datasets. In A. Flexer, G. Peeters, J. Urbano, and A. Volk, editors, *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, Delft, The Netherlands, November 4-8, 2019*, pages 99–106, 2019. URL: http://archives.ismir.net/ismir2019/paper/000009.pdf.

[13] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *Trans. Assoc. Comput. Linguistics*, 5:135–146, 2017. DOI: 10.1162/tacl_a_00051. URL: https://doi.org/10.1162/tacl%5C_a%5C_00051.

[14] D. Bollegala and J. O'Neill. A survey on word meta-embedding learning. In L. D. Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 5402–5409. ijcai.org, 2022. DOI: 10.24963/ijcai.2022/758. URL: https://doi.org/10.24963/ijcai.2022/758.

[15] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[16] J. A. Burgoyne, J. Wild, and I. Fujinaga. An expert ground truth set for audio chord recognition and music analysis. In A. Klapuri and C. Leider, editors, *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, pages 633–638. University of Miami, 2011. URL: `http://ismir2011.ismir.net/papers/OS8-1.pdf`.

[17] G. Burns. A typology of 'hooks' in popular records. *Popular Music*, 6(1):1–20, 1987. DOI: `10.1017/S0261143000006577`.

[18] E. Cambouropoulos. The local boundary detection model (LBDM) and its application in the study of expressive timing. In *Proceedings of the 2001 International Computer Music Conference, ICMC 2001, Havana, Cuba, September 17-22, 2001*. Michigan Publishing, 2001. URL: `https://hdl.handle.net/2027/spo.bbp2372.2001.021`.

[19] E. Cambouropoulos, M. A. Kaliakatsos-Papakostas, and C. Tsougras. An idiom-independent representation of chords for computational music analysis and generation. In *ICMC*, 2014.

[20] F. Carnovalini and A. Rodà. Computational creativity and music generation systems: an introduction to the state of the art. *Frontiers in Artificial Intelligence*, 3:14, 2020.

[21] H. Caselles-Dupré, F. Lesaint, and J. Royo-Letelier. Word2vec applied to recommendation: hyperparameters matter. In S. Pera, M. D. Ekstrand, X. Amatriain, and J. O'Donovan, editors, *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*, pages 352–356. ACM, 2018.

DOI: 10.1145/3240323.3240377. URL: https://doi.org/10.1145/3240323.3240377.

[22] A. Chapman. Some intervallic aspects of pitch-class set relations. *Journal of Music Theory*, 25(2):275–290, 1981. ISSN: 00222909. URL: http://www.jstor.org/stable/843652 (visited on 01/13/2023).

[23] T.-P. Chen and L. Su. Attend to chords: improving harmonic analysis of symbolic music using transformer-based models. *Transactions of the International Society for Music Information Retrieval*, 4(1), 2021.

[24] T. Chen and L. Su. Harmony transformer: incorporating chord segmentation into harmony recognition. In A. Flexer, G. Peeters, J. Urbano, and A. Volk, editors, *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, Delft, The Netherlands, November 4-8, 2019*, pages 259–267, 2019. URL: http://archives.ismir.net/ismir2019/paper/000030.pdf.

[25] S. S.-s. Cherfi, C. Guillotel, F. Hamdi, P. Rigaux, and N. Travers. Ontology-based annotation of music scores. In *Proceedings of the Knowledge Capture Conference*, K-CAP 2017, Austin, TX, USA. Association for Computing Machinery, 2017. ISBN: 9781450355537. DOI: 10.1145/3148011.3148038.

[26] A. Chiche and B. Yitagesu. Part of speech tagging: a systematic review of deep learning and machine learning approaches. *Journal of Big Data*, 9(1):1–25, 2022.

[27] S. Choudhary, T. Luthra, A. Mittal, and R. Singh. A survey of knowledge graph embedding and their applications. *CoRR*, abs/2107.07842, 2021. arXiv: 2107.07842. URL: https://arxiv.org/abs/2107.07842.

[28] J. Coates and D. Bollegala. Frustratingly easy meta-embedding - computing meta-embeddings by averaging source word embeddings. In M. A. Walker, H. Ji, and A. Stent, editors, *Proceedings of the 2018*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers), pages 194–198. Association for Computational Linguistics, 2018. DOI: 10.18653/v1/n18-2031. URL: https://doi.org/10.18653/v1/n18-2031.

[29] P. Collins and M. Schmuckler. Phrasing influences the recognition of melodies. *Psychonomic bulletin & review*, 4:254–9, June 1997. DOI: 10.3758/BF03209402.

[30] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(ARTICLE):2493–2537, 2011.

[31] G. O. Consortium et al. Creating the gene ontology resource: design and implementation. *Genome research*, 11(8):1425–1433, 2001.

[32] D. Cooke. *The Language of Music*. Oxford University Press, 1959.

[33] T. Crawford and L. Gibson. *Modern methods for musicology: prospects, proposals, and realities*. Routledge, 2016.

[34] M. S. Cuthbert and C. Ariza. Music21: a toolkit for computer-aided musicology and symbolic music data, 2010.

[35] W. B. De Haas, J. P. Magalhães, F. Wiering, and R. C. Veltkamp. Automatic functional harmonic analysis. *Computer Music Journal*, 37(4):37–53, 2013.

[36] J. de Berardinis, A. Meroño-Peñuela, A. Poltronieri, and V. Presutti. Choco: a chord corpus and a data transformation workflow for musical harmony knowledge graphs. In *Manuscript under review*, 2022.

[37] S. Decker, S. Melnik, F. Van Harmelen, D. Fensel, M. Klein, J. Broekstra, M. Erdmann, and I. Horrocks. The semantic web: the roles of xml and rdf. *IEEE Internet computing*, 4(5):63–73, 2000.

[38] T. de Clercq and D. Temperley. A corpus analysis of rock harmony. *Popular Music*, 30(1):47–70, 2011. DOI: 10.1017/S026114301000067X.

[39] W. B. de Haas, A. Volk, and F. Wiering. Structural segmentation of music based on repeated harmonies. In *2013 IEEE International Symposium on Multimedia, ISM 2013, Anaheim, CA, USA, December 9-11, 2013*, pages 255–258. IEEE Computer Society, 2013. DOI: 10.1109/ISM.2013.48. URL: https://doi.org/10.1109/ISM.2013.48.

[40] W. B. de Haas, F. Wiering, and R. C. Veltkamp. A geometrical distance measure for determining the similarity of musical harmony. *Int. J. Multim. Inf. Retr.*, 2(3):189–202, 2013. DOI: 10.1007/s13735-013-0036-6. URL: https://doi.org/10.1007/s13735-013-0036-6.

[41] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever. Jukebox: a generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

[42] G. Doras, F. Yesiler, J. Serrà Julià, E. Gómez Gutiérrez, and G. Peeters. Combining musical features for cover detection. In *Cumming J, Ha Lee J, McFee B, Schedl M, Devaney J, McKay C, Zagerle E, de Reuse T, editors. Proceedings of the 21st International Society for Music Information Retrieval Conference; 2020 Oct 11-16; Montréal, Canada.[Canada]: ISMIR; 2020. p. 279-86*. International Society for Music Information Retrieval (ISMIR), 2020.

[43] J. S. Downie, K. West, A. Ehmann, and E. Vincent. The 2005 music information retrieval evaluation exchange (mirex 2005): preliminary overview. In *6th int. conf. on music information retrieval (ismir)*, pages 320–323, 2005.

[44] A. Durant. A new day for music? digital technology in contemporary music-making, 1990.

[45] L. Ehrlinger and W. Wöß. Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48(1-4):2, 2016.

[46] M. Färber, F. Bartscherer, C. Menne, and A. Rettinger. Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web*, 9(1):77–129, 2018.

[47] S. Farrar and D. T. Langendoen. A linguistic ontology for the semantic web. *GLOT international*, 7(3):97–100, 2003.

[48] G. Fazekas, Y. Raimond, K. Jacobson, and M. Sandler. An overview of semantic web activities in the omras2 project. *Journal of New Music Research*, 39(4):295–311, 2010.

[49] A. Forte. A theory of set-complexes for music. *Journal of Music Theory*, 8(2):136–183, 1964.

[50] A. Forte. Pitch-class set analysis today. *Music analysis*, 4(1/2):29–58, 1985.

[51] A. Forte. *The American popular ballad of the golden era, 1924-1950*. Princeton University Press, 1995.

[52] A. Forte. *Tonal harmony in concept and practice*. Holt, Rinehart and Winston, 1962.

[53] B. W. Frankland and A. J. Cohen. Parsing of melody: quantification and testing of the local grouping rules of lerdahl and jackendoff's a generative theory of tonal music. *Music Perception*, 21(4):499–543, 2004.

[54] V. Fromkin, R. Rodman, and N. Hyams. *An introduction to language*. Cengage Learning, 2013.

[55] H. Gál. Theory and practice in composition. *Music & Letters*:37–49, 1942.

[56] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, and L. Schneider. Sweetening ontologies with dolce. In *International conference on knowledge engineering and knowledge management*, pages 166–181. Springer, 2002.

[57] L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, et al. The pile: an 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

[58] K. Giannos and E. Cambouropoulos. Symbolic encoding of simultaneities: re-designing the general chord type representation. In *8th International Conference on Digital Libraries for Musicology*, pages 67–74, 2021.

[59] M. Giraud, R. Groult, and F. Levé. Computational analysis of musical form. In D. Meredith, editor, *Computational Music Analysis*, pages 113–136. Springer, 2016. DOI: 10.1007/978-3-319-25931-4_5. URL: https://doi.org/10.1007/978-3-319-25931-4%5C_5.

[60] M. Good. Musicxml for notation and analysis. *The virtual score: representation, retrieval, restoration*, 12(113-124):160, 2001.

[61] D. Gusfield. *Algorithms on Strings, Trees, and Sequences - Computer Science and Computational Biology*. Cambridge University Press, 1997. ISBN: 0-521-58519-8. DOI: 10.1017/cbo9780511574931. URL: https://doi.org/10.1017/cbo9780511574931.

[62] G. Hadjeres, F. Pachet, and F. Nielsen. Deepbach: a steerable model for bach chorales generation. In *International Conference on Machine Learning*, pages 1362–1371. PMLR, 2017.

[63] N. Harley and G. Wiggins. An ontology for abstract, hierarchical music representation. In *Demo at the 16th International Society for Music Information Retrieval Conference (ISMIR 2015), Malaga, Spain*, 2015.

[64] C. Harte, M. B. Sandler, S. A. Abdallah, and E. Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. In *ISMIR 2005, 6th International Conference on Music Information Retrieval, London, UK, 11-15 September 2005, Proceedings*, pages 66–71, 2005. URL: `http://ismir2005.ismir.net/proceedings/1080.pdf`.

[65] E. P. hommeaux. Sparql query language for rdf. In 2011.

[66] H. Honing. The comeback of systematic musicology: new empiricism and the cognitive revolution, 2004.

[67] M. Horridge, N. Drummond, J. Goodwin, A. L. Rector, R. Stevens, and H. Wang. The manchester owl syntax. In *OWLed*, volume 216, 2006.

[68] A. E. Hull. Modern harmony. *London: Augener Ltd., nd*, 1915.

[69] E. J. Humphrey, J. Salamon, O. Nieto, J. Forsyth, R. M. Bittner, and J. P. Bello. JAMS: A JSON annotated music specification for reproducible MIR research. In H. Wang, Y. Yang, and J. H. Lee, editors, *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014*, pages 591–596, 2014. URL: `http://www.terasoft.com.tw/conf/ismir2014/proceedings/T106%5C_355%5C_Paper.pdf`.

[70] D. B. Huron. *The humdrum toolkit: Reference manual*. Center for Computer Assisted Research in the Humanities, 1994.

[71] J. Jones, D. de Siqueira Braga, K. Tertuliano, and T. Kauppinen. Musicowl: the music score ontology. In *Proceedings of the International Conference on Web Intelligence*, WI '17, Leipzig, Germany. Association for Computing Machinery, 2017. ISBN: 9781450349512. DOI: 10.1145/3106426.3110325.

[72] S. Kantarelis, E. Dervakos, N. Kotsani, and G. Stamou. Functional harmony ontology: musical harmony analysis with description logics. *Journal of Web Semantics*, 75:100754, 2023. ISSN: 1570-8268. DOI: https://doi.org/10.1016/j.websem.2022.100754. URL: https://www.sciencedirect.com/science/article/pii/S1570826822000385.

[73] R. Keller et al. Impro-visor. *Harvey Mudd Computer Science Department,[online] Available from: http://www. cs. hmc. edu/~ keller/jazz/improvisor/(Accessed 27 March 2013)*, 2012.

[74] H. V. Koops, W. B. de Haas, J. A. Burgoyne, J. Bransen, A. Kent-Muller, and A. Volk. Annotator subjectivity in harmony annotations of popular music. *Journal of New Music Research*, 48(3):232–252, 2019. DOI: 10.1080/09298215.2019.1613436.

[75] F. Korzeniowski and G. Widmer. Improved chord recognition by combining duration and harmonic language models. In E. Gómez, X. Hu, E. Humphrey, and E. Benetos, editors, *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23-27, 2018*, pages 10–17, 2018. URL: http://ismir2018.ircam.fr/doc/pdfs/300%5C_Paper.pdf.

[76] A. Krisnadhi, F. Maier, and P. Hitzler. Owl and rules. In *Reasoning Web International Summer School*, pages 382–415. Springer, 2011.

[77] C. L. Krumhansl and P. W. Jusczyk. Infants' perception of phrase structure in music. *Psychological Science*, 1(1):70–73, 1990. ISSN:

09567976, 14679280. URL: http://www.jstor.org/stable/40062394 (visited on 11/10/2022).

[78] A. Kurakin, I. J. Goodfellow, and S. Bengio. Adversarial machine learning at scale. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL: https://openreview.net/forum?id=BJm4T4Kgx.

[79] B. Laden. Melodic anchoring and tone duration. *Music Perception*, 12(2):199, 1994. URL: https://www.proquest.com/scholarly-journals/melodic-anchoring-tone-duration/docview/1300618540/se-2. Last updated - 2013-02-24.

[80] A. Lahnala, G. Kambhatla, J. Peng, M. Whitehead, G. Minnehan, E. Guldan, J. K. Kummerfeld, A. Çamcı, and R. Mihalcea. Chord embeddings: analyzing what they capture and their role for next chord prediction and artist attribute prediction. In *International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar)*, pages 171–186. Springer, 2021.

[81] P. H. Lang. Editorial. *The Musical Quarterly*, 32(1):131–136, 1946. ISSN: 00274631, 17418399. URL: http://www.jstor.org/stable/739569 (visited on 01/19/2023).

[82] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[83] M. Leman. Systematic musicology at the crossroads of modern music research. In *Systematic and comparative musicology: Concepts, methods, findings*, pages 89–115. Peter Lang, 2008.

[84] M. Levine. *The jazz theory book*. " O'Reilly Media, Inc.", 2011.

[85] M. Levy and M. B. Sandler. Structural segmentation of musical audio by constrained clustering. *IEEE Trans. Speech Audio Process.*, 16(2):318–326, 2008. DOI: 10.1109/TASL.2007.910781. URL: https://doi.org/10.1109/TASL.2007.910781.

[86] S. E. Lewis. Gene ontology: looking backwards and forwards. *Genome biology*, 6(1):1–4, 2005.

[87] H. Li, Z. Tang, X. Fei, K.-M. Chao, M. Yang, and C. He. A survey of audio mir systems, symbolic mir systems and a music definition language demo-system. In *2017 IEEE 14th International Conference on e-Business Engineering (ICEBE)*, pages 275–281. IEEE, 2017.

[88] J. Li, A. Sun, J. Han, and C. Li. A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1):50–70, 2020.

[89] E. Liebman and P. Stone. Artificial musical intelligence: a survey. *arXiv preprint arXiv:2006.10553*, 2020.

[90] S. R. Livingstone, C. Palmer, and E. Schubert. Emotional response to musical repetition. *Emotion*, 12(3):552, 2012.

[91] W. T. Lu, L. Su, et al. Vocal melody extraction with semantic segmentation and audio-symbolic domain transfer learning. In *ISMIR*, pages 521–528, 2018.

[92] H. M. Lukashevich. Towards quantitative measures of evaluating song segmentation. In J. P. Bello, E. Chew, and D. Turnbull, editors, *ISMIR 2008, 9th International Conference on Music Information Retrieval, Drexel University, Philadelphia, PA, USA, September 14-18, 2008*, pages 375–380, 2008. URL: http://ismir2008.ismir.net/papers/ISMIR2008%5C_219.pdf.

[93] S. Madjiheurem, L. Qu, and C. Walder. Chord2vec: learning musical chord embeddings. In *Proceedings of the constructive machine learning workshop at 30th conference on neural information processing systems (NIPS2016), Barcelona, Spain*, 2016.

[94] A. Marmoret, J. E. Cohen, N. Bertin, and F. Bimbot. Uncovering audio patterns in music with nonnegative tucker decomposition for structural segmentation. *CoRR*, abs/2104.08580, 2021. arXiv: 2104.08580. URL: https://arxiv.org/abs/2104.08580.

[95] N. Martin. The tristan chord resolved. *Intersections: Canadian Journal of Music/Intersections: revue canadienne de musique*, 28(2):6–30, 2008.

[96] M. C. McCallum. Unsupervised learning of deep features for music segmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 346–350. IEEE, 2019. DOI: 10.1109/ICASSP.2019.8683407. URL: https://doi.org/10.1109/ICASSP.2019.8683407.

[97] D. L. McGuinness, F. Van Harmelen, et al. Owl web ontology language overview. *W3C recommendation*, 10(10):2004, 2004.

[98] C. McKay, J. A. Burgoyne, J. Hockman, J. B. Smith, G. Vigliensoni, and I. Fujinaga. Evaluating the genre classification performance of lyrical features relative to audio, symbolic and cultural features. In *ISMIR*, volume 9 of number 13, pages 213–218, 2010.

[99] C. McKay and I. Fujinaga. Combining features extracted from audio, symbolic and cultural sources. In *ISMIR*, volume 14 of number 18, pages 597–602. Citeseer, 2008.

[100] L. F. Menabrea and A. Lovelace. Sketch of the analytical engine invented by charles babbage, esq., by lf menabrea, of turin, officer of

the military engineers. *Translated and with notes by AA L. Taylor's Scientific Memoirs*, 3:666–731, 1843.

[101] D. Meredith, K. Lemström, and G. A. Wiggins. Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. *Journal of New Music Research*, 31(4):321–345, 2002.

[102] A. Meroño-Peñuela, R. Hoekstra, A. Gangemi, P. Bloem, R. de Valk, B. Stringer, B. Janssen, V. de Boer, A. Allik, S. Schlobach, and K. Page. The midi linked data cloud. In *The Semantic Web – ISWC 2017*, Cham. Springer International Publishing, 2017. ISBN: 978-3-319-68204-4.

[103] G. Micchi, M. Gotham, and M. Giraud. Not all roads lead to rome: pitch representation and model architecture for automatic harmonic analysis. *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 3(1):42–54, 2020.

[104] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In Y. Bengio and Y. LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013. URL: `http://arxiv.org/abs/1301.3781`.

[105] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[106] Y. Mo, Y. Wu, X. Yang, F. Liu, and Y. Liao. Review the state-of-the-art technologies of semantic segmentation based on deep learning. *Neurocomputing*, 493:626–646, 2022.

[107] B. Mor, S. Garhwal, and A. Kumar. A systematic literature review on computational musicology. *Archives of Computational Methods in Engineering*, 27(3):923–937, 2020.

[108] R. Morris. A similarity index for pitch-class sets. *Perspectives of New Music*, 18(1/2):445–460, 1979. ISSN: 00316016. URL: `http://www.jstor.org/stable/832996` (visited on 01/13/2023).

[109] B. Motik, B. C. Grau, I. Horrocks, Z. Wu, A. Fokoue, C. Lutz, et al. Owl 2 web ontology language profiles. *W3C recommendation*, 27(61), 2009.

[110] E. Mugglestone. Guido adler's "the scope, method, and aim of musicology"(1885): an english translation with an historico-analytical commentary. *Yearbook for traditional music*, 13:1–21, 1981.

[111] R. Navigli and S. P. Ponzetto. Babelnet: building a very large multilingual semantic network. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 216–225, 2010.

[112] M. E. Newman. Power laws, pareto distributions and zipf's law. *Contemporary physics*, 46(5):323–351, 2005.

[113] H.-W. Nienhuys and J. Nieuwenhuizen. Lilypond, a system for automated music engraving. In *Proceedings of the XIV Colloquium on Musical Informatics (XIV CIM 2003)*, volume 1, pages 167–171. Citeseer, 2003.

[114] O. Nieto, G. J. Mysore, C. Wang, J. B. L. Smith, J. Schlüter, T. Grill, and B. McFee. Audio-based music structure analysis: current trends, open challenges, and applications. *Trans. Int. Soc. Music. Inf. Retr.*, 3(1):246–263, 2020. DOI: `10.5334/tismir.54`. URL: `https://doi.org/10.5334/tismir.54`.

[115] I. Niles and A. Pease. Towards a standard upper ontology. In *Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001*, pages 2–9, 2001.

[116] N. F. Noy. Semantic integration: a survey of ontology-based approaches. *ACM Sigmod Record*, 33(4):65–70, 2004.

[117] F. Pachet. A joyful ode to automatic orchestration. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(2):1–13, 2016.

[118] F. Pachet. Computer analysis of jazz chord sequence: is solar a blues?, 2000.

[119] R. Parncutt. Systematic musicology and the history and future of western musical scholarship. *Journal of interdisciplinary music studies*, 1(1):1–32, 2007.

[120] H. Paulheim. Knowledge graph refinement: a survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508, 2017.

[121] J. Pauwels, F. Kaiser, and G. Peeters. Combining harmony-based and novelty-based approaches for structural segmentation. In A. de Souza Britto Jr., F. Gouyon, and S. Dixon, editors, *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR 2013, Curitiba, Brazil, November 4-8, 2013*, pages 601–606, 2013. URL: `http://www.ppgia.pucpr.br/ismir2013/wp-content/uploads/2013/09/138%5C_Paper.pdf`.

[122] J. Pauwels, J. Martens, and M. Leman. The influence of chord duration modeling on chord and local key extraction. In X. Chen, T. S. Dillon, H. Ishibuchi, J. Pei, H. Wang, and M. A. Wani, editors, *10th International Conference on Machine Learning and Applications and Workshops, ICMLA 2011, Honolulu, Hawaii, USA, December 18-21, 2011. Volume 2: Special Sessions and Workshop*, pages 136–141. IEEE Computer Society, 2011. DOI: `10.1109/ICMLA.2011.141`. URL: `https://doi.org/10.1109/ICMLA.2011.141`.

[123] J. Pauwels, K. O'Hanlon, E. Gómez, and M. B. Sandler. 20 years of automatic chord recognition from audio. In A. Flexer, G. Peeters, J. Urbano, and A. Volk, editors, *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, Delft,*

The Netherlands, November 4-8, 2019, pages 54–63, 2019. URL: http://archives.ismir.net/ismir2019/paper/000004.pdf.

[124] G. Perle. Pitch-Class Set Analysis: An Evaluation. *Journal of Musicology*, 8(2):151–172, April 1990. ISSN: 0277-9269. DOI: 10.2307/763567. eprint: https://online.ucpress.edu/jm/article-pdf/8/2/151/193572/763567.pdf. URL: https://doi.org/10.2307/763567.

[125] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis. Mir_eval: A transparent implementation of common MIR metrics. In H. Wang, Y. Yang, and J. H. Lee, editors, *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014*, pages 367–372, 2014. URL: http://www.terasoft.com.tw/conf/ismir2014/proceedings/T066%5C_320%5C_Paper.pdf.

[126] S. M. Rashid, D. De Roure, and D. L. McGuinness. A music theory ontology. In *Proceedings of the 1st International Workshop on Semantic Applications for Audio and Music*, pages 6–14, 2018.

[127] H. Riemann. *Skizze einer neuen Methode der Harmonielehre*. Breitkopf und Härtel, 1880.

[128] P. Ristoski and H. Paulheim. Rdf2vec: rdf graph embeddings for data mining. In *International Semantic Web Conference*, pages 498–514. Springer, 2016.

[129] P. Ristoski, J. Rosati, T. Di Noia, R. De Leone, and H. Paulheim. Rdf2vec: rdf graph embeddings and their applications. *Semantic Web*, 10(4):721–752, 2019.

[130] J. Roeder. A geometric representation of pitch-class series. *Perspectives of New Music*, 25(1/2):362–409, 1987. ISSN: 00316016. URL: http://www.jstor.org/stable/833104 (visited on 01/13/2023).

[131] D. Rubinstein, E. Levi, R. Schwartz, and A. Rappoport. How well do distributional models capture different types of semantic knowledge? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 726–730, 2015.

[132] M. Sahlgren. The distributional hypothesis. *Rivista di Linguistica (Italian Journal of Linguistics)*, 20:33–53, 2008.

[133] H. Schenker. *Neue musikalische Theorien und Phantasien*, volume 2. Universal-edition ag, 1910.

[134] G. Shibata, R. Nishikimi, and K. Yoshii. Music structure analysis based on an LSTM-HSMM hybrid model. In J. Cumming, J. H. Lee, B. McFee, M. Schedl, J. Devaney, C. McKay, E. Zangerle, and T. de Reuse, editors, *Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020, Montreal, Canada, October 11-16, 2020*, pages 23–29, 2020. URL: `http://archives.ismir.net/ismir2020/paper/000005.pdf`.

[135] G. Singh, S. Bhatia, and R. Mutharaju. Owl2bench: a benchmark for owl 2 reasoners. In *International semantic web conference*, pages 81–96. Springer, 2020.

[136] J. B. L. Smith, J. A. Burgoyne, I. Fujinaga, D. D. Roure, and J. S. Downie. Design and creation of a large-scale database of structural annotations. In A. Klapuri and C. Leider, editors, *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011*, pages 555–560. University of Miami, 2011. URL: `http://ismir2011.ismir.net/papers/PS4-14.pdf`.

[137] M. J. Steedman. A generative grammar for jazz chord sequences. *Music Perception*, 2(1):52–77, 1984.

[138] C. J. Stevens. Music perception and cognition: A review of recent cross-cultural research. *Top. Cogn. Sci.*, 4(4):653–667, 2012. DOI: 10.1111/j.1756-8765.2012.01215.x. URL: https://doi.org/10.1111/j.1756-8765.2012.01215.x.

[139] N. Stringham and M. Izbicki. Evaluating word embeddings on low-resource languages. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 176–186, Online. Association for Computational Linguistics, November 2020. DOI: 10.18653/v1/2020.eval4nlp-1.17. URL: https://aclanthology.org/2020.eval4nlp-1.17.

[140] L. Su and Y.-H. Yang. Escaping from the abyss of manual annotation: new methodology of building polyphonic datasets for automatic music transcription. In *Music, Mind, and Embodiment: 11th International Symposium, CMMR 2015, Plymouth, UK, June 16-19, 2015, Revised Selected Papers 11*, pages 309–321. Springer, 2016.

[141] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a large ontology from wikipedia and wordnet. *Journal of Web Semantics*, 6(3):203–217, 2008.

[142] N. Tan, R. Aiello, and T. Bever. Harmonic structure as a determinant of melodic organization. *Memory & cognition*, 9:533–9, October 1981. DOI: 10.3758/BF03202347.

[143] D. Temperley. *The cognition of basic musical structures*. MIT press, 2004.

[144] D. Tymoczko, M. Gotham, M. S. Cuthbert, and C. Ariza. The romantext format: a flexible and standard method for representing roman numeral analyses. In *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR*, volume 2019, pages 123–129, 2019.

[145] G. Velarde, T. Weyde, and D. Meredith. An approach to melodic segmentation and classification based on filtering with the haar-wavelet. *Journal of New Music Research*, 42(4):325–345, 2013.

[146] H. Vinet. The representation levels of music information. In *International Symposium on Computer Music Modeling and Retrieval*, pages 193–209. Springer, 2004.

[147] D. Vrandečić and M. Krötzsch. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85, 2014.

[148] J. Wang, J. B. L. Smith, W. T. Lu, and X. Song. Supervised metric learning for music structure features. In J. H. Lee, A. Lerch, Z. Duan, J. Nam, P. Rao, P. van Kranenburg, and A. Srinivasamurthy, editors, *Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021, Online, November 7-12, 2021*, pages 730–737, 2021. URL: `https://archives.ismir.net/ismir2021/paper/000091.pdf`.

[149] S. Wang, W. Zhou, and C. Jiang. A survey of word embeddings based on deep learning. *Computing*, 102(3):717–740, 2020. DOI: `10.1007/s00607-019-00768-7`. URL: `https://doi.org/10.1007/s00607-019-00768-7`.

[150] R. Wason. Schenker's" notion of scale-step in historical perspective: non-essential harmonies in viennese fundamental bass theory". *Journal of Music Theory*, 27(1):49–73, 1983.

[151] R. J. Weiss and J. P. Bello. Identifying repeated patterns in music using sparse convolutive non-negative matrix factorization. In J. S. Downie and R. C. Veltkamp, editors, *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010, Utrecht, Netherlands, August 9-13, 2010*, pages 123–128. International Society for Music Information Retrieval, 2010. URL: `http://ismir2010.ismir.net/proceedings/ismir2010-23.pdf`.

[152] R. J. Weiss and J. P. Bello. Unsupervised discovery of temporal structure in music. *IEEE J. Sel. Top. Signal Process.*, 5(6):1240–1251, 2011. DOI: 10.1109/JSTSP.2011.2145356. URL: https://doi.org/10.1109/JSTSP.2011.2145356.

[153] T. Wigram and C. Gold. Music therapy in the assessment and treatment of autistic spectrum disorder: clinical application and research evidence. *Child: care, health and development*, 32(5):535–542, 2006.

[154] M. Zeng, X. Tan, R. Wang, Z. Ju, T. Qin, and T.-Y. Liu. Musicbert: symbolic music understanding with large-scale pre-training. *arXiv preprint arXiv:2106.05630*, 2021.