Friedrich-Alexander-Universität Erlangen-Nürnberg

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

Master Thesis

A Cross-Version Study on Computational Harmony Analysis of Music Recordings

submitted by

Florian Schuberth

submitted

January 15, 2021

Supervisor / Advisor

Dr.-Ing. Christof Wei
Prof. Dr. Meinard Müller

Reviewers

Prof. Dr. Meinard Müller

AUDIO LABS

International Audio Laboratories Erlangen
A joint institution of the
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and
the Fraunhofer-Institut für Integrierte Schaltungen IIS.

Fraunhofer IIS

Erklärung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form in einem Verfahren zur Erlangung eines akademischen Grades vorgelegt.

Erlangen, 15. Januar 2021

Florian Schuberth

Acknowledgements

This thesis represents the final effort of my studies to become an electrical engineer. Writing a thesis in the midst of a global pandemic provides a special challenge and in the following I would like to express my gratitude to the people who supported me during this time.

First and foremost I want to thank my supervisors Meinard Müller and Christof Weiß for their guidance and giving me the opportunity to combine the technical aspect of engineering with my personal passion for music. They introduced me to the field of Music Information Retrieval, which provides an exciting cross-disciplinary challenge, bringing together aspects from both my professional and personal interests. Furthermore, I want to thank the people who provided the data I used for my analysis within this thesis. For the Schubert dataset I want to thank Leo Brütting, Junseong Park, Hendrik Vincent Koops, Frank Zalkow, and Vlora Arifi-Müller. For the Beethoven dataset I want to thank Leo Brütting and Michael Kohl. Additionally, I want to thank Tim Zunner and Johannes Zeitler for providing the Deep Chroma features for both datasets. Publications that inspired my research are cited within the thesis.

Finally, I want to express my gratitude to my family and my friends. Thank you for your support.

Abstract

Computational harmony analysis plays an important role within the topic of *Music Information Retrieval*. It comprises a wide range of analysis tasks and potential for practical applications. Since chords constitute an essential notion for the analysis and perception of harmony, automatic chord recognition from audio recordings offers interesting insights, both from a musicological and technical standpoint. This thesis presents an evaluation of various chord recognition methods and their individual parameters, applied to two cross-version datasets of Western classical music recordings. In our experiments, we study the impact of individual parameters of various signal processing steps that are applied as part of the chord recognition systems. This includes enhancement strategies such as temporal filtering or compression, as well as the implementation of various chord models. The results show a strong interplay between algorithmic parameters and suggest their joint optimization. We use a comparison of features extracted from the audio data with symbolic baseline features to highlight the relation of musical and technical challenges involved in chord recognition. These baseline experiments investigate the problems of chord recognition, when the technical challenge of extracting pitch content from the audio data is eliminated. The results show that the musical challenge involved in chord recognition seems to outweigh the problems arising from signal processing. Finally, an in-depth analysis of the results on a track level shows the differences between evaluating across songs and versions of a cross-version dataset. For both datasets, the efficiency of chord recognition shows a higher variance across different songs than different versions. In summary, the results discussed within this thesis offer insights both into the technical aspect of automatic chord recognition and into the datasets themselves and their relevance for musicological studies.

Zusammenfassung

Die computergestützte Harmonieanalyse spielt eine wichtige Rolle im Feld der *Music Information Retrieval*. Sie umfasst ein weites Feld an Aufgaben und besitzt weitreichendes Potential für praktische Anwendungen. Akkorde bilden ein grundlegendes Konzept für die Wahrnehmung und Analyse von musikalischen Harmonien. Aus diesem Grund kann die automatisierte Akkorderkennung von Musikstücken sowohl musikwissenschaftliche als auch technische Einblicke bieten. In dieser Arbeit werden verschiedene Methoden zur Akkorderkennung und ihre Parameter im Kontext von zwei Korpora aus der klassischen Musik evaluiert. Beide Datensätze beinhalten mehrere Versionen der jeweiligen Musikstücke. In den Experimenten wird zunächst der Einfluss verschiedener algorithmischer Parameter auf die Qualität der Akkorderkennung untersucht. Die Ergebnisse zeigen Wechselwirkungen zwischen den einzelnen Parametern, welche eine gemeinsame Optimierung nahe legen. Um das Verhältnis von musikalischen und technischen Herausforderungen in der Akkorderkennung zu untersuchen, werden verschiedene Arten von Merkmalen verglichen. Der Vergleich von Merkmalen aus der klassischen Signalverarbeitung mit symbolbasierten, perfekten Merkmalen zeigt, dass die musikalische Herausforderung der Abstrahierung von Notenmerkmalen zu Akkorden einen größeren Einfluss auf die Qualität der Akkorderkennung hat, als die technische Herausforderung der Extrahierung von aussagekräftigen Merkmalen aus den Aufnahmen. Eine abschließende Detailanalyse der Ergebnisse zeigt die Qualitätsschwankungen der Akkorderkennung für einzelne Lieder und Versionen der Datensätze. Für beide Korpora zeigt sich eine größere Fluktuation der Ergebnisse für verschiedene Lieder, als für verschiedene Versionen. Die experimentellen Ergebnisse dieser Arbeit bieten Einblicke in technische und algorithmische Aspekte der automatisierten Akkorderkennung. Zudem werden musikalische Merkmale der Datensätze herausgearbeitet.

Contents

Erklärung **i**

Acknowledgements **iii**

Abstract **v**

Zusammenfassung **vii**

1 Introduction **3**

2 Background and Related Work **5**
2.1 Chord Recognition Problem 5
2.2 Chord Recognition Datasets 6
2.3 Chord Recognition Methods 7

3 Datasets **11**
3.1 Schubert Winterreise Dataset (SWD) 11
3.2 Beethoven Piano Sonatas Dataset (BSD) 14
3.3 Chord Annotations and Vocabularies 15

4 Chord Recognition Approaches **19**
4.1 Feature Extraction 19
4.2 Pre-Filtering 23
4.3 Chord Models 24
4.4 Pattern Matching, Post-Filtering and Output 26
4.5 Evaluation 27

5 Experiments and Results **29**
5.1 Effect of Individual Parameters 29
5.2 Interplay Between Different Parameters 34
5.3 Cross-Validation Splits 40
5.4 Comparison of Different Chord Vocabularies 43
5.5 Comparison of Chroma Feature Types 46

5.6 In-Depth Analysis of Results 57

6 Conclusions **71**

A Chord Statistics **75**

B Track-Wise Recall Values **85**

C Score Excerpts **93**

Bibliography **97**

Chapter 1

Introduction

Computational harmony analysis is one of the widest areas of research within *Music Information Retrieval* (MIR). It offers an algorithmic approach to extract various information, such as global or local keys, pitch activations, or chords, from music. This thesis presents a study of various approaches to automatically extract chord information from audio recordings of Western classical music.

The term harmony refers to simultaneously sounding musical notes, which are perceived as one unit by a human listener. When three or more notes are combined, one speaks of a chord. Chords are considered to be one of the main building blocks of harmony and are therefore at the central interest of musicological studies. Automatic chord recognition can be of aid in a multitude of scenarios, especially when larger bodies of data are involved. It presents a multi-disciplinary challenge, which can advance musical as well as technical research.

The complexity of automatic chord recognition tasks may vary widely, depending on the data that is analyzed and the level of detail that is expected from the output. For example, the analysis of audio recordings involves a great deal of front-end signal processing to create features that are suited for automatic evaluation. The musical challenge of correctly recognizing the played chords from pitch content is therefore extended by the technical challenge of providing meaningful features from the raw data. Conversely, the analysis of purely symbolic data, e.g., piano-roll representations, significantly reduces this technical challenge. For the experiments conducted as part of this thesis, we use audio recordings as input data.

Another influential factor is the style and historic background of music to be analyzed. In music history, different eras are characterized by the use of typical instrumentation, harmonic language, and musical structure. In these aspects, e.g., “Let It Be” by The Beatles will differ greatly from the “Mass in B Minor” by Johann Sebastian Bach. Prior knowledge of these characteristics can be beneficial when implementing chord recognition systems. The two datasets we analyze in our studies are examples of music from the late Classical and the Romantic period.

In the following, we give a description of the individual segments of this thesis. Chapter 2 provides a detailed description of the underlying problems of automatic chord recognition and an overview of related work within the field. The focus of this discussion lies on the various implementations and datasets that are commonly used in publications. In Chapter 3, we present the datasets which we use for our experiments. This includes a quick overview of their musical characteristics and a description of the raw data and annotations that are included within. Chapter 4 contains an explanation of the approaches we utilize to perform chord recognition and evaluate the results. We explain the different methods and introduce important parameters. Furthermore, we describe the evaluation metrics for comparing the recognition results to the ground truth annotations. Chapter 5 describes our practical experiments and analyzes the results in various contexts. Initially, we focus on variations of the algorithmic parameters, their interactions and their effect on the recognition results. We discuss and compare different possibilities to use the datasets for data-driven chord recognition. Furthermore, we present the impact of utilizing chord vocabularies of varying complexity and the related problems. In conjunction with the previously explained parameter variations, we compare results for different feature types. This provides an insight into the relationship of technical and musical challenges involved in chord recognition. Finally, we use an in-depth analysis of the recognition results on a singular song level to discuss the cross-version aspect and musical difficulties. To finalize the thesis, Chapter 6 summarizes the results and gives an outlook on possible subsequent research opportunities.

Chapter 2

Background and Related Work

In the following sections, we discuss the task of automatic chord recognition on the basis of scientific publications made in the field. Initially, we provide a detailed definition of the underlying problems and the modality of different chord vocabularies. Furthermore, we present corpora of music data that are commonly utilized in literature and compare them to the datasets we use for our experiments. Finally, we give an overview of different technical implementations of chord recognition systems and their efficacy.

2.1 Chord Recognition Problem

To define the problem of chord recognition, the term can be taken literal—to correctly recognize which chord is present at what time in the music. This implies two separate challenges. Firstly, segmentation of the data into temporal regions, whose start and end times correspond to the chord changes in the music. The second challenge is finding the correct chord label for each sequence. Most approaches address both challenges simultaneously by providing a chord label for each time frame, implicitly setting region borders. While the term chord *recognition* seems to be used synonymously in most publications with other terms such as chord *estimation* or *transcription*, Humphrey et al. [12] suggest a subtle differentiation. They propose that chord transcription is a more abstract task, taking into account structural information of the musical data and finding functional relations between chords. This relates it principally closer to a segmentation task than is the case for common approaches to chord recognition, which do not consider functional harmony.

Evaluating the effectiveness of a chord recognition system requires the existence of reference chord annotations, often referred to as ground truth. In most cases, these annotations are manually created by music experts, assigning a single chord label to each segment of the music recording. Due to the often ambiguous nature of musical harmony, a number of publications

focus on the influence of subjectivity on the annotation process, e.g., [12, 17, 31]. The comparison of chord labels from different experts shows a clear impact of personal preferences on the labeling process. Ni et al. [31] propose a maximum annotator agreement of around 90%. In a similar experiment, Koops et al. [17] report that an agreement between annotators was reached on only 76% of the assessed data, decreasing with increasing complexity of utilized chord vocabularies. When annotators were free to choose individual chord label complexity, the intersection of the resulting vocabularies was smaller than 20% of all vocabularies combined. These findings offer interesting insights into the "correctness" of ground truth annotations and raise concerns about the significance of automatic chord recognition results that exceed annotator consensus. Modern recognition systems may therefore have started to overfit the idiosyncracies of different chord labeling styles used by individual annotators and datasets.

The use of different chord vocabularies generally plays an important role in the context of chord recognition. In this scenario, the term vocabulary refers to the set of individual chord types the recognizer has to distinguish. Commonly used vocabularies include, e.g., the major/minor vocabulary (containing 24 different chords), or more complex variants such as a combined major/minor and 7^{th} chord vocabulary. The choice of vocabulary not only significantly impacts the complexity of the chord recognition task, but also dictates the way in which harmonies present in the music are mapped to specific chord symbols. As an example, when utilizing the major/minor vocabulary, a C minor chord with an added minor seventh note is commonly mapped to a C minor chord by the annotator. So is a C minor chord consisting only of the corresponding triad notes. This implies that an automatic chord recognition system has to label different features as the same chord. A larger and more complex vocabulary might introduce distinguishable labels for specific chords, but makes it harder to differentiate between the higher number of similar chords.

2.2 Chord Recognition Datasets

In the context of chord recognition, different datasets are used to develop, train, and evaluate recognition algorithms. These collections are essential for research in the field and create a common ground that can be used to compare approaches from various publications. For chord recognition, these datasets usually contain start and end time stamps and a label for each chord that is present in the music. The labels are usually manually created by music experts. The creation of extensive datasets is an extremely labor-intensive and time-consuming task, which involves a lot of expert knowledge and curation to make the annotations as accurate as possible. Table 3.2 shows an example for audio-aligned chord label annotations.

While researchers might create own datasets for their studies, there is a number of datasets that are commonly used within the chord recognition community. The most important one to mention here is the Beatles dataset, containing annotations for a large number of songs published by the

British band "The Beatles." The annotations are publicly available as part of the *Isophonics* dataset[1], that is curated by the Centre for Digital Music at Queen Mary University in London. It has been used in a large number of scientific and didactic publications within the field of MIR, e.g., [13, 24, 28].

In combination with a collection of songs by the bands "Queen" and "Zweieck," the Beatles dataset is used as a test dataset in the annual Music Information Retrieval Evaluation eXchange (MIREX).[2] The latter represents a community-based framework, where tasks from MIR can be evaluated in a comparable manner [7]. Next to the *Isophonics* dataset, the more recent *Billboard* dataset is also used to compare chord recognition algorithms at the MIREX challenge. It comprises a large collection of annotations for songs from popular music that were included in *Billboard* magazine's "Hot 100" rankings between 1958 and 1991 [3]. The annotations are also publicly available.[3] While there is a relatively large amount of available annotated data for chord recognition, most of it is popular music. There is a distinct lack of manually annotated datasets that contain examples from classical music recordings. The datasets that are used in this thesis are both relatively new and have therefore not been as extensively explored as the aforementioned examples. They contain annotations for influential pieces of music from the late Classical and Romantic period, both performed by multiple artists. We give a detailed description of these datasets in Chapter 3.

Classical music provides an interesting approach for cross-version analysis, since different versions usually follow the same score with the same instrumentation. In popular music, cover versions are more common, which might use different instrumentation and usually deviate from the original versions to a larger extent. A number of studies focus on cross-version approaches for the harmony analysis of classical music. For example, Konz et al. [15] exploit deviations across versions to stabilize the chord recognition process. Weiss et al. [37] explore the influence of training data-driven local key estimators across versions, songs, and annotators.

2.3 Chord Recognition Methods

Since automatic chord recognition has been and still is a very active field within MIR, there is a large number of different algorithmic approaches to extract chord information from music recordings. In the following, we provide a thematic categorization of various approaches and enhancement methods found in literature.

In 1999, Fujishima [9] proposed a matching of hand-crafted, binary chord template vectors with chroma features extracted from audio signals. The matching was implemented using the Euclidean distance or the cosine similarity as measures, picking the chord template with highest

[1] http://www.isophonics.net/datasets
[2] https://www.music-ir.org/mirex/
[3] https://ddmal.music.mcgill.ca/research/The_McGill_Billboard_Project_(Chord_Analysis_Dataset)/

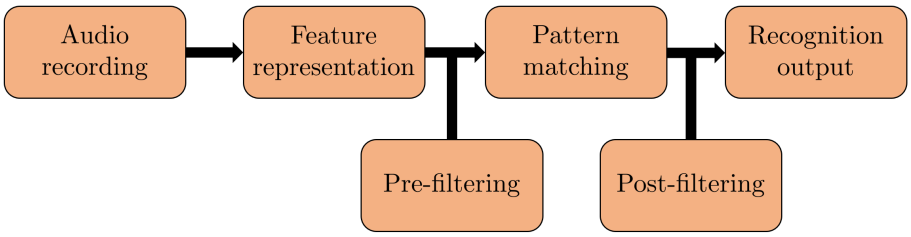


Figure 2.1. Schematic overview of the common stages of automatic chord recognition systems [28].

similarity as recognition result for each time frame. While this approach has been published over 20 years ago, the basic idea or parts of it can still be found in most modern chord recognition methods.
Generally, the process can be broken down into two main steps: the extraction of suitable features from the music-based input data and the matching of these features with some form of chord templates. Enhancement methods that are applied before the pattern matching stage are referred to as pre-filtering in literature. When applied during or after the pattern matching, these enhancements are denoted post-filtering. Figure 2.1 provides a schematic visualization of the structure. It should be noted that there are numerous publications using symbolic music representations as input data. In this thesis, the applied methods and presented state-of-the-art will be restricted to chord recognition from audio recordings.

In [9], the pitch class profile (PCP) is used as feature representation. It is a twelve-dimensional vector, representing the salience of the twelve semitone pitch classes[4] within the audio data. This representation does not contain octave information. The PCP is also referred to as *chroma*, which is the term we will use throughout this thesis. Chroma vectors have been found to be a suitable feature for many MIR tasks, including chord recognition. They can be obtained in different ways. Commonly, they are acquired by first computing a pitch representation with, e.g., the constant-Q transform (CQT) or a binned Short-time Fourier transform (STFT) and then summing up elements of the same pitch class.
In literature, efforts were made to improve chroma features used for chord recognition by, e.g., making them more robust to timbre changes [29], using Non-negative Matrix Factorization (NMF) for prior note transcription [21], reducing negative influences of overtones [23], or refining the frequency resolution by using spectral reassignment [32]. Comprehensive comparisons of various strategies for chroma feature enhancement can be found in [5] and [13]. Both studies showed that the use of different chroma features and pre-filtering methods heavily impacts the results of chord recognition. Especially logarithmic compression and overtone removal by suitably weighting the pitch features were found to be beneficial pre-filtering strategies.
Next to traditional signal processing methods, techniques from machine learning and especially deep-learning have been popularized more recently to extract features from audio recordings.

[4] i.e., {C, C♯, D, D♯, E, F, F♯, G, G♯, A, A♯, B}

In [24] and [39], Convolutional Neural Network (CNN) architectures are used to extract features from CQT-based spectral representations of the audio data. In [18], a Deep Neural Network (DNN) is implemented to extract chroma features from quarter-tone STFT spectrograms. The studies showed an increase in chord recognition quality when using features obtained with deep-learning strategies, compared to traditional signal processing chroma features.

Fujishima [9] used twelve-dimensional, binary, hand-crafted pitch class chord templates to represent different chords.[5] While this implementation is quite demonstrative and can be well-used for didactic purposes [28], it is rarely found in state-of-the-art chord recognition systems. Modern approaches usually rely on some form of data-driven chord models, mostly trained by supervised learning strategies. An intuitive implementation is the averaging of all feature vectors that are equally annotated to create averaged chord templates. More sophisticated methods not only use the average statistics of training data, but also their variance. The most popular choice for such a model are multivariate Gaussian distributions, as used in, e.g., [5, 6, 35]. Each chord model is defined by a mean vector and a covariance matrix, the number of dimensions is usually chosen to match the feature space. This way, chord similarity values for each feature vector can be obtained by evaluating the probability density function of each Gaussian chord model. Mean vector and covariance matrix can easily be calculated directly from the labeled training data. By using multiple, weighted Gaussian distributions for each chord model, Gaussian Mixture Models (GMM) provide a more sophisticated approach with higher capacity to fit the training data. This comes at the cost of a higher computational complexity for training the chord models, which is usually done by using the Expectation Maximization (EM) algorithm, see [26, 35]. In a comprehensive study by Cho and Bello [5], the comparison of Gaussian chord models of various complexity showed no prominent gain in chord recognition effectiveness when using higher-order GMMs. While more complex chord models do provide better results, the differences can be largely offset by a suitable choice of features.

In [9], cosine similarity and Euclidean distance are used as a similarity measure to compare feature vectors with chord templates. The similarity-maximizing template was picked as recognition output for each time frame independently. Since the frame rate is often higher than the chord change rate in music, temporal post-filtering techniques such as median or average filtering can be applied to the similarity representation before picking the output. In modern chord recognition systems, the pattern matching and post-filtering steps are often combined by considering chord sequences instead of singular chords. For this, the most popular method in literature is the implementation of Hidden Markov-models (HMM). Their use for chord recognition was first proposed by Sheh and Ellis [35], inspired by algorithms used in the field of speech recognition. HMMs are used to model chords as hidden states, with sets of initial, transition, and emission probabilities. Feature vectors are viewed as an observable output sequence. In most cases,

[5] e.g., C major chord template: $(1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0)^{\mathrm{T}}$

Viterbi decoding is applied to find the chord (state) sequence with the highest likelihood, see, e.g., [5, 6, 21, 35]. The transition probability set is used to model musical relations between chords. It can be acquired by implementing the EM algorithm, though Cho and Bello [5] showed that the use of an untrained, uniform, diagonal enhanced transition matrix with high self-transition probability values leads to similar chord recognition effectiveness. As an alternative to HMMs, Conditional Random Fields (CRF) with Viterbi decoding are often used for chord sequence decoding, especially when combined with DNN architectures, see, e.g., [19, 22, 39]. Other models that are used in literature to include temporal dependencies within chord sequences are Recurrent Neural Networks (RNN) [24], dynamic Bayesian networks [23], or weighted acyclic harmonic graphs [34].

A good way to comparably measure the quality of automatic chord recognition algorithms is the annual MIREX challenge, as mentioned in Section 2.2. The task is to recognize chords from the *Isophonics* and *Billboard* datasets with five different chord vocabularies of increasing size and complexity. As an evaluation measure, the Chord Symbol Recall (CSR) is used, as described in [33]. It is calculated by dividing the duration of correctly recognized segments by the total duration of annotated segments. The CSR is similar to our recall measure, which we describe in Section 4.5. While we use a quantized time axis based on sampled time frames, the CSR is calculated on a continuous time axis. The best algorithms from the past few years achieved a CSR of around 85% for the more simple chord vocabularies[6] and around 70% for the most complex vocabulary.[7] As previously mentioned, these values are only evaluated on popular music recordings. Hence, there is a lack of common ground for chord recognition in classical music.
This thesis does not represent an effort to further optimize the quality of state-of-the-art automatic chord recognition algorithms, but to offer insights into the characteristics of the analyzed audio recordings and selected algorithmic parameters.

[6] root note only and major/minor only
[7] major/minor + 7^{th} chords + inversions

Chapter 3

Datasets

In this chapter, we describe our two analyzed datasets in detail. This includes an overview of all relevant annotation modalities that are included, a short musical categorization, and statistical information about the chord labeling. Furthermore, we describe the individual chords included in the three different chords vocabularies we use for our experiments.

3.1 Schubert Winterreise Dataset (SWD)

Winterreise (eng. *Winter Journey*, D.911/op. 89) is a song cycle for solo voice and piano, written by Austrian composer Franz Schubert in 1827. It represents one of the most popular examples of the *art song*, a musical genre characteristic for the Romantic period. It consists of 24 songs, the lyrics are based on poems by Wilhelm Müller. The Schubert Winterreise dataset (SWD) is a multimodal dataset, comprising multiple representations of the raw data and several annotations. It is publicly available.[1] The raw data includes sheet music, MIDI[2] representation of the score, lyrics in text format, as well as audio recordings of nine different performances of the full song cycle. The annotations comprise musical keys, score- and audio-aligned chord labels, and structural segmentation. Figure 3.1 a) shows an overview of the different songs and versions of the SWD. The IDs of the nine different versions are `AL98`, `FI55`, `FI66`, `FI80`, `HU33`, `OL06`, `QU98`, `SC06`, and `TR99`, representing short handles of the singers' names and the year of the recording. The song IDs are `D911-01` to `D911-24`. The SWD contains a total number of 216 individual tracks. With the term track, we refer to the recording of a single song in a specific version.

[1] https://zenodo.org/record/3968389#.X93yY9hKiUk
[2] Musical Instrument Digital Interface, symbolic representation of musical data

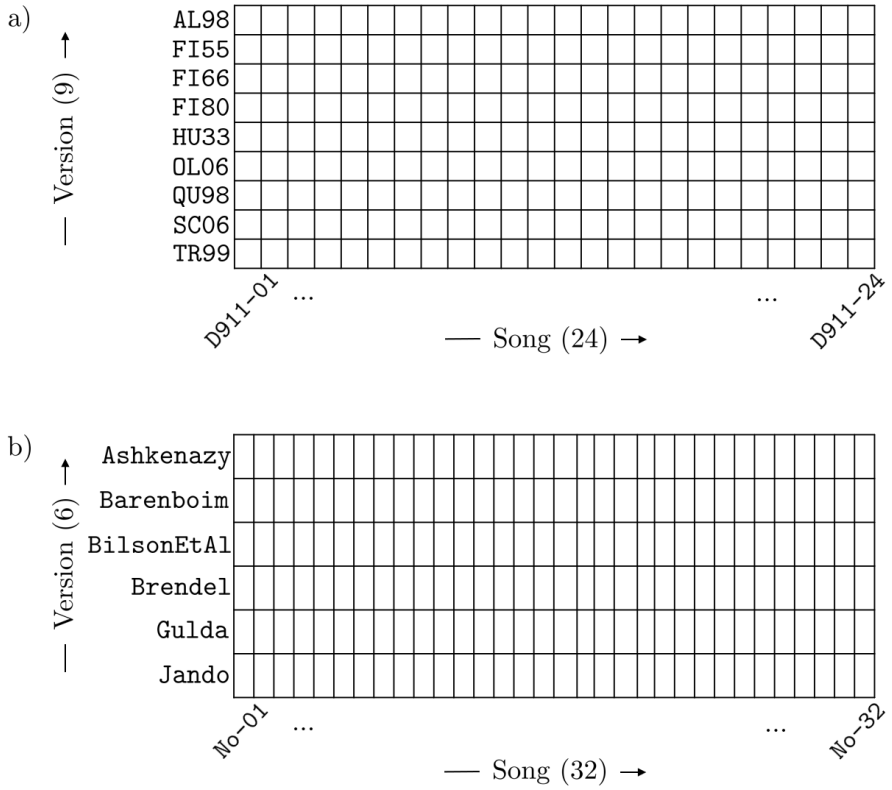


Figure 3.1. Schematic representation of the number and IDs of songs and versions contained in each dataset. a) For SWD. b) For BSD.

In the context of this thesis, we use the term *version* as a descriptor of the different performances. While some of the modalities—such as recording quality and setting, individual tempo, dynamics, or global keys—might differ between the performances, all of them are musically similar and follow the score closely. This makes them comparable on a musical time axis, allowing for cross-version analysis. Furthermore, the instrumentation is always the same, consisting of one singer and one piano.

Table 3.1 shows an exemplary excerpt from the score-aligned chord annotations included in the SWD. The chord labels are given in different granularities, the start and end points are denoted in measure positions. In the latter, the decimals represent musically linear, precise positions within a measure, i.e., the decimals describe different musical beats, depending on the current musical time signature. As an example, in a 4/4 time measure, "10.000" would denote beat 1 of measure 10, "10.500" would denote beat 3. By utilizing manually created measure annotations, the score-aligned chord annotations were aligned to each version's audio using a sophisticated synchronization pipeline. For further insights into the synchronization process and general information about the SWD, we refer to the accompanying journal paper [38]. Table 3.2 shows an example of the audio-aligned chord annotations, specifying start and end time of each chord label in the corresponding version's audio in seconds.

Start	End	Shorthand	Extended	Major/minor	Major/minor+inversion
...	...	...	...	...	...
9.000	9.999	D:hdim7/C	D:(♭3,♭5,♭7)/C	D:min	D:min/C
10.000	10.499	C:min	C:(♭3,5)	C:min	C:min
10.500	10.750	C:min/G	C:(♭3,5)/G	C:min	C:min/G
...	...	...	...	...	...

Table 3.1. Example of score-aligned chord annotations from `D911-01`. The global key corresponds to the original score. Start and end point of the chord segments are given as measure positions.

Start	End	Shorthand	Extended	Major/minor	Major/minor+inversion
...	...	...	...	...	...
19.74	22.08	C:hdim7/A♯	C:(♭3,♭5,♭7)/A♯	C:min	C:min/A♯
22.08	23.3	A♯:min	A♯:(♭3,5)	A♯:min	A♯:min
23.3	23.86	A♯:min/F	A♯:(♭3,5)/F	A♯:min	A♯:min/F
...	...	...	...	...	...

Table 3.2. Example of audio-aligned chord annotations from `D911-01` in version `QU98`. Note that the global key of this particular version deviates from the original score. Start and end point of the chord segments are given in seconds, corresponding to the audio recording.

AL98	FI55	FI66	FI80	HU33	OL06	QU98	SC06	TR99	
1.55	4.20	3.86	2.99	1.75	1.31	**4.87**	2.51	**0.38**	[dB]

Table 3.3. Sound intensity difference between sung and instrumental parts for each version in `D911-01`. Higher values indicate a louder singing voice compared to the piano accompaniment. The highest and lowest values are marked.

Table 3.3 shows a quantification of how loud the singing voice is recorded in comparison to the piano accompaniment. For this, we evaluate the average sound intensity of the audio recordings of `D911-01` for each version. We compare the sound intensity of audio segments containing piano and singing voice to the sound intensity of segments only containing piano notes. This results in some kind of "SNR"[3] value for each version. The higher the value, the louder the singing voice is compared to the piano accompaniment. It can be seen that the loudness difference is biggest in version `QU98`, with a value of 4.87 dB. The smallest difference is found in version `TR99`, with a value of 0.38 dB. It has to be noted that we computed these values using a quite simple approach and only for one out of the 24 songs. Therefore, they might not perfectly represent the actual intensities throughout the complete dataset. Nevertheless, the evaluation offers a rough

indication of the differences between versions. It also corresponds to our subjective impression when listening to the different recordings.

The SWD represents an effort to create an extensive, precisely annotated, classical music dataset. Despite it being published recently, numerous researchers have used the underlying data for their publications, further developing and contributing to the SWD. We refer to the harmonic analysis by Absil [1], providing the basis for the annotations. Further, we refer to Koops [16], Grohganz [10], and Brütting [2], exploring harmonic and structural characteristics of the SWD.

3.2 Beethoven Piano Sonatas Dataset (BSD)

The German composer Ludwig van Beethoven wrote 32 piano sonatas between the years 1795 and 1822. These works represent highly influential pieces of classical piano music literature. In contrast to Schubert's Winterreise, the sonatas have been composed over a period of 27 years and are therefore more musically heterogeneous. While the first pieces can be classified as typical examples of sonatas from the Classical period, later works exhibit harmonical and structural similarities with music from the Romantic period. The Beethoven piano sonatas dataset (BSD) comprises audio recordings and annotations for six different performances of the first movements of all 32 sonatas. This means that the BSD includes a total of 192 individual tracks. Figure 3.1 b) shows an overview of the songs and versions contained within the BSD. While denoting the movements as "songs" is inaccurate from a musicological standpoint, we use the term to achieve a consistent terminology. The version IDs are `Ashkenazy`, `Barenboim`, `BilsonEtAl`, `Brendel`, `Gulda`, and `Jando`, named after the performing pianists. The song IDs are `No-01` to `No-32`. The song IDs are in chronological order.

The BSD is not publicly available in its entirety. For the audio recordings and measure annotations we refer to Jiang et al. [14]. The score-based chord annotations are published by Chen et al. [4] and can be accessed freely.[4] The latter were used to create the audio-aligned chord annotations used in in our experiments. The synchronization process was similar to the one used for the SWD. The annotation style is the same for both datasets, an example is given in Tables 3.1 and 3.2.

In contrast to the SWD, some of the versions contain a different number of repetitions in certain songs, namely `No-01`, `No-02`, and `No-06`. This makes the comparison of versions on a musical time axis more complicated for these songs. Apart from that, all versions follow the same structure in the remaining songs. The instrumentation is always the same, consisting of a solo piano. The total duration of the audio recordings of all versions combined amounts to roughly 23 hours, which is over double the duration of the SWD at roughly eleven hours. The average

[3] Signal-to-noise ratio

[4] https://github.com/Tsung-Ping/functional-harmony

playtime of a single track in the BSD is 7 min 4 s, the average track duration in the SWD is 2 min 58 s.

3.3 Chord Annotations and Vocabularies

For this thesis, we use the chord annotations described above as a ground truth for the evaluation of different chord recognition methods. As labels, we are using the "Extended" and "Major/minor" columns in the annotations, as seen in Tables 3.1 and 3.2. To provide an insight into the annotation styles which are included in the datasets, we also show the remaining columns. We process the annotations in three different granularities, resulting in references for three different chord vocabularies. We refer to these vocabularies as **major/minor**, **triad**, and **seventh** from here on out. For the major/minor vocabulary we use the "Major/minor" column, for triad and seventh we use the "Extended" column. The "Major/minor" style is mostly derived from the third note of the "Extended" column, but in some cases no third note is present. In these cases, further musical context was used to assign the chord quality for the "Major/minor" column. With the term quality, we refer to the chord type, such as, e.g, major, minor, or diminished.
The major/minor vocabulary contains 24 different chords, comprised of one major and one minor chord for each of the twelve different root notes from C to B. We denote major chords with the suffix capital **M**, e.g., a C major chord is referred to as CM. We symbolize minor chords with a small **m**. The triad vocabulary contains 40 different chords, extending the major/minor vocabulary with diminished and augmented chords. It contains twelve additional diminished chords, denoted by the suffix **dim**. Since there are only four distinguishable augmented chords when utilizing enharmonically equivalent pitch classes,[5] we include only the augmented chords for the root notes C, C♯, D, and D♯ in the vocabulary. We denote them with the suffix **aug**. The seventh vocabulary is the largest and most complex vocabulary we use in our experiments, containing 91 different chords. Within it, we include the complete triad vocabulary and extend it with 51 different 7^{th} chords, split up in five different qualities. The first quality is the dominant seventh chord, denoted by the suffix **M7**. It consists of a major triad with an added minor seventh note. Next is the minor seventh chord, consisting of a minor triad with an added minor seventh note. We symbolize it with the suffix **m7**. By adding a major seventh note to a major triad, the major seventh chord is obtained. We denote it with the suffix **maj7**. The half-diminished seventh chord is constructed by adding a minor seventh note to a diminished triad. We symbolize it with the suffix **hdim7**. The last chord quality we include in the seventh vocabulary is the diminished seventh chord, constructed by adding a diminished seventh note on top of a diminished triad. It can also be obtained by stacking three minor triads on the root note. We denote it with the suffix **dim7**. There are only three distinguishable dim7 chords,[6] so

[5] e.g., Caug contains the same notes as Eaug and G♯aug
[6] e.g., Cdim7 contains the same notes as D♯dim7, F♯dim7, and Adim7

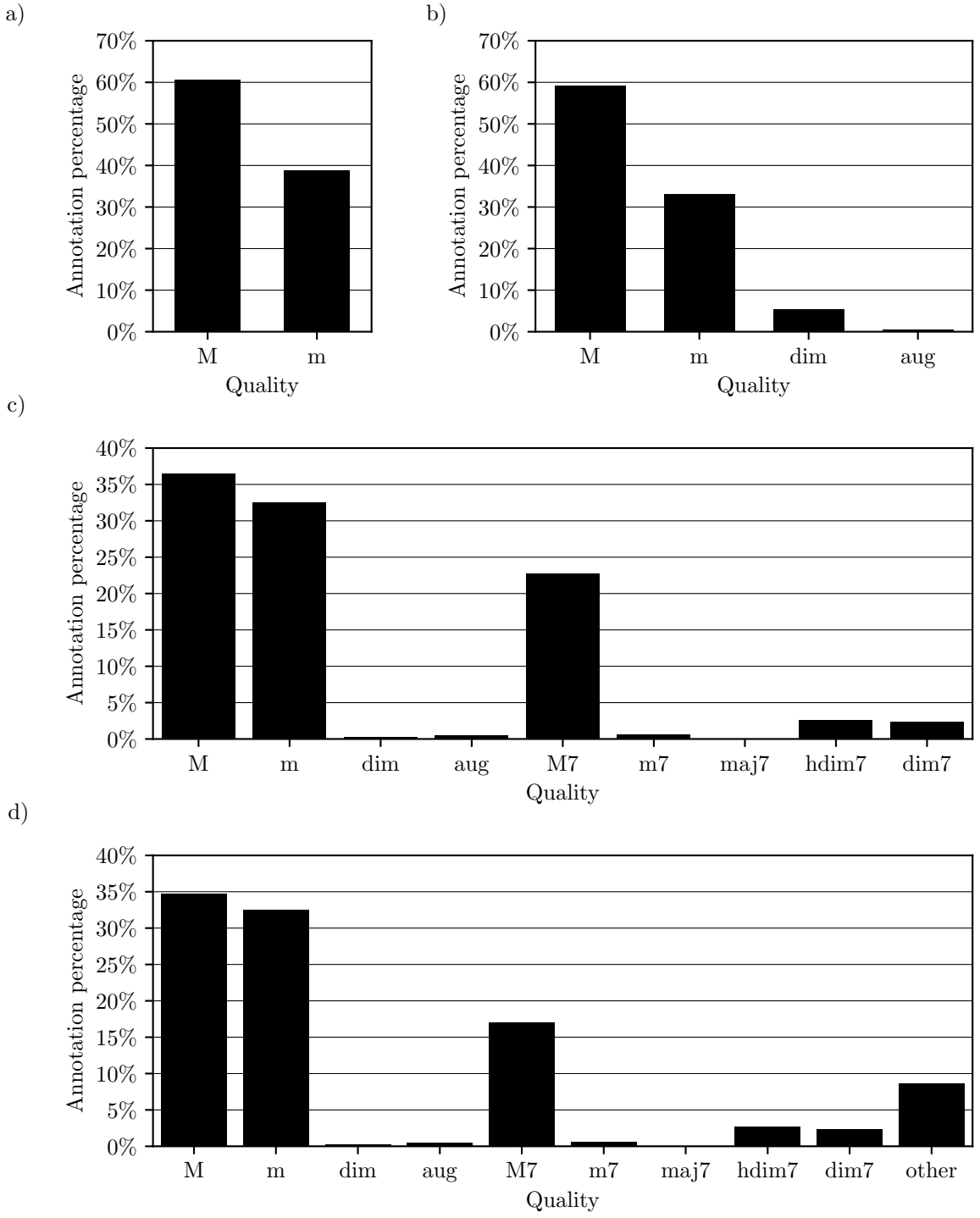


Figure 3.2. Distribution of chord quality annotations for SWD. a) Major/minor vocabulary. b) Triad vocabulary. c) Seventh vocabulary. d) Seventh vocabulary without any mapping or reduction.

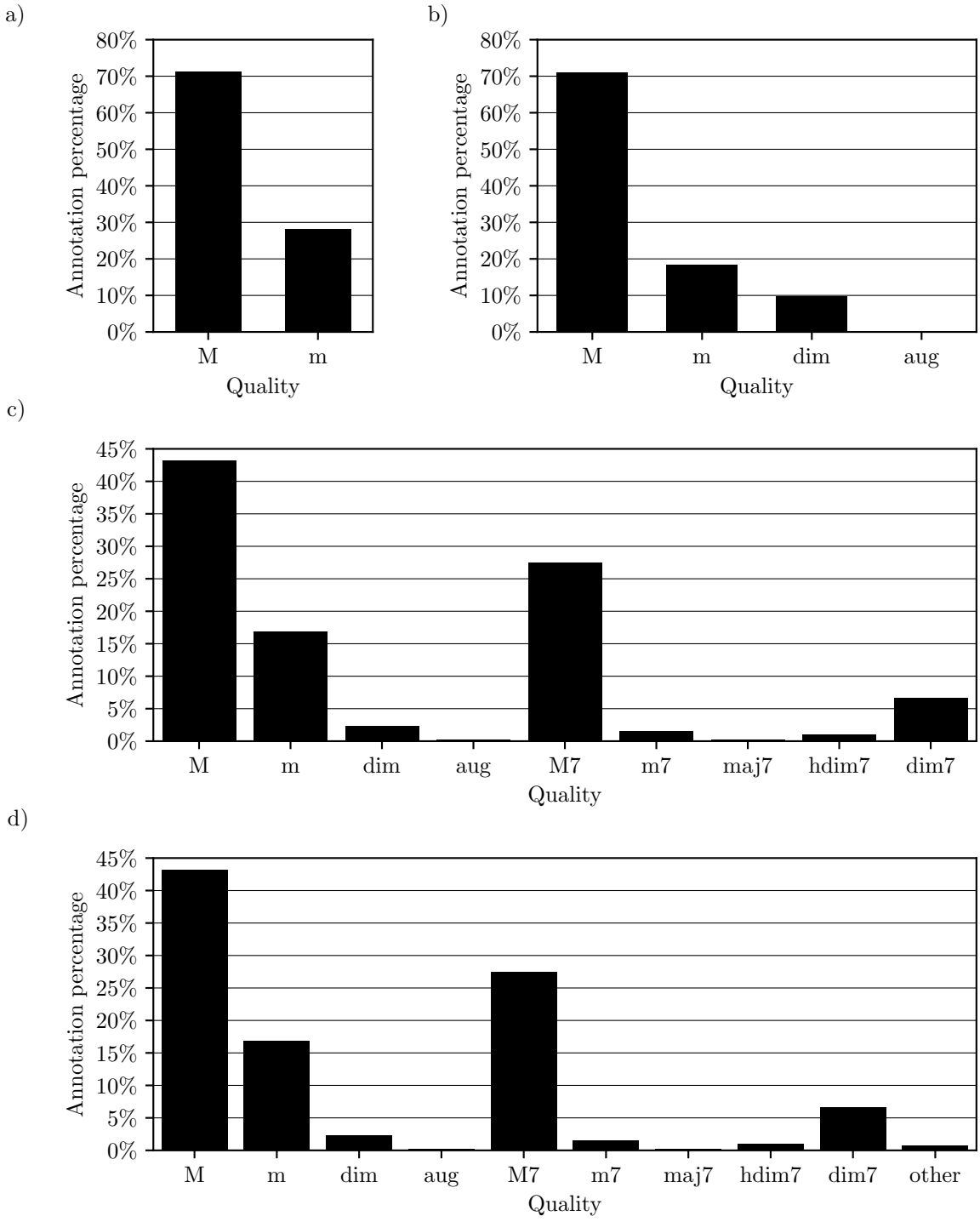


Figure 3.3. Distribution of chord quality annotations for BSD. a) Major/minor vocabulary. b) Triad vocabulary. c) Seventh vocabulary. d) Seventh vocabulary without any mapping or reduction.

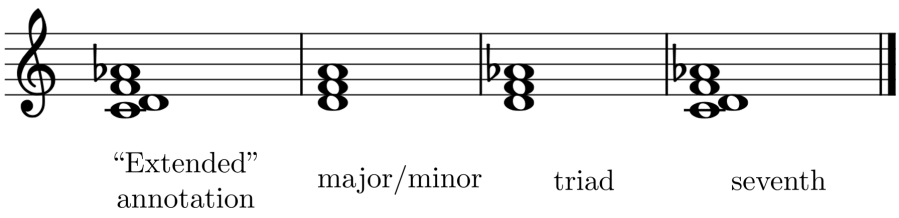


Figure 3.4. Different levels of chord reduction and mapping we apply for our three chord vocabularies. An annotated "D:(b3, b5, ♭7)" chord is mapped to Dm for major/minor, reduced to Ddim for triad, and represented accurately as Dhdim7 for seventh vocabulary.

we only include the dim7 chords for the root notes C, C♯, and D. Note that in the major/minor and triad vocabulary, we use only triads, i.e., chords consisting of three different notes. The additional chords we use in the seventh vocabulary comprise four different notes. For creating the reference chord labels for the triad and seventh vocabulary, we parse the "Extended" column from the annotations. It specifies the chords as described by Harte et al. [11]. The root note is directly given and the quality is implied by denoting the included chord notes in terms of intervals. As an example, a Cdim7 chord is given as "C:(b3, b5, 𝄫7)."

When parsing the annotations, we implement different types of musical mapping and reduction of chord labels. For the major/minor vocabulary, we classify all chords containing a major third note as major, all chords containing a minor third note as minor. This means, chords constructed of diminished triads are mapped to minor quality and chords constructed of augmented triads are mapped to major quality. For the triad vocabulary, the triad quality of a chord is classified accurately, but we reduce additional chord notes. In the seventh vocabulary, we reduce any chord notes except triad notes and the 7^{th} interval, such as 9^{th} or 11^{th} intervals. In Figure 3.4 we show an example for a "D:(b3, b5, ♭7)" chord.
In Figures 3.2 and 3.3, we show statistics of the distribution of different chord qualities within each vocabulary and dataset. Subfigures a) show the distribution in the major/minor vocabulary, b) the distribution for the triad vocabulary, and c) the distribution for the seventh vocabulary. The percentages show the share of each chord quality in terms of the total duration of the dataset, not the occurrences of each label in the annotated segments (which differ in duration). Subfigures d) show the "strict" distribution of all considered chord qualities when there is no mapping or reduction applied. We classify chords as "other," which are not labeled specifically as one of the qualities we consider within the three vocabularies. This means that the comparison of Subfigures a), b), and c) with d) shows the corresponding level of mapping or reduction we implement. In Appendix A, we show statistics of the occurrence of each individual chord in each dataset.

Chapter 4

Chord Recognition Approaches

In the following, we give a technical description of the different approaches for automatic chord recognition we use in our experiments. We introduce the underlying methods, important parameters, and evaluation metrics. The structure of this chapter is inspired by the order of implementation.

4.1 Feature Extraction

As a first step, we extract feature vectors from the audio recordings. In Section 2.3, we give an overview of commonly used feature types. For this thesis, we implement and compare six different **chroma feature** types, denoted as $\mathcal{C}_{\text{CQT}}$, $\mathcal{C}_{\text{STFT}}$, $\mathcal{C}_{\text{IIRT}}$, $\mathcal{C}_{\text{deep}}$, $\mathcal{C}_{\text{score}}$, and $\mathcal{C}_{\text{annot}}$. The first four are "real" features, actually extracted from the audio recordings. The latter two are baseline features, taken from symbolic representations of the input data and used for comparison. $\mathcal{C}_{\text{CQT}}$, $\mathcal{C}_{\text{STFT}}$, and $\mathcal{C}_{\text{IIRT}}$ are based on log-frequency spectrograms extracted with traditional signal processing methods, obtained by using the time-frequency transforms STFT, CQT, and a transformation based on infinite impulse response filterbanks (IIRT). These log-frequency spectrograms capture spectral energy according to the MIDI pitches of the twelve-tone equal temperament. The center frequency $f_{\text{pitch}}(p)$ of a MIDI pitch $\{p \in \mathbb{N} \mid 0 \leq p \leq 127\}$ is

$$f_{\text{pitch}}(p) = 2^{(p-69)/12} \cdot 440\,\text{Hz}. \tag{4.1}$$

For a time frame $n \in \mathbb{Z}$, $\mathcal{P}(n, p)$ describes the salience of a pitch band in the audio recordings. It can be obtained from CQT and IIRT directly, as these transforms use logarithmic frequency axes. For STFT, which produces linearly spaced frequency coefficients, we have to bin the coefficients accordingly. Note that for lower pitch bands, fewer STFT coefficients are available. For a more detailed description of the transforms we refer to [27, 28]. We refer to the pitch-like features

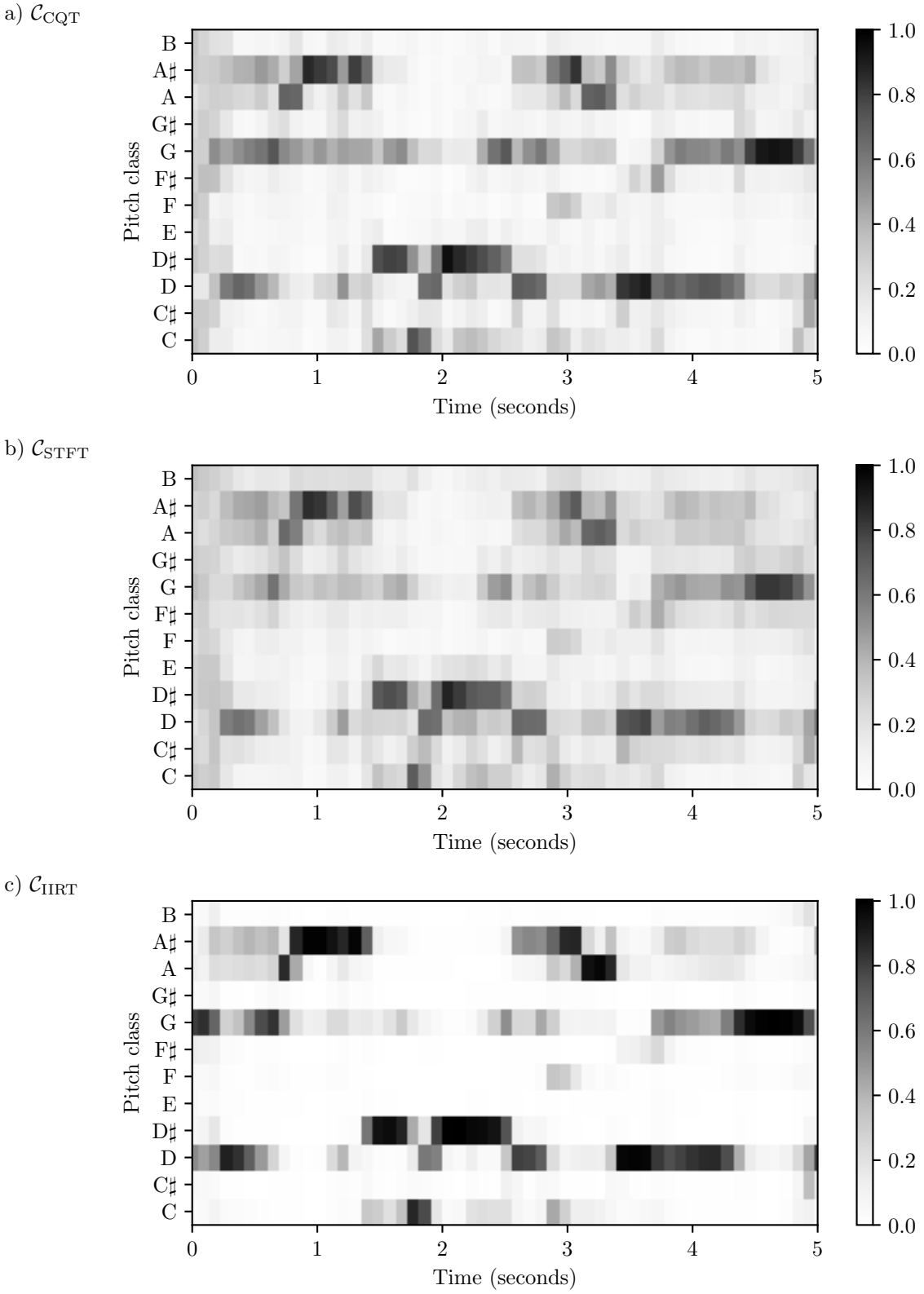


Figure 4.1. Normalized chroma features of the first five seconds of song `D911-22`, version `AL98`. a) $\mathcal{C}_{\text{CQT}}$. b) $\mathcal{C}_{\text{STFT}}$. c) $\mathcal{C}_{\text{IIRT}}$.

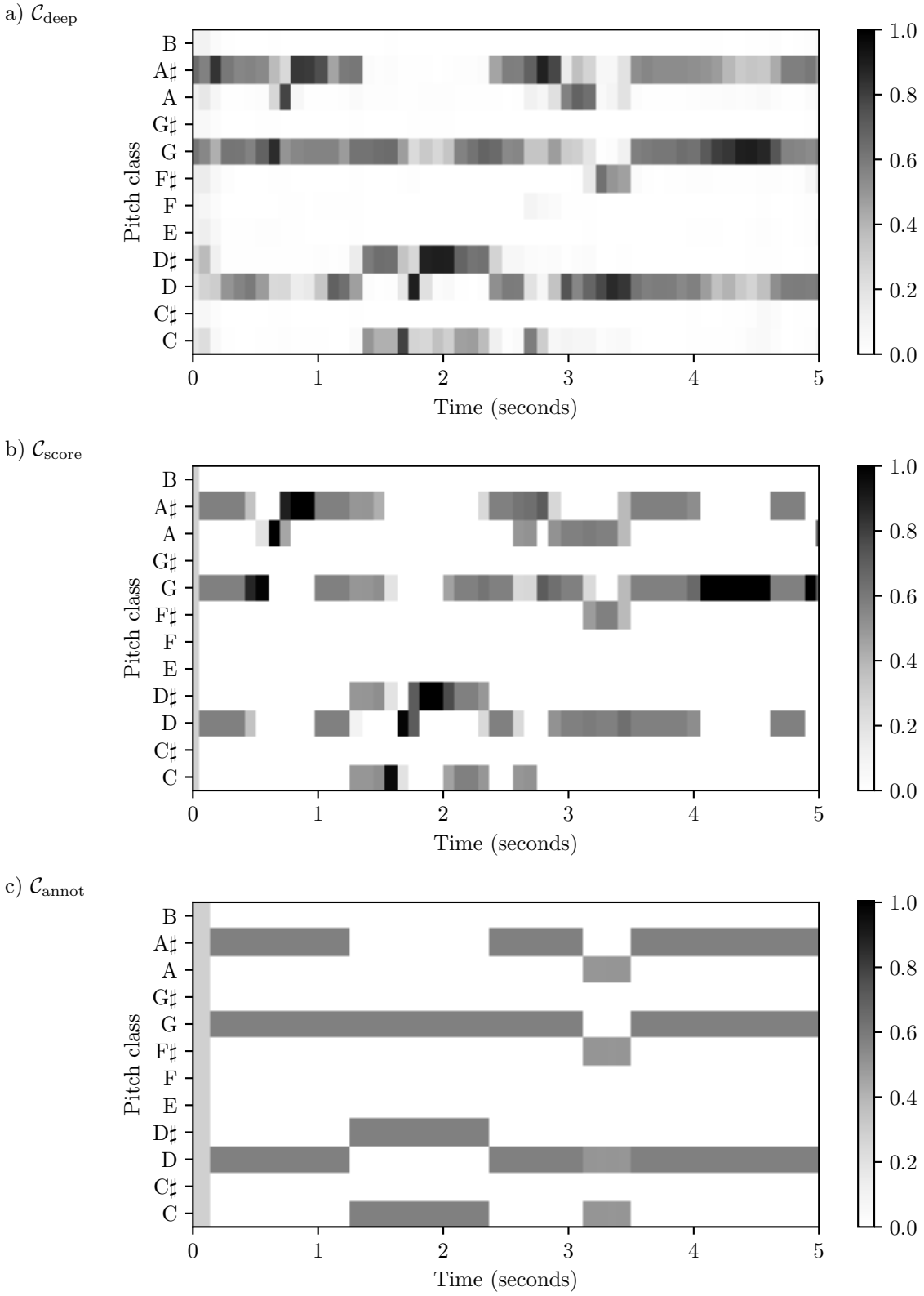


Figure 4.2. Normalized chroma features of the first five seconds of song `D911-22`, version `AL98`. a) $\mathcal{C}_{\text{deep}}$. b) $\mathcal{C}_{\text{score}}$. c) $\mathcal{C}_{\text{annot}}$.

obtained from the CQT, STFT, and IIRT as $\mathcal{P}_{\text{CQT}}$, $\mathcal{P}_{\text{STFT}}$, and $\mathcal{P}_{\text{IIRT}}$, respectively. For all three transforms we use a hop size of 2048 samples, resulting in a feature rate of roughly 10.8 Hz for the input sample rate of 22 050 Hz. We need to choose a power of two as hop size for the CQT, for comparability we use the same hop size for all three variants. For STFT and IIRT we use a window size of 4096 samples. The window size for CQT is chosen automatically as a function of the respective pitch bands. To account for a possible deviation from the standard tuning at 440 Hz, we perform tuning estimation to adjust the frequency coefficients of the three transforms accordingly. We choose a number of three bins per semitone for the CQT to achieve better pitch band separation. For both CQT and IIRT we restrict the MIDI pitches to a range of 24–108, which corresponds to a range of 7 octaves from C1 to C8 in Western pitch notation. With STFT we set all frequency coefficients corresponding to pitch bands outside this range to zero.

While the pitch-like features represent the salience of individual pitch bands in the audio, chroma features sum up this information for the set of twelve individual **pitch classes** {C, C♯, ..., B} by discarding the octave information. For this, we sum up the values of all pitch coefficients corresponding to the same pitch class. Doing this for all time frames n results in the chromagram $\mathcal{C}(n, c)$ with pitch class $\{c \in \mathbb{N} \mid 0 \leq c \leq 11\}$. Analogously to the pitch-like features, we refer to the chroma features obtained from STFT, CQT, and IIRT as $\mathcal{C}_{\text{CQT}}$, $\mathcal{C}_{\text{STFT}}$, and $\mathcal{C}_{\text{IIRT}}$, respectively. For the extraction of $\mathcal{C}_{\text{CQT}}$, $\mathcal{C}_{\text{STFT}}$, and $\mathcal{C}_{\text{IIRT}}$ from the audio data, we use the Python implementation of `librosa` [25].

$\mathcal{C}_{\text{deep}}$ are chroma features extracted by means of deep-learning techniques. The architecture is based on a musically motivated CNN, which uses a Harmonic CQT (HCQT) of the audio recordings as input data. It is trained with over 150 hours of classical music data. The training data comprises the Saarland Music Data [30], the MusicNet database [36], the MAPS dataset [8], a dataset featuring three versions of Wagner's *Der Ring des Nibelungen* [20], and unpublished solo piano recordings. Additionally, both the SWD and the BSD are used for training. For the extraction of chroma features for the SWD, the SWD itself is omitted from the training data. The same is true for the BSD. For training $\mathcal{C}_{\text{deep}}$, symbolic, binary chroma features are used, similar to $\mathcal{C}_{\text{score}}$. A detailed description of the chroma extractor and training strategies is given in Zeitler [40], which is based on the work of Zunner [41]. $\mathcal{C}_{\text{deep}}$ for both datasets is available to us with a feature rate of 10 Hz.

The two baseline feature types we use next to the audio-based chroma features are denoted as $\mathcal{C}_{\text{score}}$ and $\mathcal{C}_{\text{annot}}$. We compute $\mathcal{C}_{\text{score}}$ from the audio-aligned pitch annotations that are included in the datasets, generated from symbolic scores. $\mathcal{C}_{\text{score}}$ can be seen as a baseline "perfect" chroma. We compute $\mathcal{C}_{\text{annot}}$ by converting the chord annotations into chroma vectors. For this, we use the labels from the "Extended" annotation style (see Table 3.2), root note and interval notes indicate the corresponding pitch classes. Note that we do not implement mapping or reduction of the annotations for the computation of $\mathcal{C}_{\text{annot}}$. This means that $\mathcal{C}_{\text{annot}}$ differs from the ground

truth we use for the evaluation of each chord vocabulary. We use $\mathcal{C}_{\text{annot}}$ to evaluate how well our chord recognizers can derive the chord labels of each vocabulary from the exact "Extended" annotations. We parse both baseline chromas with a feature rate of 10 Hz, the same as $\mathcal{C}_{\text{deep}}$. The annotation data for $\mathcal{C}_{\text{score}}$ is given with feature rate of 50 Hz in the dataset. To avoid aliasing issues, we apply moving mean filtering prior to the downsampling to 10 Hz. This means that $\mathcal{C}_{\text{score}}$ does not maintain its binary form after processing. All six chroma types are ℓ^2-normalized to eliminate dynamic differences between time frames.
Figures 4.1 and 4.2 show a comparison of all six chroma types for the first five seconds of song `D911-22` in version `AL98`. Figure 4.1 shows a comparison of the three signal processing chromas. We can see that $\mathcal{C}_{\text{IIRT}}$ seems to provide the best separation between pitch classes and is least prone to noisy components. It is similar to the perfect chroma $\mathcal{C}_{\text{score}}$. $\mathcal{C}_{\text{CQT}}$ and especially $\mathcal{C}_{\text{STFT}}$ seemingly contain more noise components, but still capture the correct notes. In Figure 4.2 we can see that $\mathcal{C}_{\text{deep}}$ is highly similar to $\mathcal{C}_{\text{score}}$. Subfigures 4.2 b) and c) show the effect of normalizing chroma vectors with a different number of equally valued entries. $\mathcal{C}_{\text{score}}$ and $\mathcal{C}_{\text{annot}}$ do not maintain their initial, binary form after processing.

4.2 Pre-Filtering

Pre-filtering refers to various enhancement methods that we use to improve the features. For our experiments, we implement and adjust three different pre-filtering strategies. The three methods are **logarithmic compression**, **pitch weighting**, and **median filtering**.
As the name suggests, logarithmic compression is a signal processing method to compress the dynamic range of feature values. The motivation of applying compression is similar to the idea of using normalization to eliminate dynamic differences between feature vectors of different time frames. For chord recognition, it is not relevant how loud a certain note is played in the audio, but only which notes are played at all. We apply logarithmic compression by computing

$$\mathcal{P}^{\log}(n, c) = \log\left(1 + \gamma \cdot \mathcal{P}(n, c)\right) \tag{4.2}$$

$$\mathcal{C}^{\log}(n, c) = \log\left(1 + \gamma \cdot \mathcal{C}(n, c)\right) \tag{4.3}$$

with $\gamma \in \mathbb{R}_{>0}$, for pitch or chroma features, respectively. For $\mathcal{C}_{\text{CQT}}$, $\mathcal{C}_{\text{STFT}}$, and $\mathcal{C}_{\text{IIRT}}$, we apply logarithmic compression to the pitch features, for $\mathcal{C}_{\text{deep}}$ we apply it to the chroma features directly. The variable γ is the parameter controlling the level of compression. Higher γ means stronger compression, reducing relative differences between strong and weak components in the features. We weight the pitch-like features in our range from MIDI pitch 24–108 with a Gaussian window, centered at MIDI pitch 60 (C4) with a standard deviation of 15 pitches. The idea is to emphasize the frequency range where the main harmonic content is expected to be located, while attenuating frequencies at the extremes of the considered pitch range. We apply pitch weighting

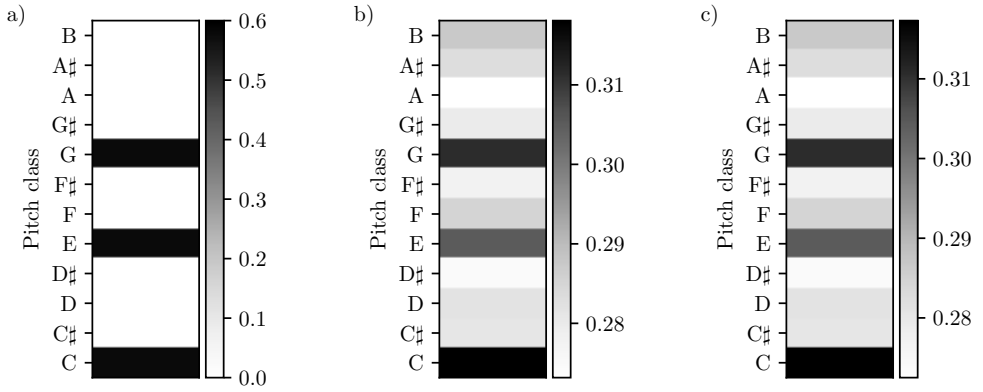


Figure 4.3. Examples of the three different chord models: binary templates, averaged templates and mean vectors of the Gaussian models. a) Normalized $\mathbf{t}_{\mathrm{b}}^{\mathrm{CM}}$. b) Normalized $\mathbf{t}_{\mathrm{a}}^{\mathrm{CM}}$. c) $\boldsymbol{\mu}^{\mathrm{CM}}$. We acquired b) and c) from the SWD with major/minor vocabulary, $\mathcal{C}_{\mathrm{CQT}}$, and $\gamma = 10^6$.

after logarithmic compression for $\mathcal{C}_{\mathrm{CQT}}$, $\mathcal{C}_{\mathrm{STFT}}$, and $\mathcal{C}_{\mathrm{IIRT}}$.
As a last pre-filtering step, we implement temporal smoothing by means of moving median filtering. The idea behind this is the reduction of irrelevant short-time fluctuations in the chroma features, such as signal processing or recording artifacts. We apply the moving median filter

$$\mathcal{C}^{\mathrm{filt}}(n,c) = \mathrm{median}\left\{\mathcal{C}(i,c), \mathcal{C}(i+1,c), ..., \mathcal{C}(j,c)\right\}, \qquad (4.4)$$
$$i = n - \left\lfloor \frac{l_{\mathrm{filt}} - 1}{2} \right\rfloor, j = n + \left\lceil \frac{l_{\mathrm{filt}} - 1}{2} \right\rceil$$

directly to the chroma features, with filter length $l_{\mathrm{filt}} \in \mathbb{Z}$. Finally, we normalize all chroma features with respect to the ℓ^2-norm.
We discuss the effect of the individual pre-filtering strategies in Section 5.1, including variation and optimization of the parameters for logarithmic compression γ and median filtering l_{filt}.

4.3 Chord Models

Algorithms for chord recognition rely on models that describe the chords to be recognized in the feature space. The exact representation is usually influenced by the form of features that are used. It is possible to use hand-crafted, musically informed chord templates as well as data-driven, trained templates. In this thesis, we implement three different chord models.
The first variant are **binary templates**. They are hand-crafted, twelve-dimensional vectors that each describe a specific chord. The individual entries represent the twelve pitch classes and are set to 0 or 1, according to the respective chord notes. As an example, the template vector

$$\mathbf{t}_{\mathrm{b}}^{\mathrm{CM}} = (1,0,0,0,1,0,0,1,0,0,0,0,0)^{\intercal} \qquad (4.5)$$

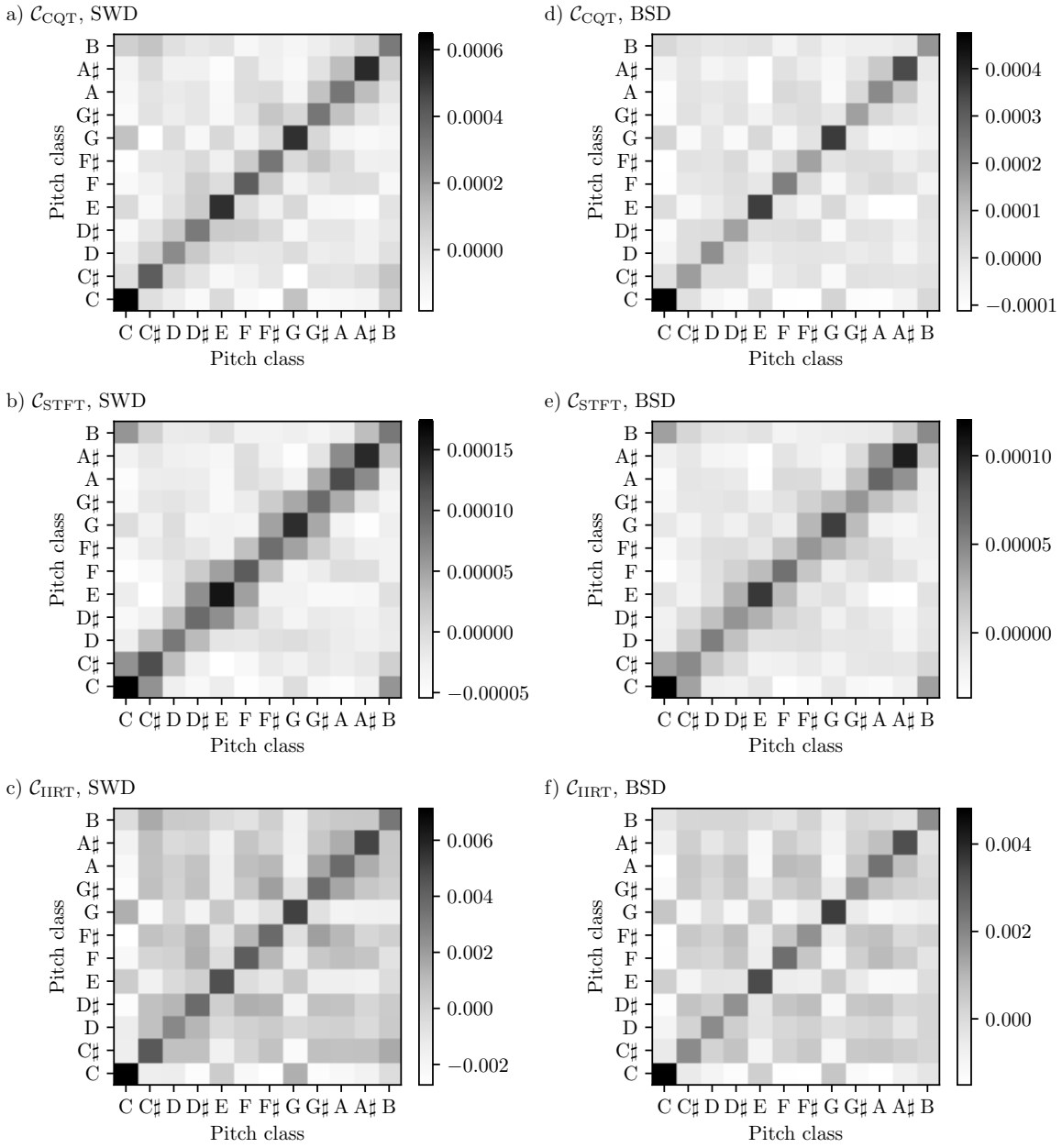


Figure 4.4. Visualization of $\mathbf{\Sigma}^{\mathrm{CM}}$ for different chroma features and datasets. a) SWD with $\mathcal{C}_{\mathrm{CQT}}$. b) SWD with $\mathcal{C}_{\mathrm{STFT}}$. c) SWD with $\mathcal{C}_{\mathrm{IIRT}}$. d) BSD with $\mathcal{C}_{\mathrm{CQT}}$. e) BSD with $\mathcal{C}_{\mathrm{STFT}}$. d) BSD with $\mathcal{C}_{\mathrm{IIRT}}$. The covariance matrices are trained for the major/minor vocabulary with $\gamma = 10^6$.

describes the binary template for the C major chord, containing ones for the C, E, and G pitch classes. We denote the binary templates as $\mathbf{t}_{\mathrm{b}}^{(\mathrm{chord})}$. Figure 4.3 a) shows a visualization of the $\mathbf{t}_{\mathrm{b}}^{\mathrm{CM}}$ template. Before calculating the similarity measure which we describe in the following section, the binary templates are normalized with respect to the ℓ^2-norm.

Secondly, we use **averaged templates**. They rely on labeled training data and present a data-driven approach. We obtain the averaged templates by calculating the mean of each pitch

class of all feature vectors that are labeled as the same chord. Instead of training the templates for each chord individually, we only train the C chord model for each quality. Accordingly, we cyclically shift all feature vectors labeled with that quality to the root note C. After training, we again cyclically shift the C chord template to obtain the templates for the remaining root notes. This strategy ensures that each chord template of the same quality receives the same amount of training, indifferent of occurrences within the training data. It also augments the training data for individual chords. The templates are trained on the already normalized feature vectors. We denote the averaged templates as $\mathbf{t}_{\mathrm{a}}^{(\mathrm{chord})}$. Figure 4.3 b) shows a visualization of the $\mathbf{t}_{\mathrm{a}}^{\mathrm{CM}}$ template. Before calculating similarity measures, the averaged templates are normalized with respect to the ℓ^2-norm.

As a third method, we implement **Gaussian models**. While the averaged templates are trained with the mean statistics of the feature vectors, Gaussian models additionally describe their variance statistics. Each chord model is represented by a twelve-dimensional, multivariate Gaussian distribution, characterized by a mean vector $\boldsymbol{\mu}^{(\mathrm{chord})} \in \mathbb{R}_{>0}^{12}$ and a covariance matrix $\boldsymbol{\Sigma}^{(\mathrm{chord})} \in \mathbb{R}^{12\times 12}$. We acquire the mean vectors in the same way as the averaged templates. For $\mathcal{C}_{\mathrm{CQT}}$, $\mathcal{C}_{\mathrm{STFT}}$, $\mathcal{C}_{\mathrm{IIRT}}$, $\mathcal{C}_{\mathrm{deep}}$ they are in fact identical. In both cases, we train on the already normalized feature vectors. $\mathcal{C}_{\mathrm{score}}$ and $\mathcal{C}_{\mathrm{annot}}$ both contain a large number of zero entries, which leads to numerical problems when computing the covariance matrix. For that reason, we add a small random value to the chroma features before training the Gaussian models. This means that the averaged templates and the Gaussian mean vectors slightly differ for $\mathcal{C}_{\mathrm{score}}$ and $\mathcal{C}_{\mathrm{annot}}$. We obtain the covariance matrix $\boldsymbol{\Sigma}^{(\mathrm{chord})}$ for each model by computing the covariance of all 12 pitch classes across the entire training data, treating each pitch class as a random variable and each feature vector as an observation. Again, we do this only once for the C chord of each quality, followed by a cyclic shift to acquire the chord models for the remaining root notes. Figure 4.3 c) shows a visualization of $\boldsymbol{\mu}^{\mathrm{CM}}$. In contrast to the binary and averaged templates, the mean vectors and covariance matrices of the Gaussian models are not normalized. Figure 4.4 shows a visualization of $\boldsymbol{\Sigma}^{\mathrm{CM}}$ for different chroma features and datasets, trained for the major/minor vocabulary with $\gamma = 10^6$.

4.4 Pattern Matching, Post-Filtering and Output

As the final step of most chord recognition pipelines, a pattern matching stage is implemented, combined with metrics to pick the final output for each time frame. During or after the matching stage, we use temporal smoothing strategies, referred to as post-filtering.

For pattern matching, we use two different implementations for our experiments. These are inspired by the chord model that is used for the respective method. For the approach with binary and averages templates, we use a similarity measure based on the **inner product of normalized vectors**. For each time frame, we compute the similarity of each considered chord

template with the current feature vector. For the approach with Gaussian models, we evaluate the **probability density function** of the Gaussian distribution defined by the respective chord model. This results in a value for each chord model and feature vector of each time frame. For both metrics, higher values signify a higher similarity between respective model and feature vector.

To decide upon the final output of the chord recognition, an intuitive approach would be to simply pick the chord corresponding to the model with the **maximum similarity** for each time frame. We also implement this method, but only use it as a baseline comparison. It does not include any consideration of the relationships between time frames. Our main approach is based on HMMs, describing chords as latent state variables, with **Viterbi decoding** to directly determine the output chord label sequence. In the HMM approach, we use the feature vectors as observation sequence. We model the underlying chords as hidden states. As initial state probabilities, we use a uniform distribution across all considered chords. The similarity values we acquire in the pattern matching stage are seen as emission probabilities. For the transition probabilities, which model the likelihood of transitioning from one chord to another, we use uniform, diagonal enhanced matrices, following the findings of Cho and Bello [5]. These matrices contain constant, high values on the main diagonal, which represent the self-transition probability of each state. We set the remaining transition probabilities to a constant, low value. Each row and column in the self-transition probability matrix sums up to 1. The parameter $\{p_{\text{self}} \in \mathbb{R} \mid 0 \leq p_{\text{self}} \leq 1\}$ describes the self-transition probability and controls the likelihood of each state transitioning to itself. In our scenario, this represents the tendency of our chord recognition system to stay on the same output for successive time frames. Finally, we use Viterbi decoding to find the chord (state) sequence, that fits the features (observation sequence) in the best way. This sequence contains a chord for each time frame. We use the corresponding chord labels as output. For an in-depth description of HMMs in the context of chord recognition, we refer to [5, 28].

The use of HMMs and Viterbi decoding introduces a relationship between successive time frames and can be seen as a form of post-filtering. To denote this variant, we use $\texttt{HMM}_{(\text{chordmodel})}$, jointly describing the chord model, pattern matching, and output decision. With $\texttt{HMM}_{\text{b}}$, we describe the use of binary templates, inner product as similarity measure, and Viterbi decoding. $\texttt{HMM}_{\text{a}}$ describes the same with averaged templates. Furthermore, we use $\texttt{HMM}_{\text{G}}$ to denote the method using Gaussian models, probability density function as similarity measure, and Viterbi decoding. With `BT` we describe the baseline method, using binary templates, inner product, and maximum similarity without HMMs.

4.5 Evaluation

To assess the effectiveness of our chord recognition methods, we compare the recognition results with the chord annotations. To ensure comparability, we parse the audio-aligned annotations

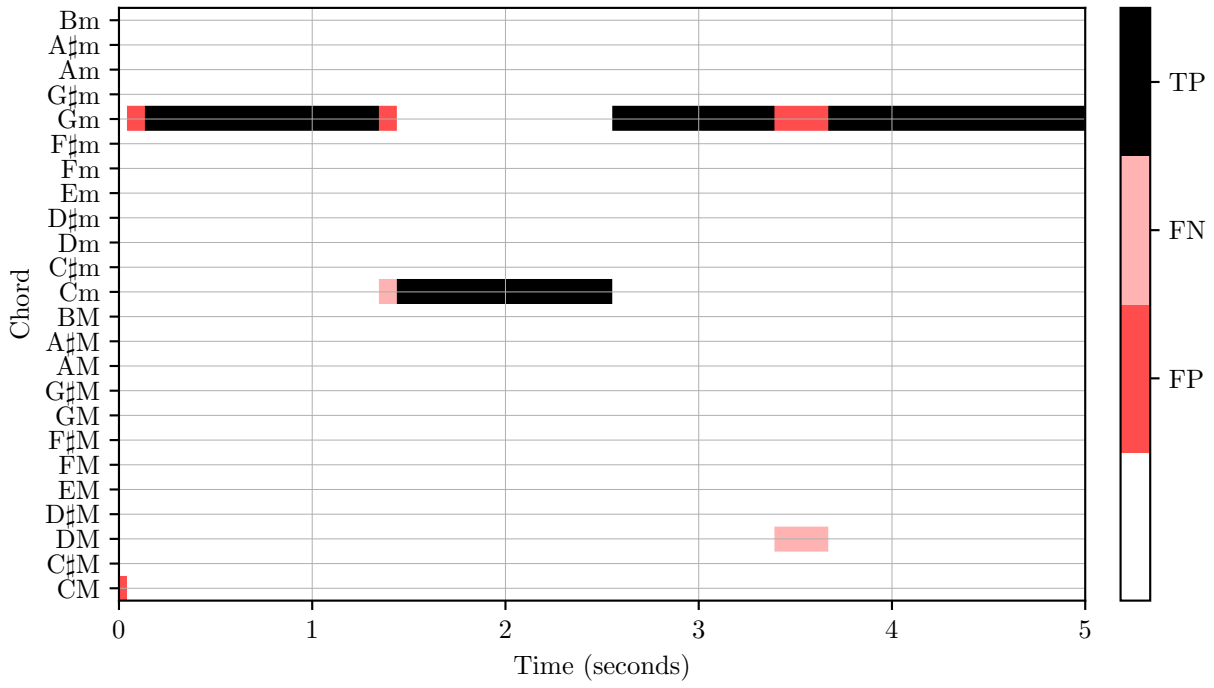


Figure 4.5. Frame-wise evaluation measures, with the recognition result from $\texttt{HMM}_\text{G}$ for the first five seconds of `D911-22` in version `AL98`.

(given with start and end time in seconds) on a frame level with the same sample rate as the input features. For each frame, we classify a correctly identified chord label as **true positive (TP)**, wrongfully identified labels as **false positive (FP)**, and we classify reference labels that are not identified by the recognizer as **false negative (FN)**. Figure 4.5 shows an example of the frame-wise evaluation, with the recognition result from $\texttt{HMM}_\text{G}$, using $\mathcal{C}_\text{CQT}$. We show the first five seconds of `D911-22` in version `AL98`.

The chord recognition methods we use do not implement a "no chord" label. This means that time frames, where no chord label is annotated, by default produce a FP, but not a FN. We can see an example for this at the beginning of the plot in Figure 4.5. The first chord is annotated with a start time of 0.22 s, corresponding to the silence at the beginning of the recording. Since our main interest lies in the correct identification of actually annotated chords, we implement a **recall** measure R for evaluation on a higher level. It is calculated after

$$\text{R} = \frac{\#\text{TP}}{\#\text{TP} + \#\text{FN}}, \tag{4.6}$$

dividing the number of TP by the total number of annotated frames. This way, we only evaluate the quality of our chord recognition system for annotated time frames. The recall measure allows for an evaluation exceeding the time frame level, e.g., on a track or even dataset level.

Chapter 5

Experiments and Results

In this chapter, we present the results of our practical experiments. In Section 5.1, we focus on the influence of algorithmic parameters on the features and the recognition results. Section 5.2 describes the relationships and interplay between the different parameters. In Section 5.3, we describe different ways to split the datasets for training data-driven methods and the influence on recognition quality. Furthermore, we describe the impact of using the three different chord vocabularies and show results for each of them. Section 5.5 combines the previous variations with different choices of feature types. We compare the results for each chroma type to discuss musical and technical challenges involved in chord recognition. Finally, in Section 5.6, we describe the previous findings on a track and measure level, across versions and songs. When considered as useful, we discuss the results for our two datasets BSD and SWD in parallel.

5.1 Effect of Individual Parameters

Section 4.2 describes the various pre-filtering methods we apply to enhance the feature representations. In the following, we focus on the influence of the algorithmic parameters for pitch weighting, moving median filtering, and logarithmic compression. We use pitch weighting to put emphasis on the center of the pitch range. For this we weight the pitches with a Gaussian window, centered at pitch 60 (C4). Figure 5.1 visualizes the impact of this pre-filtering method. Subfigure a) shows an example of $\mathcal{P}_{\mathrm{CQT}}$ for `D911-22` without pitch weighting, b) shows the example with weighting. The features are averaged across all time frames of the song, smoothed with a filter length of 1.5 octaves, and max-normalized. We visualize the features across all versions of the SWD. Even without weighting we can see that the most energy of harmonic content is located roughly in the middle of our considered pitch range. The figure shows a decrease towards both extremes of the range for all versions. Since all versions closely follow the

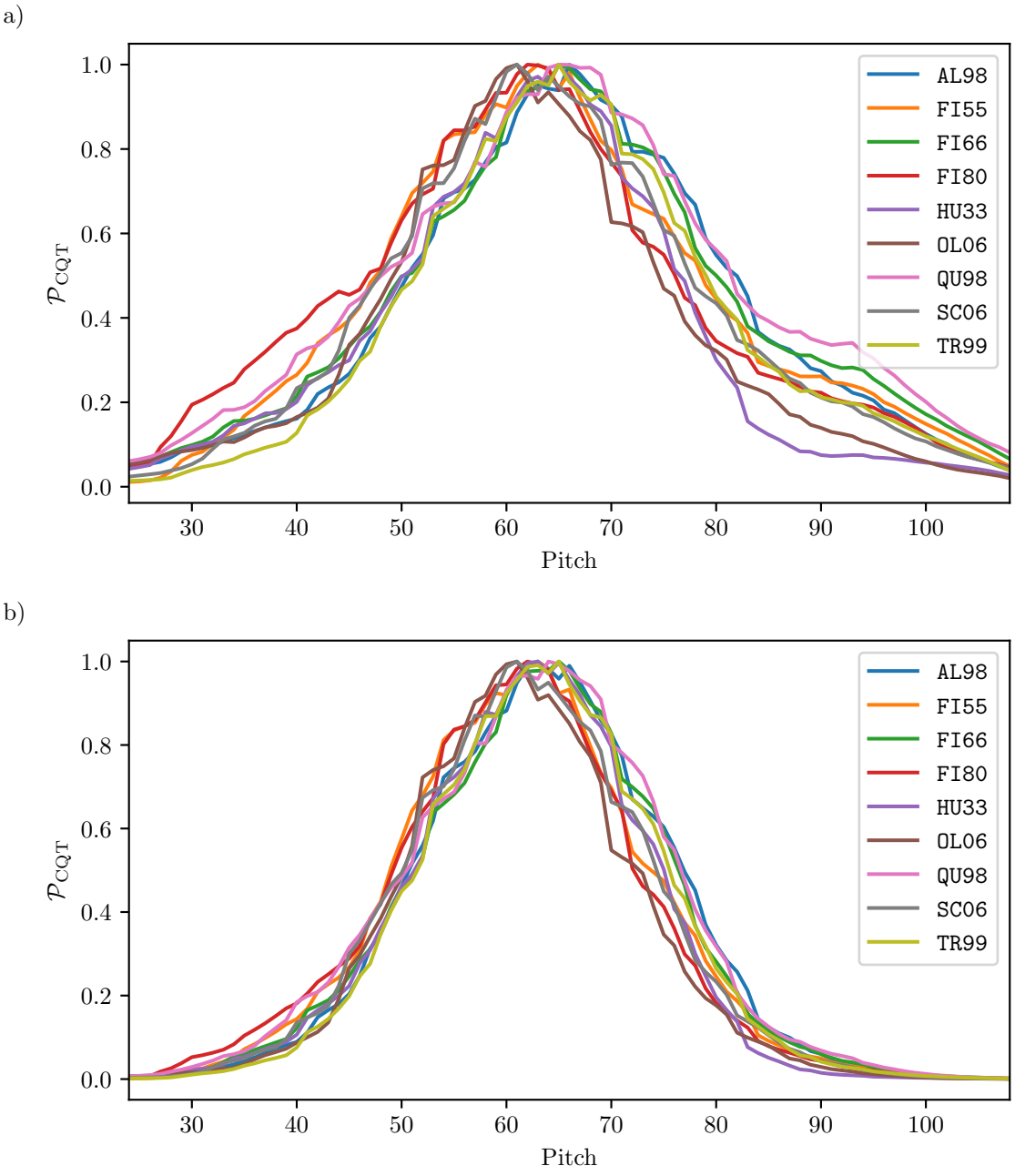


Figure 5.1. Influence of weighting on the pitch features. We show $\mathcal{P}_{\mathrm{CQT}}$ for `D911-22`, averaged across all time frames, smoothed with a filter length of 1.5 octaves. The values for each version are max-normalized. a) No pitch weighting. b) Weighted with a Gaussian window, centered at pitch 60.

score,[1] we can assume that the differences in harmonic content are mostly caused by differences in timbre and the individual recording situations. Since our interest for chord recognition lies in the correct identification of notes that are played, regardless of acoustic characteristics, we want to reduce these differences. In Subfigure b) we can see that our weighting procedure reduces

[1] apart from small global key differences, maximum 1 whole tone

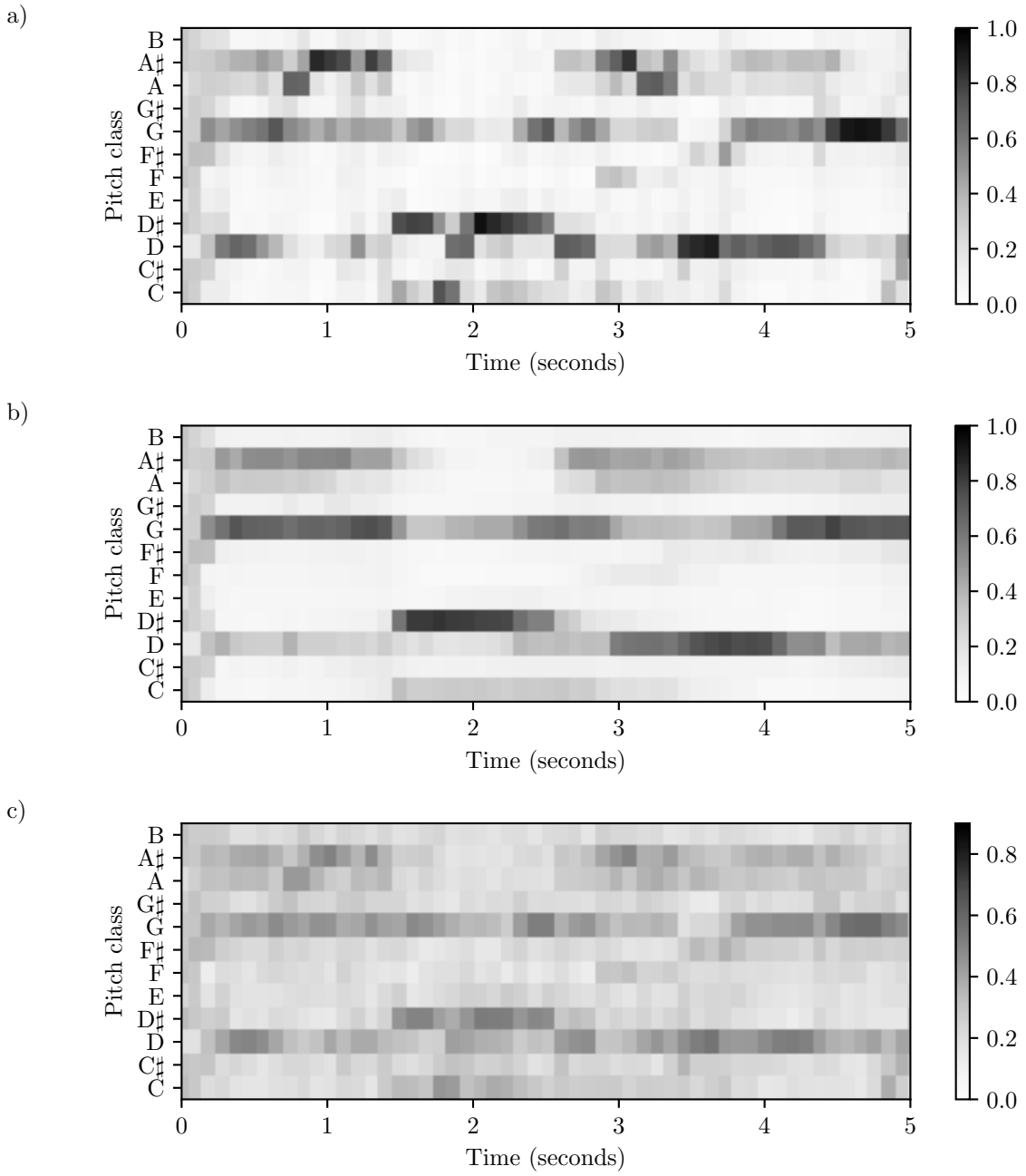


Figure 5.2. Influence of pre-filtering on the chroma features. Normalized $\mathcal{C}_{\mathrm{CQT}}$ of the first five seconds of song `D911-22`, version `AL98`. a) No pre-filtering. b) With median filtering, $l_{\mathrm{filt}} = 11$. c) With logarithmic compression, $\gamma = 100$.

the differences of harmonic content between the individual versions. It also further reduces the values towards the ends of the pitch range. Overall, our experiments showed consistently better results with pitch weighting, so we applied it for all further experiments shown in this thesis.

Next, wo focus on the influence of moving median filtering to apply temporal smoothing to the chroma features. With the parameter l_{filt} we specify the filter length as a number of time

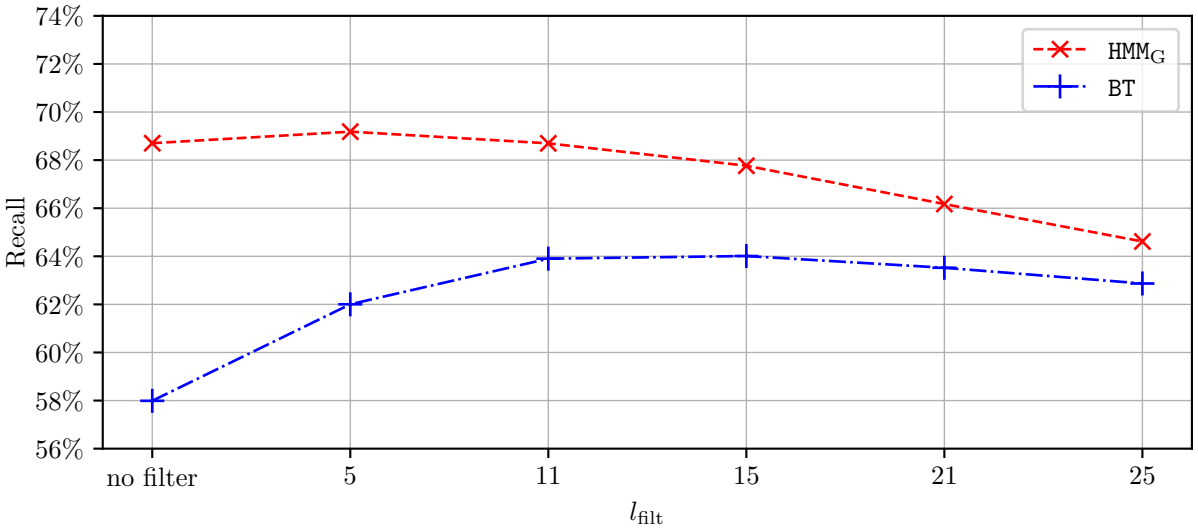


Figure 5.3. Recall values for complete SWD as a function of l_{filt}. We used `BT` and $\texttt{HMM}_{\text{G}}$ as chord recognition methods with $\mathcal{C}_{\text{CQT}}$ and major/minor chord vocabulary.

frames. In Figure 5.2, we visualize the effect of median filtering chroma features. It shows a comparison of the first five seconds of song `D911-22`, version `AL98`, with $\mathcal{C}_{\text{CQT}}$. Subfigure a) shows the unfiltered version, b) shows the median filtered chroma features with $l_{\text{filt}} = 11$. The idea of temporal smoothing for chord recognition is the elimination of irrelevant local fluctuations, such as transitional or melodic notes, which do not belong to the underlying chord. In our example we can see a clear reduction of short notes in the chroma features. The chord notes are emphasized.[2] Obviously, median filtering also reduces chord notes if they appear for a short time. Additionally, temporal smoothing can blur the timing of note onsets. Therefore, a suitable value for l_{filt} has to be found to achieve a good trade-off. In Figure 5.3, we show a sweep of l_{filt} from 5 to 25. We use the recall measure to show recognition results evaluated on the complete SWD dataset. Furthermore, we compare the results with `BT` and $\texttt{HMM}_{\text{G}}$, both using $\mathcal{C}_{\text{CQT}}$ and major/minor vocabulary. We can see a maximum gain in recall of approximately six percent points for `BT` with $l_{\text{filt}} = 15$ and a maximum gain of only approximately one percent point for $\texttt{HMM}_{\text{G}}$ with $l_{\text{filt}} = 5$. These results are in line with the findings of Cho and Bello [5], who reported no significant gain from pre-filtering in combination with post-filtering.

As the final pre-filtering parameter, we focus on the impact of logarithmic compression with parameter γ. We use it to reduce the dynamic range of the feature vectors. Figure 5.2 shows the impact of logarithmic compression on the chroma features. We again use our running example of the first five seconds of song `D911-22`, version `AL98`, with $\mathcal{C}_{\text{CQT}}$. Subfigure a) shows the uncompressed chroma vectors, c) shows the chroma features with compression, $\gamma = 100$. The comparison shows a clear reduction of dynamic range. Furthermore, we see that logarithmic compression also enhances low values that might correspond to noise. While the compressed

[2] annotated chords can be seen in Figure 4.5

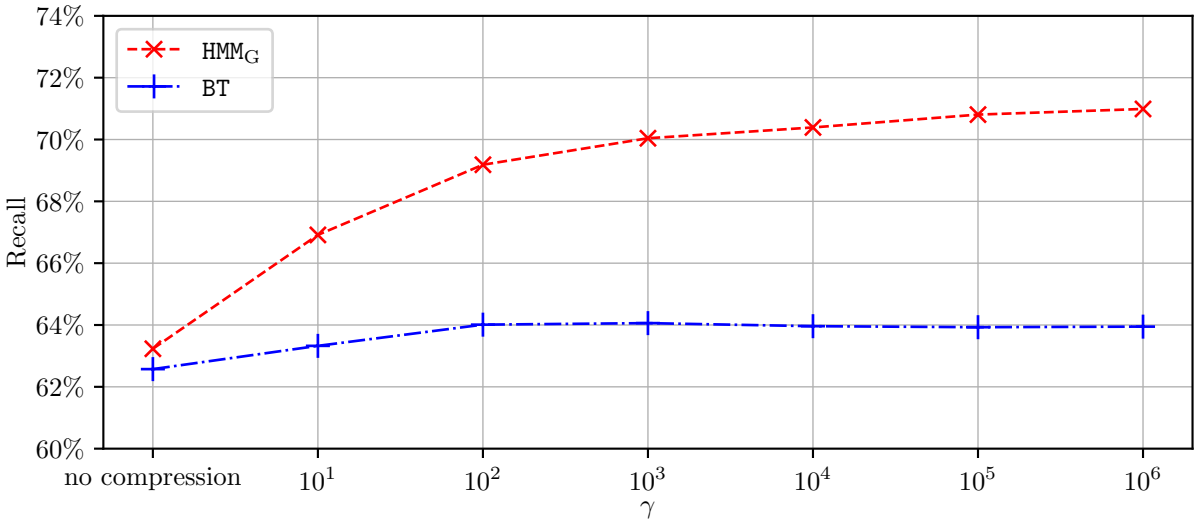


Figure 5.4. Recall values for complete SWD as a function of γ. We used `BT` and $\texttt{HMM}_\text{G}$ as chord recognition methods with $\mathcal{C}_\text{CQT}$ and major/minor chord vocabulary.

features seem less descriptive to a human observer, logarithmic compression is a common pre-filtering strategy for chord recognition tasks, as the enhanced components can be beneficial for the pattern matching. In Figure 5.4 we again show the evaluation of the recognition results for the complete SWD, using `BT` and $\texttt{HMM}_\text{G}$. We sweep γ from 10 to 10^6. It can be seen that logarithmic compression has a strictly positive influence on the results of both methods. Especially for $\texttt{HMM}_\text{G}$, the figure shows a gain of approximately eight percent points for $\gamma = 10^6$ as compared to no logarithmic compression. The impact for `BT` is lower, with a maximum gain of approximately 1.5 percent points for $\gamma = 10^3$. This shows that the enhanced components from compression can be utilized especially well by the Gaussian models.

In our experiments, we apply post-filtering in the form of HMMs with Viterbi decoding. As the controlling parameter, we use the self-transition probability p_self of the uniform transition matrix. The higher the value is, the higher is the likelihood of our chord recognizer returning the same chord label in consecutive time frames. It can be seen as a control for the recognizer's "stiffness." Since a chord usually lasts longer than one frame in the audio recordings, we expect high values for p_self to be beneficial for the chord recognition task. In Figure 5.5 we show the recognition results for the complete SWD with $\texttt{HMM}_\text{b}$ and $\texttt{HMM}_\text{G}$, using $\mathcal{C}_\text{CQT}$. We sweep the parameter p_self from 10% to 90% in steps of ten percentage points. The results show that $\texttt{HMM}_\text{G}$ seems to benefit from a high value of p_self. In fact, we obtain best results with a self-transition probability close to 100%, as we show in Section 5.2. Surprisingly, the results for $\texttt{HMM}_\text{b}$ show the reverse behavior. The recall drops from approximately 69% at $p_\text{self} = 10\%$ to approximately 58% at $p_\text{self} = 90\%$. One possible explanation for this behavior might be a lack of discriminating power of the binary templates. The similarity values might be too low to "overcome" the high self-transition probability, resulting in the chord recognizer staying on the same chord erroneously.

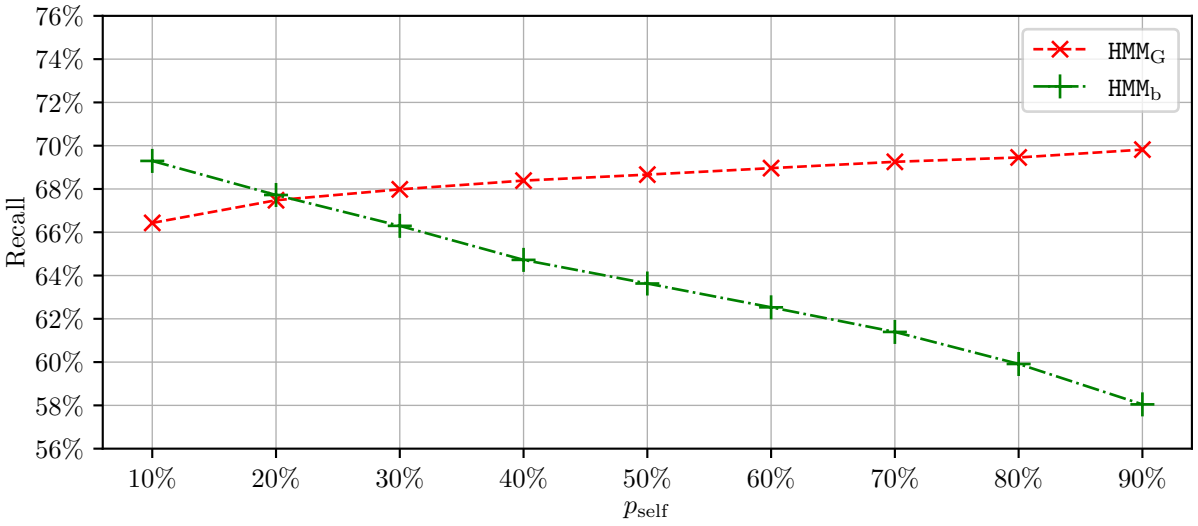


Figure 5.5. Recall values for complete SWD as a function of p_{self}. We used HMM$_{\text{b}}$ and HMM$_{\text{G}}$ as chord recognition methods with $\mathcal{C}_{\text{CQT}}$ and major/minor chord vocabulary.

The results we presented so far were obtained with methods that use either binary templates or Gaussians as chord models. We acquired all previously presented results for HMM$_{\text{G}}$ with training on the neither split, which we discuss in Section 5.3. In general, we consistently achieved the best results using Gaussian models in combination with HMMs and Viterbi decoding. When using averaged templates, we achieved the worst results. Thus, the capacity of the Gaussian models to capture the statistics of the training data seems to outperform the capacity of the averaged templates. Even without the benefit of any training, the use of binary templates with HMMs can achieve results close to the Gaussians, when suitable pre-filtering and post-filtering strategies are applied. In the following, we focus on presenting the results obtained with HMM$_{\text{G}}$.

5.2 Interplay Between Different Parameters

In the previous chapter we presented the influence of individual parameters on the chord recognition process and quality. Figures 5.3, 5.4, and 5.5 showed the quality of different recognition methods as a function of l_{filt}, γ, and p_{self}, respectively. For each figure, we set the values of the remaining parameters to a fixed value. To acquire a better understanding of the interplay between the different parameters, we move away from the one-dimensional approach of sweeping a single parameter individually. Instead, we jointly sweep two parameters, implementing a two-dimensional grid search. As mentioned before, we focus on the method HMM$_{\text{G}}$. The two most influential parameters for this method are γ and p_{self}, the parameters controlling the strength of logarithmic compression and post-filtering through HMMs, respectively.

Figure 5.6 shows the results for a grid search of γ and p_{self}, evaluated on the SWD. We use HMM$_{\text{G}}$ with $\mathcal{C}_{\text{deep}}$ as features. We sweep the values of γ in a range from 10 to 10^6, and also include

the results without logarithmic compression. We sweep p_{self} in a non-linear fashion, ranging from 93% to 99.99999999%. Each cell in the grid search matrix represents a recall value for the complete SWD. We underline each cell with a grayscale value that represents the recall value. Additionally, we show the values as text, rounded to the nearest integer percentage. The cell with the highest recall value is highlighted with white border lines. Subfigure a) shows the results with the major/minor vocabulary, b) shows the results with the triad vocabulary, and c) shows the results for the seventh vocabulary.
The figure shows that the best values with $\mathcal{C}_{\text{CQT}}$ for the SWD are achieved with large values for both γ and p_{self}. With all three vocabularies, the optimal value for logarithmic compression is $\gamma = 10^6$. The optimal value for p_{self} shows a minor change across vocabularies, but is located at $p_{\text{self}} > 99.9999\%$ in all three cases. We can also see that the chord recognition quality is insensitive to small deviations from the optimal parameter values. The grayscale difference between the cell with the optimal value and neighboring cells is practically imperceptible. The figure shows a clear drop in recognition quality for the more complex vocabularies, especially for the seventh vocabulary. We discuss these findings in more detail in Section 5.4.
Figure 5.7 shows the results for the BSD, again for $\mathcal{C}_{\text{CQT}}$. We can see that the optimal values for p_{self} for each vocabulary are lower than for the SWD. For major/minor, we obtained the best results for $p_{\text{self}} = 93\%$. Again, we can see that the recognition quality is robust to small deviations from the optimal values for both p_{self} and γ. For the more complex vocabularies, this seems to change. While the results are robust across variations of p_{self}, they exhibit a strong dependency on the strength of logarithmic compression. Additionally, lower values for γ achieve better results with an optimal value of $\gamma = 100$ for the triad vocabulary and $\gamma = 10$ for the seventh vocabulary. For both datasets, our experiments revealed a similar behavior for $\mathcal{C}_{\text{STFT}}$ and $\mathcal{C}_{\text{IIRT}}$, compared with the results we show for $\mathcal{C}_{\text{CQT}}$.
In Figure 5.8 we can see the grid search results for the SWD with $\mathcal{C}_{\text{deep}}$, the feature type acquired with deep-learning techniques. Again, we show the results for the three vocabularies in Subfigures a), b), and c). We can see that the use of $\mathcal{C}_{\text{deep}}$ seems to enable a higher overall chord recognition quality. We discuss this finding in more detail in Section 5.5. With respect to the parameters, the results show an overall small sensitivity to parameter changes. With the exception of using no logarithmic compression at all, the recall values lie within a range of approximately four percent points across the whole parameter range, for each vocabulary respectively. The optimal parameter values stay the same for all three vocabularies at $\gamma = 10$ and $p_{\text{self}} = 99.99999999\%$.
As our final grid search example, Figure 5.9 shows the results for the BSD with $\mathcal{C}_{\text{deep}}$. We can see that the optimal values for p_{self} are again in a lower region than for the SWD. For the major/minor and triad vocabularies the optimal value is $p_{\text{self}} = 93\%$, for the seventh vocabulary it is $p_{\text{self}} = 99\%$. For all three vocabularies, we can see little variation of the results across parameter changes, except for using no logarithmic compression at all. In both datasets, we can see overall better results and a higher robustness for $\mathcal{C}_{\text{deep}}$, compared to $\mathcal{C}_{\text{CQT}}$. For both chroma

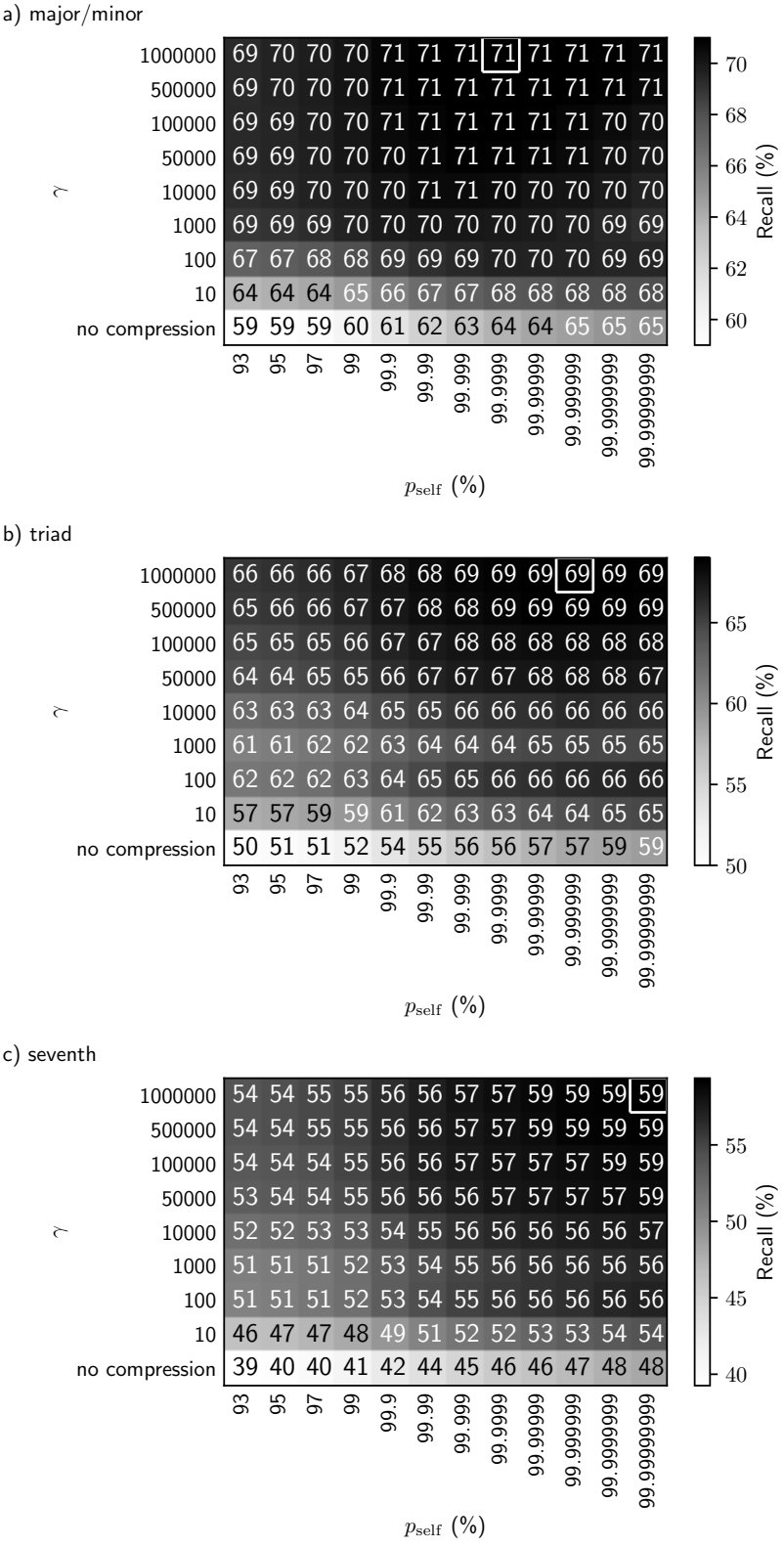


Figure 5.6. Grid search of the parameters γ and p_{self}. We show the results for SWD with $\texttt{HMM}_{\text{G}}$ and $\mathcal{C}_{\text{CQT}}$. The recall text values are rounded to the nearest integer, the underlying grayscale values accurately represent the numerical recall values. The highest value is highlighted. a) With major/minor vocabulary. b) With triad vocabulary. c) With seventh vocabulary.

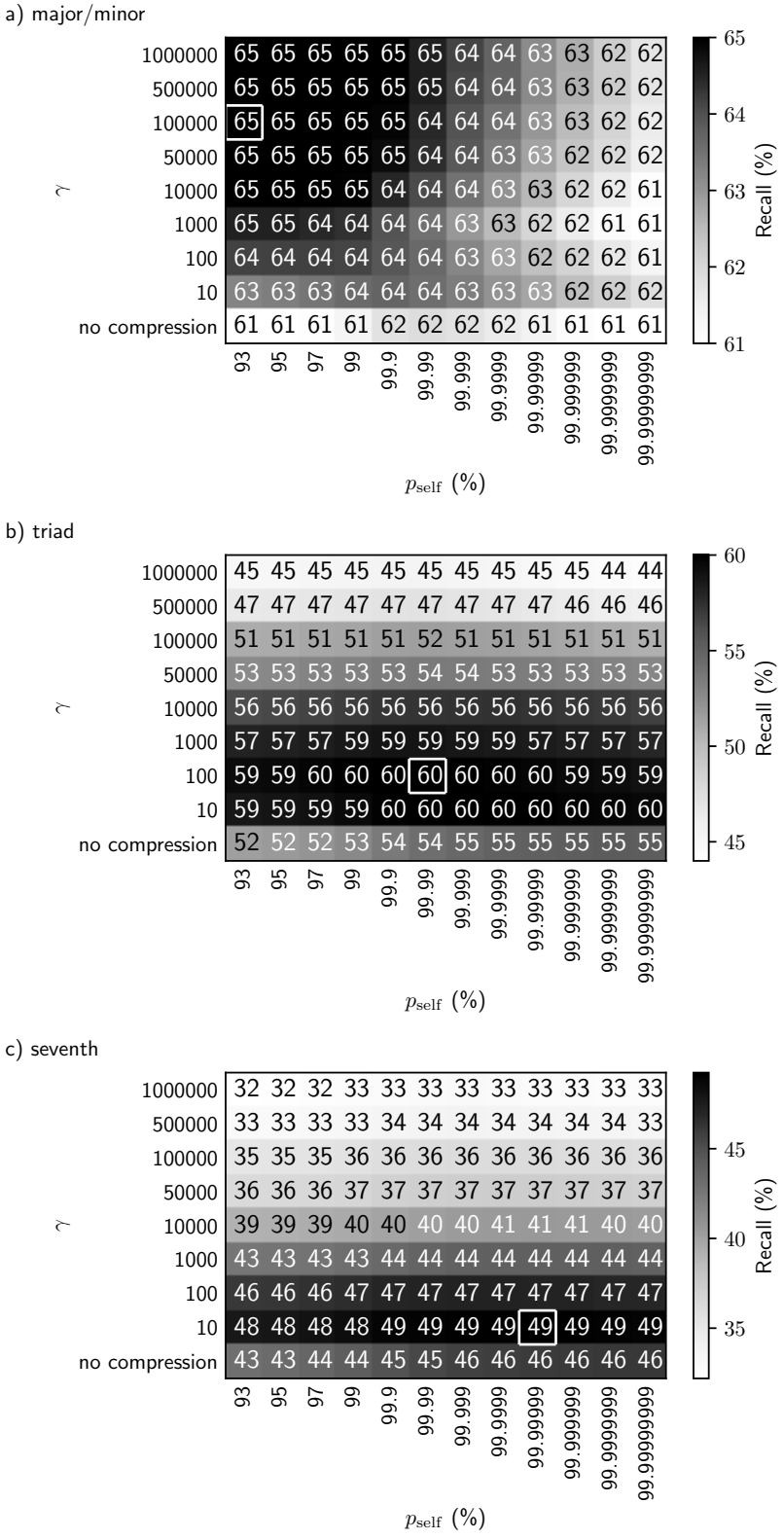


Figure 5.7. Grid search of the parameters γ and p_{self}. We show the results for BSD with $\texttt{HMM}_{\text{G}}$ and $\mathcal{C}_{\text{CQT}}$. The recall text values are rounded to the nearest integer, the underlying grayscale values accurately represent the numerical recall values. The highest value is highlighted. a) With major/minor vocabulary. b) With triad vocabulary. c) With seventh vocabulary.

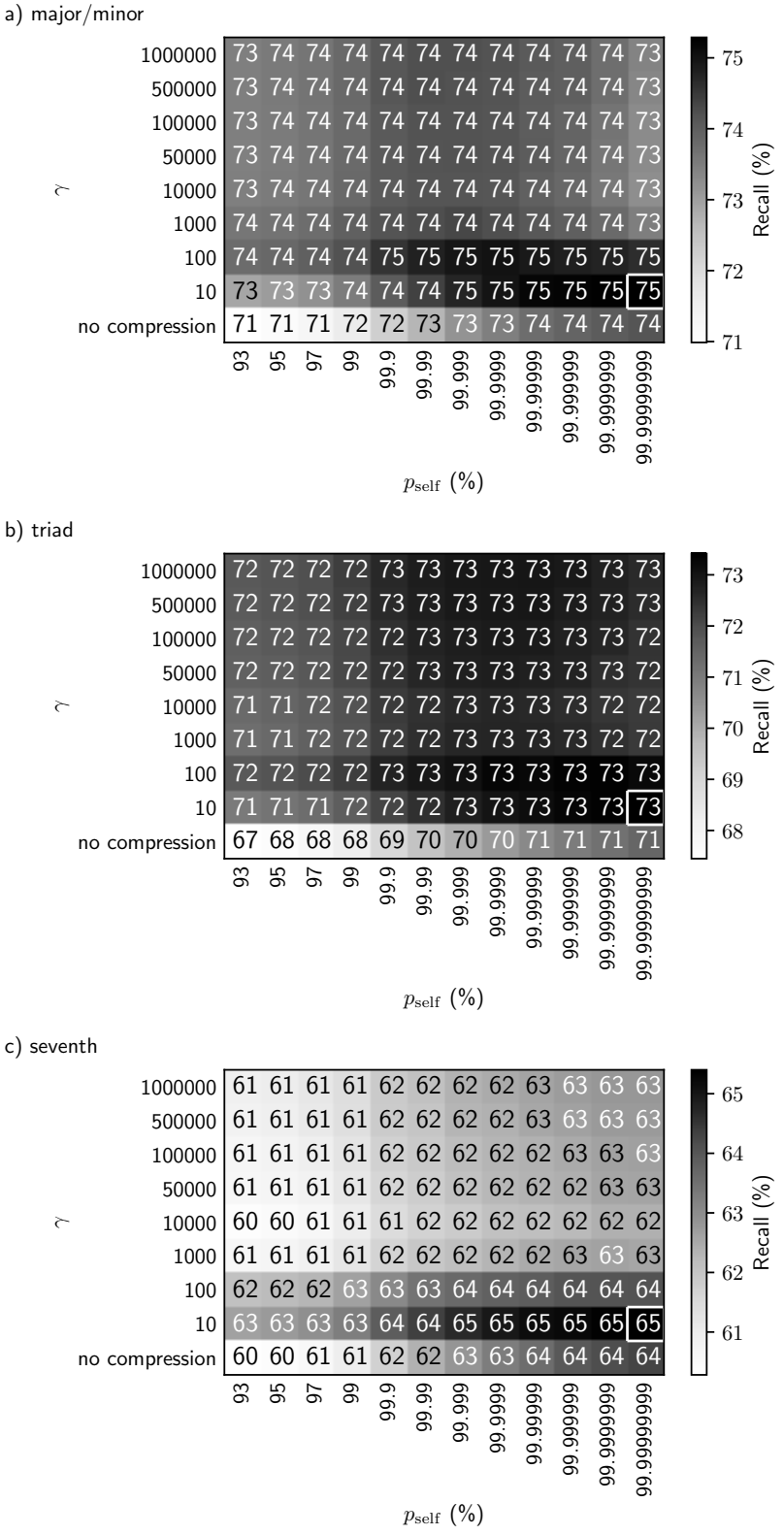


Figure 5.8. Grid search of the parameters γ and p_{self}. We show the results for SWD with $\texttt{HMM}_{\text{G}}$ and $\mathcal{C}_{\text{deep}}$. The recall text values are rounded to the nearest integer, the underlying grayscale values accurately represent the numerical recall values. The highest value is highlighted. a) With major/minor vocabulary. b) With triad vocabulary. c) With seventh vocabulary.

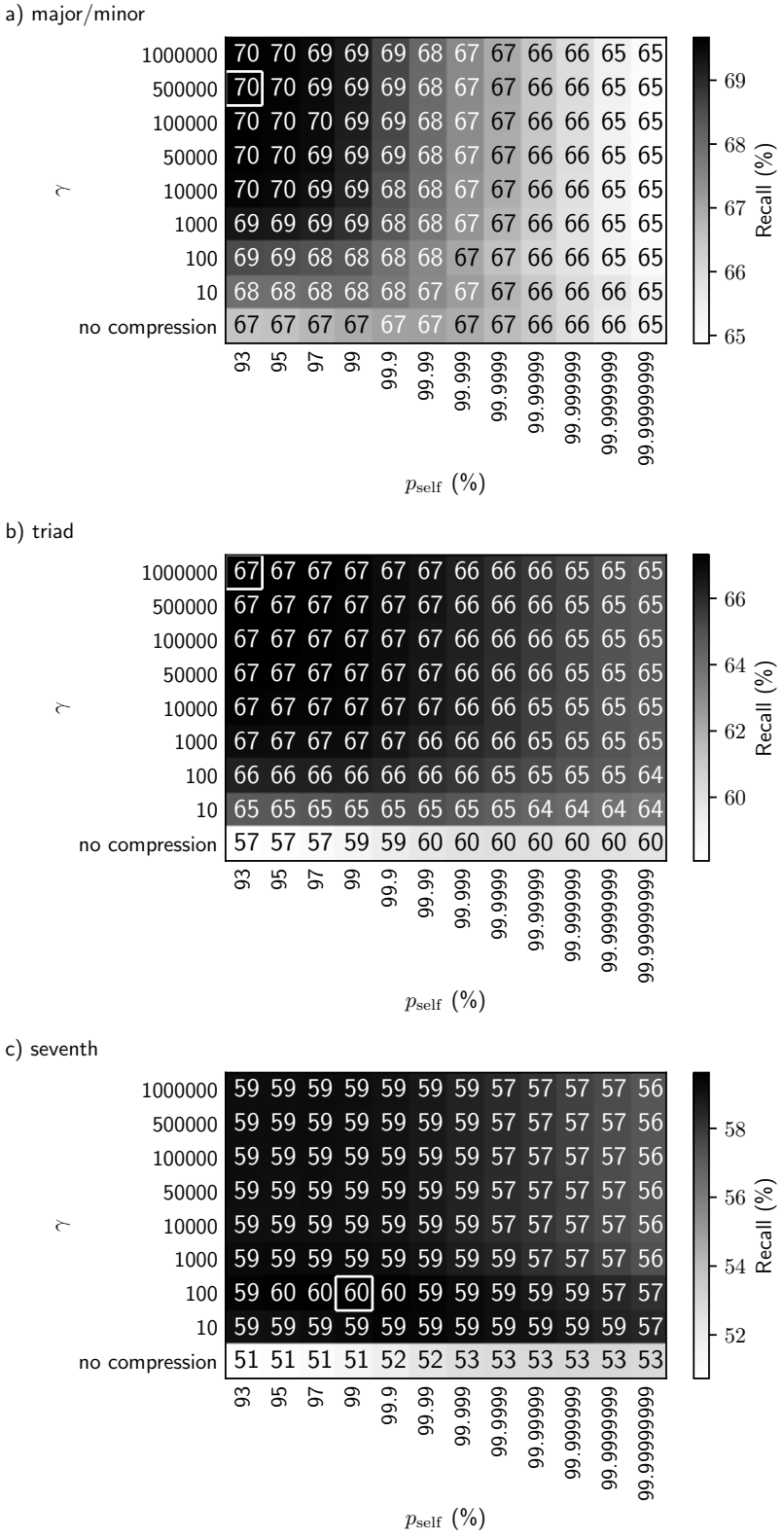


Figure 5.9. Grid search of the parameters γ and p_{self}. We show the results for BSD with $\texttt{HMM}_{\text{G}}$ and $\mathcal{C}_{\text{deep}}$. The recall text values are rounded to the nearest integer, the underlying grayscale values accurately represent the numerical recall values. The highest value is highlighted. a) With major/minor vocabulary. b) With triad vocabulary. c) With seventh vocabulary.

feature types, the results for the SWD show a better overall chord recognition quality than for the BSD. The recall differences between the two datasets consistently lie within a range of five to ten percent points.
Let us summarize. While it is important to set meaningful ranges for the different algorithmic parameters, we cannot report a substantial gain in chord recognition quality for micro-adjustments of the parameters. This indicates a high robustness of our chord recognition methods. Generally, the performance always seems to benefit from applying logarithmic compression. Furthermore, the values for p_{self} should be set to a high level, especially for the SWD. For the BSD, slightly lower self-transition probabilities provide better results. A possible explanation for this finding might be the slower tempo and harmonic rhythm of Schubert's Winterreise. Individual chords tend to last longer than in the BSD. This translates to a higher number of consecutive time frames with the same chord label in our scenario, suggesting higher values for p_{self}. If we compare the results for the different vocabularies, we can see that the highest optimal value for p_{self} can be found for the seventh vocabulary. This is the case for all four figures. A likely explanation for this finding is the high number of chords in the vocabulary. It contains more chords which are similar to the one that is actually annotated. A high self-transition probability can prevent the chord recognizer from oscillating between these similar chords. The results of this section were all acquired with a neither split for training the Gaussian models, which we explain in the following section.

5.3 Cross-Validation Splits

In the following, we present the different dataset splits we used to train, optimize, and evaluate the chord recognizers. Since we mostly use the same dataset for simultaneously training and testing the recognition methods, we have to implement suitable data splits. Additionally, the different splits can offer insights into the generalization of our methods across songs, versions, or both, as well as across entirely different datasets.

Figure 5.10 shows a schematic visualization of the variants we used to split the datasets. We denote them version, song, neither, and cross-dataset split. For each variant, we split the data into three subsets, denoted training, validation, and test set. As the name suggests, for the **version split** we split the dataset along the version axis. This means that each subset contains the full number of songs, but only a restricted, non-overlapping number of versions. For the SWD, the training set contains five different versions, the validation set contains one version, and the test set contains three different versions. For the BSD, the training set contains three different versions, the validation set contains one version, and the remaining two versions are used for the test set. To acquire test results for the complete datasets, we use three-fold cross-validation, shuffling the versions contained in the subsets for each fold. As an example, in the BSD version

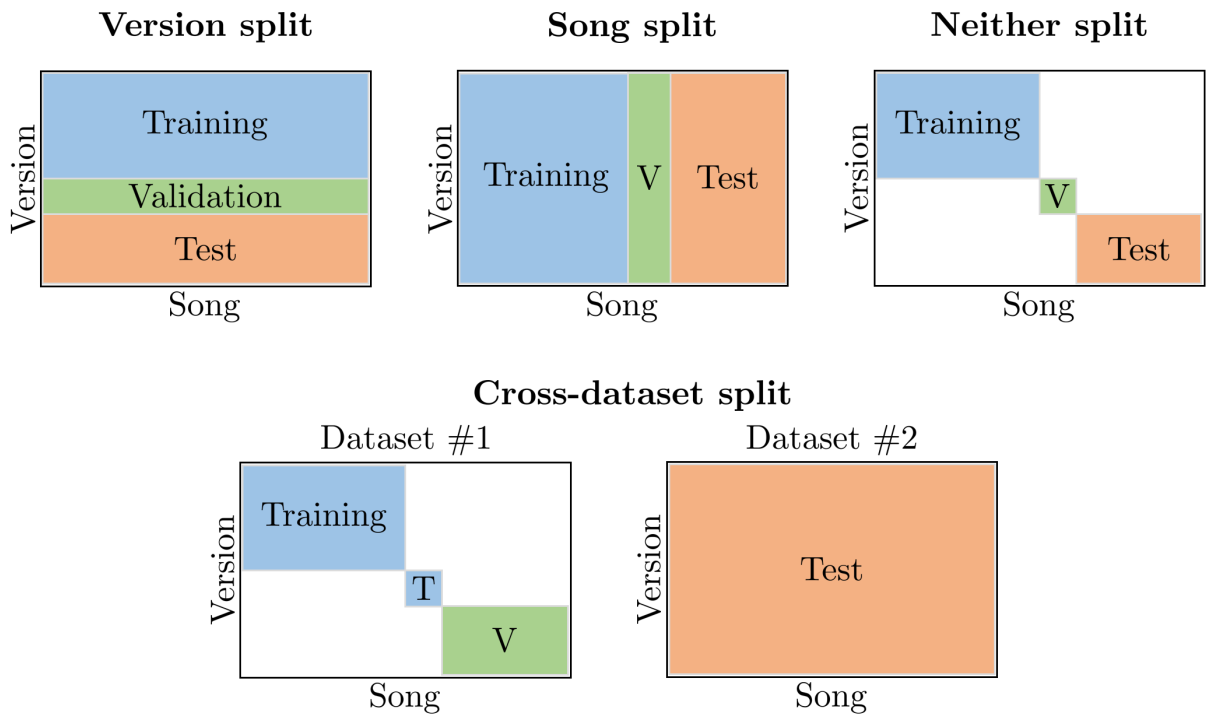


Figure 5.10. Schematic representation of the four data splits we used to train, validate, and test within and across datasets [37]. We denote them version, song, neither, and cross-dataset split.

split we use the versions `Ashkenazy` and `Barenboim` in the first fold to get test results for the method we trained and optimized using the remaining versions in the training and validation sets. In the second fold, we use versions `BilsonEtAl` and `Brendel` for testing and the remaining versions for training and validation. In the final fold, we test on versions `Gulda` and `Jando`, finally obtaining test results for the complete dataset. The version split can used to test the generalization of the chord recognizers across versions. In each split variant, we choose the subsets for each fold so the combined test sets cover the complete datasets in a non-overlapping fashion. The training data in the different folds may overlap. The order of the data included in the individual folds is chosen alphabetically. An exception for this is the song split of the BSD, where we randomly assigned the order of the songs for the subsets. We did this to create training diversity across the early and late sonata movements.

For the **song split**, we split the datasets across the song axis. This means that each set contains all versions, but only a subset of the songs. For the SWD, the training set contains 13 different songs, the validation set contains three different songs, and the test set contains eight different songs. We again use three-fold cross-validation to acquire results for the complete dataset. For the BSD, the training set contains 13 different songs, the validation set contains three different songs and the test set contains 16 different songs. This allows for only a two-fold cross-validation. We use the song split to test the generalization across unknown songs.

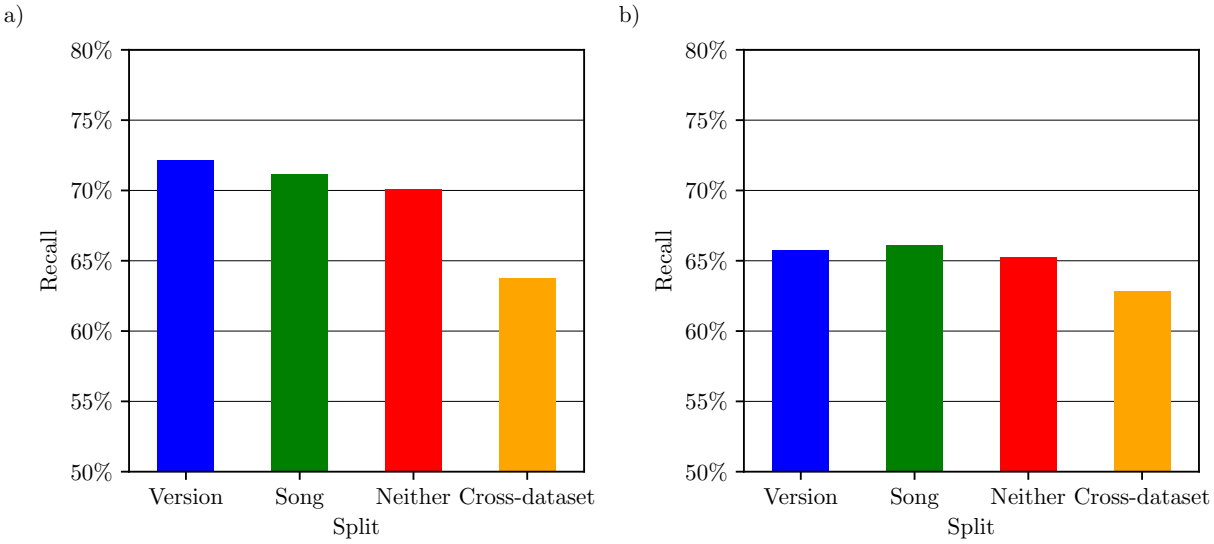


Figure 5.11. Recall values across the different split variants. We show the results for $\texttt{HMM}_{\text{G}}$ with $\mathcal{C}_{\text{CQT}}$, major/minor chord vocabulary, and optimized parameters. a) For SWD. b) For BSD.

In the **neither split**, we split the datasets across both axes simultaneously. This ensures that our test set neither contains any of the songs, nor any of the versions we used for training and validation. Therefore, the neither split represents the strictest separation between training and test data within each dataset. In the SWD, the training set of the neither split contains 19 songs in four different versions. The validation set contains two songs in two different versions, and the test set contains three songs in three different versions. For the BSD, the training set contains 24 songs in three versions, the validation set contains four songs in one version, and the test set contains four songs in two versions. For both datasets, the neither split allows for a 24-fold cross-validation. It represents the most general scenario for training and testing within the same dataset.

In addition to the three splits within each dataset, we used a **cross-dataset** split to evaluate the generalization from one dataset to another. As shown in 5.10, we split the first dataset into training and validation set and then use the complete second dataset as test set. The split of the training dataset corresponds to the neither split, but we add the former validation set to the training set and use the former test set as validation set. We again use 24-fold cross-validation on the training dataset and then evaluate the results for the whole test set at once. When we report results for the cross-dataset split for SWD, we refer to the results with training on the BSD and testing on the SWD, vice versa for the cross-dataset results for BSD.

For all splits, we use the training set to train the Gaussian chord models with a given set of parameters. The trained Gaussians are then used to perform chord recognition on the validation set. Subsequently, we use the results of the validation set to optimize our method parameters.

We jointly optimize the three parameters l_{filt}, γ, and p_{self}. For each parameter, we consider three different values within a suitable range. This amounts to a total number of 27 different parameter combinations. We evaluate each of the combinations on the validation set and pick the parameter combination which produces the best results. Subsequently, we use the optimal parameters to train the Gaussians on both the training and the validation set combined. Finally, we use the trained Gaussians and the optimal parameters to perform chord recognition on the test set. For the cross-dataset split, the procedure is slightly different, since we have an optimal parameter combination for each of the 24 folds, but only one fixed test set. Hence, we use the parameter combination that was picked most often. The considered parameter values are $\{1, 11, 21\}$ for l_{filt}, and $\left\{10, 10^4, 10^6\right\}$ for γ. For the SWD, we used the values $\{99\%, 99.99\%, 99.9999\%\}$ for p_{self}, for the BSD we used the values $\{93\%, 99\%, 99.99\%\}$. These parameter values are based on the findings we presented in the previous sections. The results we showed there were acquired with the neither split, but without any optimization. We used the given parameters to train the Gaussians on the combined training and validation set.

In Figure 5.11 we show a comparison of the results obtained with the different data splits for both datasets. The recall values are obtained from $\texttt{HMM}_{\text{G}}$ with $\mathcal{C}_{\text{CQT}}$, major/minor chord vocabulary, and optimized parameters. For both datasets, we can see that the cross-dataset split led to the lowest chord recognition quality with a recall of approximately 64% for SWD and 63% for BSD, which is the result we expected. The difference in results between the three inner-dataset splits lies within a range of approximately two percent points for both datasets, respectively. For the SWD, we report the best results for the version split with a recall of approximately 72%, with song and neither split at approximately 71% and 70%, respectively. The results for the BSD show a smaller gap between the cross-dataset split and the other three splits. Here, the song split achieves a slightly higher recall value of approximately 66%, as compared to the version and neither split with a recall of just over 65%. In the following sections, we report results across different splits, feature types, and vocabularies. In Section 5.6 we give possible explanations for the differences in recognition quality between the splits.

5.4 Comparison of Different Chord Vocabularies

In this section, we show a comparison of the chord recognition results with different chord vocabularies. In our experiments, we use three different vocabularies, the major/minor, triad, and seventh vocabulary. Recapitulating our definition from Section 3.3, the major/minor vocabulary contains 24 different chords, twelve major and twelve minor chords for each root note. The triad vocabulary contains 40 different chords, twelve major, minor, and diminished chords as well as four different augmented chords for root notes C, C♯, D, and D♯. The seventh vocabulary is the largest vocabulary, containing 91 different chords. It comprises all chords from

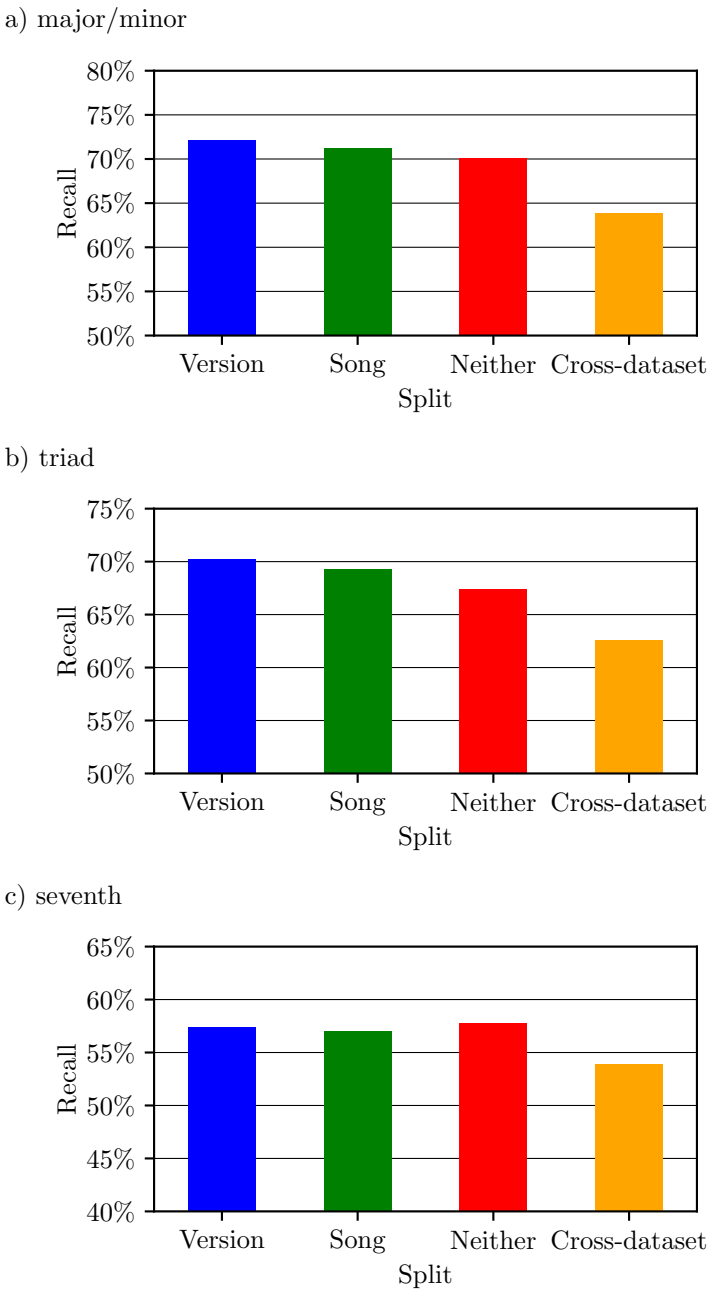


Figure 5.12. Recall values across the different vocabularies and split variants for SWD. We show the results for $\texttt{HMM}_{\text{G}}$ with $\mathcal{C}_{\text{CQT}}$ and optimized parameters. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

the triad vocabulary, with twelve additional chords for M7, m7, maj7, and hdim7, respectively. Additionally, it contains three dim7 chords for root notes C, C♯, and D. For the evaluation of the results with different chord vocabularies, the chord labels from the annotations are parsed accordingly, with varying levels of mapping and reduction. A detailed description of this process was given in Section 3.3.

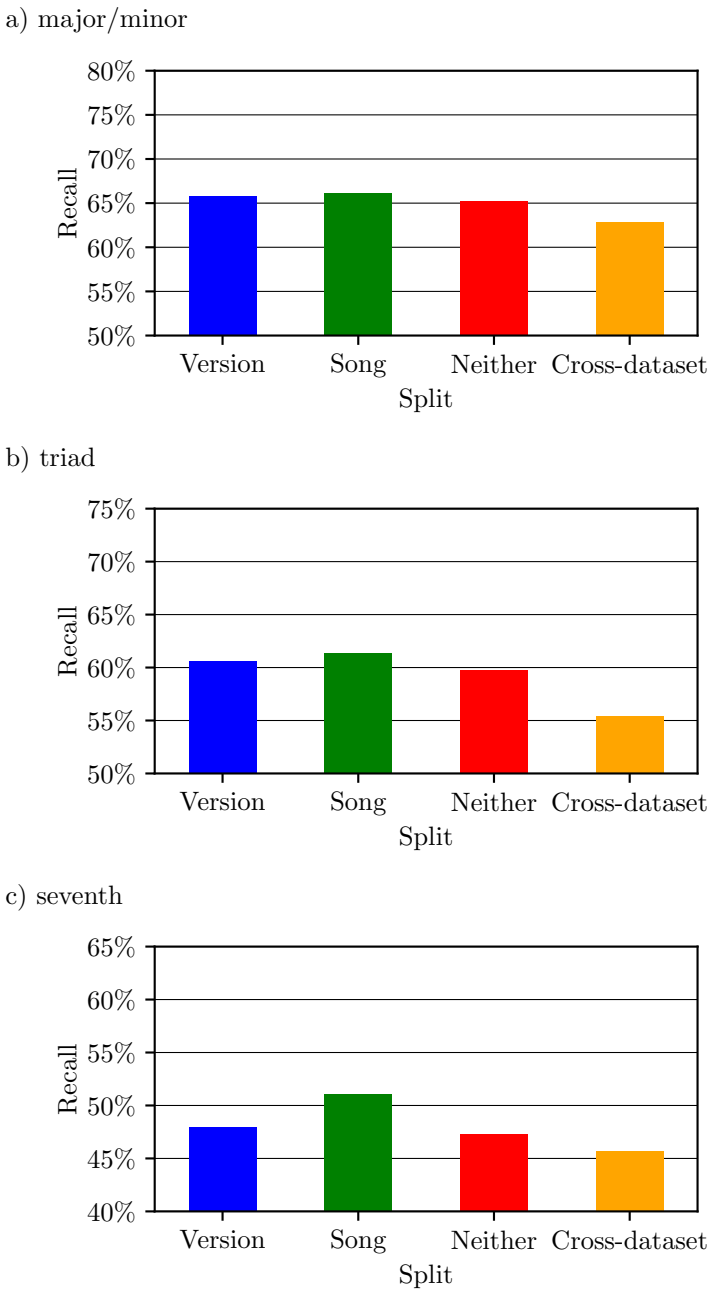


Figure 5.13. Recall values across the different vocabularies and split variants for BSD. We show the results for $\texttt{HMM}_{\text{G}}$ with $\mathcal{C}_{\text{CQT}}$ and optimized parameters. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

In Figure 5.12 we show a comparison of the results for the three vocabularies. The recognition results are obtained with $\texttt{HMM}_{\text{G}}$ and $\mathcal{C}_{\text{CQT}}$ on the SWD. The parameters are optimized as described previously. We report individual recall values for each of the four different splits. Subfigure a) shows the results for major/minor, which we already discussed in the previous section. For the inner-dataset splits we achieve results of 70–72%, with the cross-dataset the chord recognition

quality drops to approximately 64%. In Subfigure b) we can see similar relationships between the split variants for the triad vocabulary. Overall, the quality of the chord recognition decreases by roughly one to three percent points for each split. The version split again provides the best result with a recall of just over 70%. The reduction in recognition quality for the triad vocabulary in comparison to the major/minor vocabulary corresponds to our expectations, since the higher number of chords increases the overall complexity of the chord recognition task. Still, the difference in quality is relatively small. In Section 5.6 we show that the song-wise recall actually increases in some cases when using the triad vocabulary.
In Subfigure c), we show the results for the seventh vocabulary. Compared to the other vocabularies, we can see a prominent decrease in recognition quality. The recall values for the version and song split drop by approximately twelve to 13 percent points compared to the triad vocabulary results. The decrease for the neither and cross-dataset split is slightly lower at approximately nine to ten percent points. Here, the neither split slightly outperforms the other split variants. We are not surprised by the decrease in quality when we use the seventh vocabulary. With a number of 91 chords it is larger than the other two vocabularies, which signifies a large increase in complexity for the chord recognition task.

Figure 5.13 shows the results for the BSD, obtained with the same methods and for the same vocabularies. In Subfigure a) we see the results discussed in the previous section. When we compare the results for the triad vocabulary in b) with the major/minor results, we again see a decrease in chord recognition quality. Compared to the SWD, the drop in recall is larger for the BSD with approximately five percent points for the inner-dataset splits and approximately eight percent points for the cross-dataset split. Furthermore, we see a similar decrease in quality when comparing triad and seventh vocabulary for both SWD and BSD. The drop in recall values ranges from ten to 13 percent points across all four split variants. Across all three vocabularies, the song split provides the highest recall values for the BSD, especially for the seventh vocabulary.

5.5 Comparison of Chroma Feature Types

In this section we show the chord recognition results with different feature types. We compare three chroma features obtained with traditional signal processing methods, $\mathcal{C}_{\text{CQT}}$, $\mathcal{C}_{\text{STFT}}$, and $\mathcal{C}_{\text{IIRT}}$. Additionally, we report the results with $\mathcal{C}_{\text{deep}}$, where the features are extracted by means of deep-learning with a CNN [40]. Furthermore, we show the results for the two baseline features $\mathcal{C}_{\text{score}}$ and $\mathcal{C}_{\text{annot}}$. $\mathcal{C}_{\text{score}}$ is based on the audio-aligned MIDI representation of the score, which is part of the datasets. It can be seen as a perfect chroma, exclusively containing entries for all notes that are played. $\mathcal{C}_{\text{annot}}$ is based on the "Extended" column of the chord annotations. All annotated chord notes for each chord label are converted to chroma vectors. We implement no mapping or reduction of chords, so the notes contained in $\mathcal{C}_{\text{annot}}$ differ from the chord annotations

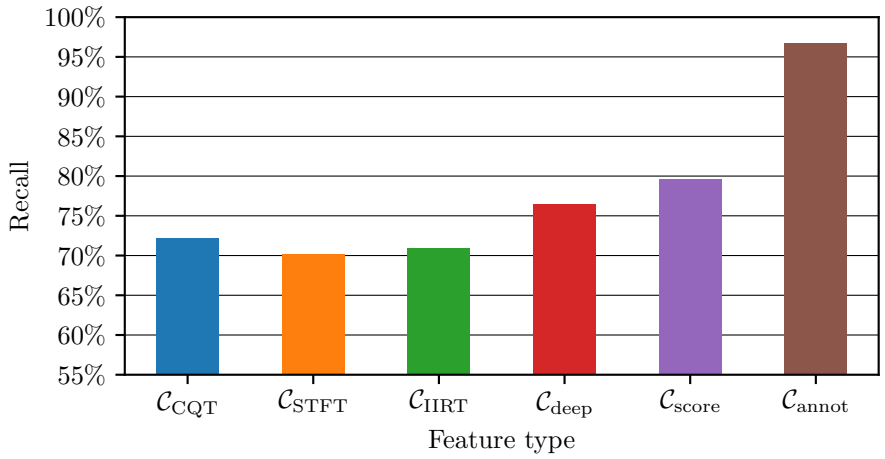
we used for our three vocabularies. Visual examples of all six chroma types can be seen in Figures 4.1 and 4.2.

In Figures 5.14–5.21 we show the results for the different feature types on the dataset level. All values were acquired with $\texttt{HMM}_{\text{G}}$ and optimized parameters, as discussed in Section 5.3. In each figure, Subfigure a) shows the results for the major/minor vocabulary, b) shows the results for the triad vocabulary, and c) shows the results for the seventh vocabulary. We show the results for SWD and BSD next to each other on double pages, with the results for SWD on the left hand side and the results for BSD on the right hand side. We consecutively show the results for our four different data split variants. This means, Figures 5.14 and 5.15 show the results for the version split, Figures 5.16 and 5.17 show the results for the song split, Figures 5.18 and 5.19 show the results for the neither split, and finally, Figures 5.20 and 5.21 show the results for the cross-dataset split. In these eight figures, we combine all modalities that we discussed previously. They represent our final discussion of chord recognition results on the dataset level, before we move on to an in-depth discussion on more detailed levels in Section 5.6.

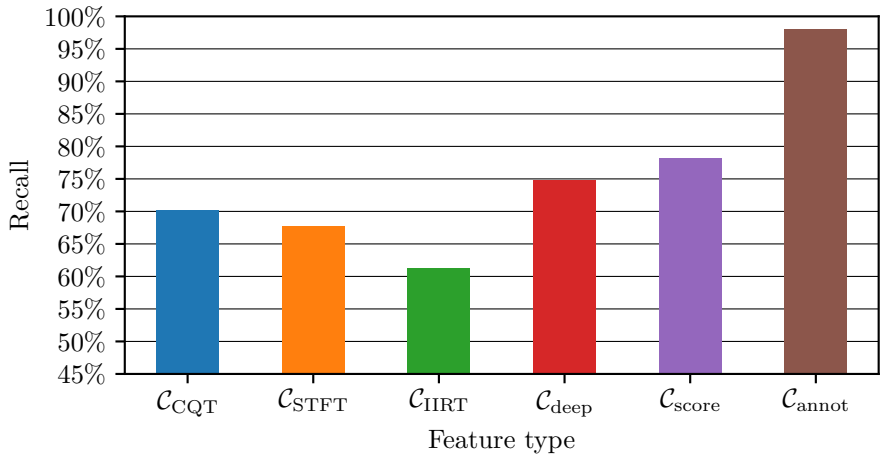
Let us first focus on the differences between the three signal processing chroma types $\mathcal{C}_{\text{CQT}}$, $\mathcal{C}_{\text{STFT}}$, and $\mathcal{C}_{\text{IIRT}}$. Across all split variants, vocabularies, and both datasets we can see that the chord recognition quality with these chroma types is lower than the quality with $\mathcal{C}_{\text{deep}}$ and the two baseline chromas. For the SWD, the best results between the three signal processing chromas were obtained with $\mathcal{C}_{\text{CQT}}$. This is the case across all three vocabularies and all four data splits. The overall best value for $\mathcal{C}_{\text{CQT}}$ for SWD is obtained with the major/minor vocabulary in the version split, with a recall of approximately 72%. The results across vocabularies were already discussed in the previous section, with a small decrease in recall for the triad vocabulary and a larger drop for the seventh vocabulary. The lowest value for $\mathcal{C}_{\text{CQT}}$ for the SWD is obtained with cross-dataset split and seventh vocabulary, with a recall of approximately 54%. For the major/minor vocabulary, all three signal processing chromas produce comparable results. With the higher complexity of the triad and seventh vocabulary, the results for $\mathcal{C}_{\text{IIRT}}$ show a larger decrease than $\mathcal{C}_{\text{CQT}}$ and $\mathcal{C}_{\text{STFT}}$ for the SWD. The recall drops from approximately 71% with version split and major/minor vocabulary to approximately 43% with cross-dataset split and seventh vocabulary. The results from $\mathcal{C}_{\text{STFT}}$ exhibit a stability across vocabularies that is comparable with $\mathcal{C}_{\text{CQT}}$, but provide overall slightly lower results for the SWD. The best recall value for $\mathcal{C}_{\text{STFT}}$ is approximately 70% with version split and major/minor vocabulary, the lowest value of approximately 53% is obtained with seventh vocabulary and neither and cross-dataset split.

For the BSD, the three signal processing chromas also provide lower recall values than the remaining three chroma types. In contrast to the SWD, $\mathcal{C}_{\text{STFT}}$ consistently outperforms $\mathcal{C}_{\text{CQT}}$ and $\mathcal{C}_{\text{IIRT}}$. This is true for all modalities, except for the cross-dataset split, where $\mathcal{C}_{\text{CQT}}$ produces marginally better results. This indicates that features from $\mathcal{C}_{\text{CQT}}$ possess a large discriminating power for the audio recordings of the SWD. The best values for the BSD are generally lower than

a) major/minor



b) triad



c) seventh

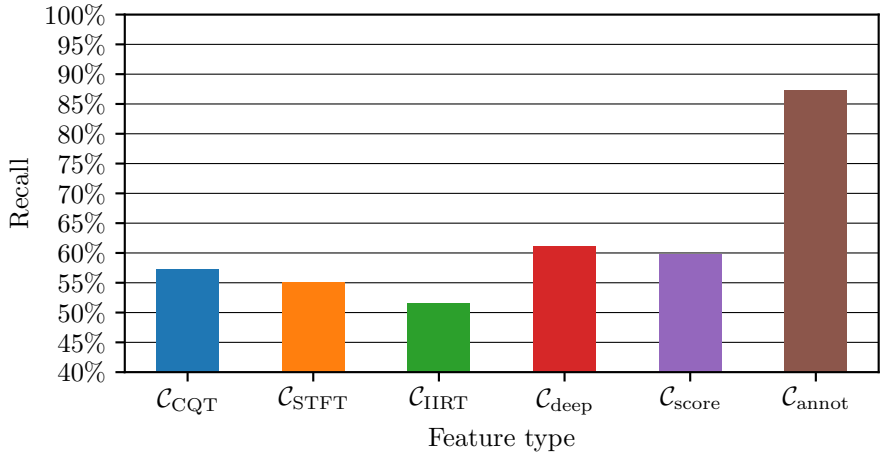
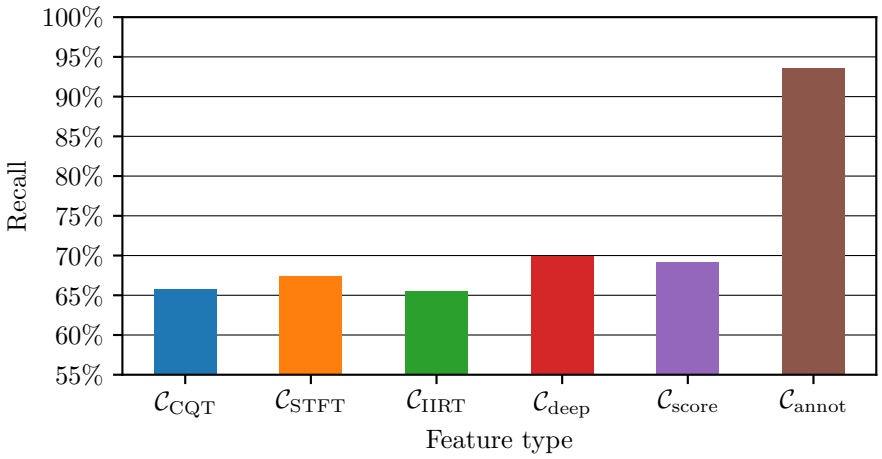
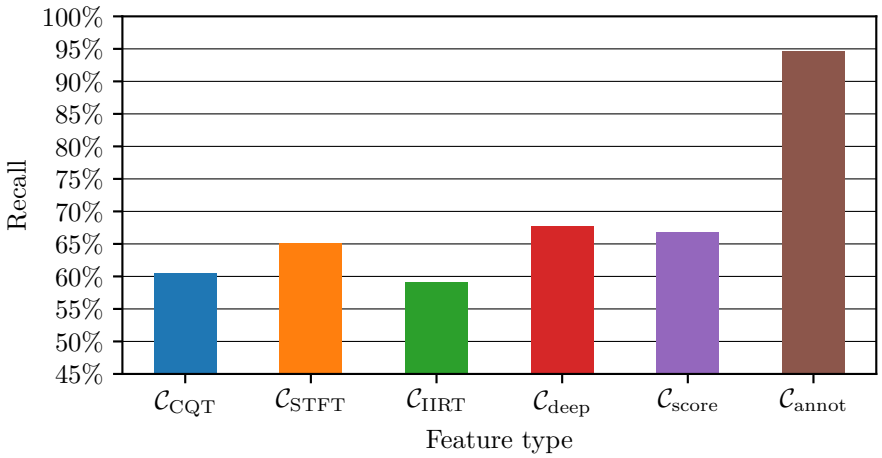


Figure 5.14. Recall values across the different feature types for **SWD**, **version split**. We show the results for $\texttt{HMM}_{\text{G}}$ with optimized parameters. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

a) major/minor



b) triad



c) seventh

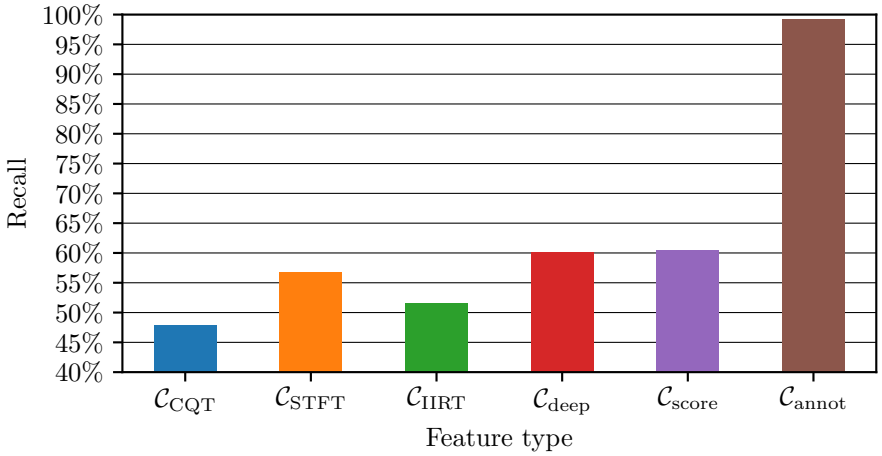
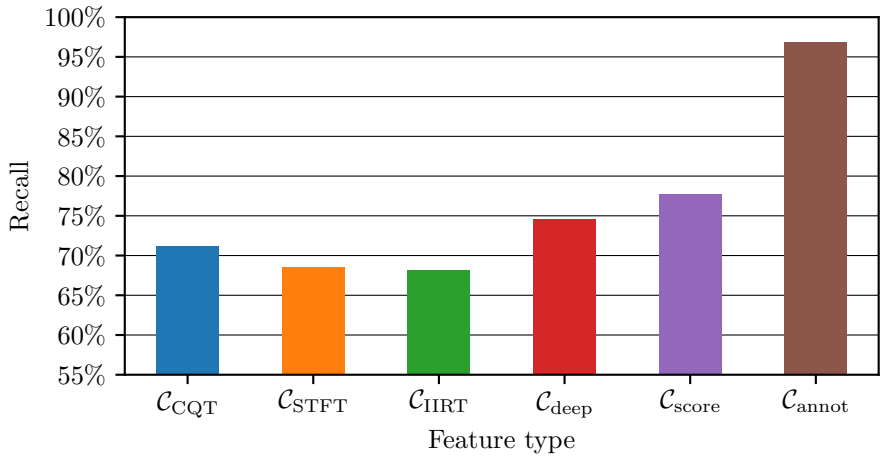
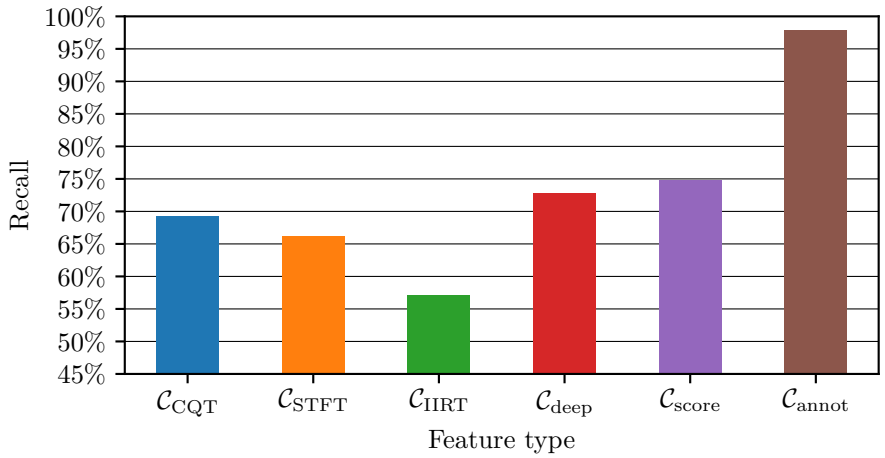


Figure 5.15. Recall values across the different feature types for **BSD**, **version split**. We show the results for $\texttt{HMM}_{\text{G}}$ with optimized parameters. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

a) major/minor



b) triad



c) seventh

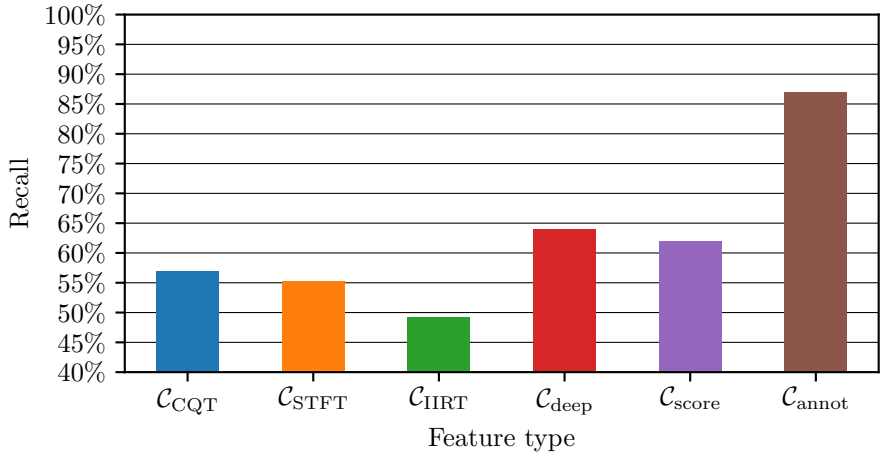
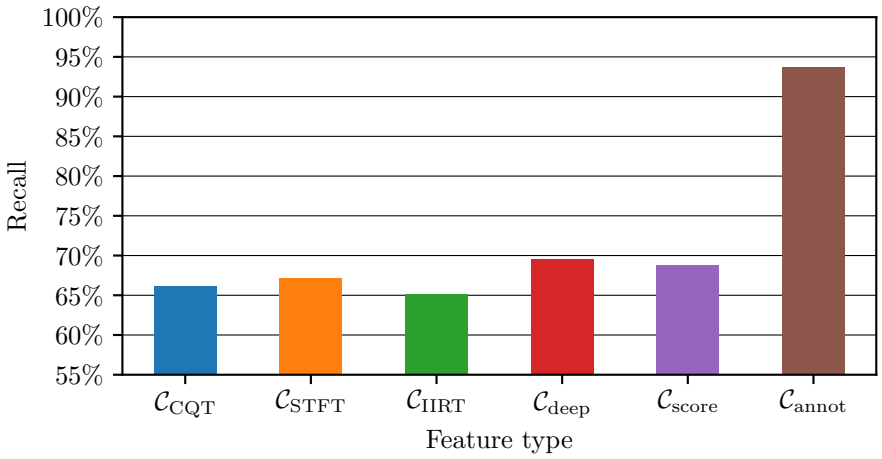
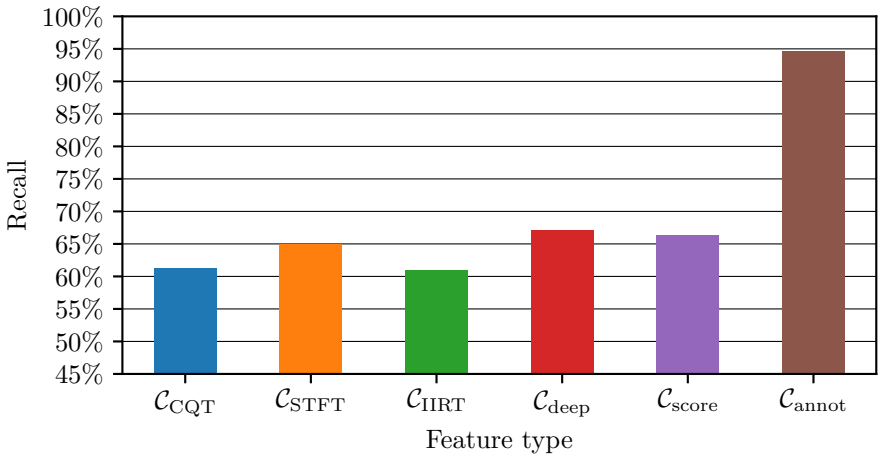


Figure 5.16. Recall values across the different feature types for **SWD**, **song split**. We show the results for $\texttt{HMM}_{\text{G}}$ with optimized parameters. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

a) major/minor



b) triad



c) seventh

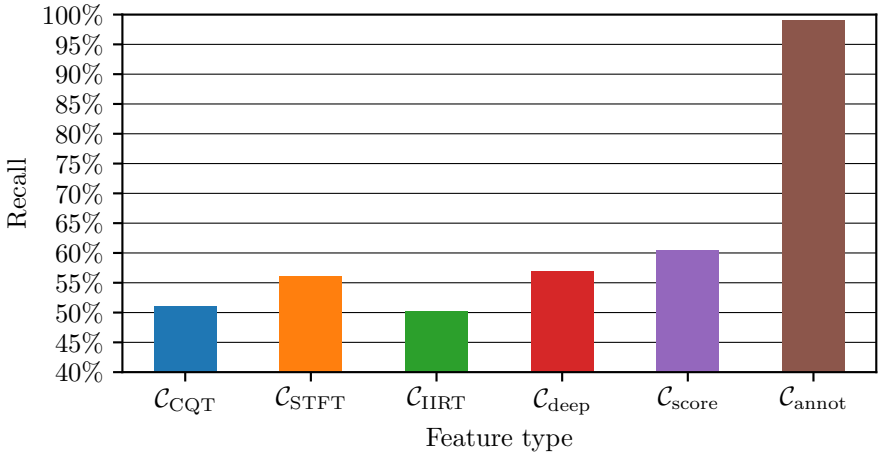
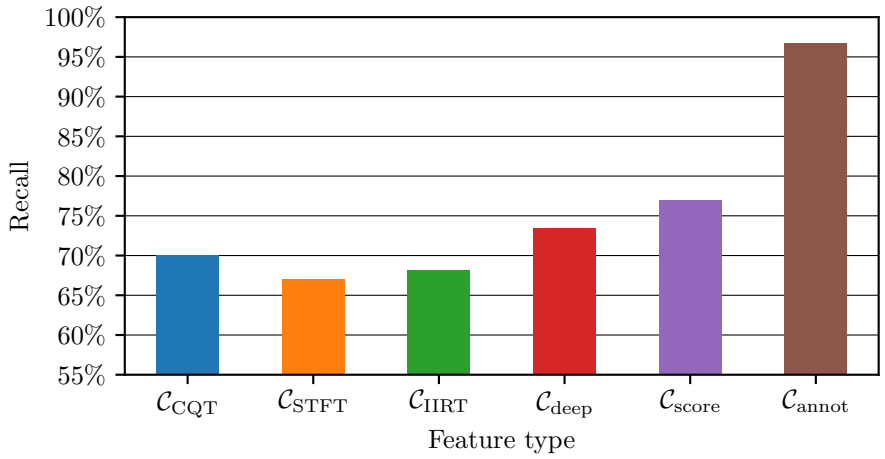
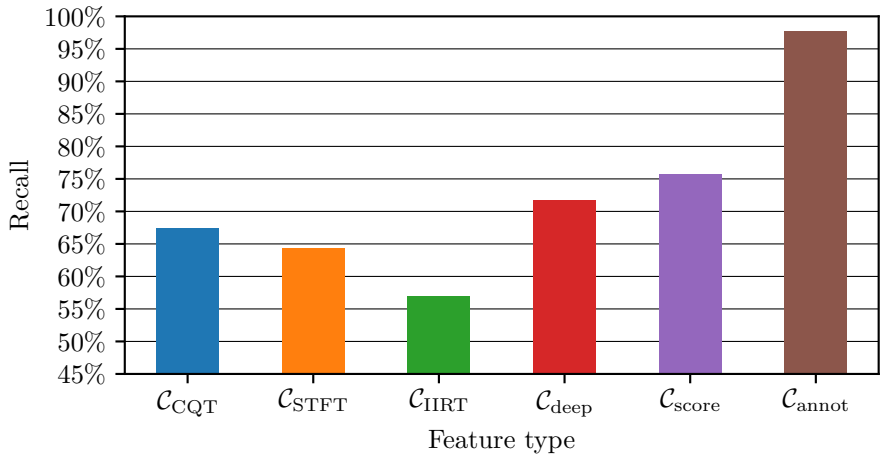


Figure 5.17. Recall values across the different feature types for **BSD**, **song split**. We show the results for $\texttt{HMM}_{\text{G}}$ with optimized parameters. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

a) major/minor



b) triad



c) seventh

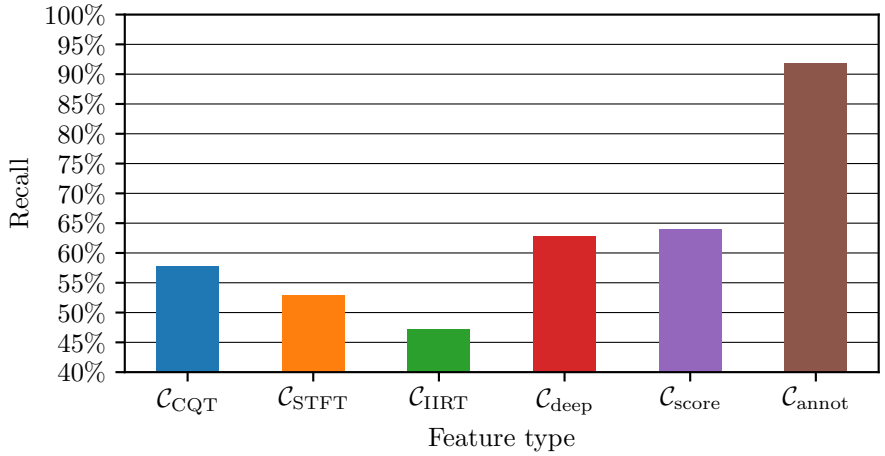
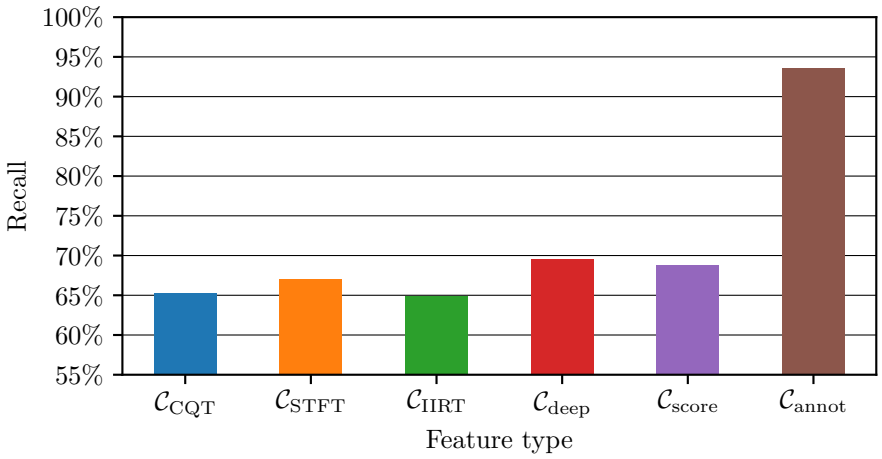
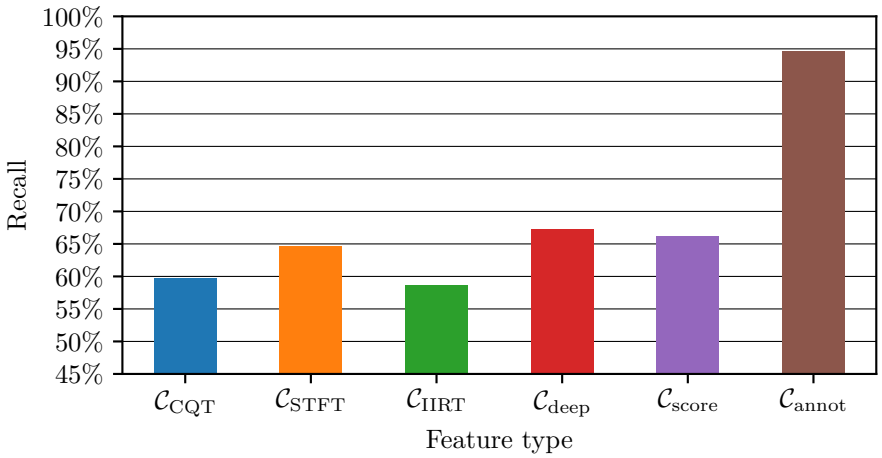


Figure 5.18. Recall values across the different feature types for **SWD**, **neither split**. We show the results for $\texttt{HMM}_{\text{G}}$ with optimized parameters. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

a) major/minor



b) triad



c) seventh

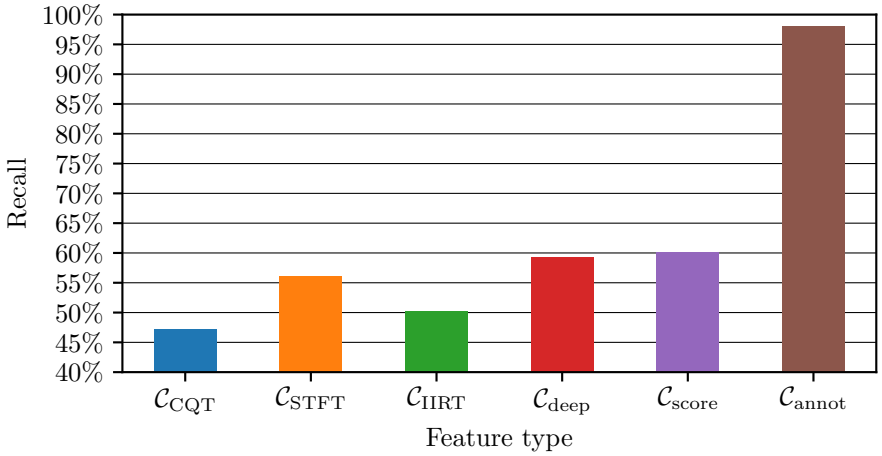
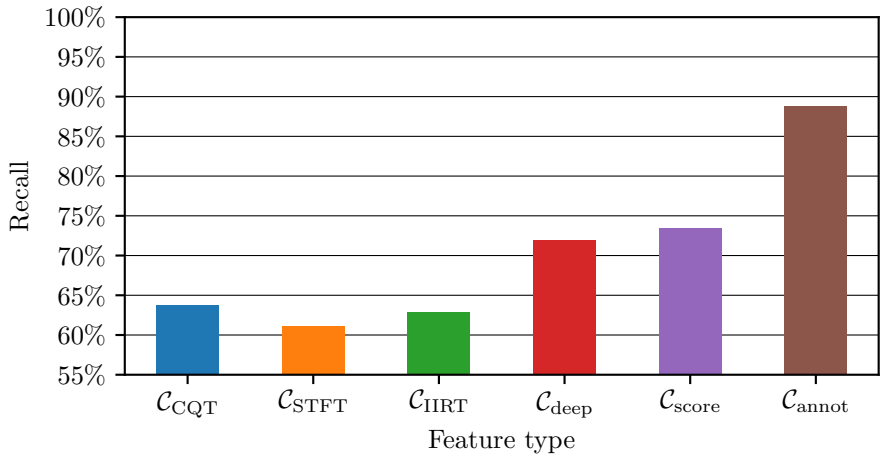
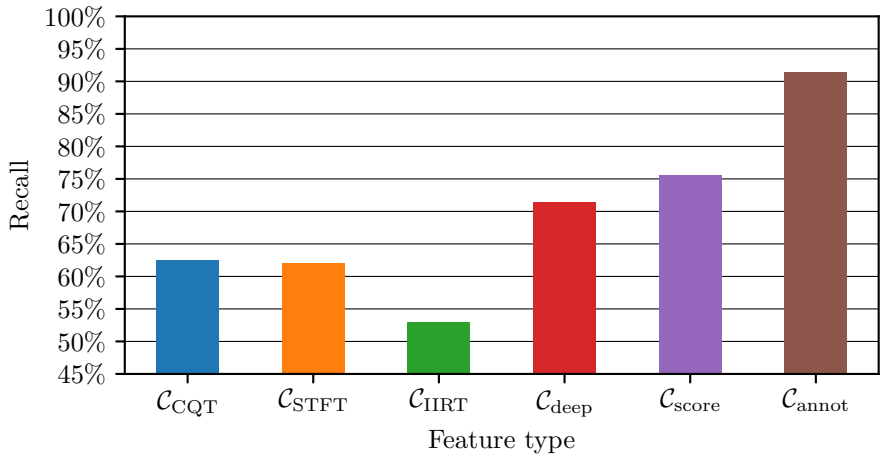


Figure 5.19. Recall values across the different feature types for **BSD**, **neither split**. We show the results for $\texttt{HMM}_{\text{G}}$ with optimized parameters. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

a) major/minor



b) triad



c) seventh

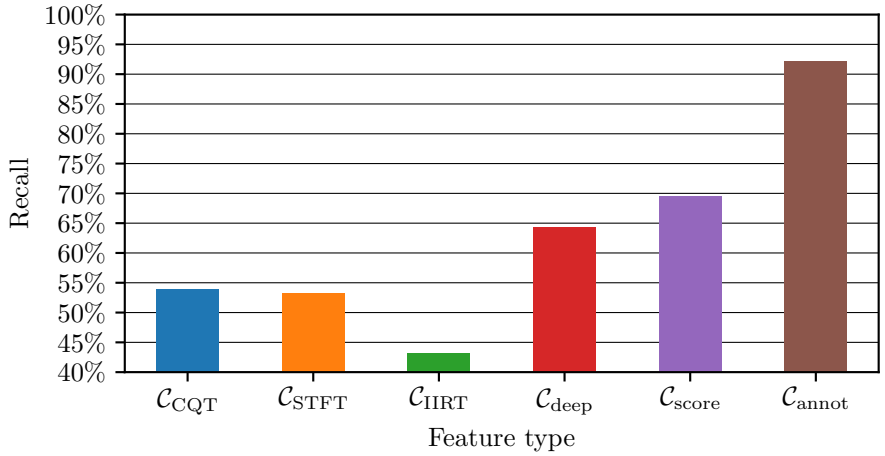
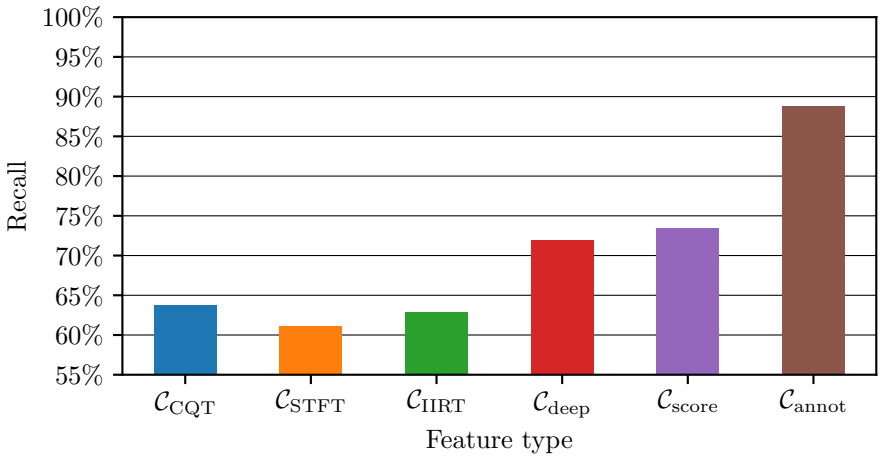
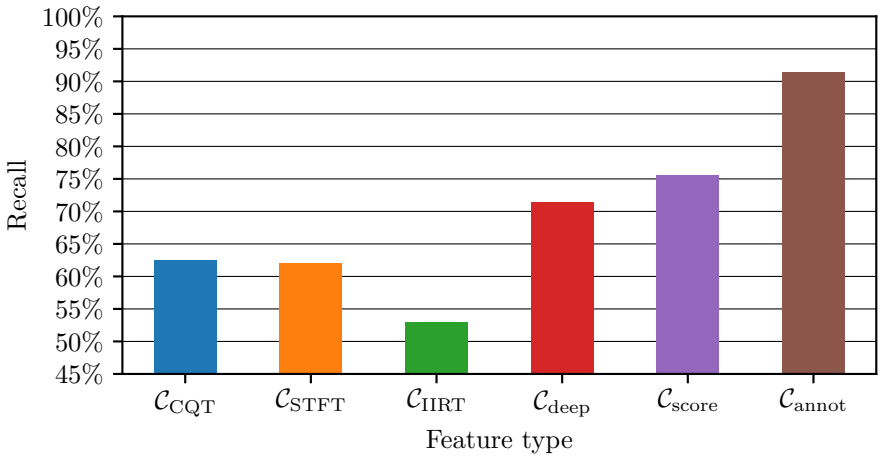


Figure 5.20. Recall values across the different feature types for **SWD**, **cross-dataset split**. We show the results for $\texttt{HMM}_{\text{G}}$ with optimized parameters. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

a) major/minor



b) triad



c) seventh

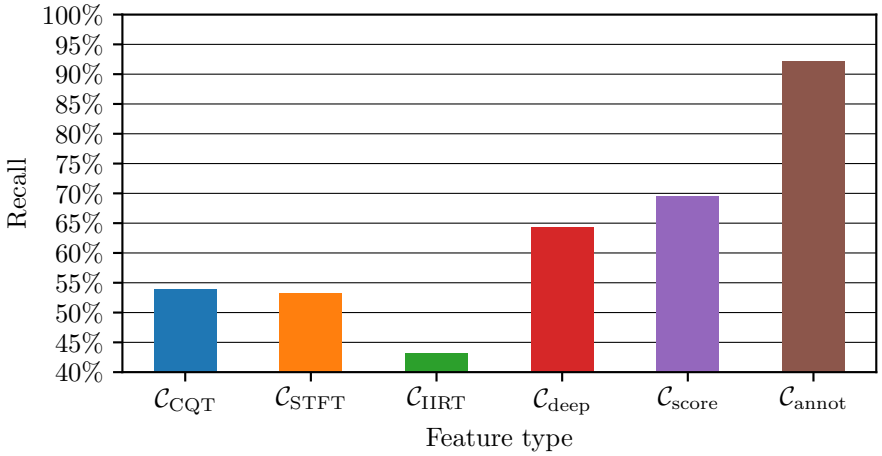


Figure 5.21. Recall values across the different feature types for **BSD**, **cross-dataset split**. We show the results for $\texttt{HMM}_{\text{G}}$ with optimized parameters. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

for the SWD, with a recall of approximately 67% for version and song split with major/minor vocabulary and $\mathcal{C}_{\text{STFT}}$. $\mathcal{C}_{\text{CQT}}$ and $\mathcal{C}_{\text{IIRT}}$ produce comparable results for the BSD, which are consistently lower than for $\mathcal{C}_{\text{STFT}}$. Like in the SWD, $\mathcal{C}_{\text{STFT}}$ produces the most consistent results across vocabularies and splits for the BSD, with a range of 53–67% recall from best to worst result. The overall worst result for the BSD is 43% recall for $\mathcal{C}_{\text{IIRT}}$, seventh vocabulary, and cross-dataset split.

In both datasets, chord recognition with $\mathcal{C}_{\text{deep}}$ consistently outperforms the results with the three signal processing chromas. In some cases, it even outperforms the results with the symbol-based chroma $\mathcal{C}_{\text{score}}$. For the SWD, the highest recognition quality with $\mathcal{C}_{\text{deep}}$ at approximately 77% recall can be obtained with the version split and major/minor vocabulary. Interestingly, the worst overall result for $\mathcal{C}_{\text{deep}}$ of approximately 61% is also acquired with the version split, with the seventh vocabulary. Especially for the more complex vocabularies, remarkable results can be achieved with $\mathcal{C}_{\text{deep}}$. Across all split variants, the recognition quality for the triad vocabulary decreases only slightly, compared to the major/minor vocabulary. The maximum recall difference is approximately two percent points. Even for the seventh vocabulary, a recall value of 64% can be achieved for the song and cross-dataset splits. For the BSD, the use of $\mathcal{C}_{\text{deep}}$ also produces higher recall values than the signal processing chromas. Here, the cross-dataset split provides the best results. The overall highest recall value for the BSD with $\mathcal{C}_{\text{deep}}$ is approximately 72% for the cross-dataset split with major/minor vocabulary. Furthermore, for the seventh vocabulary a maximum value of 64% recall can be achieved, similar to the SWD. The lowest overall recall value for $\mathcal{C}_{\text{deep}}$ in the BSD is 57% for the song split with seventh vocabulary. Generally, a gain in chord recognition quality can be achieved with $\mathcal{C}_{\text{deep}}$, compared to the signal processing chroma types. This comes at the cost of a much higher implementational effort. Additionally, the network for the deep chroma extractor has to be trained extensively. We presented the datasets used for training the chroma extractor in Section 4.1.

The results for $\mathcal{C}_{\text{score}}$ provide insight into chord recognition from symbolic data for both of our datasets. By using MIDI-based features, we eliminate the technical challenge of extracting chroma features from the actual audio recordings. We are still left with the musical challenge of assigning a chord label for the notes that are played. Hence, the difference in chord recognition quality between using $\mathcal{C}_{\text{score}}$ and the previously discussed feature types reveals the impact of the challenge of extracting pitch class content on the recognition process. For the SWD, $\mathcal{C}_{\text{score}}$ consistently provides better results than the three signal processing chroma features. The highest overall result is a recall of 79% for the version split with major/minor vocabulary. For the major/minor and triad vocabularies, the results for $\mathcal{C}_{\text{score}}$ are consistently five to ten percent points higher than with the signal processing chromas. For the seventh vocabulary, the difference is slightly lower with approximately five percent points. The exception here is the cross-dataset split, where $\mathcal{C}_{\text{score}}$ provides overall good results, similarly to the previously discussed $\mathcal{C}_{\text{deep}}$. Generally, there is only a slight difference between the results with $\mathcal{C}_{\text{deep}}$ and $\mathcal{C}_{\text{score}}$. Since the deep chroma extractor

was trained with $\mathcal{C}_{\text{score}}$ (of different datasets), this similarity is not surprising. In some cases, $\mathcal{C}_{\text{deep}}$ even provides better results. This reinforces the potential of chroma features obtained with deep-learning methods. For the BSD, $\mathcal{C}_{\text{score}}$ also provides better results than the signal processing chromas, but the difference is smaller. Across all vocabularies and splits, the maximum difference in recall is approximately two to five percent points. Again, the cross-dataset split provides an exception with a substantially larger recall difference. In general, the use of $\mathcal{C}_{\text{score}}$ increases the chord recognition quality up to approximately ten percent points in recall as compared to the signal processing chroma features. Still, the best result reached only 79% for major/minor vocabulary, leaving substantial room for improvement. From these results we can infer that the musical challenge of deriving correct chord labels from the played notes outweighs the technical challenge of acquiring feature vectors and the imperfections that come with it.

Going one step further, the use of $\mathcal{C}_{\text{annot}}$ provides a substantial reduction in musical complexity for the chord recognition task. By utilizing only the annotated chord notes, we further eliminate "musical noise" for the chord recognition, such as figurational notes. We are still left with the challenge of reducing and mapping the accurately annotated chord notes to the respective chord labels of our three different chord vocabularies. Across all splits, vocabularies, and both datasets we can see that the results for $\mathcal{C}_{\text{annot}}$ are substantially better than for all other feature types. The recall is consistently above 85%, reaching values up to approximately 99%. For both datasets, the figures show an increase in recognition quality when using the triad vocabulary as compared to the major/minor vocabulary. This result is most likely caused by the lower level of chord label mapping that is necessary for the triad vocabulary. For the seventh vocabulary, the recall reaches values of approximately 99% for the BSD with all three inner-dataset splits. Conversely, the recognition quality decreases for the SWD when using the seventh vocabulary. This discrepancy is most likely produced by the difference in harmonic language between the two datasets. As we show in Figures 3.2 and 3.3, parsing the chord annotations involves a lower level of reduction for the BSD than for the SWD. This can be seen in the amount of "other" chords, which we show in Subfigure d). The chords we include in the seventh vocabulary are better suited for the harmonic language of the BSD than for the SWD. Note that the chord annotations for the SWD tend to provide more detail than the annotations for the BSD. This could also influence our results. Overall, the experimental results with $\mathcal{C}_{\text{annot}}$ offer an interesting insight into the musical challenge of chord recognition and the influence of the chosen chord vocabulary.

5.6 In-Depth Analysis of Results

In the following, we discuss the chord recognition results on more detailed levels. In the previous sections, we reported recall values on the dataset level. Now, we report results for the individual

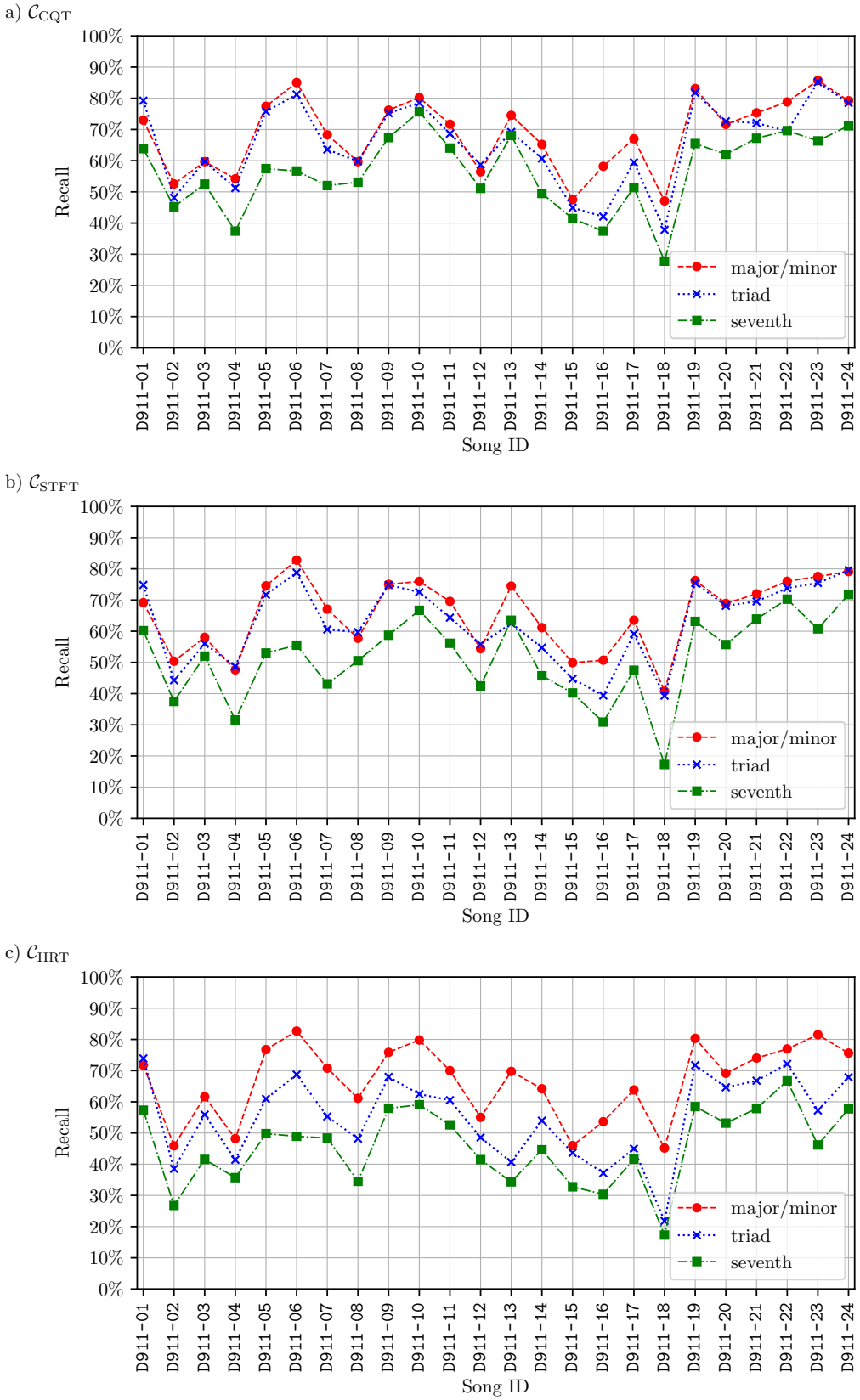


Figure 5.22. Song-wise recall values for SWD averaged across versions with $\texttt{HMM}_{\text{G}}$ and neither split. Results for major/minor, triad, and seventh vocabulary. a) $\mathcal{C}_{\text{CQT}}$. b) $\mathcal{C}_{\text{STFT}}$. c) $\mathcal{C}_{\text{IIRT}}$.

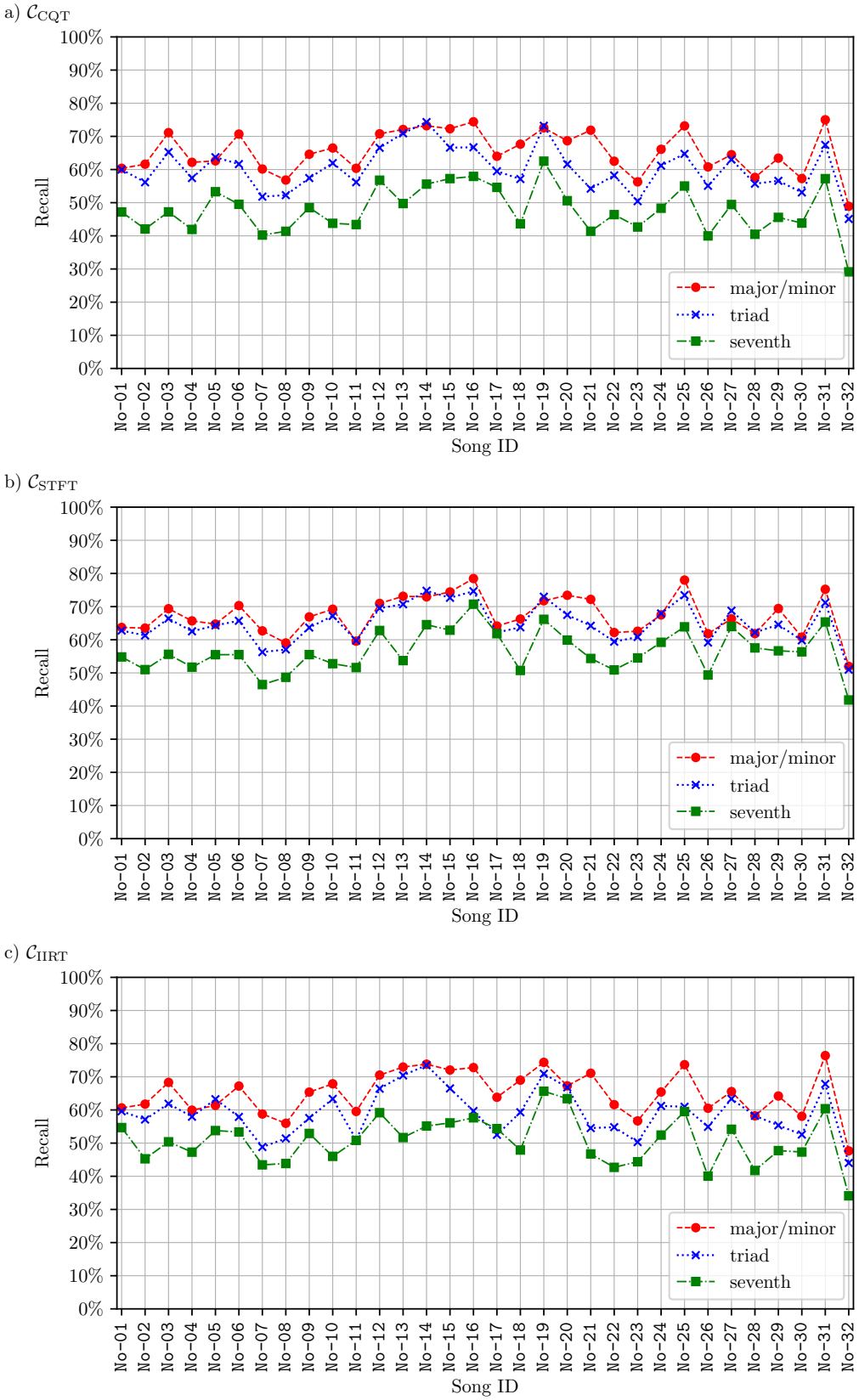


Figure 5.23. Song-wise recall values for BSD averaged across versions with $\texttt{HMM}_{\text{G}}$ and neither split. Results for major/minor, triad, and seventh vocabulary. a) $\mathcal{C}_{\text{CQT}}$. b) $\mathcal{C}_{\text{STFT}}$. c) $\mathcal{C}_{\text{IIRT}}$.

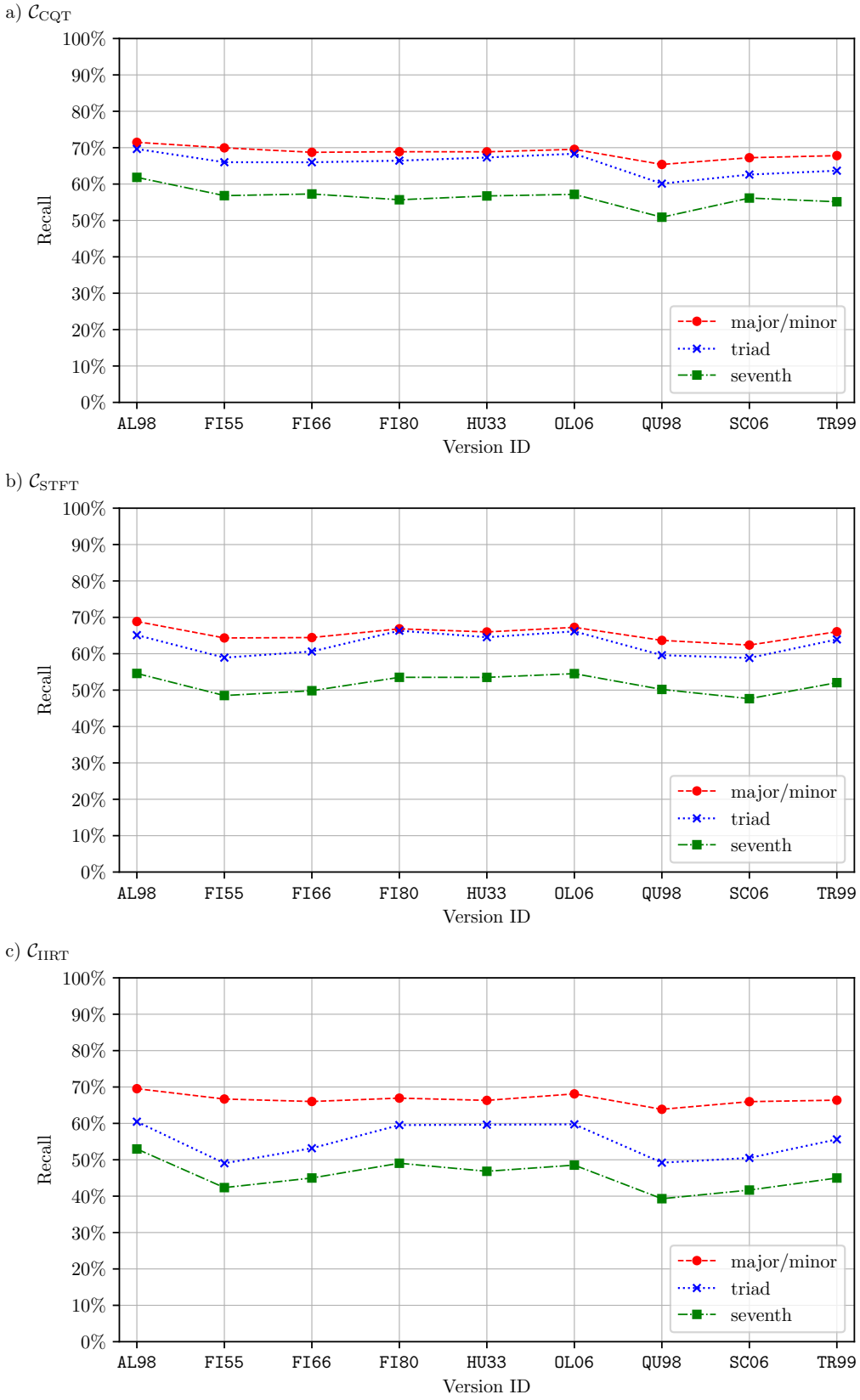


Figure 5.24. Version-wise recall values for SWD averaged across songs with $\texttt{HMM}_{\text{G}}$ and neither split. Results for major/minor, triad, and seventh vocabulary. a) $\mathcal{C}_{\text{CQT}}$. b) $\mathcal{C}_{\text{STFT}}$. c) $\mathcal{C}_{\text{IIRT}}$.

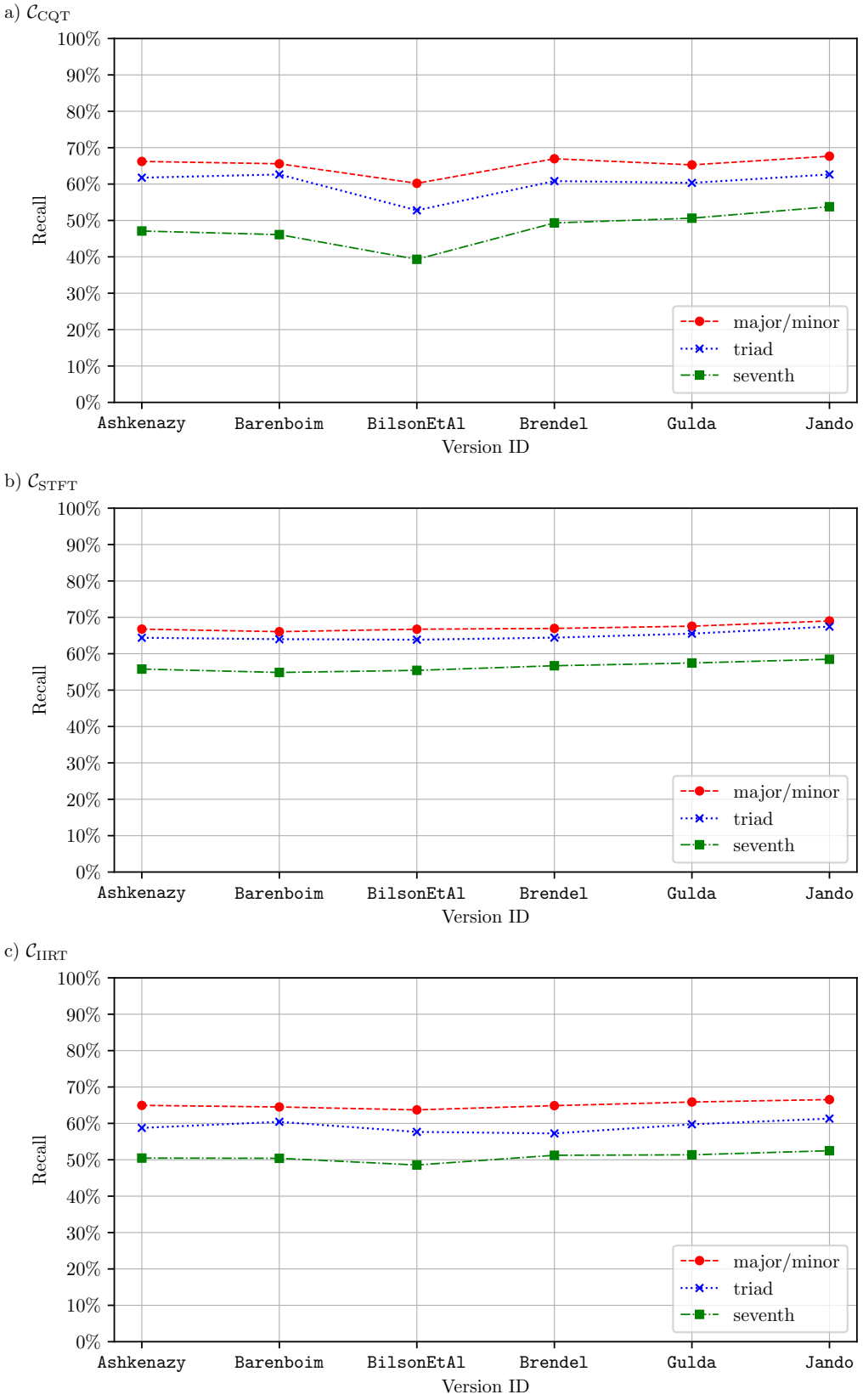


Figure 5.25. Version-wise recall values for BSD averaged across songs with $\mathtt{HMM}_{\mathrm{G}}$ and neither split. Results for major/minor, triad, and seventh vocabulary. a) $\mathcal{C}_{\mathrm{CQT}}$. b) $\mathcal{C}_{\mathrm{STFT}}$. c) $\mathcal{C}_{\mathrm{IIRT}}$.

songs and versions of each dataset. Subsequently, we move on to the track level and finally we report results on the measure level of individual songs.

Figures 5.22 and 5.23 show the recall values for the individual songs of both datasets. Every individual value is averaged across all versions of the respective dataset. It is important to note that we use averaged track-wise recall values. The results we presented so far were frame-wise recall values. This means that in the following figures, the recall value for each individual track is weighted equally, although the length of each track might differ. We report the results for $\mathcal{C}_{\text{CQT}}$ in Subfigure a), $\mathcal{C}_{\text{STFT}}$ in b), and $\mathcal{C}_{\text{IIRT}}$ in c). In each plot we show the values for all three chord vocabularies as individual lines with different line styles, colors, and markers.
For the SWD, we can clearly see a high variance in chord recognition quality across the individual songs. This is the case for all three chroma types. For some songs, such as `D911-06` or `D911-19`, recall values of over 80% can be achieved. The worst results are obtained for `D911-18`, with a major/minor performance of approximately 50% and recall values below 30% for the seventh vocabulary. With the exception of a small number of outliers, the results for all three chroma types follow similar trends along the song axis. The same is true for the results for the major/minor and triad chord vocabularies. Both lines exhibit similar trends, with slightly smaller values for the triad vocabulary. Noteworthy exceptions are `D911-01`, where the triad vocabulary provides better results across all three chroma types, and `D911-16`, where the results for the triad vocabulary show an opposite trend as compared to the major/minor results. Since all three chroma types follow similar trends for all songs and we acquired comparable results for $\mathcal{C}_{\text{score}}$, the recall differences between individual songs can likely be traced back to musical characteristics. As an example, song `D911-18`, *Der stürmische Morgen* (eng. *The Stormy Morning*), is a fast paced song with a restless character. It contains a large number of broken chords and purely melodic structures with occasional chromatic ornaments. In contrast, song `D911-06`, *Wasserflut* (eng. *Flood*) is a slow and solemn song with a more melancholic character. It contains many sustained chords in the piano accompaniment, the singing voice often exhibits broken triads containing the underlying chord notes. Musical characteristics such as these can have a large influence on the chord recognition quality.

For the BSD, the results across individual songs exhibit a smaller variance compared to the SWD. Across all three chroma variants, the recall values for major/minor and triad vocabulary mostly lie between 50–80%, the values for the seventh vocabulary mostly lie between 40–70%. Again, the results for all three vocabularies largely follow the same trends along the song axis for all three chroma variants. The best results are achieved for `No-31`, the worst results are obtained for `No-32`. A likely reason for the smaller differences between songs is the longer duration and larger musical heterogeneity of each song in the BSD, as compared to the SWD. Most of the sonata movements contain sections of varying musical complexity and characteristics, so the difficulty

of chord recognition might be "evened out" along the full duration of each song. In the SWD, individual songs exhibit a more homogeneous musical character and are substantially shorter.

In Figures 5.24 and 5.25, we show the recall values for the individual versions of both datasets. Each value is averaged across all songs of the respective dataset. We again report the values for $\mathcal{C}_{\mathrm{CQT}}$, $\mathcal{C}_{\mathrm{STFT}}$, and $\mathcal{C}_{\mathrm{IIRT}}$ in each subfigure respectively, with an individual line for each of the three chord vocabularies. For both datasets, the results across versions exhibit more prominent fluctuations than the results across songs. In the SWD, the average recall for `AL98` is slightly higher compared to the other versions. The lowest values are achieved for versions `FI55`, `QU98`, and `SC06`. For each vocabulary type, the maximum difference across versions is approximately five to ten percent points for each chroma variant. In the BSD, the differences between versions are on a similarly low level. Here, the results with $\mathcal{C}_{\mathrm{CQT}}$ exhibit slightly larger fluctuations, compared to $\mathcal{C}_{\mathrm{STFT}}$ and $\mathcal{C}_{\mathrm{IIRT}}$. For the latter, the maximum difference across versions for each vocabulary is approximately five percent points, for $\mathcal{C}_{\mathrm{CQT}}$ the maximum difference is approximately ten percent points. For all three chroma and vocabulary variants, version `Jando` provides slightly higher recall values. For $\mathcal{C}_{\mathrm{CQT}}$ and $\mathcal{C}_{\mathrm{IIRT}}$ version `BilsonEtAl` produces the lowest recall values.

In Figures 5.26 and 5.27, we show the recall values for each individual track in both datasets for $\mathcal{C}_{\mathrm{CQT}}$. Additionally, we report the averaged results across each song, each version, and all tracks. The average across rows corresponds to the song-wise recall values we showed in Subfigures 5.22 a) and 5.23 a), the average across columns corresponds to the version-wise recall values we showed in Subfigures 5.24 a) and 5.25 a). The recall values are underlined with corresponding grayscale values. In Appendix B, we additionally show the track-wise recall values for $\mathcal{C}_{\mathrm{STFT}}$, $\mathcal{C}_{\mathrm{IIRT}}$, and $\mathcal{C}_{\mathrm{deep}}$. Previously, we reported only small differences in chord recognition quality for different versions. While this is true for the average across all songs, the track-wise recall values reveal larger local differences. As an example, song `D911-24` in version `QU98` in the SWD produces lower recall values than the other versions of that song. This is true for all three chord vocabularies. In the BSD, similar occurrences can be found. For instance, version `BilsonEtAl` produces lower recall values for songs `No-07` and `No-26` than the other versions for all three chord vocabularies. From these results we can conclude that a track-wise analysis offers deeper insights into the local recall fluctuations between individual versions, songs, chroma variants, and vocabularies.

For a more in-depth analysis of recognition results and common difficulties, we can analyze individual songs on the measure level. The use of a musical time axis such as measures allows us to compare the results obtained from different versions. In Figures 5.28, 5.29, and 5.30, we present three short examples, two from the SWD, and one from the BSD. In Appendix C, we show corresponding excerpts from the original score. To convert our frame-based time axis to the measure axis, we used the audio-aligned measure annotations. We used a temporal resolution of 1/4 measures and linearly interpolated the time between measure borders, ignoring the actual

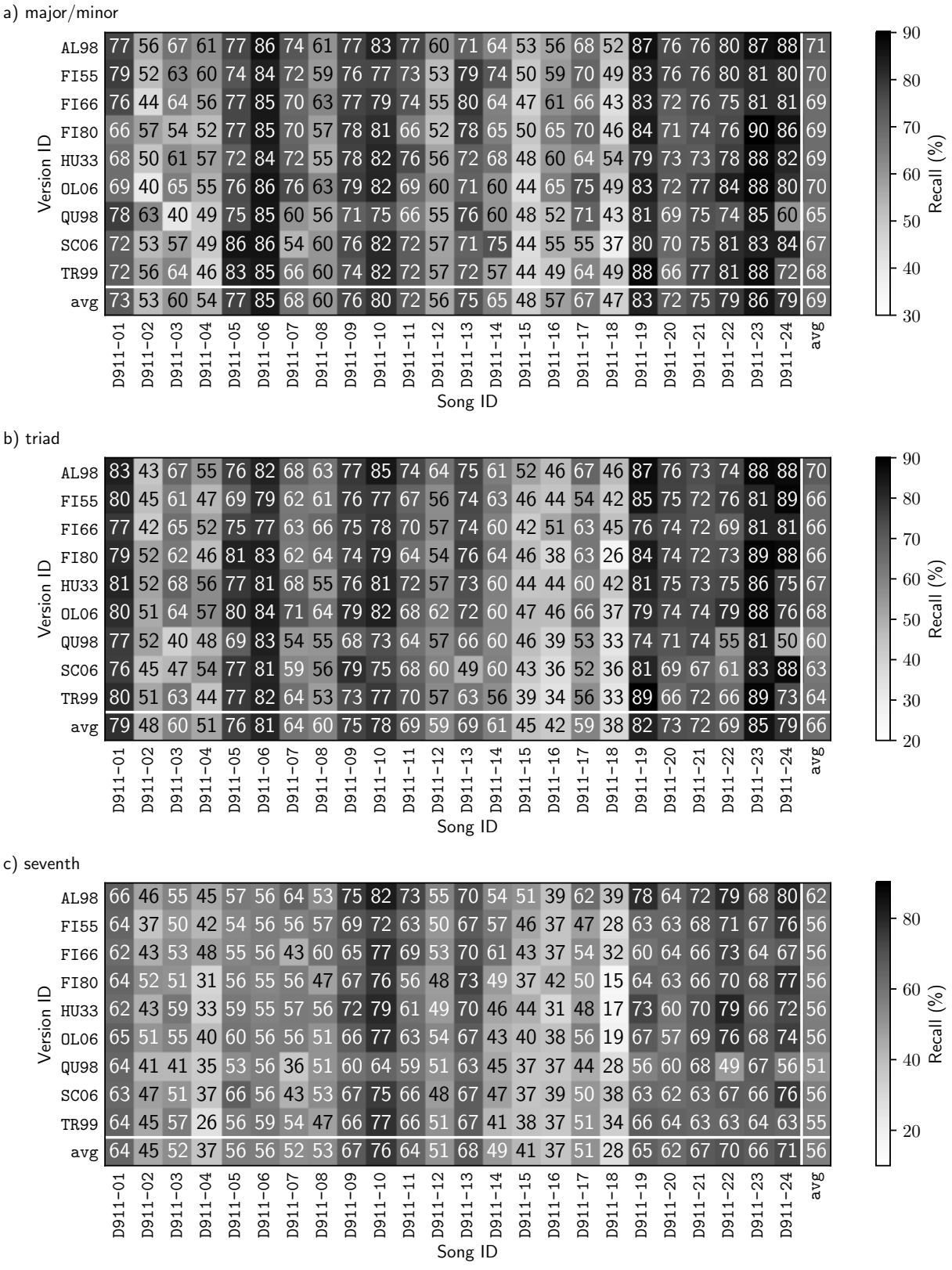


Figure 5.26. Track-wise recall values for SWD with $\texttt{HMM}_{\text{G}}$, $\mathcal{C}_{\text{CQT}}$, and neither split. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

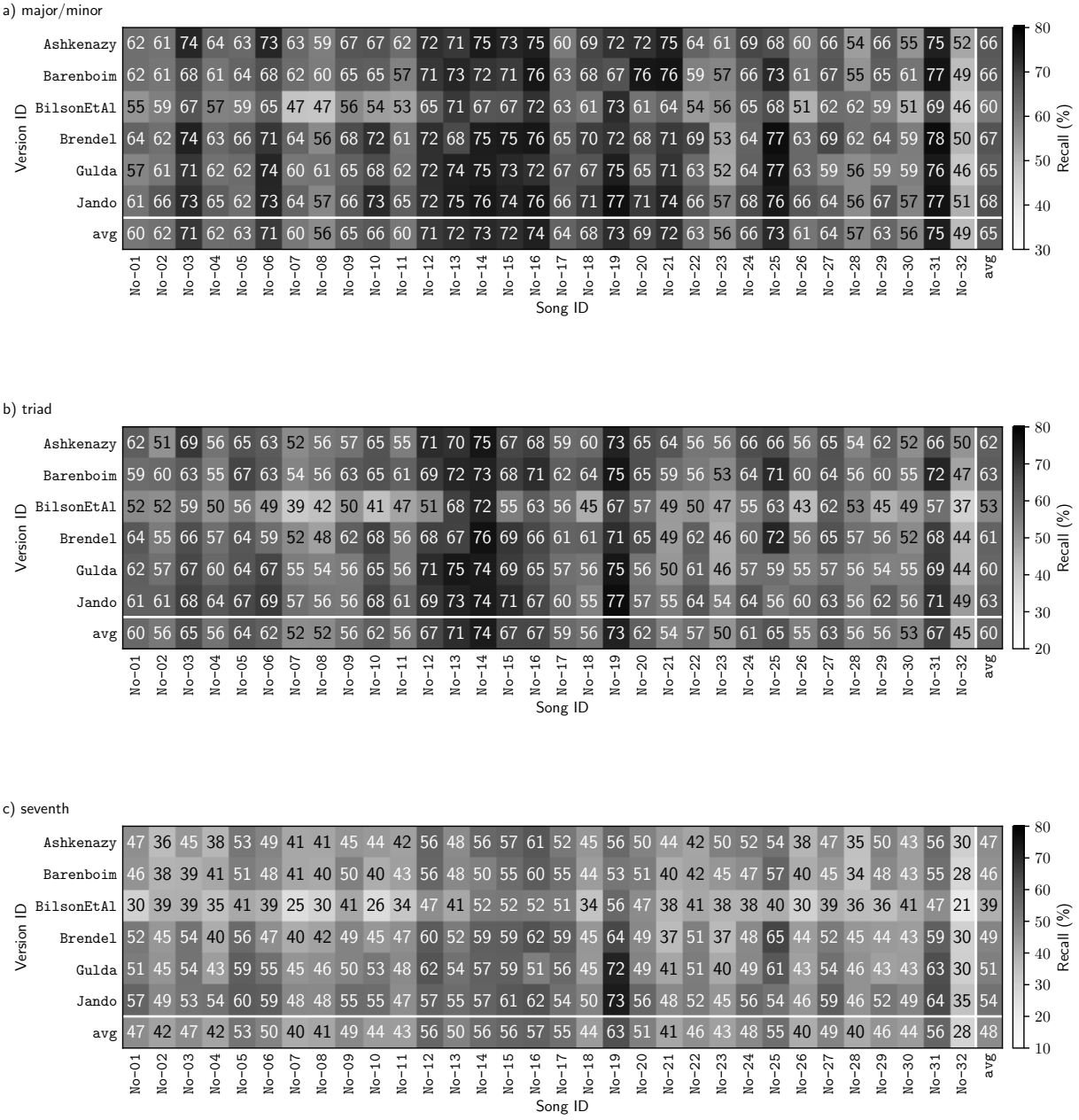


Figure 5.27. Track-wise recall values for BSD with $\texttt{HMM}_{\text{G}}$, $\mathcal{C}_{\text{CQT}}$, and neither split. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

musical time signature. To obtain one chord label per 1/4 measure segment, we implemented a majority vote among all time frames falling into that temporal segment. Additionally, we transposed the results from each individual version and the chord annotations to the global key of the original score to obtain comparable results across versions. In the figures, we show the ground truth annotations marked in black. Incorrect results from the chord recognition for each version are marked in different shades of red. The darker the shade is, the more versions commonly produced the same erroneous result. For the SWD, the darkest shade is obtained when all 9 versions made the same mistake, for the BSD, the darkest shade is obtained for all 6 versions. This presentation enables us to simultaneously see which errors are made in the chord recognition and if these errors were made commonly for multiple versions.

In Figure 5.28, we present the first nine measures of song `D911-05`, *Der Lindenbaum* (eng. *The Linden Tree*) from the SWD for the triad chord vocabulary. The results were acquired with $\mathcal{C}_{\text{IIRT}}$ and the neither split. A common error in chord recognition is the correct recognition of the root note but a false classification of the chord quality. Examples for this error can be seen in measures two, four, five, and nine. It is especially prominent in measure two, where the figure shows that all versions produced a confusion of the BM chord with a Bm chord. Generally, chord recognizers tend to incorrectly recognize chords that share one or multiple chord notes with the actually played chord. For instance, in measure six we can see that for all versions, the annotated F♯m chord is incorrectly recognized as a D♯dim chord. The triad notes of F♯m are F♯, A, and C♯, the triad notes of D♯dim are D♯, F♯, and A. Hence, both chords share two out of three triad notes. "Musically understandable" errors such as these are among the most common mistakes in chord recognition. The more chords a chord vocabulary contains, the more similar chords are available which can produce such confusions.
Figure 5.29 shows measures 50–59 of song `D911-22`, *Mut!* (eng. *Have Courage!*) from the SWD. Here, we want to highlight two additional errors that commonly occurred in our evaluations. Due to the high self-transition probability we implement when using $\texttt{HMM}_{\text{G}}$, temporally short chord changes are sometimes not recognized. This is especially true, when the subsequent chord change goes back to the initial chord. An example for this can be seen in measures 51 and 58. In both cases, the chord change to FM and DM, respectively, is not recognized. Song `D911-22` is a fast paced song with a 2/4 time signature, therefore the chord change lasts only for a single, short beat. This mistake is made in all versions of this example. The second common error we want to highlight can be seen in measures 55 and 56. Here, DM is recognized as Daug for some versions. We often noticed confusions from major to augmented chords, as well as from minor to diminished chords. A likely reason for this mistake might be the training of our chord models. The feature vectors we use for training augmented and diminished chord models often contain a strong perfect fifth note as well as the "correct" augmented or diminished fifth note. This might be an artifact caused by the harmonics of the root note. Since major and augmented as

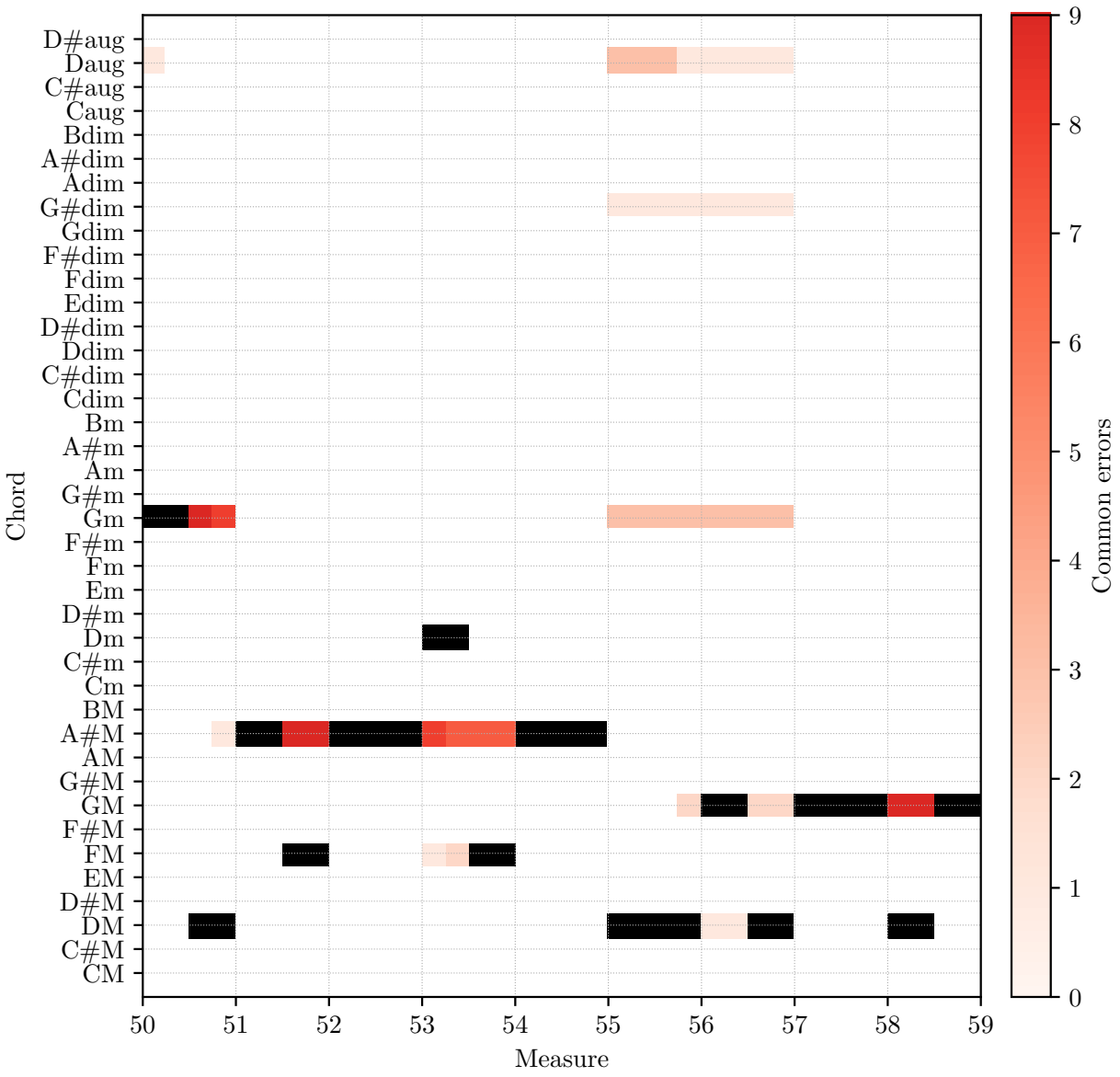


Figure 5.28. Chord recognition results for $\mathcal{C}_{\text{IIRT}}$ on the measure level with triad vocabulary. We present the first nine measures of song `D911-05`, *Der Lindenbaum* (eng. *The Linden Tree*) from the SWD. Ground truth annotations are marked black, incorrect chord recognition results are marked red. The shade of red indicates, how many versions produce the same error.

well as minor and diminished chords contain the same third interval, this often led to a quality confusion. Obviously, this was only the case when using the triad and seventh vocabulary.
In Figure 5.30, we present the first nine measures of song `No-14`, famously known as *Mondscheinsonate* (eng. *Moonlight Sonata*) from the BSD for the major/minor chord vocabulary. Here, we can again see some of the common errors we mentioned previously. In measures one, two, and seven, a confusion from C♯m to C♯M occurs. In the case of measure seven, it even occurs for all versions simultaneously. In measure three, we can see a confusion from DM to F♯m. This

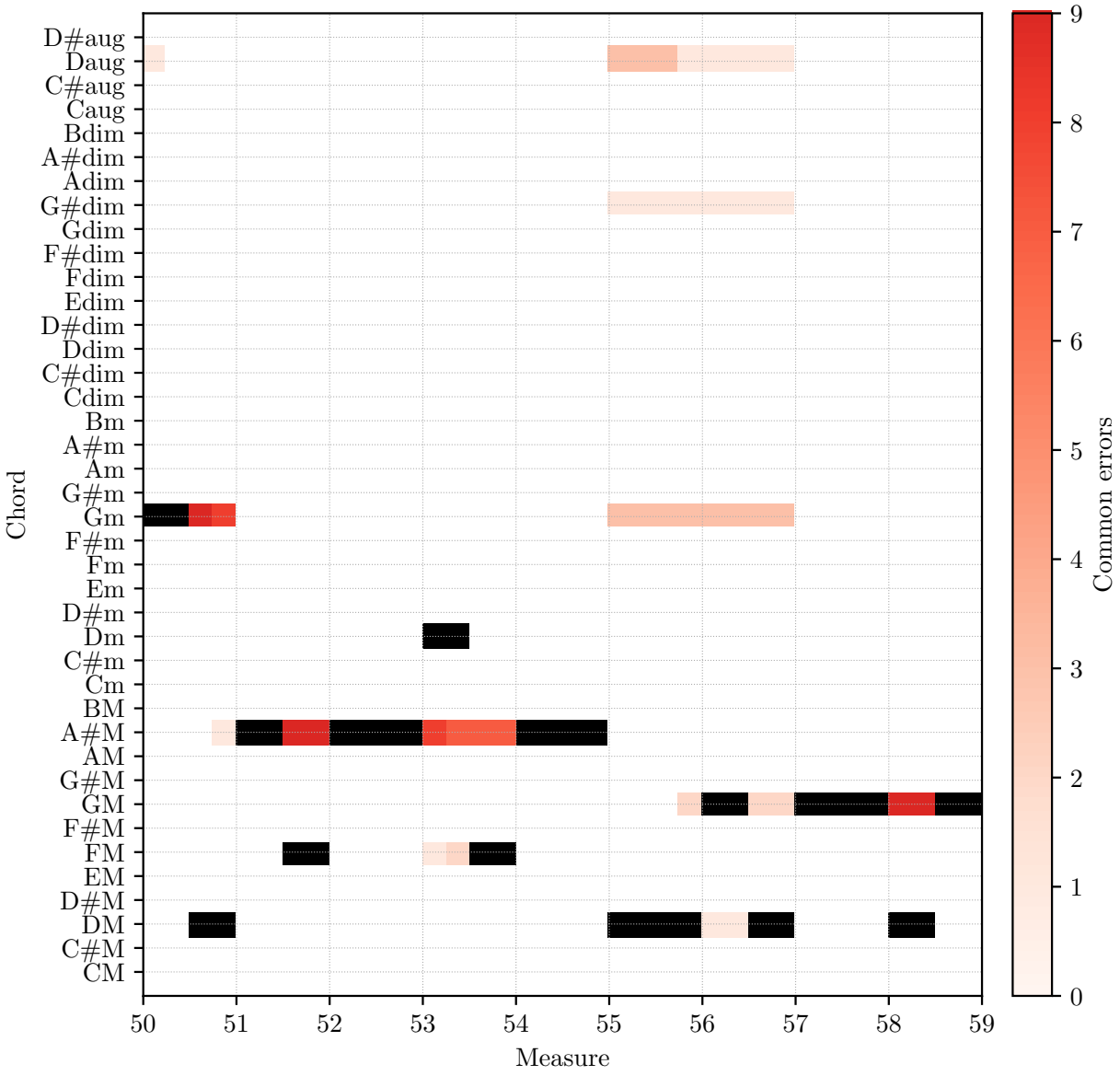


Figure 5.29. Chord recognition results for $\mathcal{C}_{\text{IIRT}}$ on the measure level with triad vocabulary. We present measures 50–59 of song `D911-22`, *Mut!* (eng. *Have Courage!*) from the SWD. Ground truth annotations are marked black, incorrect chord recognition results are marked red. The shade of red indicates, how many versions produce the same error.

confusion is understandable, since F♯m contains F♯ and A, which are two of the three triad notes of DM. In measure eight, an EM chord is incorrectly recognized as a G♯m chord in five out of six versions. Again, both chords share two triad notes.

In order to evaluate how often errors occur for multiple versions simultaneously, we also used the measure-based time axis. In Figure 5.31, we show a visualization of the amount of 1/4 measure segments where errors occur for one or more versions. The occurrences are accumulated,

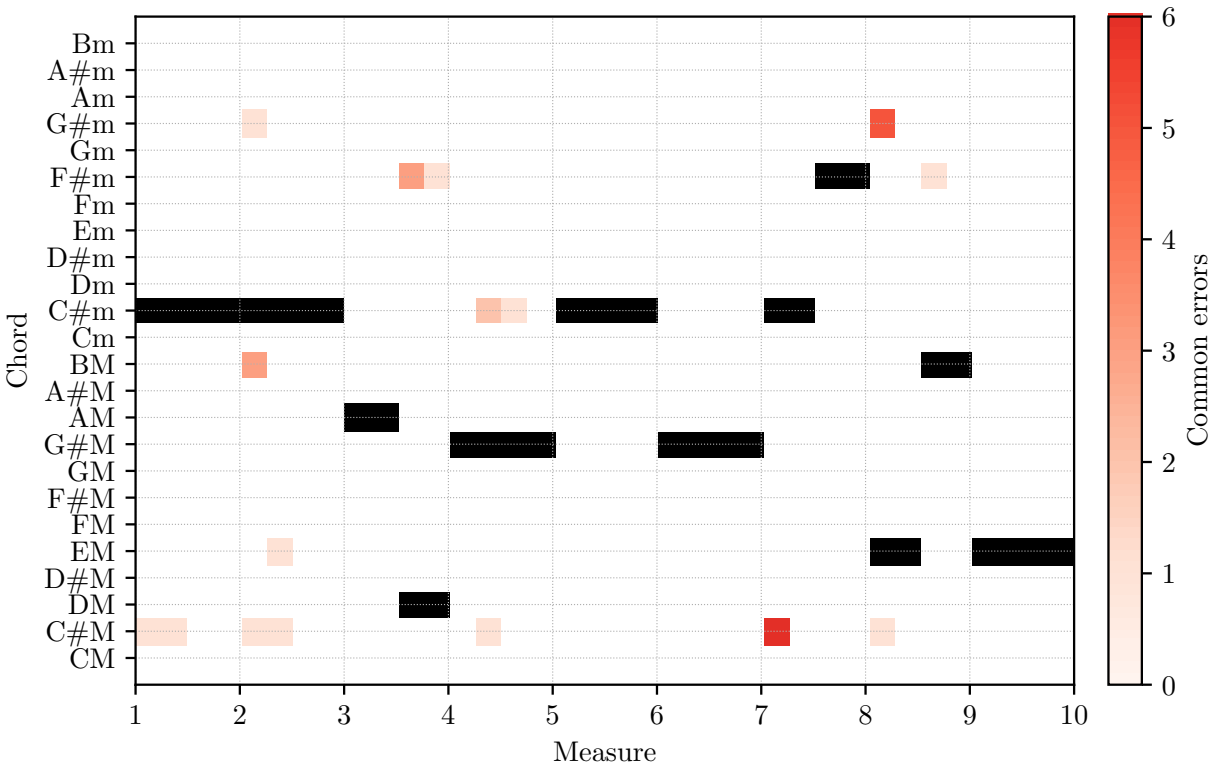


Figure 5.30. Chord recognition results for $\mathcal{C}_{\text{CQT}}$ on the measure level with major/minor vocabulary. We present the first nine measures of song `No-14`, famously known as *Mondscheinsonate* (eng. *Moonlight Sonata*) from the BSD. Ground truth annotations are marked black, incorrect chord recognition results are marked red. The shade of red indicates, how many versions produce the same error.

beginning from points where all versions simultaneously produce an error. The numbers are evaluated for the major/minor vocabulary with $\mathcal{C}_{\text{CQT}}$ and the neither split. In Subfigure a) we show the results for the SWD, in Subfigure b) we show the results for the BSD. For both datasets, we can see that in approximately 25% of all 1/4 measure segments where an error occurs, all versions simultaneously produce an error. It should be noted that we do not evaluate whether or not they produce the same error. In the SWD, we can see that for approximately 22% of all segments where an error occurs, only one version produces an error. For the BSD this is the case for approximately 25%. These observations highlight the potential of cross-version fusion strategies. Figure 5.31 shows that in approximately 50–60% of all segments with errors, the majority of versions produce the correct recognition result. Fusing the results for each measure segment across versions, using, e.g., majority voting, could help to overcome a substantial part of errors.

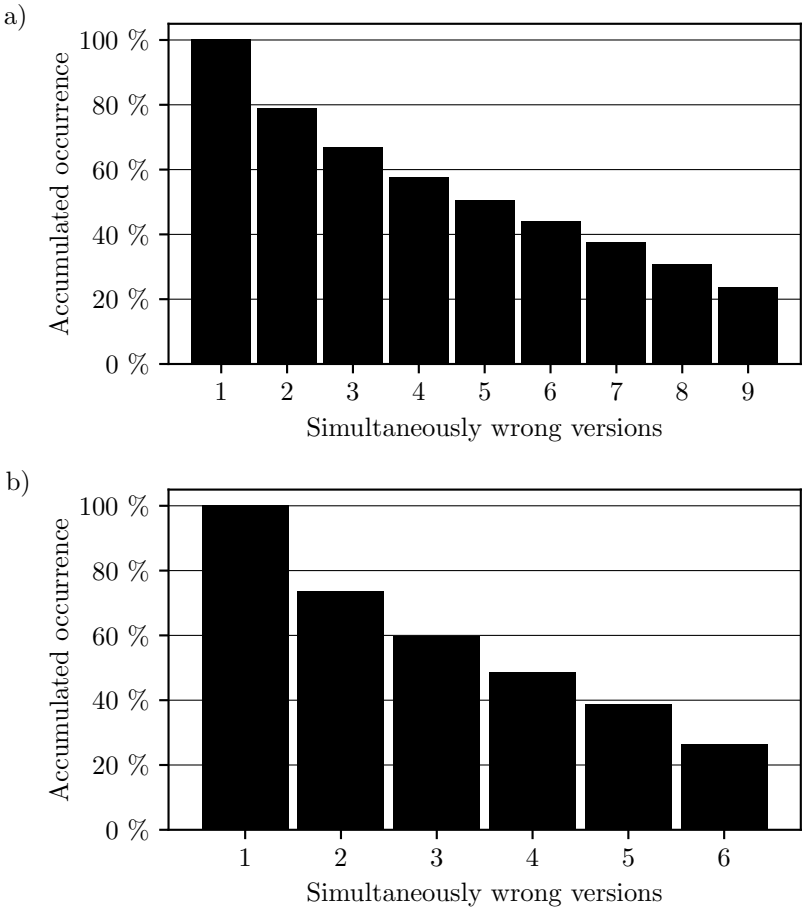


Figure 5.31. Cumulative representation of the 1/4 measure segments where errors occur. Each bar represents the percentage, where "at least x versions produced an error simultaneously". a) For SWD. b) For BSD.

Chapter 6

Conclusions

In this thesis we presented a study of automatic chord recognition in the context of audio recordings of classical music. We analyzed two corpora of music from the late Classical and Romantic period, the Schubert Winterreise Dataset (SWD) and the Beethoven Piano Sonatas dataset (BSD). Since both datasets contain multiple versions of the same songs, the datasets are well-suited for a cross-version analysis.

In our experiments, we initially focused on the influence of parameters for pitch weighting, logarithmic compression, and moving median filtering as pre-filtering methods as well as the influence of the self-transition probability of HMMs as post-filtering method. Our experiments showed that logarithmic compression and a suitable choice of the self-transition probability have a major influence on the chord recognition results. While it is important to set the parameter values within a meaningful range, our experiments showed no prominent gain from micro-adjusting each parameter. With a two-dimensional grid search, we showed the interplay between parameter variations for logarithmic compression and the self-transition probability. Variations of one parameter can cause a change of the optimal value for the other parameter. This suggests a joint optimization of parameter values.

To train our chord models and optimize parameter values, we applied different data splits for cross-validation. We split along the song, version, and dataset axis. For the SWD, splitting along the version axis provided slightly better results than the other split variants, implying a better generalization across versions than across songs. A likely reason for this is the high diversity of musical characteristics of the individual songs of the SWD. Training on all songs therefore provides better generalization than training on all versions. For the BSD, the song split provided the best results, implying the opposite case. For both datasets, neither and cross-dataset split produced the worst results.

We evaluated the chord recognition results for three different vocabularies, the major/minor, triad, and seventh vocabulary. We reached the best recognition quality for the major/minor

vocabulary, which corresponds to our expectations. Yet, the results for the triad vocabulary were only slightly lower with a decrease in recall of approximately five percent points. Furthermore, we reported a more prominent reduction in chord recognition quality for the seventh vocabulary. The recall values decreased by approximately 15 percent points compared to the results with the major/minor vocabulary.

The comparison of different chroma feature types constitutes one of the central contributions of this thesis. We compared the three signal processing chromas $\mathcal{C}_{\mathrm{CQT}}$, $\mathcal{C}_{\mathrm{IIRT}}$, and $\mathcal{C}_{\mathrm{STFT}}$, the deep-learning chroma $\mathcal{C}_{\mathrm{deep}}$, and the two symbolic baseline chromas $\mathcal{C}_{\mathrm{score}}$ and $\mathcal{C}_{\mathrm{annot}}$. While all three signal processing chromas produced comparable results, our experiments showed a slightly higher efficiency with $\mathcal{C}_{\mathrm{CQT}}$ for the SWD and with $\mathcal{C}_{\mathrm{STFT}}$ for the BSD. Generally, the signal processing chromas produced maximum recall values of around 70%. With $\mathcal{C}_{\mathrm{deep}}$ we achieved an increase in recall of about five percent points. This shows the potential of applying deep-learning techniques for feature extraction in chord recognition. To examine the impact of the technical challenge of extracting features from the audio recordings, we performed chord recognition with the score-based chroma $\mathcal{C}_{\mathrm{score}}$. The results for $\mathcal{C}_{\mathrm{score}}$ showed an increase in recall of approximately five to ten percent points as compared to the signal processing chromas. Compared to $\mathcal{C}_{\mathrm{deep}}$, there was no clear increase in recognition quality and in some cases even a slight decrease. From these results we can conclude that the musical challenge of abstracting notes to a chord label has a larger impact on the chord recognition effectiveness than the technical challenge of extracting the note information from the audio recordings. A further reduction of the musical challenge by using $\mathcal{C}_{\mathrm{annot}}$ showed a clear increase in recognition quality, with recall values close to 100% in some selected scenarios.

The in-depth analysis of recognition results on a more detailed level revealed insights into the cross-version and cross-song differences of both datasets. For both datasets, the analysis of recall values for the individual songs averaged across versions showed large fluctuations. This was the case for all three chord vocabularies. Especially in the SWD, different songs obtained highly varying results. In contrast, the analysis of recall values for individual versions averaged across songs revealed only slight differences. None of the versions produced prominently higher or lower recall values than the others. While this is true when averaged across all songs, a track-wise evaluation of recall values showed numerous cases where the results for individual songs varied largely for different versions. With the in-depth analysis of recognition results on the measure level, we highlighted errors that commonly occurred across different versions. In approximately 25% of the segments where errors occurred, either one version or all of the versions produced an error, respectively. Furthermore, we showed that in approximately 50% of segments where errors occurred, the majority of versions obtained the correct recognition result, highlighting the potential of cross-version fusion strategies.

While our studies offered several insights into chord recognition in the context of cross-version classical music recordings, the datasets provide broad opportunity for further research. As an

example, the SWD can be used to study the impact of a singing voice on chord recognition. Furthermore, our studies were mostly focused on results on a complete dataset level. Our in-depth analysis showed that both datasets offer further potential for research on more detailed levels, investigating characteristics of individual songs and versions.

Appendix A

Chord Statistics

We present statistics of the individual chords contained in the chord vocabularies for both datasets. We give a detailed description of the chord vocabularies in Section 3.3.

Figure A.1. SWD, statistics of the individual chords of the major/minor vocabulary.

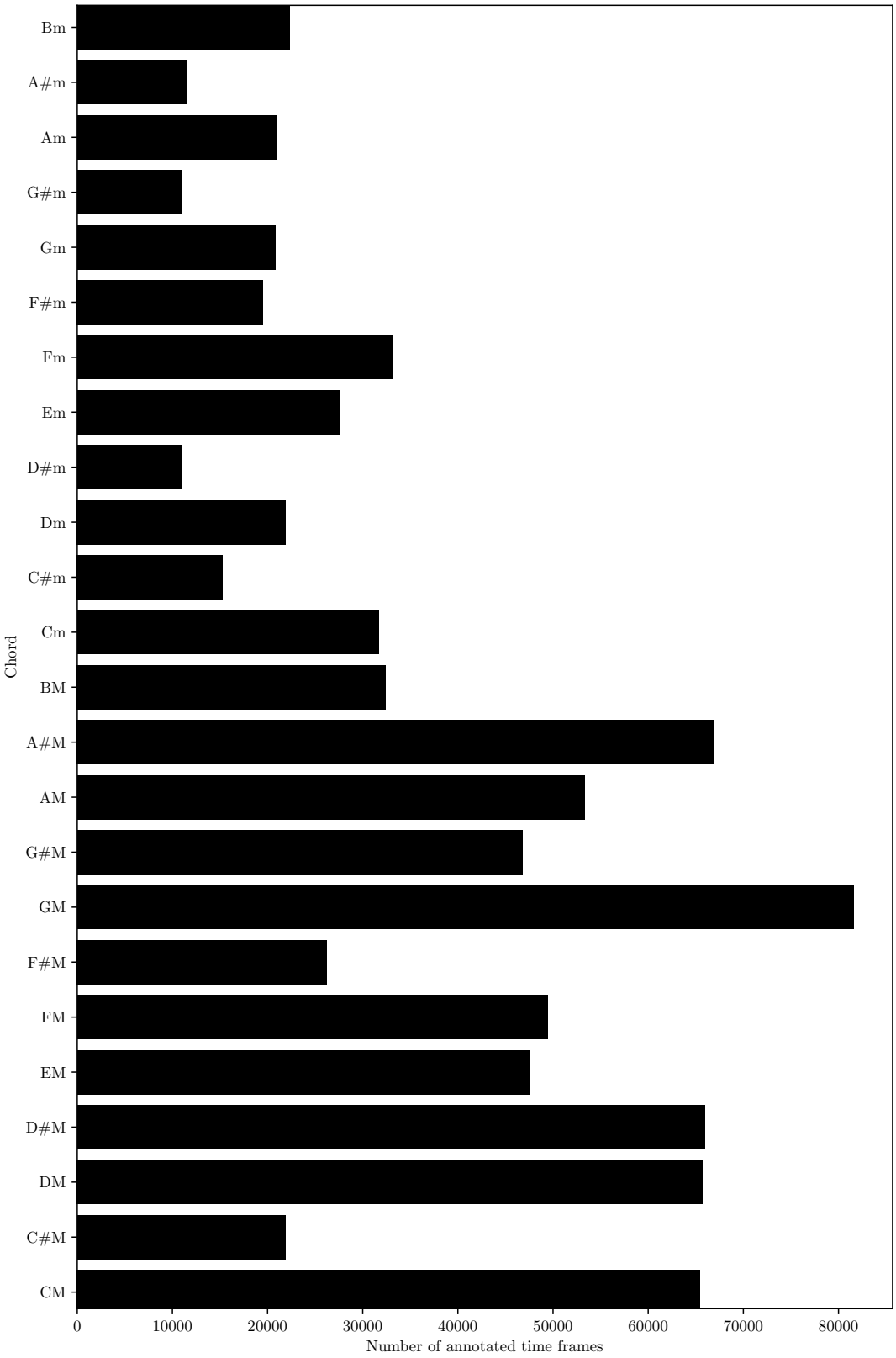


Figure A.2. BSD, statistics of the individual chords of the major/minor vocabulary.

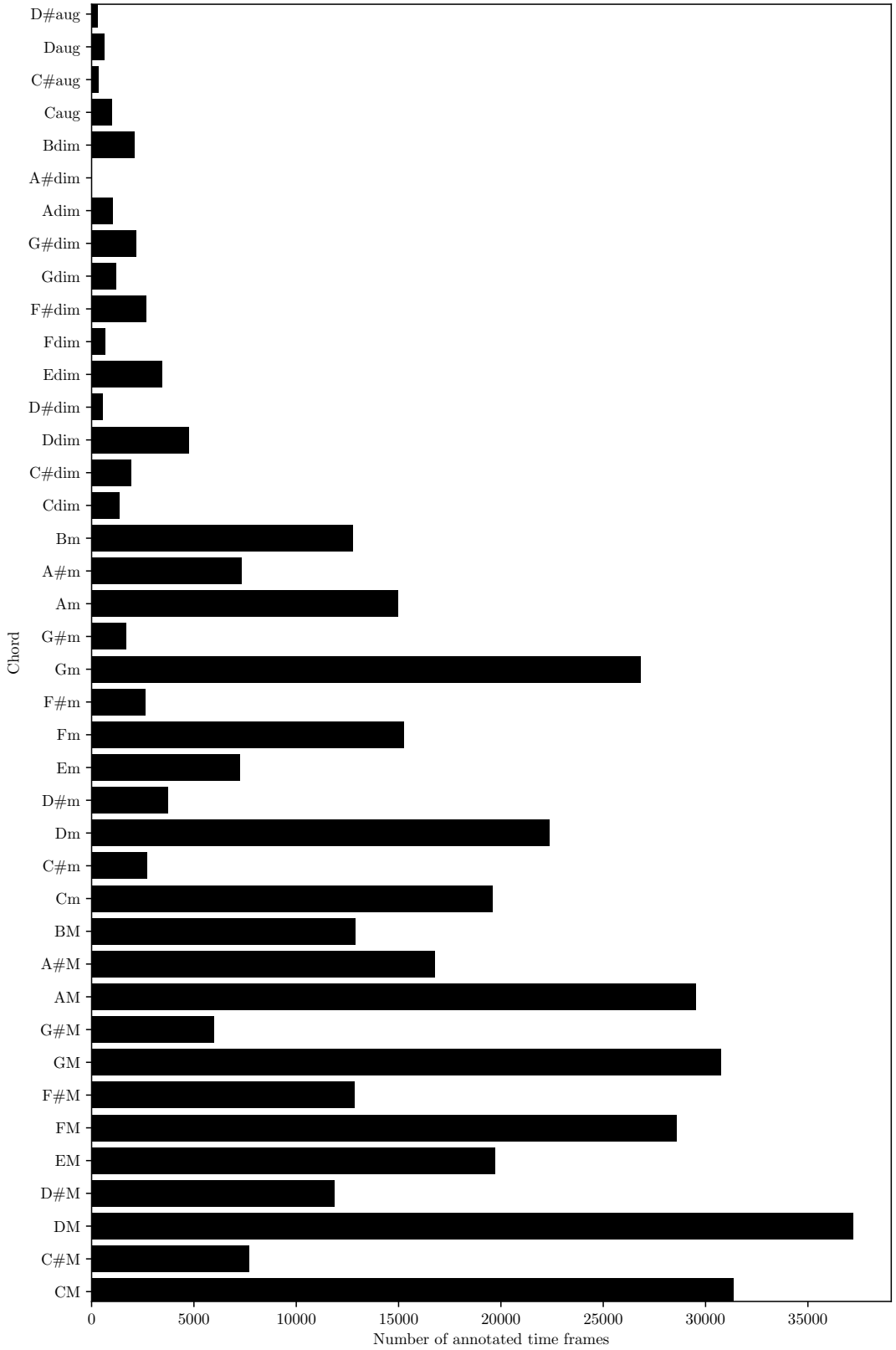


Figure A.3. SWD, statistics of the individual chords of the triad vocabulary.

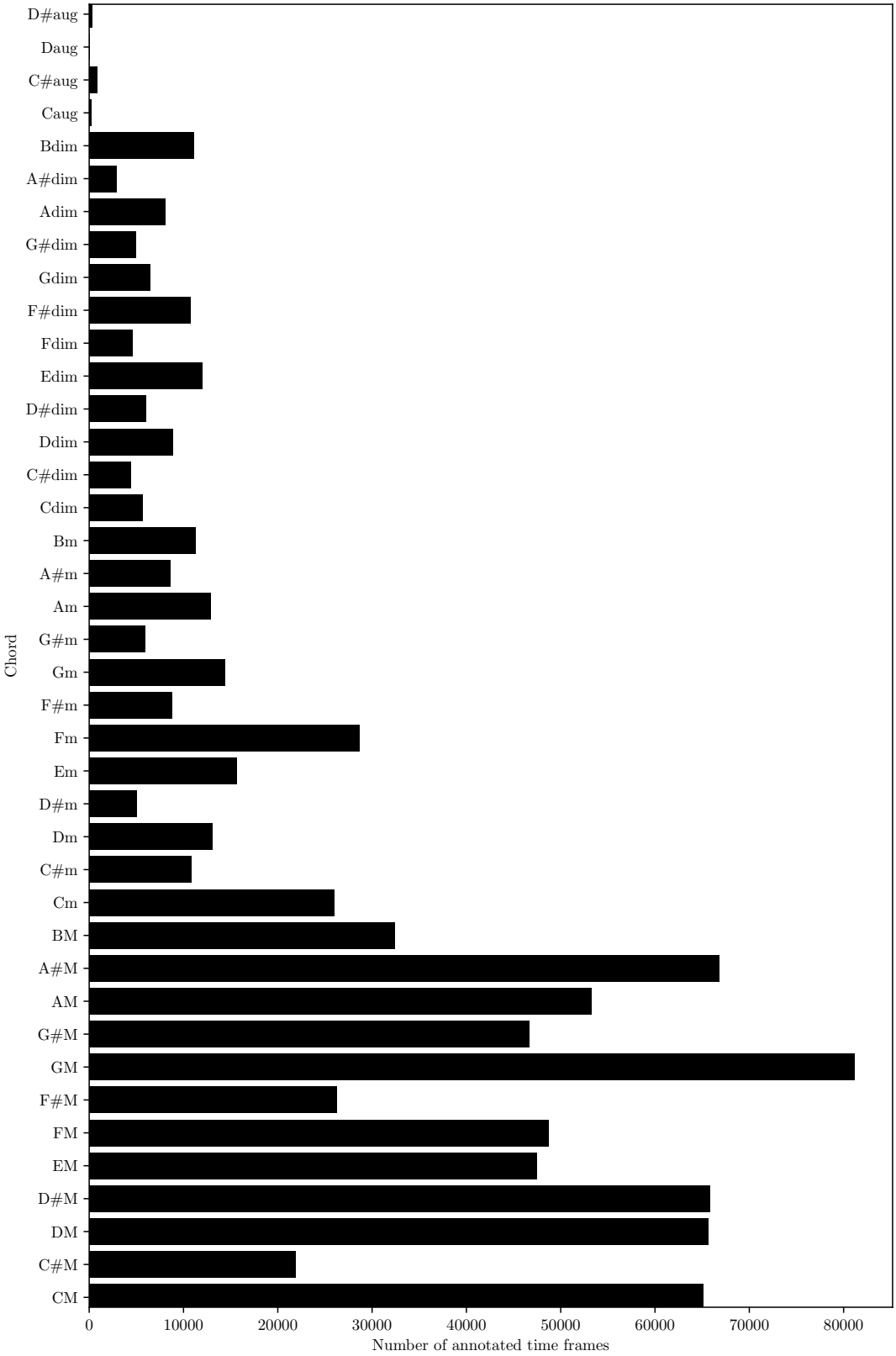


Figure A.4. BSD, statistics of the individual chords of the triad vocabulary.

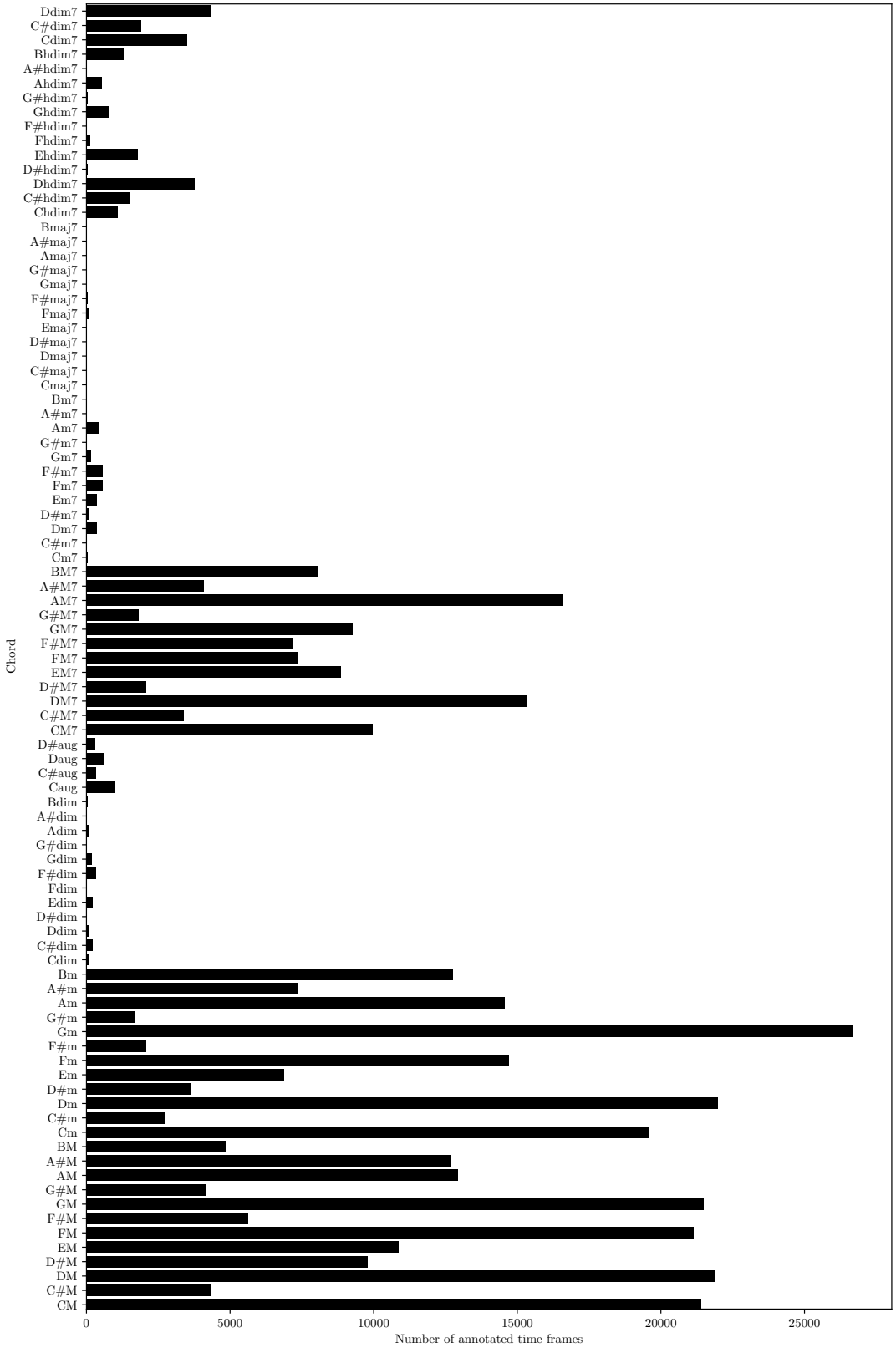


Figure A.5. SWD, statistics of the individual chords of the seventh vocabulary.

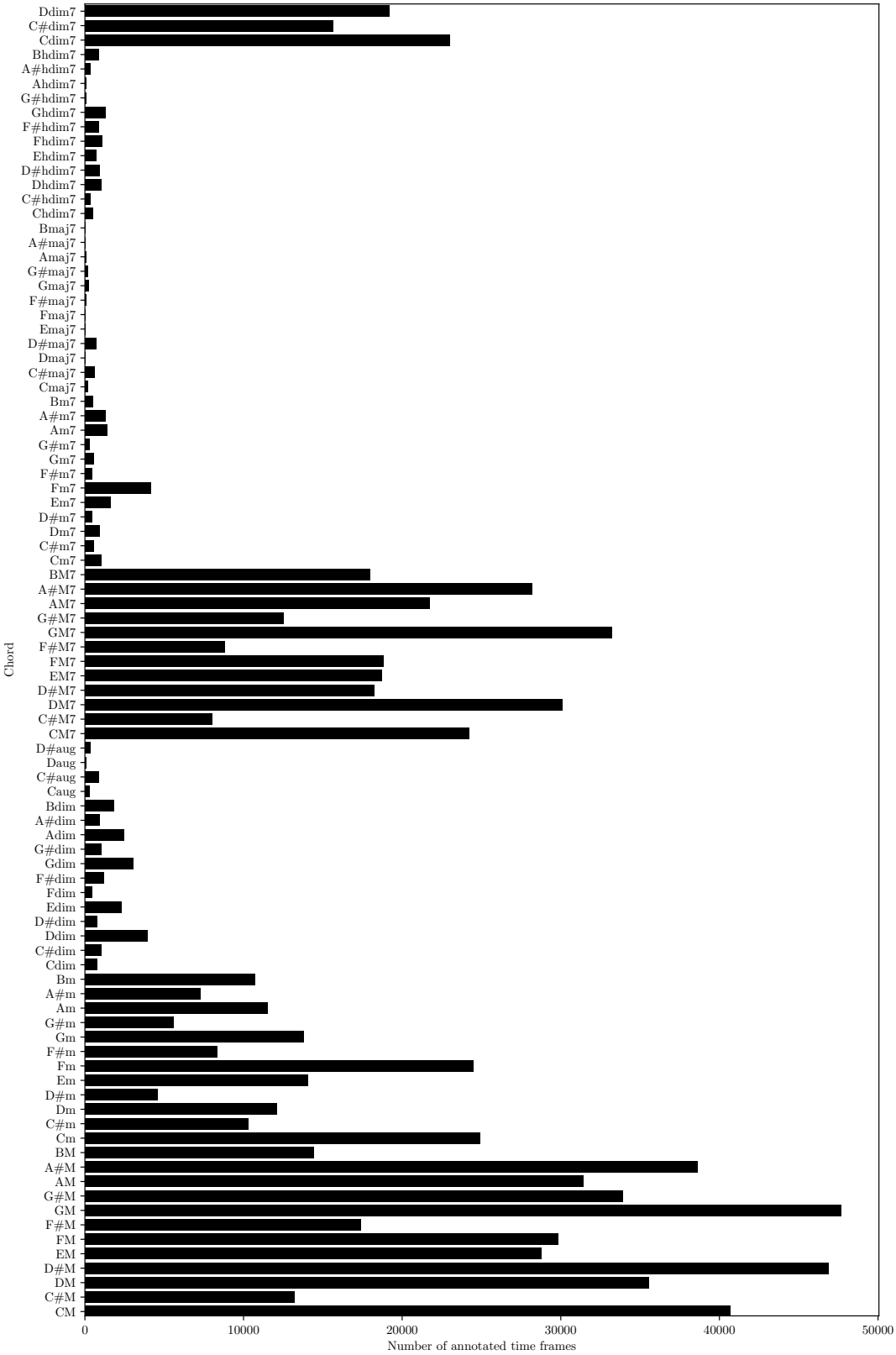


Figure A.6. BSD, statistics of the individual chords of the seventh vocabulary.

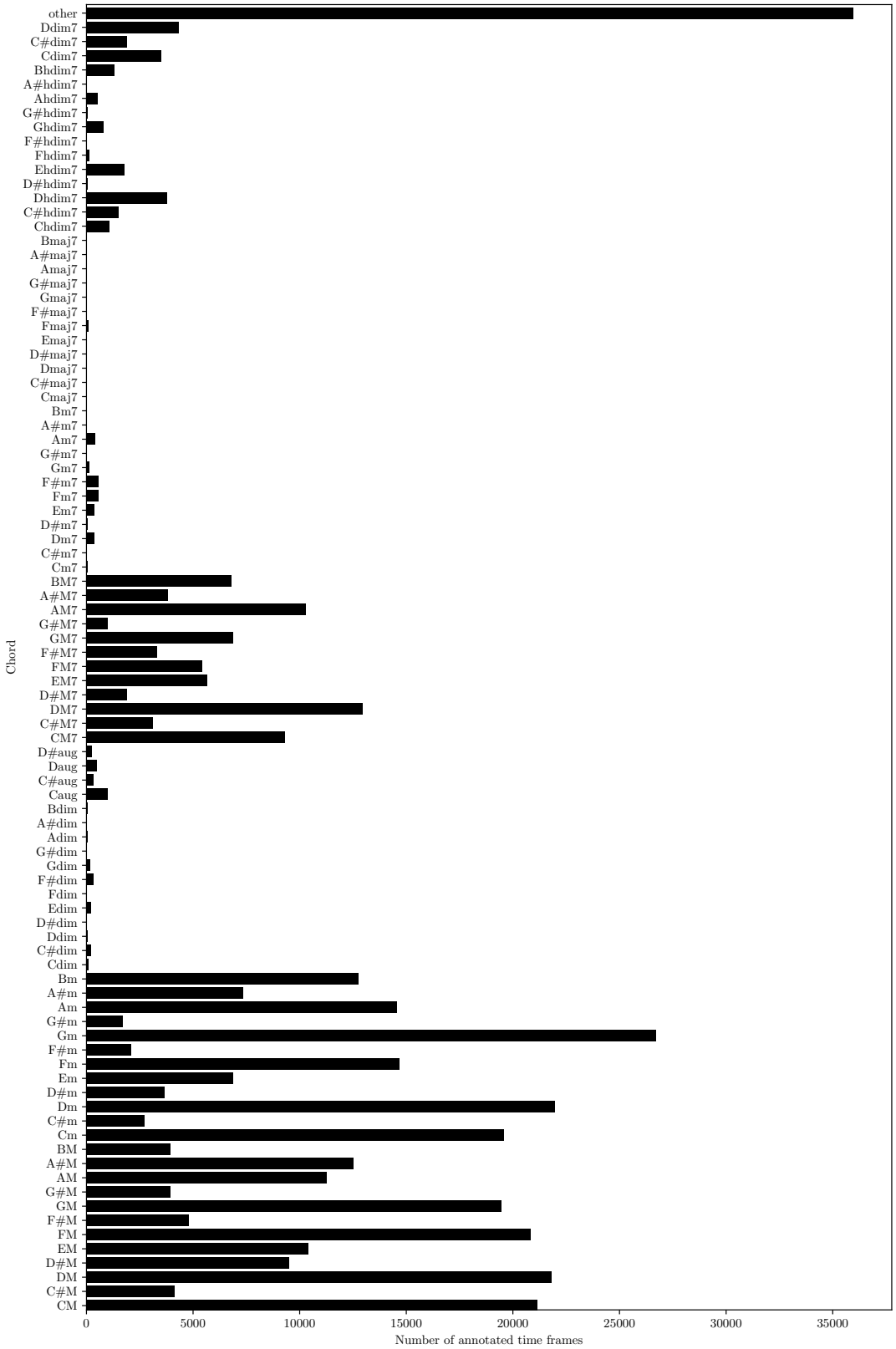


Figure A.7. SWD, statistics of the individual chords of the seventh vocabulary without any mapping or reduction.

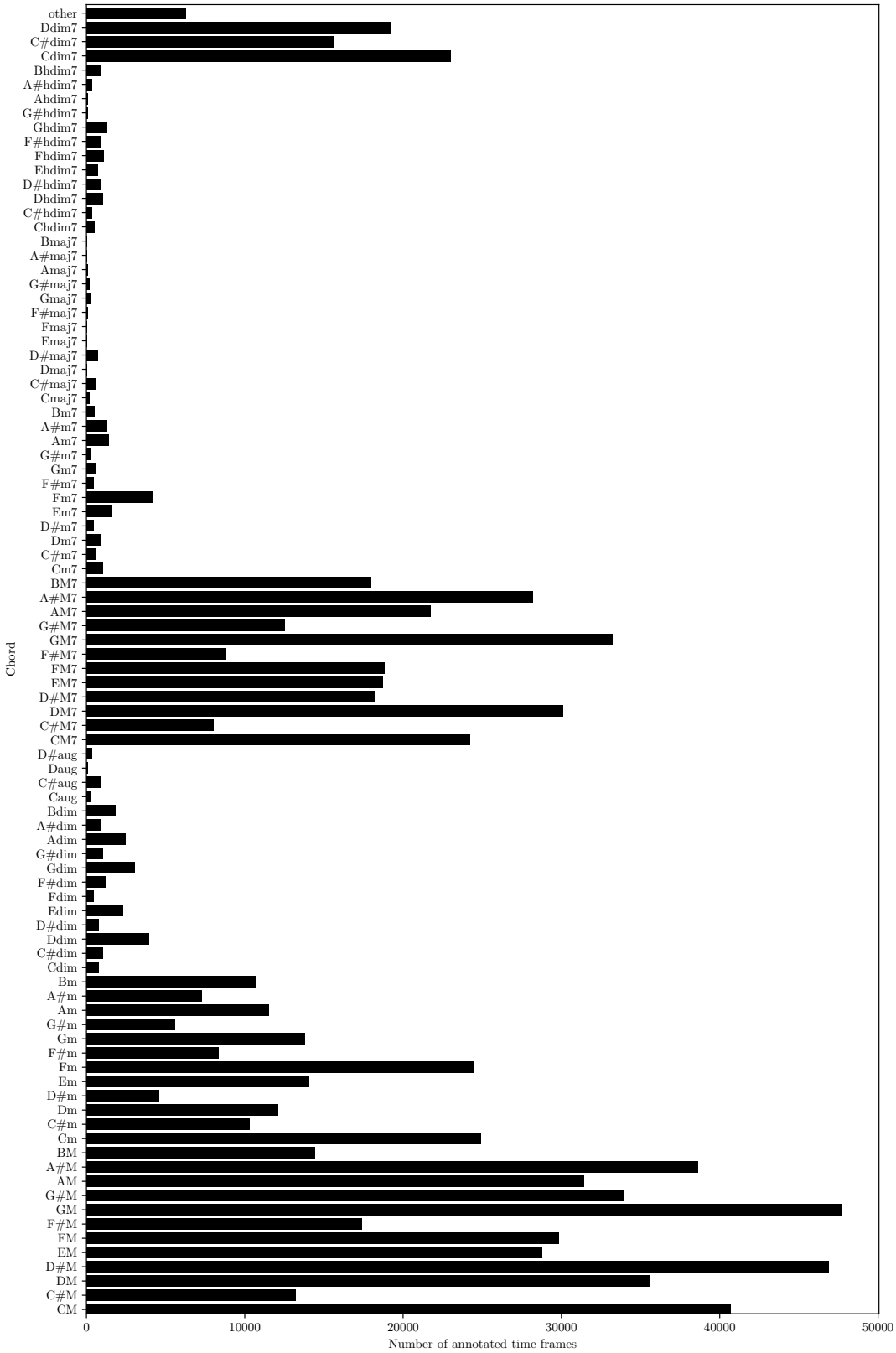


Figure A.8. BSD, statistics of the individual chords of the seventh vocabulary without any mapping or reduction.

Appendix B

Track-Wise Recall Values

We present track-wise recall values with different chroma feature types for both datasets. We give a detailed description of the plots in Section 5.6.

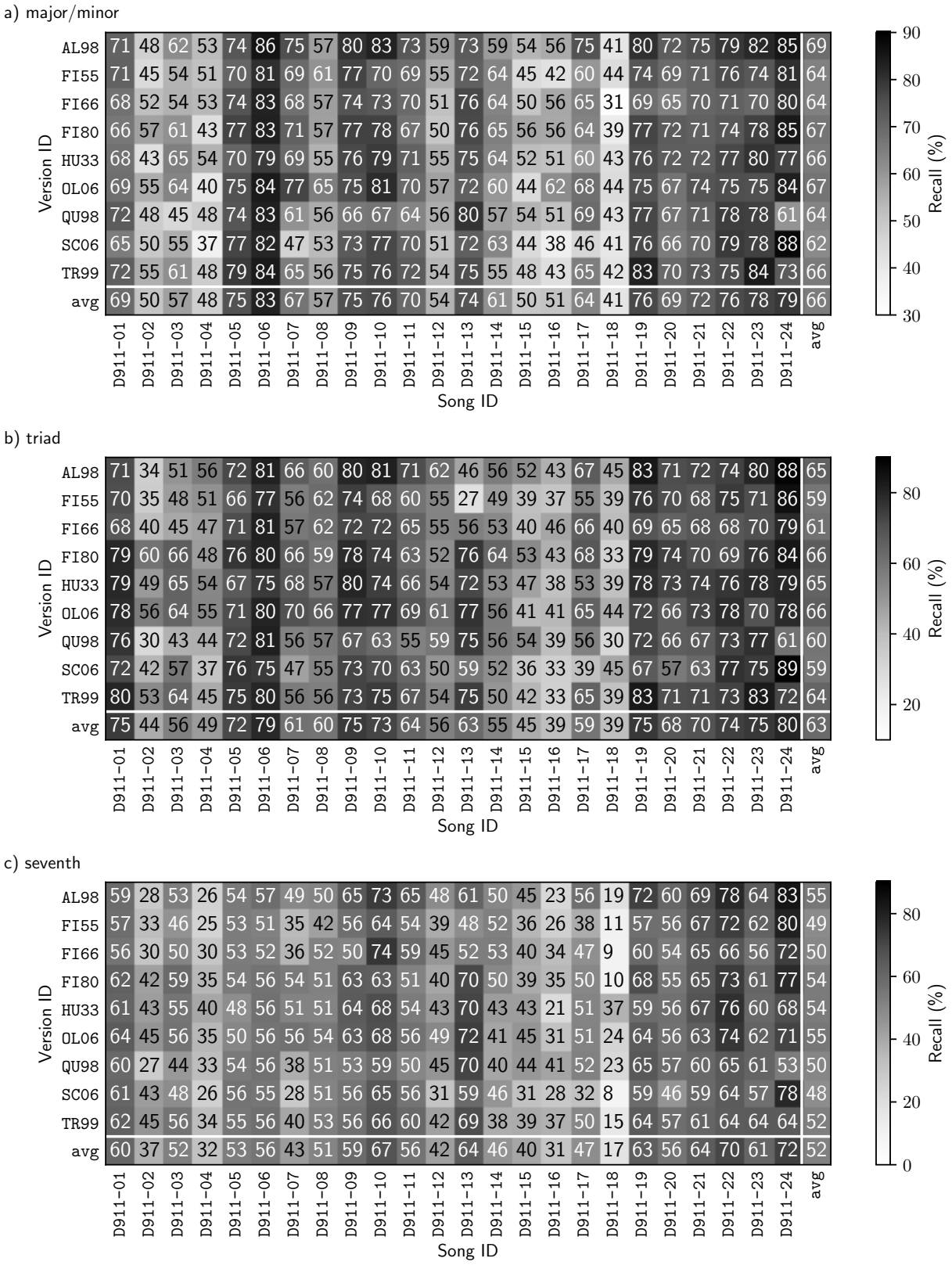


Figure B.1. Track-wise recall values for SWD with $\texttt{HMM}_{\text{G}}$, $\mathcal{C}_{\text{STFT}}$, and neither split. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

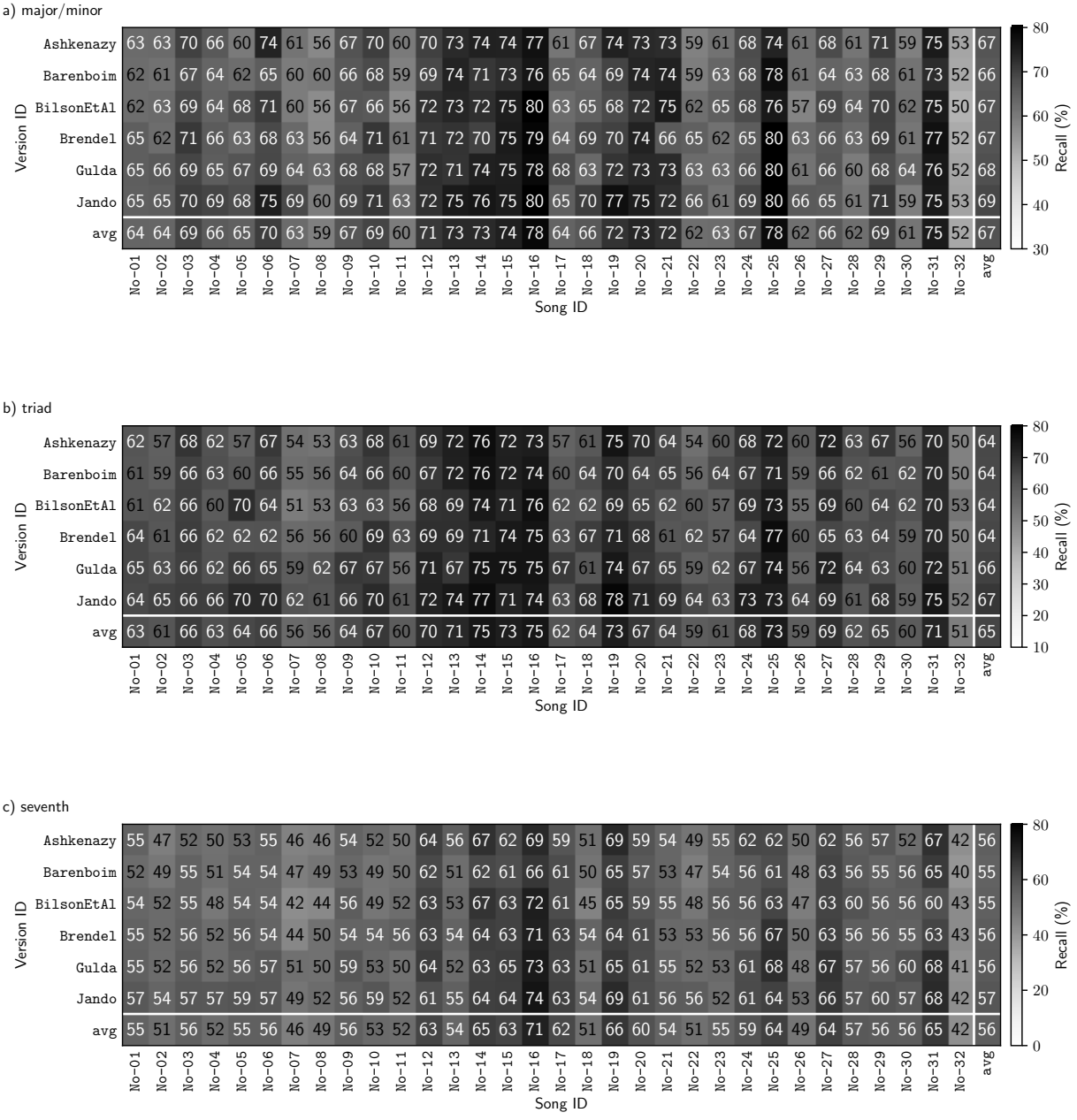


Figure B.2. Track-wise recall values for BSD with $\texttt{HMM}_{\text{G}}$, $\mathcal{C}_{\text{STFT}}$, and neither split. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

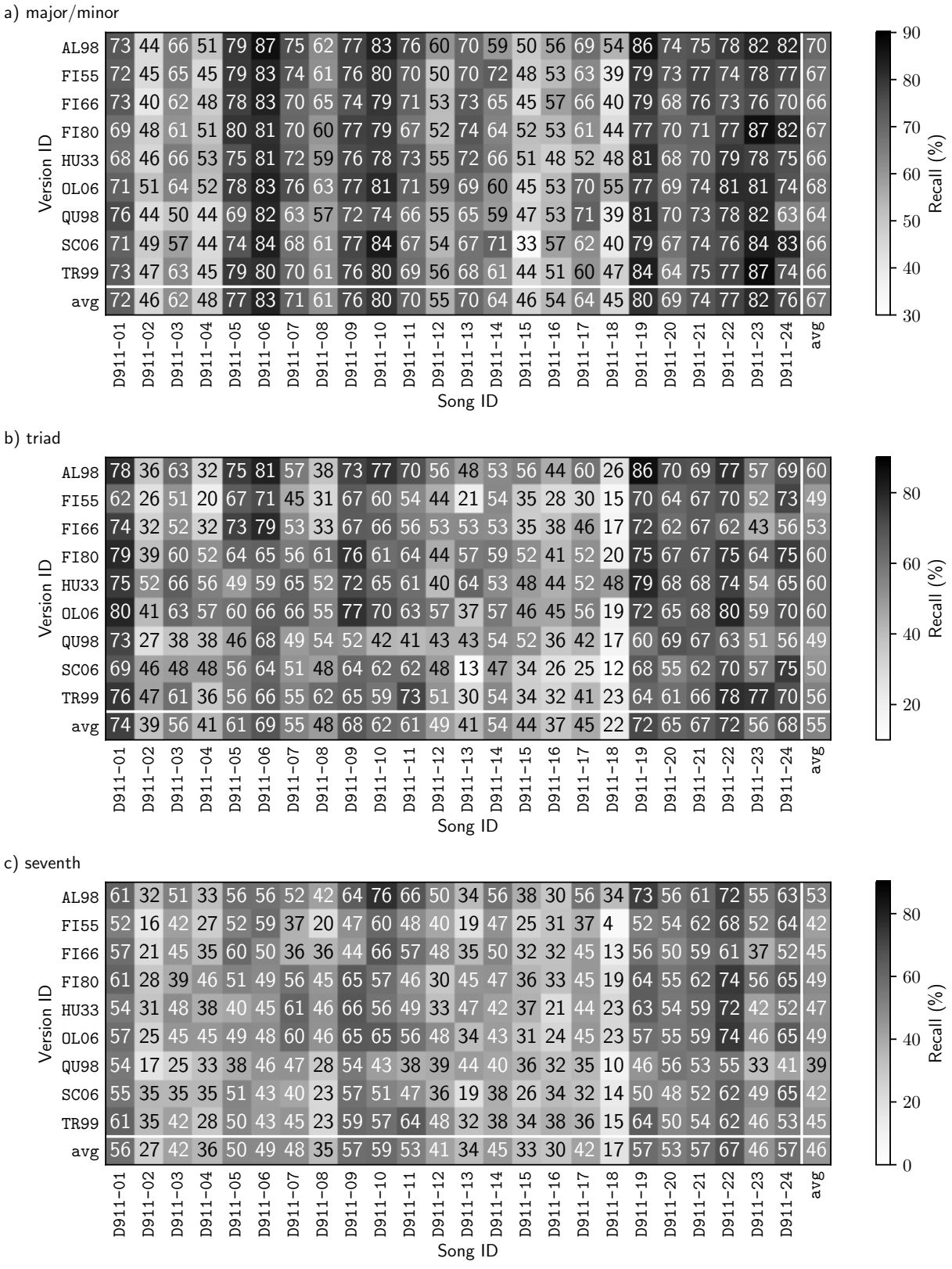


Figure B.3. Track-wise recall values for SWD with $\texttt{HMM}_{\text{G}}$, $\mathcal{C}_{\text{IIRT}}$, and neither split. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

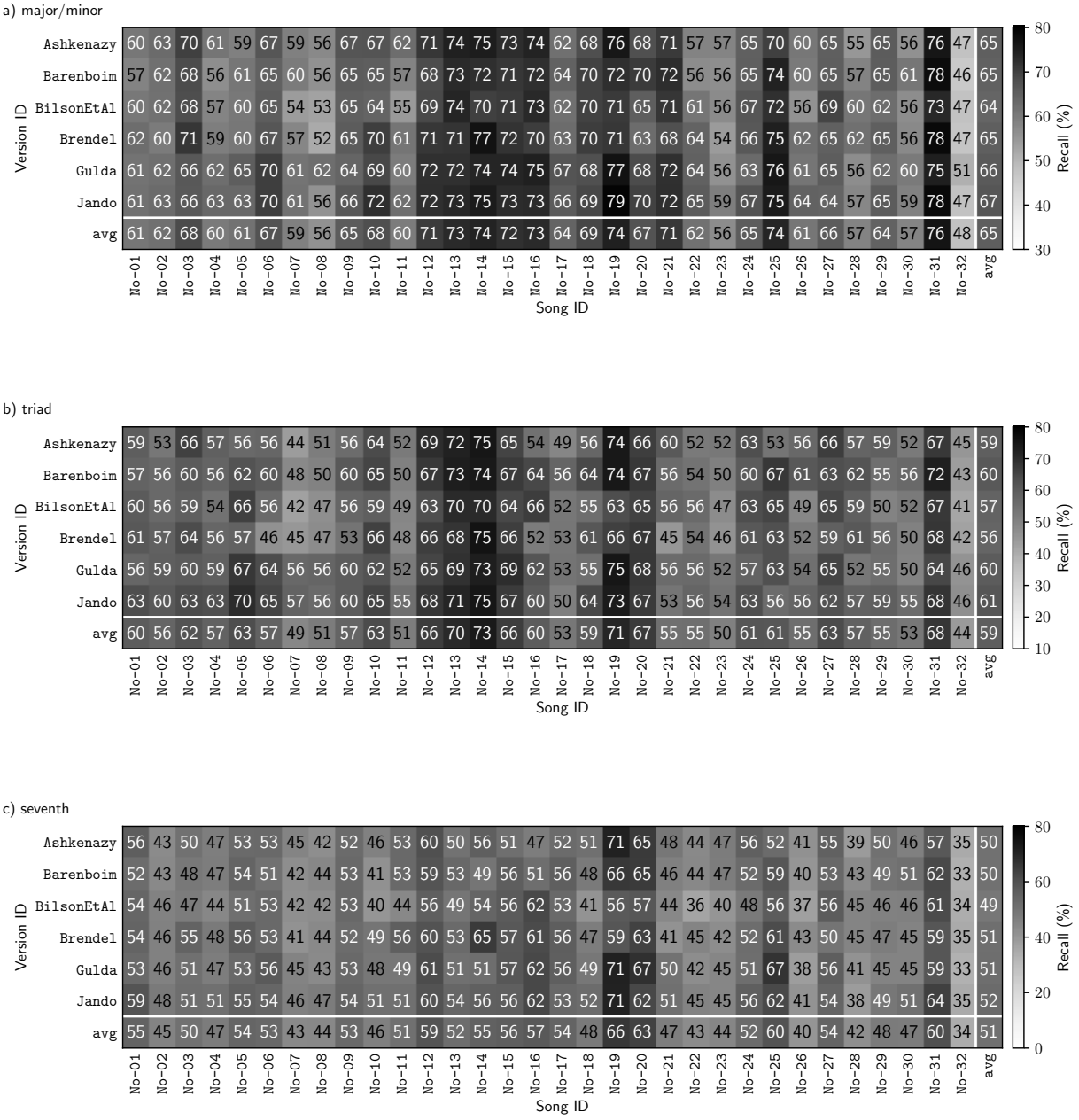


Figure B.4. Track-wise recall values for BSD with $\texttt{HMM}_{\text{G}}$, $\mathcal{C}_{\text{IIRT}}$, and neither split. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

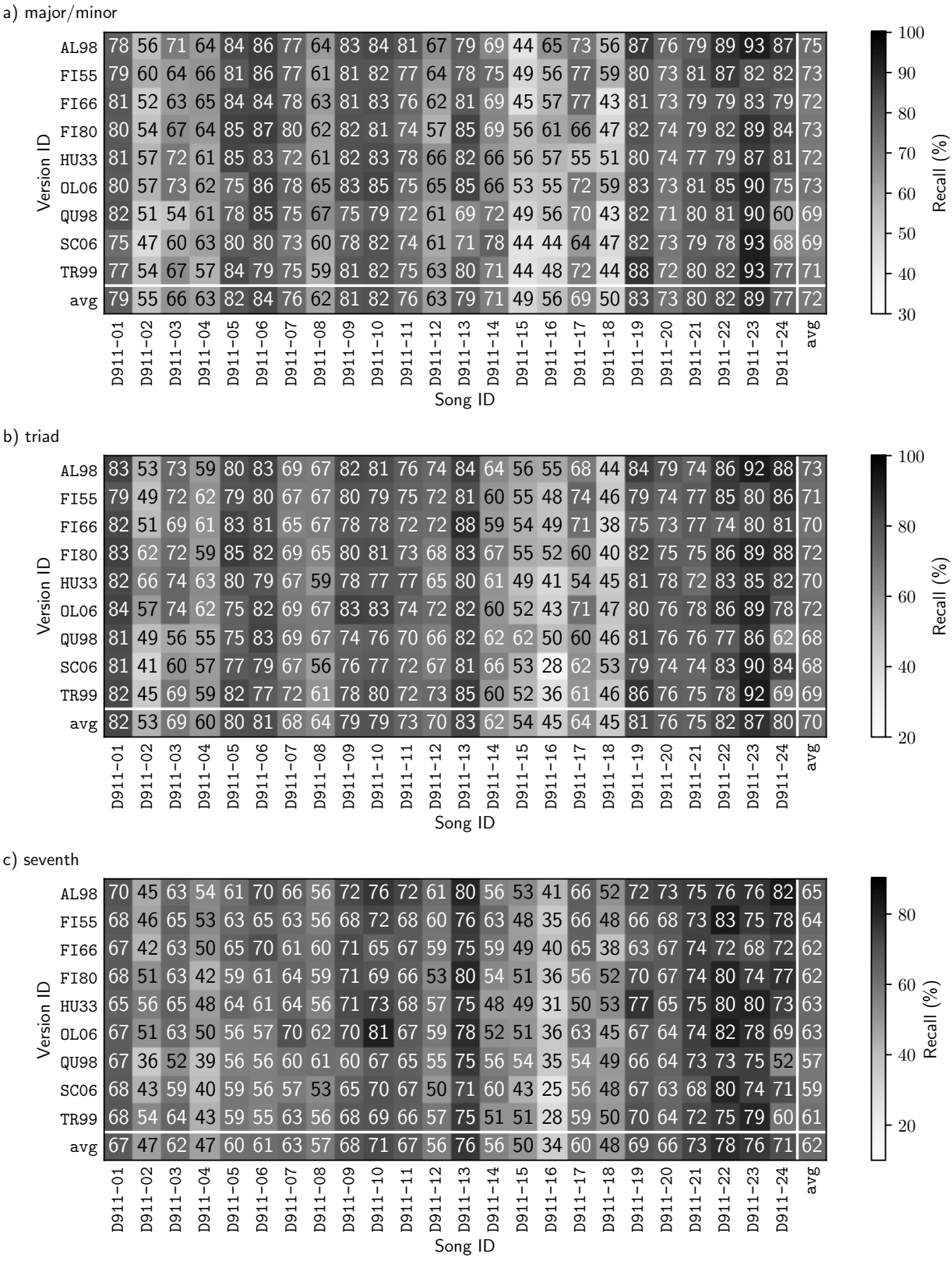


Figure B.5. Track-wise recall values for SWD with $\texttt{HMM}_{\text{G}}$, $\mathcal{C}_{\text{deep}}$, and neither split. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

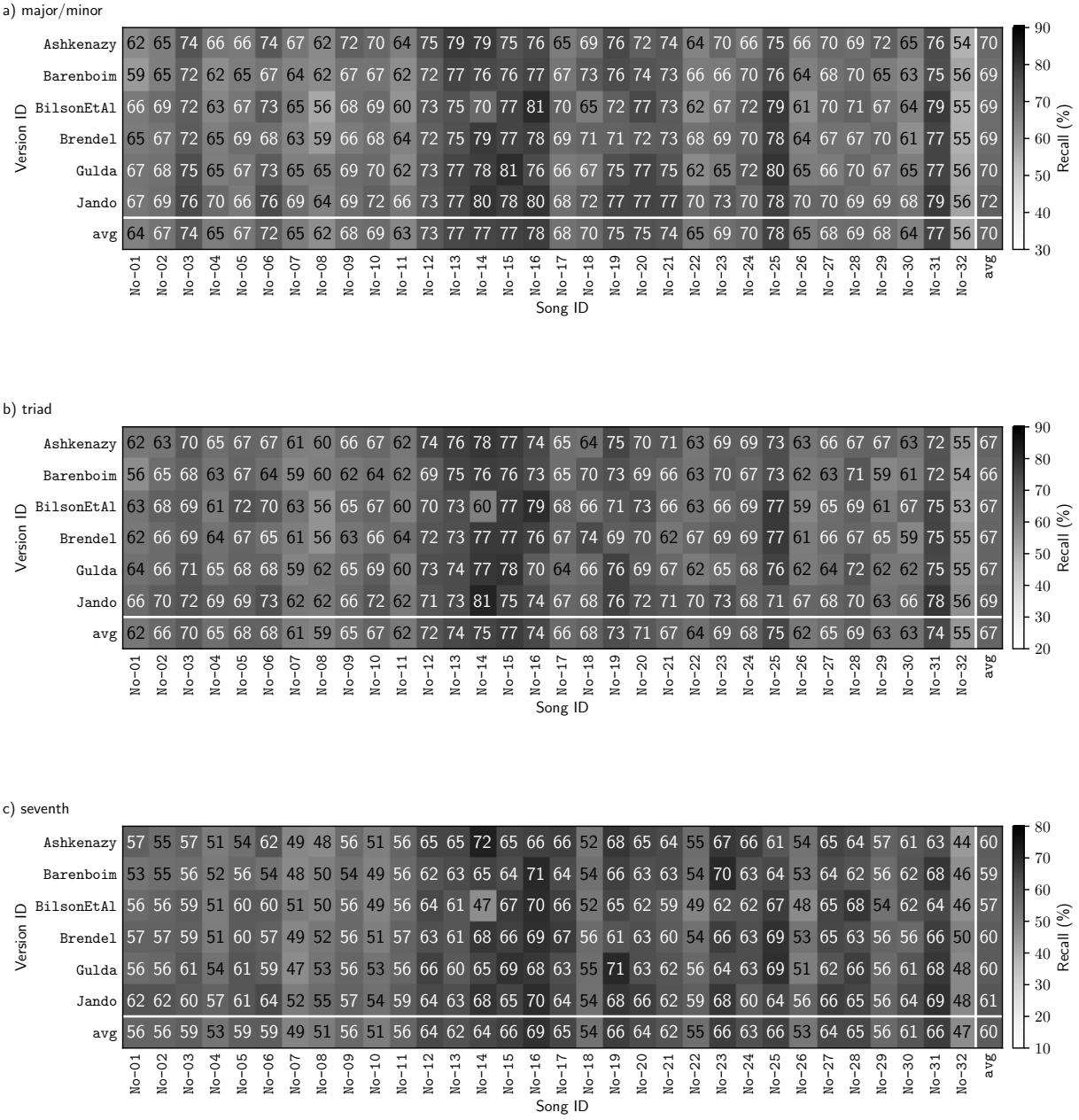


Figure B.6. Track-wise recall values for BSD with $\texttt{HMM}_{\text{G}}$, $\mathcal{C}_{\text{deep}}$, and neither split. a) For major/minor vocabulary. b) For triad vocabulary. c) For seventh vocabulary.

Appendix C

Score Excerpts

We present score excerpts that correspond to the musical pieces we discuss in Section 5.6.

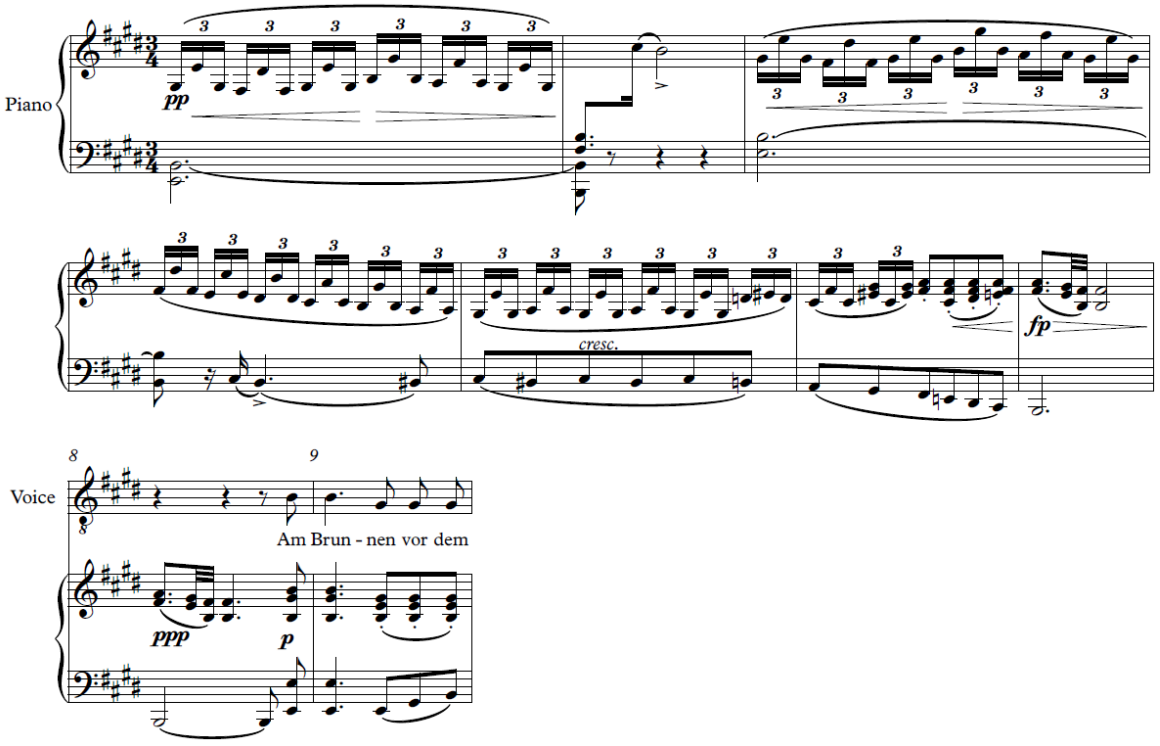


Figure C.1. Score excerpt from song `D911-05`, *Der Lindenbaum* (eng. *The Linden Tree*) from the SWD.

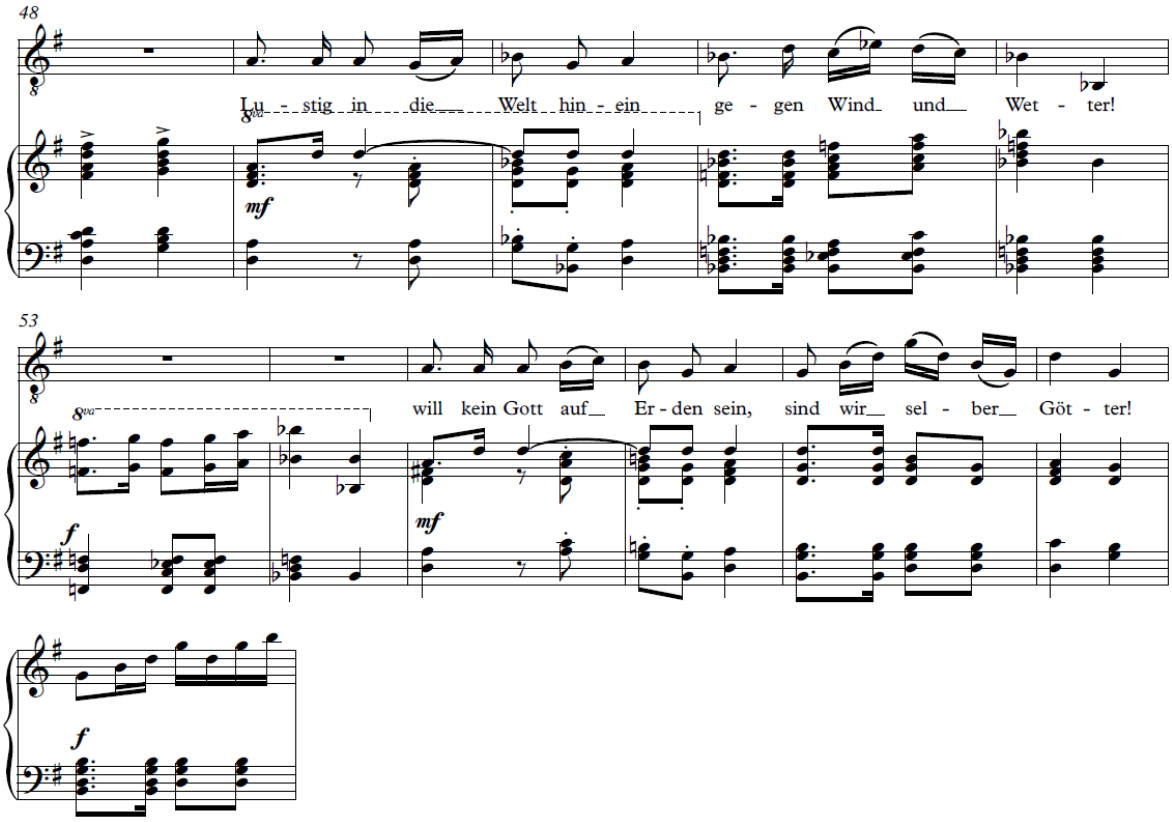


Figure C.2. Score excerpt from song `D911-22`, *Mut!* (eng. *Have Courage!*) from the SWD.

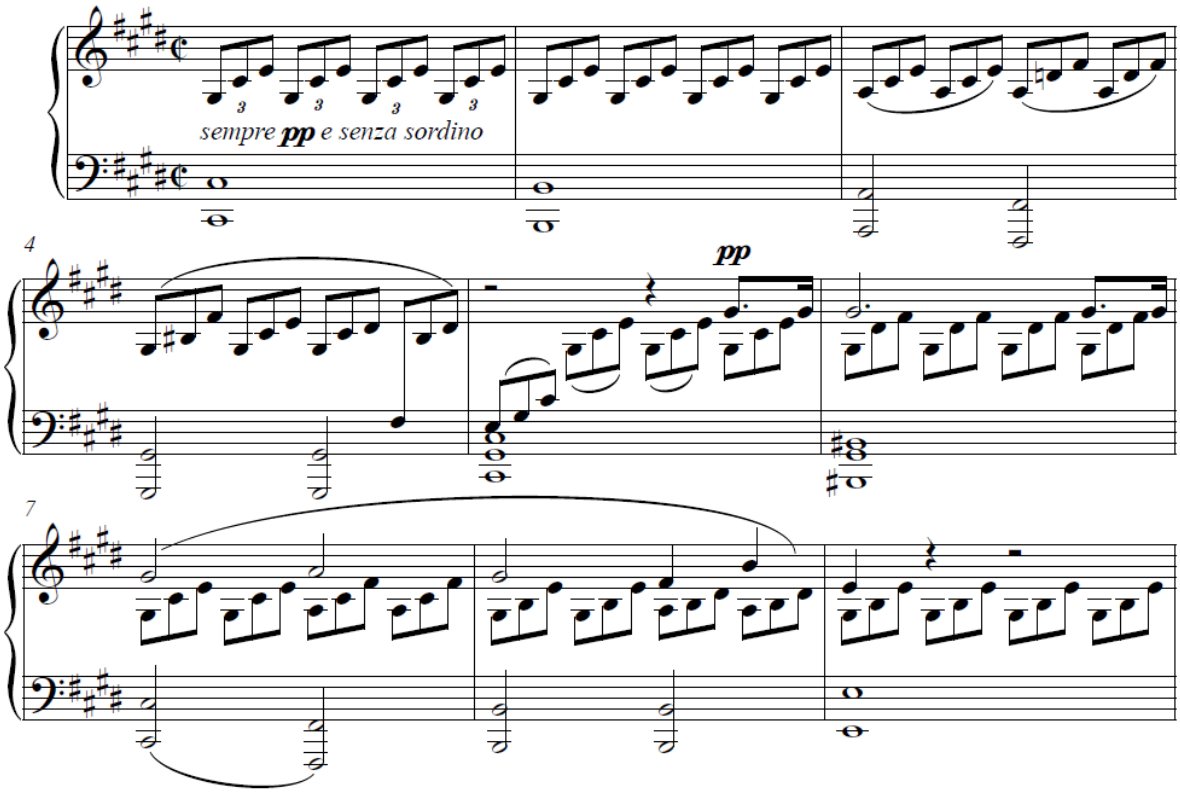


Figure C.3. Score excerpt from song `No-14`, famously known as *Mondscheinsonate* (eng. *Moonlight Sonata*) from the BSD.

Bibliography

[1] F. G. J. Absil, *Musical Analysis – Visiting the Great Composers*, Frans Absil Music, 6th ed., 2017.

[2] L. Brütting, *Hierarchical tonal analysis of music signals*, Master's thesis, Bachelor Thesis, Friedrich-Alexander-University of Erlangen-Nuremberg, Erlangen, 2019.

[3] J. A. Burgoyne, J. Wild, and I. Fujinaga, *An expert ground truth set for audio chord recognition and music analysis*, in Proceedings of the 12$^{\text{th}}$ International Society for Music Information Retrieval Conference (ISMIR), 2011.

[4] T. Chen and L. Su, *Functional harmony recognition of symbolic music data with multi-task recurrent neural networks*, in Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR), Paris, France, 2018, pp. 90–97.

[5] T. Cho and J. P. Bello, *On the relative importance of individual components of chord recognition systems*, IEEE/ACM Transactions on Audio, Speech, and Language Processing, 22 (2014), pp. 477–492.

[6] T. Cho, R. J. Weiss, and J. P. Bello, *Exploring common variations in state of the art chord recognition systems*, in Proceedings of the Sound and Music Computing Conference (SMC), Barcelona, Spain, 2010, pp. 1–8.

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

[8] V. Emiya, R. Badeau, and B. David, *Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle*, IEEE Transactions on Audio, Speech, and Language Processing, 18 (2010), pp. 1643–1654.

[9] T. Fujishima, *Realtime chord recognition of musical sound: A system using common lisp music*, in Proceedings of the International Computer Music Conference (ICMC), Beijing, China, 1999, pp. 464–467.

[10] H. Grohganz, *Algorithmen zur strukturellen Analyse von Musikaufnahmen*, PhD thesis, University of Bonn, Germany, 2015.

[11] C. Harte, M. B. Sandler, S. Abdallah, and E. Gómez, *Symbolic representation of musical chords: A proposed syntax for text annotations*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), London, UK, 2005, pp. 66–71.

[12] E. J. Humphrey and J. P. Bello, *Four timely insights on automatic chord estimation*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Málaga, Spain, 2015, pp. 673–679.

[13] N. Jiang, P. Grosche, V. Konz, and M. Müller, *Analyzing chroma feature types for automated chord recognition*, in Proceedings of the AES Conference on Semantic Audio, Ilmenau, Germany, 2011.

[14] N. Jiang and M. Müller, *Automated methods for analyzing music recordings in sonata form*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Curitiba, Brazil, 2013, pp. 595–600.

[15] V. Konz and M. Müller, *A cross-version approach for harmonic analysis of music recordings*, in Multimodal Music Processing, M. Müller, M. Goto, and M. Schedl, eds., vol. 3 of Dagstuhl Follow-Ups, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2012, pp. 53–72.

[16] H. V. Koops, *Computational Modelling of Variance in Musical Harmony*, PhD thesis, Utrecht University, Utrecht, The Netherlands, 2019.

[17] 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 (2019), pp. 232–252.

[18] F. Korzeniowski and G. Widmer, *Feature learning for chord recognition: The deep chroma extractor*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), New York City, USA, 2016, pp. 37–43.

[19] ——, *A fully convolutional deep auditory model for musical chord recognition*, in Proceedings of the 26th IEEE International Workshop on Machine Learning for Signal Processing (MLSP), Salerno, Italy, 2016.

[20] M. Krause, F. Zalkow, J. Zalkow, C. Weiss, and M. Müller, *Classifying leitmotifs in recordings of operas by Richard Wagner*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Montréal, Canada, 2020, pp. 473–480.

[21] S. Maruo, K. Yoshii, K. Itoyama, M. Mauch, and M. Goto, *A feedback framework for improved chord recognition based on nmf-based approximate note transcription*, in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2015, pp. 196–200.

[22] K. Masada and R. Bunescu, *Chord recognition in symbolic music: A segmental crf model, segment-level features, and comparative evaluations on classical and popular music*, Transactions of the International Society for Music Information Retrieval (TISMIR), 2 (2019), pp. 1–13.

[23] M. Mauch and S. Dixon, *Approximate note transcription for the improved identification of difficult chords*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Utrecht, The Netherlands, 2010, pp. 135–140.

[24] B. McFee and J. P. Bello, *Structured training for large-vocabulary chord recognition*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Suzhou, China, 2017, pp. 188–194.

[25] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, *Librosa: Audio and music signal analysis in python*, in Proceedings of the 14th Python in Science Conference (SciPy), Austin, Texas, USA, 2015, pp. 18–25.

[26] T. K. Moon, *The expectation-maximization algorithm*, IEEE Signal Processing Magazine, 13 (1996), pp. 47–60.

[27] M. Müller, *Information Retrieval for Music and Motion*, Springer Verlag, 2007.

[28] ——, *Fundamentals of Music Processing*, Springer Verlag, 2015.

[29] M. Müller and S. Ewert, *Towards timbre-invariant audio features for harmony-based music*, IEEE Transactions on Audio, Speech, and Language Processing, 18 (2010), pp. 649–662.

[30] M. Müller, V. Konz, W. Bogler, and V. Arifi-Müller, *Saarland music data (SMD)*, in Late-Breaking and Demo Session of the International Society for Music Information Retrieval Conference (ISMIR), Miami, USA, 2011.

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

[32] K. O'Hanlon and M. B. Sandler, *Comparing cqt and reassignment based chroma features for template-based automatic chord recognition*, in Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 2019, pp. 860–864.

[33] J. Pauwels and G. Peeters, *Evaluating automatically estimated chord sequences*, in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, Canada, 2013, pp. 749–753.

[34] T. Rocher, M. Robine, P. Hanna, and L. Oudre, *Concurrent estimation of chords and keys from audio*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Utrecht, The Netherlands, 2010, pp. 141–146.

[35] A. Sheh and D. P. W. Ellis, *Chord segmentation and recognition using EM-trained hidden Markov models*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Baltimore, MD, USA, 2003, pp. 185–191.

[36] J. Thickstun, Z. Harchaoui, and S. M. Kakade, *Learning features of music from scratch*, in Proceedings of the 5th International Conference on Learning Representations (ICLR), Toulon, France, 2017.

[37] C. Weiss, H. Schreiber, and M. Müller, *Local key estimation in music recordings: A case study across songs, versions, and annotators*, IEEE/ACM Transactions on Audio, Speech & Language Processing, 28 (2020), pp. 2919–2932.

[38] C. Weiss, F. Zalkow, V. Arifi-Müller, H. Grohganz, H. V. Koops, A. Volk, and M. Müller, *Schubert Winterreise dataset: A multimodal scenario for music analysis*, ACM Journal on Computing and Cultural Heritage (JOCCH), (2020, in press).

[39] Y. Wu and W. Li, *Automatic audio chord recognition with MIDI-trained deep feature and BLSTM-CRF sequence decoding model*, IEEE/ACM Transactions on Audio, Speech & Language Processing, 27 (2019), pp. 355–366.

[40] J. Zeitler, *Extracting Tonal Features for Music Analysis Using Deep Learning*, Internship Report, Friedrich-Alexander-University of Erlangen-Nuremberg, Erlangen, 2020.

[41] T. Zunner, *Deep Learning Techniques for Tonal Analysis of Music Recordings*, Internship Report, Friedrich-Alexander-University of Erlangen-Nuremberg, Erlangen, 2020.