A Unified Generative and Discriminative Approach to Automatic Chord Estimation for Music Audio Signals

Yiming WU

Abstract

This thesis describes a statistical approach to automatic chord estimation (ACE) for music signals. The chord is an important mid-level representation of polyphonic music that lies between the musical intentions of humans and actual musical sounds, and the ACE task has been one of the fundamental research topics in the music information retrieval (MIR) community.

There are two major approaches to ACE. The *discriminative* approach is based on a chord classification model implemented by a deep neural network (DNN) that directly estimates a chord sequence from a music signal. The performance of this approach is bounded by the amount of annotated data used for supervised training. In contrast, the *generative* approach is based on a probabilistic latent variable model that represents the generative process of a music signal from a chord sequence. To solve the inverse problem analytically, only classical models such as hidden Markov models (HMMs) have mainly been used.

To overcome these limitations, we propose two techniques for neural ACE. First, we make effective use of an sufficient amount of external MIDI data for better representation of music signals. Second, we propose a unified generative and discriminative approach that connects a deep classification model to a deep generative model through a latent chord sequence following a chord language model in a variational autoencoding manner. This enables semi-supervised training of the classification model regularized by the generative model.

In Chapter 3, we propose a DNN-based feature extraction method that predicts the existence of twelve pitch classes (chroma vectors) in higher (vocal), lower (bass), and wider (accompaniment) frequency ranges as acoustic features fed to a chord classification model. Specifically, we train a DNN in a supervised

manner by using a large amount of MIDI-formatted music scores and the corresponding synthesized music signals as input-output pairs. We experimentally show that a chord classification model trained with the proposed chroma vectors outperforms the one trained with conventional chroma vectors.

In Chapter 4, we propose a unified generative and discriminative approach on the basis of amortized variational inference (AVI). Specifically, we formulate a deep generative model that represents the generative process of chroma vectors from discrete labels and continuous features. Given chroma vectors as observed data, the posterior distributions of the latent labels and features are computed approximately by using deep classification and recognition models, respectively. These three models form a variational autoencoder (VAE) and can be trained jointly in a semi-supervised manner. We experimentally show that the regularization of the classification model on the basis of the language model of chord labels and the generative model of chroma vectors improves the performance of ACE even under a fully supervised condition.

In Chapter 5, we integrate the VAE-based regularized training method with multi-task learning for joint chord and key estimation. Specifically, we introduce key labels as additional latent variables and formulate a hierarchical generative model of keys, chords, and chroma vectors reflecting the typical process of music composition. We comprehensively investigate possible architectures of each component of the VAE: the separated, shared, and hierarchical architectures of the chord and key classification models, and the uniform, Markov, and autoregressive architectures of the language model. Through the comparison of different combinations, we experimentally show that the VAE-based multi-task learning improves chord estimation as well as key estimation.

Chapter 6 concludes this thesis and briefly discusses future work. The limited expression capability of chroma vectors and the limited performance improvement obtained by the VAE-based semi-supervised training are the main remaining issues of this study. We therefore discuss the potentials of higher-dimensional feature representations, advanced temporal models, higher-level language models, and latent variable disentanglement.

Acknowledgments

This work was accomplished at Speech and Audio Processing Laboratory, Graduate School of Informatics, Kyoto University. I express my gratitude to all people who helped me and this work.

At first, I would like to express my special thanks and appreciation to my supervisor Associate Professor Kazuyoshi Yoshii. I would not have had the opportunity to work on this valuable research topic if he had not suggested me to come to Kyoto University. During my research, he was always very dedicated to discussing our research topics, and gave me insightful comments for advancing this work. Without his continuing engagement and generous support, this work would not have been completed.

I also express my special thanks and appreciation to Professor Tatsuya Kawahara. He gave me a lot of essential comments on my research in our laboratory meetings.

Furthermore, I express my special thanks and appreciation to the members of my dissertation committee, Professor Hisashi Kashima and Professor Ko Nishino for their time and valuable comments and suggestions.

This work cannot be accomplished without generous support of Assistant Professor Eita Nakamura. He gave me specific advice for implementing the mathematical models in this work, from his deep knowledge of mathematics.

I also deeply thank both current and past members in Speech and Audio Processing Lab. I am grateful for comments and supports from Assistant Professor Koji Inoue, Dr. Kohei Sekiguchi, Dr. Tristan Carsault, Dr. Ryo Nishikimi, Mr. Sei Ueno, the members of the music group, and the other members.

This work was supported by the Japan Society for the Promotion of Science

(JSPS) with their financial support as a Fellowship for Young Scientists (DC2).

Last but not least, I am truly grateful to my family for their support of my long student life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter describes the research background of automatic chord estimation from music signals and explains our approach.

1.1 Background

When composing, enjoying, playing, or studying music, we treat music under various levels of abstraction. The actual audio, often recorded as a waveform, is the lowest-level representation of music. The musical symbols, which we usually see in musical scores, are the higher-level abstraction of music. The musical concepts in western music theory, namely beats, chords, and musical keys, act as the mid-level representations of polyphonic music that lie between the highly abstract musical intentions of humans and actual musical sounds.

Automatic music transcription (AMT), which aims to infer the symbolic representations behind a music signal, has been the fundamental research topic in the field of music information retrieval (MIR). For example, it forms the basis of music content visualization (e.g., *Songle* [1]), and higher-level MIR tasks such as genre classification [2] and cover song retrieval [3]. Recently, AMT methods have also been used for creative applications such as automatic DJ or mashup creation [4].

Among the symbolic music representations, the chord sequence is a dominant mid-level representation. In the Western music tradition, chords are the main components of music annotations that describe the **harmony**, the simulta-

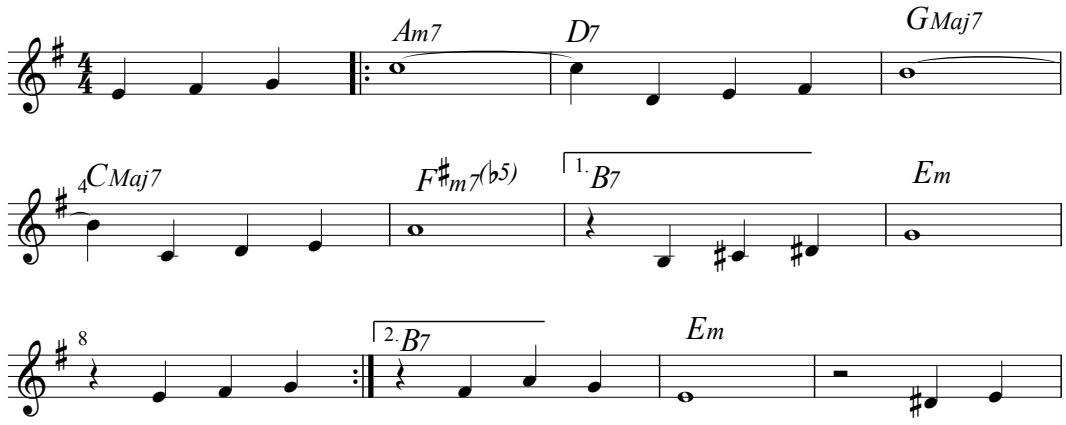


Figure 1.1: The lead sheet of *Autumn Leaves*.

neous sound of different pitches played in the music. A chord symbol represents a certain combination of multiple pitches. Unlike musical notes, chords do not tell the actual pitches, but abstractly represent the harmonic content evolving over time. For example, in lead sheets (Fig.1.1, a form of music notation consisting of melody, chords, and lyrics), chords are shown to musicians to roughly suggest the intentions of how musical notes should be arranged and played in each bar. How chords are used is highly dependent on the music style; Chord progression in jazz music is often highly complicated, and in popular and rock music it tends to be more straightforward and repetitive.

This thesis addresses automatic chord estimation (ACE), a sub-task of AMT that aims to estimate the chord sequence behind a music signal. The diversity and complexity of the acoustic characteristics of music signals make the ACE task very challenging. To address the problem, previous studies have focused on two main aspects of the chord transcription process:

1. Extracting a sequence of audio features from a music signal
2. Estimating a sequence of chord labels from a sequence of audio features

audio feature extraction aims at transforming a raw audio signal into a compact representation that emphasizes the property of the audio signal that is relevant to chords. The mainstream of audio features for ACE is the *chroma* representation,

which is usually calculated by combining various signal processing techniques. To estimate a chord sequence, a classification model that represents the relations between audio features and chord labels is designed and trained in a supervised manner using various machine learning techniques

1.2 Data Representations

We explain acoustic and symbolic representations of the tonal aspects of music used for ACE.

1.2.1 Chords

A chord is defined as a group of several notes that sound simultaneously. Although some researchers regard a combination of two notes as a chord, in most cases a chord is supposed to contain at least three notes. The combination of the notes is specified with a *chord label*, which is the combination of the **root note**, the lowest note in the stacked notes, and the **quality**, the position of notes in relation to each other.

A chord with three notes is called a *triad*. A triad is the most basic category in chord theory. In particular, the triads that consist of three notes that are stacked in thirds are the most important, namely the *major triads*, *minor triads diminished triads*, and *augmented triads*. In a major triad, the interval between the root note and the second note is a major third, and the interval between the second and the third note is a minor triad; In a minor triad, the two intervals are a minor third and a major third, respectively. A diminished triad consists of two minor thirds, and an augmented triad consists of two major thirds. In addition, the *suspended* chords, which consist of a major second and a perfect fourth interval, is also used occasionally in popular musics.

Chord with four notes is called a *tetrad*. Generally, tetrad chords are defined on the basis of the triad chords, *i.e.*, stacking an additional note above the triad chord. For example, the *seventh* chord is defined by stacking a third interval above a triad chord. Because the additional note brings instability to the har-

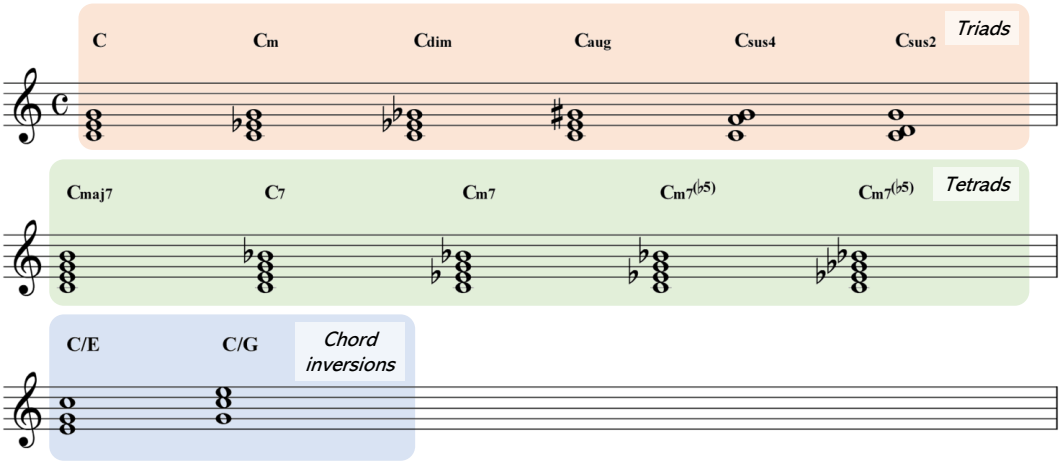


Figure 1.2: The definitions of triad and tetrad chords, and chord inversions.

mony, tetrad chords have richer acoustic characteristics than triad chords, and is often used as an accent in jazz and pop music.

When the bass note of the harmony is not the root note, an additional bass note annotation is added after the chord symbol, forming a *transposed chord*. Chord inversion is another technique of music composition that introduces instability to a chord progression. Fig. 1.2 lists the definitions of different chord labels with root note C. Since there are 12 possible root notes, the total number of chord labels can be up to 216 if all of the qualities and transpositions are considered.

1.2.2 Keys

The key is a musical concept that is closely related to the chord concept. Specifically, a key is regarded as a set of notes that are often used in the music. These notes are typically described using a *musical scale*, a sequence of notes within a single octave that are ordered by ascending pitch (Fig. 1.3). Similar to chord label, a key label is specified with the first note of the scale (or the *key note*) and the scale type. Fig. 1.3 shows the *major scale* and the (natural) *minor scale*, the two most basic types of musical scales.

Because the key specifies the notes used in the music, it is closely related to

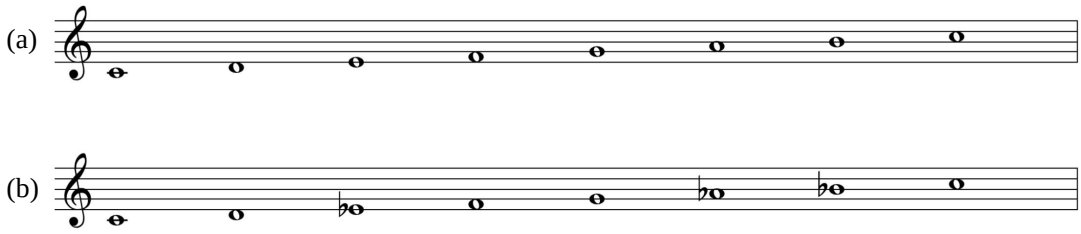


Figure 1.3: (a) C major scale. (b) C minor scale.

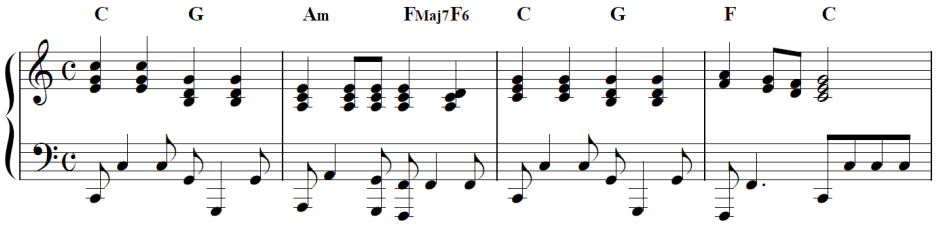
the usage of chords. Keys and chords are both necessary when analyzing the structure and emotion of the music. The chords with notes that are included in the scale (or the *diatonic chords*) are more likely to be used, and the non-diatonic chords are less likely to be used [5]. Moreover, the key specifies the *functionality* of each chord in the music. For example, the diatonic chord whose root note is the same as the key note is called the *tonic chord*. The tonic chord creates a stable feeling, and is usually used at the beginning and end of the music (or the musical sentence). The diatonic chords whose root notes are the fifth and fourth note of the musical scale are called the *dominant* and *subdominant* chords. These chords create different levels of tension; The arrangement of chords that evolve between tension and release can create a sense of drama to the music.

Key transitions occur much less frequently than chord transitions. In most popular songs, the keys remain unchanged during the whole songs, and thus the functionality of the chords also remain unchanged in these songs. When the key transition occurs, the possibility and functionality of the chords would change correspondingly.

1.2.3 Representations of Chord and Key Labels

In conventional studies on ACE, the chord progression of a musical piece is represented as a series of chord labels that is time-synchronized with the music audio. Each label represents the harmony inherited in a specific section of the piece. In the common datasets used for ACE, the chord information of a musical piece is annotated as a list of chord labels and the corresponding timestamps with respect to the music audio (Fig. 1.4). In a typical music processing situation

Original score



Chord annotation

0.000000	0.175157	N
0.175157	1.852358	C
1.852358	3.454535	G
3.454535	5.126371	A:min
5.126371	5.950680	F:maj7
5.950680	6.774988	F:maj6
6.774988	8.423605	C
8.423605	10.014172	G
10.014172	11.651179	F
11.651179	13.392675	C
…		

Figure 1.4: The original score of the intro of the Beatles song "Let it be" and the chord annotation with respect to its original recording.

where the audio signal is transformed to a frame-level representation (*e.g.*, a sequence of chroma vectors), the annotations are given at the frame level as well, so that each label is assigned to an audio frame.

Similarly, the key can also be represented as a label sequence. If we assume that the key always remains unchanged during a song, the key can be annotated using a single label. Otherwise, the time-synchronized representation should be used as well.

In a computational formulation, a chord/key label is often represented with a binary vector. The most common form of the chord label representation is the one-hot vector, a D-dimensional vector (D is the total number of chords), where the dimension corresponding to the represented chord is set to 1, while the other dimensions are set to 0. Another representation focuses on the hierarchical structure of the definitions of chords, and represents a chord with multiple one-hot vectors that tell the root note, triad type, additional notes, transposition, and so on [6, 7]. Alternatively, one can also use a *template vector* [8], a multi-hot binary vector whose dimensions represent the activation states of the notes

(pitch classes). One problem of using the template vector-based representation is the ambiguity of chord labels; Different chords might be represented with the same template vector when the chords share the same set of pitch classes.

1.2.4 Representations of Music Signals

Because the ACE task focuses on the pitches in the music audio, a common ACE system transforms the audio signal of the music into a 2-dimensional time-frequency representation (such as short-time Fourier transform spectrogram), which represents the evolution of sound energy on each frequency bin along time. More specifically, the time-frequency representation is a time series of spectrum vectors, each spectrum representing the energy of each frequency in a short snippet of the audio signal (as known as *audio frame*). Typically, it is still difficult to distinguish chords from the raw spectrogram of music audio, for it may contain various elements that are irrelevant to chords, such as percussive sounds, overtones, main melodies and other decorative pitches. Therefore, ACE studies try to extract higher-level audio representation from the audio spectrogram.

The chroma vector [9] is the most commonly-used audio feature representation for ACE methods. The main idea of chroma vector is to aggregate all pitch salience information in the music audio into 12 *pitch classes*. The motivation of calculating chroma vectors is related to human's auditory characteristics on the *chromatic scale*; Notes that belong to the same pitch class are perceived as similar, while notes that belong to different pitch classes are perceived as dissimilar. Practically, when we recognize a chord from the sound of multiple notes, we often focus on the pitch classes of the notes, and take less interest in the vertical placement of the notes in relation to each other (known as the *voicing*). Therefore, chroma vectors can preserve sufficient information on the harmonic state of music audio with a small number of dimensions.

1.3 Approaches

In this thesis, we present two approaches to ACE that make effective use of existing non-paired music data (MIDI files and non-annotated music signals). First, we introduce a data-driven audio feature extraction method that significantly enhances the pitch salience information crucial for ACE. Second, we propose a unified discriminative and generative approach to ACE that enables both *supervised* and *unsupervised* training.

1.3.1 DNN-Based Chroma Vector Extraction

In Chapter 3, we propose a data-driven method for effective audio feature extraction. Transforming the raw audio signal into audio feature representation is the first step of chord estimation. As discussed in Chapter 2, the chroma representation, which describes how the salience of the 12 pitch classes vary across the duration of the audio, has been commonly employed as the audio feature representation in the related studies. A variety of techniques were proposed for precisely extracting the pitch class information in the music audio. The earlier methods were based on expert knowledge on frequency-domain audio signal analysis, combining multi-pitch estimation, sound source separation, and noise compression techniques. Due to the diversity and complexity of the acoustic characteristics of music signals, designing a good feature extraction method that is useful for ACE has been very challenging.

The proposed data-driven method omits these feature engineering techniques. Instead of careful feature engineering, we implement a feature extraction model using a deep neural network that directly outputs the pitch class salience from audio spectrogram. The feature extraction model is intended to automatically learn the optimal feature extraction process using a large amount of training data. We find that music data in Musical Instrument Digital Interface (MIDI) format is very suitable for constructing a large-scale training dataset for such purpose. More specifically, from the note information in a *general MIDI* music data, both the music audio and its target chroma vectors can be automatically

obtained. Compared with spectral analysis-based feature extraction methods, the data-driven feature extraction method is robust to disturbances by unrelated sound such as percussive sound, and the pitch class salience is more distinct. As a result, the proposed feature extraction method can improve ACE performance of chord classification models without collecting additional annotated music data.

1.3.2 VAE-Based Joint Generative and Discriminative Modeling

In Chapter 4, we propose a new training methodology for DNN-based ACE method on the basis of the variational autoencoder (VAE). Conventional DNN-based ACE methods take a *discriminative* approach; A deep model is trained to directly convert audio features into the posterior of chord labels. Discriminative approach makes chord label inference very straightforward, but also oversimplifies how a person transcribes music. In such methods, the classifier is usually trained for learning a frame-wise audio-to-label mapping without considering the characteristics of chord label sequences and the generative process of music. In addition, the classifier needs to be trained in a fully supervised manner, and the performance of ACE heavily depends on the amount, diversity, and quality of annotated music signals because of the nature of supervised training.

We formulate ACE from a *generative* approach. More specifically, we formulate a deep *generative* model that represents the complex generative process of audio feature (observed variables) from the discrete chord labels and continuous latent features (latent variables). The deep *classification* and *recognition* models are then introduced for inferring the distributions of the latent variables given the observation, in the manner of variational inference. These three components forms a variational autoencoder (VAE) [10], which can be jointly trained in a (semi-)supervised manner where the generative model acts as a regularizer for the classification model.

1.3.3 Multi-Task Learning-Based Joint Chord and Key Estimation

In Chapter 5, we extend the VAE proposed in Chapter 4 to introduce key labels and present a joint chord and key estimation method. The main limitations of existing work on joint estimation of multiple musical elements are the lack of sufficient supervised data, and the absence of a solid formulation representing the relations between different musical concepts. Chapter 4 shows the advantage of VAE-based training that the prior of the latent variables could act as a *language model* that effectively regularizes the label classification model during unsupervised training. On the basis of this finding, we further examine whether the proposed method can apply to automatic estimation of multiple music concepts that are semantically related.

Specifically, we add latent variables representing the musical key labels, and formulate a *hierarchical* generative model representing the generative process from keys to chords and that from chords to chroma vectors. The deep classification model and the language models that represent the relations of multiple musical labels are then introduced. When formulating the VAE, we consider various approaches to implement the classification model and the language models. By comparing the combinations of different implementations, we evaluate the effectiveness of the joint estimation method.

1.4 Organization

Chapter 2 reviews related work on automatic chord estimation for music signals. Chapter 3 presents a deep feature extraction model trained with large-scale MIDI data, and also presents a large-vocabulary ACE method on the basis of the extracted features. Chapter 4 presents a semi-supervised ACE method on the basis of a VAE. Chapter 5 presents a joint chord and key estimation method on the basis of an extended VAE. Chapter 6 concludes this thesis with future directions.

Chapter 2

Literature Review

This chapter reviews related works on automatic chord estimation. Generally, ACE studies focus on different aspects of ACE systems (Fig. 2.1), including audio feature extraction, model formulation and training, chord label inference, and performance evaluation. We first review the methods for audio feature extraction, and then discuss different approaches to formulating and training a chord classification model. Finally, we discuss the evaluation metrics in the ACE studies.

2.1 Audio Feature Extraction

This section discusses the very first step of ACE systems, namely feature extraction from audio signals.

2.1.1 Chroma Feature

The chroma vectors has been the standard audio representation for ACE research since it was proposed by Fujishima [9]. Many variants of chroma vectors have been proposed by earlier ACE studies. The Main steps for calculating chroma vectors from music audio are shown in Fig. 2.2.

Like other audio processing methods, the first step of the calculation is transforming the audio signal into the frequency domain to obtain the frequency representation called *spectrogram*. *Short Time Fourier Transform* (STFT), which computes the frequency magnitude in a sliding window across the signal, is the

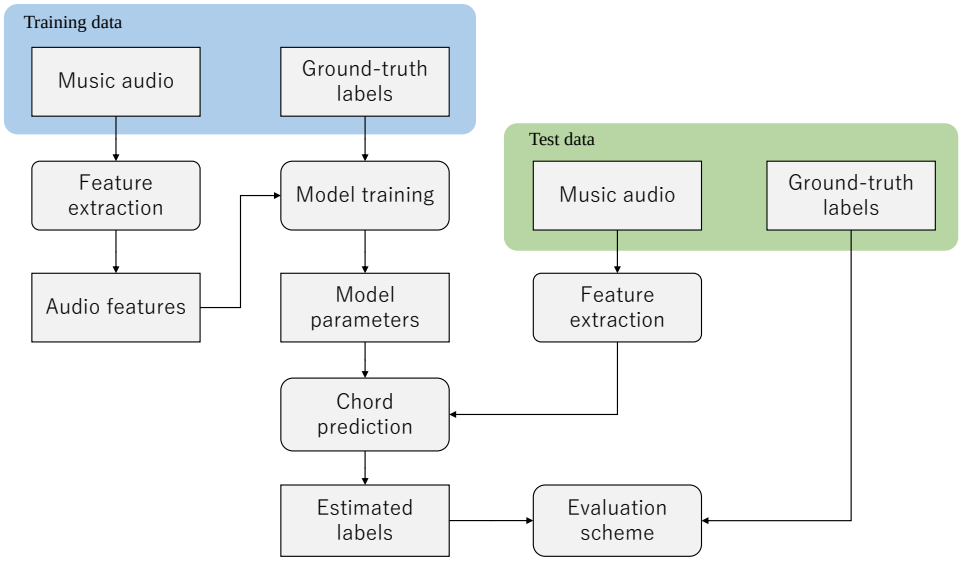


Figure 2.1: The components and workflow in typical ACE studies. Data are shown as rectangles, and processes are shown as rounded rectangles.

standard way for frequency domain transformation. However, STFT does not suit well to many music analysis tasks because of the fixed-length window; Short windows will result in poor frequency resolution, which means frequencies with long wavelengths cannot be distinguished, whilst long windows will result in poor time resolution. Moreover, because human's perception of pitches is more closely related to the logarithmic frequency scale, the linear frequency scale of STFT spectrogram is unsuitable for describing pitch-based information. As a solution to the frequency scale issue, some studies further transform the frequency range into logarithmic scale [11], by applying a certain log-filterbank on the audio signal. Another way to obtain the log-scale spectrogram is *Constant-Q Transform* (CQT) [12]. By making use of frequency-dependent window lengths, CQT partially resolves the problem of fixed-length window, and becomes a popular choice for frequency domain representation.

A key element of feature calculation is to filter out various irrelevant information in music audio, such as percussive noise, overtones, or timbral variations. Harmonic Percussive Source Separation (HPSS) algorithm [13, 14] is often used

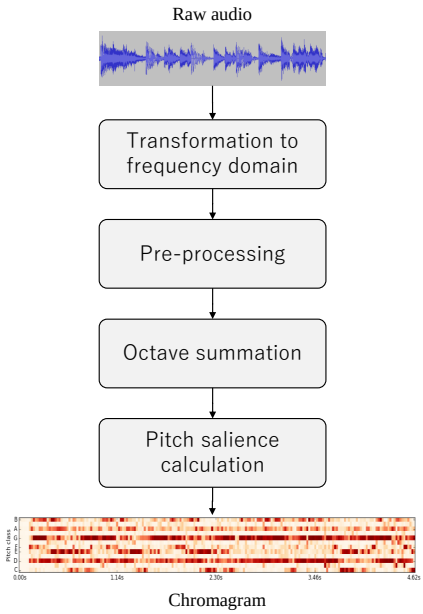


Figure 2.2: The common steps to convert music audio into chroma feature.

for filtering out the percussive components in the spectrogram. To deal with the overtones, some musical pitch tracking techniques are used to transform the magnitude spectrogram into pitch salience representations. For example, the *NNLS Chroma* [11] is obtained by solving a *Non-Negative Least Squares* problem on the log-scaled spectrogram of music audio. Similarly, the Harmonic Pitch Class Profile (HPCP) [15] is computed from the pitch salience information estimated by performing a peak tracking process on the spectrogram.

The pre-processed spectrogram or pitch salience is transformed into chroma feature by octave summation. The summed values are often affected by the dynamics of the spectrogram/salience values along time. To mitigate the effect of the dynamics and enhance the active pitch classes, some feature processing techniques can be applied to the summed vector, such as **logarithmic compression**.

In many cases, a single-octave chroma feature with only 12 dimensions is not capable of expressing the harmonic state of the music. Especially, describing harmony states with only 12 pitch classes may cause ambiguity when the chord vocabulary includes complicated chord types. Some studies [11, 16] employed

multi-channel chroma, which extends the chroma feature to represent the pitch class salience of multiple frequency bands with $12n$-dimension vectors ($n \in \mathbf{N}$). In particular, the *bass chroma*, which represents the pitch-class salience of the lower frequency band, plays an important role in chord recognition, because the bass note is an important basis for defining a chord.

2.1.2 Data-Driven Feature Representation

Instead of carefully designing the feature extraction process on the basis of explicit knowledge, some studies tried to obtain a good feature representation by training a machine-learning model with actual music data. Korzeniowski's *deep chroma extractor* [8] is a DNN-based chroma feature estimator that takes log-scaled spectrogram as input and outputs 12-dimension vectors representing the corresponding pitch class saliences. To train the DNN, the chord annotations for music data are transformed into *chord template vectors* that tell the pitch classes that the chords contain. The template vectors are then regarded as the ground-truth chroma feature for training the DNN. Some recent studies even discarded the explicit feature representation, and let the DNN implicitly learn a proper feature representation at its hidden layers [17]. This approach usually requires a larger amount of training data to guarantee the generalized performance. The study of deep chroma extractor showed the advantage of data-driven feature representations: it effectively discards the irrelevant information in music audio. Therefore, the data-driven feature can significantly improve chord estimation performance, compared with the hand-crafted features. Moreover, the data-driven feature may also be useful in any other MIR applications that use similar feature representations (*e.g.*, structural segmentation, key estimation, and cover song identification).

2.2 Automatic Chord Estimation

This section discusses the various approaches to formulating a chord classification model in the existing ACE studies. Formulation of the chord classification

model, which defines how audio features and chord labels are related, is the core of ACE studies. Practically, conventional ACE studies have been focusing on data-driven methods on the basis of statistical models, and we roughly categorize them into *generative* and *discriminative* approaches.

2.2.1 Generative Approach

Early studies took the *generative* approach on the basis of a classical probabilistic latent variable model. In the probabilistic framework, inferring the chord label sequence $\mathbf{Y}$ from observed sequence $\mathbf{X}$ is regarded as a maximum a posteriori probability (MAP) estimation as follows:

$$\mathbf{Y} = \underset{\mathbf{Y}}{\operatorname{argmax}}\, p(\mathbf{Y}|\mathbf{X}). \tag{2.1}$$

When $p(\mathbf{Y}|\mathbf{X})$ is intractable, we can transform the formulation using the Bayes' theorem:

$$p(\mathbf{Y}|\mathbf{X}) = \frac{p(\mathbf{X}|\mathbf{Y})p(\mathbf{Y})}{p(\mathbf{X})} \propto p(\mathbf{X}|\mathbf{Y})p(\mathbf{Y}), \tag{2.2}$$

where $p(\mathbf{X}|\mathbf{Y})p(\mathbf{Y})$ represents the generative process from $\mathbf{Y}$ to $\mathbf{X}$. Therefore, Eq. 2.1 is formulated as follows:

$$\mathbf{Y} = \underset{\mathbf{Y}}{\operatorname{argmax}}\, p(\mathbf{X}|\mathbf{Y})p(\mathbf{Y}). \tag{2.3}$$

In particular, Hidden Markov model (HMM) [18] has been the most common method for assigning chord labels to audio frames [19]. In an HMM, the latent variables (*i.e.*, chord labels) are represented as discrete latent states. At every time step, it emits an instance of the observed variable under the current latent state, and then transits from the current state to another state following first- or higher-order Markov process. The frame-wise emission probability of the latent states formulates the posterior $p(\mathbf{X}|\mathbf{Y})$ in (2.3) as follows:

$$p(\mathbf{X}|\mathbf{Y}) = \prod_{n=1}^{N} p(\mathbf{x}_n|\mathbf{y}_n), \tag{2.4}$$

and the transition probability between the latent states formulates the prior $p(\mathbf{Y})$ as follows:

$$p(\mathbf{Y}) = p(\mathbf{y}_1) \prod_{n=2}^{N} p(\mathbf{y}_n|\mathbf{y}_{n-1}). \tag{2.5}$$

As is shown in the formulations, an HMM contains two assumptions on the observed and latent variables: the hidden variables form a Markov chain, and each observed variable is conditionally independent to all others.

Various techniques have been proposed to improve chord estimation by HMM. For example, according to musical theory, the usage of chords and the tendency of chord transition depends on the keys [5]. Therefore, it is reasonable to introduce an additional latent variable representing musical s. Lee and Slaney did this by training multiple HMMs corresponding to different keys [20]. Mauch and Dixon proposed a dynamic Bayesian network (DBN) that represents the hierarchical structure over metrical positions, musical keys, bass notes, and chords, and formulates the generative process of 2-channel chroma feature [21]. Similarly, Ni *et al.*proposed harmony progression analyzer (HPA) that uses a DBN whose latetnt states represent chords, inversions, and musical keys [22].

Some studies have focused on the temporal characteristics of frame-level chord labels. To represent repetitions of chord patterns, for example, multi-order HMMs [23] and duration explicit HMMs [24] have been proposed, where the results of ACE were found to be intensive to the frame-level transition probabilities of chord labels. Although such a frame-level language model is incapable of learning typical chord progressions at the symbol level, it is still effective for encouraging the continuity of chord labels at the frame level.

2.2.2 Discriminative Approach

DNNs have intensively been used as a powerful discriminative model for directly estimating the posterior probability $p(\mathbf{X}|\mathbf{Y})$. Humphrey and Bello [17] firstly proposed to use a convolutional neural network (CNN) for learning an effective representation of raw audio spectrograms, and provide the chord posteriors at

the output layer. Since then, a number of DNN-based discriminative ACE methods have been proposed that use Convolutional Neural Networks [6], Recurrent Neural Netoworks [7,25–27], or transformers [28]. In general, these DNN-based methods have outperformed the HMM-based generative methods [29].

The limitation of the discriminative approach is that it often oversimplifies the process that a person transcribes music. The transcription models simply focus on learning the audio-to-label mapping, while the label-to-audio generative process is ignored. Consequently, the trained models often suffer from estimation errors that obviously conflict with the common sense on musical scores, *e.g.*, over-frequent label transitions. To remedy this problem, *chord language models* are usually integrated with the classification model to provide temporally-coherent chord labels at the inference stage. An HMM or a conditional random field (CRF) [29,30], for example, can be used for estimating the optimal path of chord labels from the estimated posterior probabilities. This is similar to the DNN-HMM hybrid model for automatic speech recognition (ASR) [31] that integrates a DNN acoustic model and an HMM language model. Besides, recurrent neural networks (RNNs), which can represent longer-term dependencies of label sequences, have also been used as language models [32]. Unlike HMMs, RNN-based language models do not have efficient algorithm to infer the optimal label sequence; instead, beam-search algorithm is used in these cases to infer a near-optimal sequence.

Chen [24] pointed out that in the frame-level formulation, the main effect of HMM-based language model is simply smoothing the label sequence, *i.e.*, reducing the label transitions. The transition probabilities between latent states, which represent the tendencies of chord progression, have much less impact on the chord estimation performance than the smoothing effect. Korzeniowski *et al.*further pointed out that the frame-based temporal models, either HMM-based or RNN-based models are incapable of expressing the musical structure within the musical chords. RNN-based models are, nevertheless, better at modeling chord sequence at the symbol level (*i.e.*, the chord label transits at every time step) [33,34]. Korzeniowski conjectured that chord estimation performance

would be improved if the symbol-wise language model is applied to chord inference. In his later work [26], he tried to integrate the frame-wise acoustic model with the symbol-wise language model by introducing another sequence model, namely duration model, that predicts how long a chord symbol would repeat itself at the frame level.

Furthermore, it is a critical issue for deep learning-based discriminative methods to collect enough amount of training data with sufficient annotation quality. Specific to the chord estimation field, time-synchronized ground-truth chord labels are necessary for supervised training. However, annotating those time-synchronized labels is not only time consuming and tedious, but also needs professional music knowledge. Although the label quantity can be increased by employing more annotators or applying data augmentation [29], the bias of keys, chord types, and music genres may still affect the performance of supervised training. To mitigate the annotation cost, Wu [27] proposed an HMM-based forced alignment approach that estimates the timing information of the non-aligned chord annotations.

2.2.3 Autoencoding Formulation

Autoencoding formulation is considered to be an effective approach for applying generation-based formulation in DNN-based methods. Unlike the HMM-based generative models, deep temporal models (*e.g.*, recurrent neural networks) do not assume that latent sequences are Markovian, or that the observed variables are conditionally independent. In these cases, the posteriors of the latent variables with respect to the deep generative models are usually intractable. The autoencoding formulation solves this problem by formulating a cyclic architecture using a pair of deep models. More specifically, the latent variables are firstly estimated from the observed variable by the recognition model (as known as the *encoder*). Then, a learnable generator (as known as *decoder*) tries to correctly reconstruct the observed variable given the estimated latent variable. By jointly optimizing the encoder and the decoder, the encoder is encouraged to output the correct latent variable so that the decoder can correctly reconstruct

the observation.

Integration of deep generative and discriminative models has actively been explored in the context of unsupervised or semi-supervised machine learning. In the field of Automatic Speech Recognition (ASR), some studies tried to jointly train a speech-to-text model with a text-to-speech model to improve the performance of ASR by using both annotated and non-annotated speech audio signals [35]. For the MIR studies, Choi and Cho [36] proposed an unsupervised drum transcription method which formulated a synthesizer with concrete drum sound samples, and trains a deep transcription model to minimize the distance between the observed and synthesized audio.

Specifically, the *variational autoencoder* (VAE) [10] have been the most popular strategy to jointly optimize the two models. The original VAE formulates a latent variable model as follows:

$$p(\mathbf{X}, \mathbf{Z}) = p(\mathbf{X}|\mathbf{Z})p(\mathbf{Z}), \tag{2.6}$$

where $\mathbf{X}$ is the observed variable and $\mathbf{Z}$ is the latent variable. For the latent variable's posterior $p(\mathbf{Z}|\mathbf{X}) = p(\mathbf{X}|\mathbf{Z})p(\mathbf{Z})/p(\mathbf{X})$ is analytically intractable under the DNN-based formulation, a recognition model $q(\mathbf{Z}|\mathbf{X})$ is introduced to estimate the true posterior $p(\mathbf{Z}|\mathbf{X})$ in the framework of amortized variational inference (AVI).

In a VAE, the joint optimization of the generative model $p(\mathbf{X}|\mathbf{Z})$ and the recognition model $q(\mathbf{Z}|\mathbf{X})$ is done by optimizing the variational lower bound $\mathcal{L}_{\mathbf{X}}$ of the log-marginal likelihood $\log p(\mathbf{X})$ defined as follows:

$$\begin{aligned}
\log p(\mathbf{X}) &= \log \int p(\mathbf{X}, \mathbf{Z}) d\mathbf{Z} \\
&= \log \int q(\mathbf{Z}|\mathbf{X}) \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z}|\mathbf{X})} d\mathbf{Z} \\
&\geq \int q(\mathbf{Z}|\mathbf{X}) \log \frac{p(\mathbf{X}, \mathbf{Z})}{q(\mathbf{Z}|\mathbf{X})} d\mathbf{Z} \\
&\stackrel{\text{def}}{=} \mathcal{L}_{\mathbf{X}},
\end{aligned} \tag{2.7}$$

where the equality holds, *i.e.*, $\mathcal{L}_{\mathbf{X}}$ is maximized, if and only if $q(\mathbf{Z}|\mathbf{X}) = p(\mathbf{Z}|\mathbf{X})$. Since the gap between $\log p(\mathbf{X})$ and $\mathcal{L}_{\mathbf{X}}$ in (2.7) is given as the KL divergence

from $q(\mathbf{Z}|\mathbf{X})$ to $p(\mathbf{Z}|\mathbf{X})$, the minimization of the KL divergence is equivalent to the maximization of $\mathcal{L}_{\mathbf{X}}$. Practically, the target of training the generative and recognition model is to maximize the following approximation of $\mathcal{L}_{\mathbf{X}}$ using Monte Carlo integration:

$$\begin{aligned}\mathcal{L}_{\mathbf{X}} &= \mathbb{E}_{q(\mathbf{Z}|\mathbf{X})}[\log p(\mathbf{Z}) - \log q(\mathbf{Z}|\mathbf{X})] - \mathbb{E}_{q(\mathbf{Z}|\mathbf{X})}[\log p(\mathbf{X}|\mathbf{Z})] + \log p(\mathbf{X}) \\ &\approx \frac{1}{I}\sum_{i=1}^{I} \log p(\mathbf{X}|\mathbf{Z}_i) - KL(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})),\end{aligned} \tag{2.8}$$

where $\mathbf{Z}_i$ is a sample drawn from $q(\mathbf{Z}|\mathbf{X})$, and the number of samples I is usually set to 1 in a typical VAE implementation.

Its extension, known as the Conditional VAE (CVAE) [37], enables semi-supervised training for deep classification models by introducing the label as an additional latent variable. By introducing the latent label $\mathbf{Y}$, the generative process is described as follows:

$$p(\mathbf{X}, \mathbf{Y}, \mathbf{Z}) = p(\mathbf{X}|\mathbf{Y}, \mathbf{Z})p(\mathbf{Y})p(\mathbf{Z}). \tag{2.9}$$

Let $q(\mathbf{Y}|\mathbf{X})$ and $q(\mathbf{Z}|\mathbf{X})$ be the variational approximations for the latent variables $\mathbf{Y}, \mathbf{Z}$. The training objective is to maximize the ELBO of both $\log p(\mathbf{X}, \mathbf{Y})$ (supervised condition) and $\log p(\mathbf{X})$ (unsupervised condition), which is derived in the similar way with (2.7) and (2.8):

$$\mathcal{L}_{\mathbf{X},\mathbf{Y}} \approx \frac{1}{I}\sum_{i=1}^{I} \log p(\mathbf{X}|\mathbf{Z}_i, \mathbf{Y}) - KL(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})), \tag{2.10}$$

$$\mathcal{L}_{\mathbf{X}} \approx \frac{1}{I}\sum_{i=1}^{I} \log p(\mathbf{X}|\mathbf{Z}_i, \mathbf{Y}_i) - KL(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})) - KL(q(\mathbf{Y}|\mathbf{X})||p(\mathbf{Y})), \tag{2.11}$$

where $\mathbf{Y}_i$ is the sample drawn from $q(\mathbf{Y}_i|\mathbf{X})$, and the bound on the marginal likelihood for the entire dataset is:

$$\mathcal{L} = \sum_{\mathbf{X}} \mathcal{L}_{\mathbf{X}} + \sum_{\mathbf{X} with \mathbf{Y}} \mathcal{L}_{\mathbf{X},\mathbf{Y}}. \tag{2.12}$$

The classification model $q(\mathbf{Y}|\mathbf{X})$ contributes only to the unsupervised objective (2.11), and is not trained as a label estimator. To remedy this, the objective function is extended to include the classification loss term:

$$\mathcal{L}' = \mathcal{L} + \sum_{\mathbf{X} with \mathbf{Y}} \log q(\mathbf{Y}|\mathbf{X}). \tag{2.13}$$

In this way, the classification model is optimized in two ways:

1. Under the supervised condition, the classification model is trained to maximize the likelihood of the ground-truth labels *w.r.t*the model (the standard maximum likelihood estimation framework);
2. Under the unsupervised condition, the classification model is trained so that the generative model can correctly reconstruct the observed variable when conditioned by the estimated label. In other words, the generative model acts as a *regularizer* for the classification model to improve the classification performance.

2.2.4 Multi-Task Learning

The key of multi-task AMT is to properly formulate the semantic relations between different musical elements. As is mentioned in this section, HMMs have been extensively applied in earlier studies that model the hierarchical generative process of labels and features. Lee and Slaney [20] proposed joint training of multiple HMMs corresponding to different keys, for the likelihood and transition probabilities are considered to be dependent to keys. Given an unknown sequence of chroma feature, the chord sequence and musical key can be jointly determined by inferring the optimal sequences from all the HMMs, and selecting the one with the highest likelihood. In other studies, the relations between the musical concepts are hierarchically formulated as the emission probabilities conditioned by the latent states of HMMs [21, 22, 38]. This approach explicitly reflects our basic musical knowledge, *e.g.*, the usage of chord labels depend on the key of the music [5], chord transitions are more likely to occur at beat positions. Papadopoulo and Peeters [39] proposed a joint chord and downbeat

estimation model which focuses on the relationship between downbeats and chord transition positions.

For DNN-based methods, one common approach to deal with multiple musical elements is multi-task learning. In this approach, a DNN is trained to jointly estimate the posteriors of multiple target variables, so that it can implicitly learn the relations between different musical concepts, and improve the prediction reliability. A typical approach to multi-task learning is to decompose a single task into several sub-tasks. For example, Böck *et al.* [40] proposed a multi-task approach for simultaneous estimation of tempo, beat, and downbeat. Yang *et al.* [41] proposed a multi-task DNN that jointly estimates chords and root notes. Similarly, Mcfee and Bello [6] employed a structured training strategy for DNN-based music chord estimation, which jointly estimates root notes, bass notes and pitch classes as well as chord classes. The harmony transformer proposed by Chen and Su [28] jointly estimates chord sequence and chord transition positions to improve the estimation performance.

Some studies have explored the application of multi-task learning for joint estimation of different musical concepts. Böck *et al.* [42] used multi-task learning for joint estimation of tempos and beats, and showed its benefit for beat tracking. Jiang *et al.* [43] used crowd-sourced data to train a DNN-based multi-task classification model that jointly estimates keys, chords, beats, and melody scales. In the MIREX2019 competition, the multi-task classification improved the key estimation performance. These multi-task methods do not explicitly formulate the hierarchical semantic structures of the musical symbols.

2.3 Evaluation Metrics

The performance of an ACE method is typically measured by the overlap rate between the estimated and annotated chord labels. A standard metric employed by Music Information Retrieval Evaluation eXchange (MIREX) is called *chord symbol recall* (CSR):

$$CSR = \frac{T_{overlap}}{T_{annotation}}, \tag{2.14}$$

Table 2.1: Durations of Chord Qualities in the Common Datasets

Quality shorthand	Duration [h]
maj	**41.93**
min	**11.19**
aug	0.16
dim	0.21
sus2	0.25
sus4	1.70
maj6	0.93
min6	0.28
7	6.72
maj7	2.24
min7	6.00
minmaj7	0.03

where $T_{overlap}$ is the total duration of segments where the annotation equals the estimation, and $T_{annotation}$ is the total duration of annotated segments. Because the length of music can have a wide variety, CSR is often weighted by the length of the song when computing an average for a given corpus, which is referred to as the *weighted chord symbol recall* (WCSR). Some automation tools have been released for calculating the overlap rate [44].

The chord vocabulary is a key factor of chord estimation. The majority of related works adopted the most basic vocabulary, which includes only major and minor triads, plus a non-chord label. Because common annotation datasets use a much larger chord vocabulary, the chord labels should be reduced to major and minor triads following certain rules [45] when formulating and evaluating the ACE methods on the basic vocabulary. Larger chord vocabularies, which include rare triads, tetrads, and/or chord inversions, are very challenging evaluation metrics. The main difficulty in ACE with such vocabulary is distinguishing similar chords that share multiple pitch classes, or even the same roots (*e.g.*, *Cmaj* and *C7*). These chords sound very similar, so they are often difficult to distinguish on the basis of the audio feature; Moreover, the subjectivity of annotators leads to the inconsistency of chord annotations [46]. Some studies try to mitigate the difficulties by reflecting the musical knowledge about the

consistent tones and taxonomy of chord labels [6,7,47] into objective functions and/or by using an even-chance training scheme [48].

ACE studies use some common chord annotation datasets for evaluating the effectiveness of the proposed methods:

1. **Isophonics** [45]: 217 songs from the albums by The Beatles, Queen, and Zweieck;
2. **RWC-Popular** [49]: 100 Japanese and American style pop songs;
3. **Robbie Williams Dataset** [50]: 65 songs from the albums by Robbie Williams;
4. **uspop2002** [51]: 186 songs of American popular songs
5. **McGill Billboard Dataset** [52]: 726 songs that are randomly selected from Billboard charts.

In these datasets, the ground-truth chord progressions of these songs are annotated with the label syntax specified in [45]. One important property of these datasets is the imbalanced distribution of chord qualities: as shown in Table 2.1, the *major* and *minor* triads make up the majority of the duration, while the duration of other chord qualities are much shorter. In machine-learning based methods, the imbalance of label categories in training data has large impact on recognition performance. Lack of samples of chord annotations other than *major* and *minor* triads is another reason why large-vocabulary chord estimation is challenging.

Chapter 3

DNN-Based Chroma Vector Extraction

This chapter presents the data-driven feature extraction method from music audio. We construct a large set of time-synchronized MIDI-audio pairs to train a DNN feature extraction model, which is then used to estimate the pitch class saliences of real-world music audio recordings. The extracted multi-channel chromagram is then used for implementing a higher performance ACE model. Furthermore, we show a simple but effective approach for recognizing difficult chord types on the basis of the proposed feature representation.

3.1 Introduction

Most of the recent works on ACE tend to rely on purely data-driven methods on the basis of general purpose machine learning models. In these works, data pairs of music recordings and corresponding (time-synchronized) chord annotations are collected and used for supervised training. Thanks to DNN's high generalization capabilities, the trained chord classification models are able to achieve good performance without careful feature engineering. However, the performance is highly dependent on the amount of training data, which is usually hard to annotate or collect in sufficient quantities.

To overcome the limitation, we try to make use of music data in Music Instrument Digital Interface (MIDI) format as supplementary training data, to

improve the performance of ACE model at low cost [30]. Raffel [53] pointed out the potentials of MIDI files. A piece of music recorded as the MIDI format can provide detailed annotations of transcription, keys and meters, as long as they are matched to the corresponding audio recordings. Moreover, one can automatically synthesize the music audio using any sound library that supports General MIDI. In other words, MIDI music data can describe musical works in both *continuous* signal domain and *discrete* musical symbol domain. We can thus easily generate a large amount of audio-annotation pairs with guaranteed annotation correctness, and use them for machine-learning models.

Specifically, we formulate a DNN-based feature extraction model that estimates the frame-wise pitch class salience given an audio spectrogram. This model is trained using a large set of music audio synthesized from MIDI files, and the ground-truth pitch class salience, which is calculated from the note information. Although the model is trained on synthesized music data, it works well on real-world music recordings. We experimentally show in this chapter that the proposed data-driven model can introduce a strong domain knowledge into the chord classification model, and is able to effectively improve chord estimation performance.

The idea of using MIDI-synthesized music data for ACE task is also employed in Lee's work [19]. In his method, a symbolic music harmony analysis tool is used to transform the note information in the MIDI files to chord annotations. The generated annotations are used for training the HMM chord classification model, together with the synthesized music audio. Similarly, we use the note information to obtain the annotation for the synthesized audio, but the annotation format is the pitch-class salience representation.

3.2 Method

3.2.1 Preprocessing

Each audio signal (Either MIDI-synthesized audio or real-world recording) is first downsampled to 22,050Hz and transformed into log-frequency spectro-

gram via Constant-Q Transform (CQT), which is computed over 7 octaves (from note C1 to B7) with 12 bins per octave (the distance of neighboring bins is one semitone) and 2048 samples of hop size. CQT is calculated on the music audio multiple times with different minimum frequencies that are harmonically scaled: $h \cdot f_m in$ to obtain a multi-channel spectral representation, so that the harmonics are aligned across the channels. This stacked representation is called Harmonic CQT (H-CQT) [54], which has been applied in DNN-based pitch salience estimation task.

The calculated H-CQT spectrogram is normalized before it is used as model input. We first transform it into the log scale:

$$\mathbf{S}_{log} = ln(\mathbf{S} + \epsilon), \tag{3.1}$$

where $\mathbf{S}$ represents the raw CQT spectrogram and ϵ is a small number for avoiding zero value in log calculation. We then apply global mean-variance normalization to reduce the variance of overall spectral energy between different music pieces.

$$\mathbf{S}_{norm} = \frac{\mathbf{S}_{log} - mean(\mathbf{S}_{log})}{var(\mathbf{S}_{log})}. \tag{3.2}$$

Finally, the preprocessed H-CQT spectrogram is sent to the deep feature extraction model as the input matrix.

3.2.2 Target Representation

Our goal is to train a DNN that can transform the spectrogram S_norm into the desirable chroma representation. We let $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ be the sequence of chroma vectors corresponding to the music spectrogram, where $\mathbf{x}_n \in \{0, 1\}^D$ is a D-dimensional binary vector representing the existence of D pitch classes at frame n. Specifically, we let $\mathbf{x}_n$ include 3 different pitch class information ($D = 36$): the pitch class of bass note, top note, and middle notes. To convert a MIDI file into the ground-truth feature vector $\mathbf{X}$, first the note information is transformed into frame-wise piano-roll representation (the notes of percussive instruments are discarded). Then, note activations at each frame are reflected in the corresponding feature vector (Fig. 3.1):

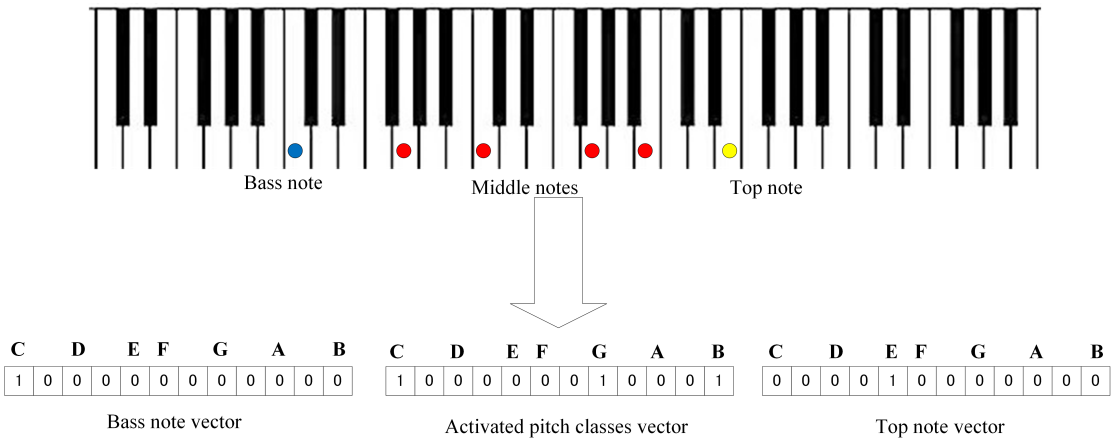


Figure 3.1: MIDI note activation of a time frame is represented in three 12-dimension vectors, indicating the pitch classes of current bass note, middle notes and top note, respectively.

- The pitch class of the highest note is assigned 1 in the **top note** channel;
- The pitch class of the lowest note is assigned 1 in the **bass note** channel;
- The pitch classes of all active notes except the highest and lowest notes are alligned 1 in the **middle note** channel.

We make full use of the advantage of MIDI data to provide more meaningful and far less error-prone annotations for feature learning.

Each value in the DNN's output represents a Bernoulli distribution that tells the probability whether the corresponding pitch class is present or not. The training objective of the DNN is to minimize the cross entropy between the output probabilities and the ground-truth values. After the DNN is trained, it can be used to effectively estimate the pitch class information of music recordings.

3.2.3 DNN for Feature Extraction

We formulate a fully convolutional neural network to construct a learnable pitch-class salience prediction model. As illustrated in Fig.3.2, the network is constructed with 7 convolutional layers. The first layer takes the multi-channel H-CQT spectrogram as the input, and convolves it with 64 learnable filters of size 3×11. Each kernel covers a pitch interval of 3 semitones, and a wider time

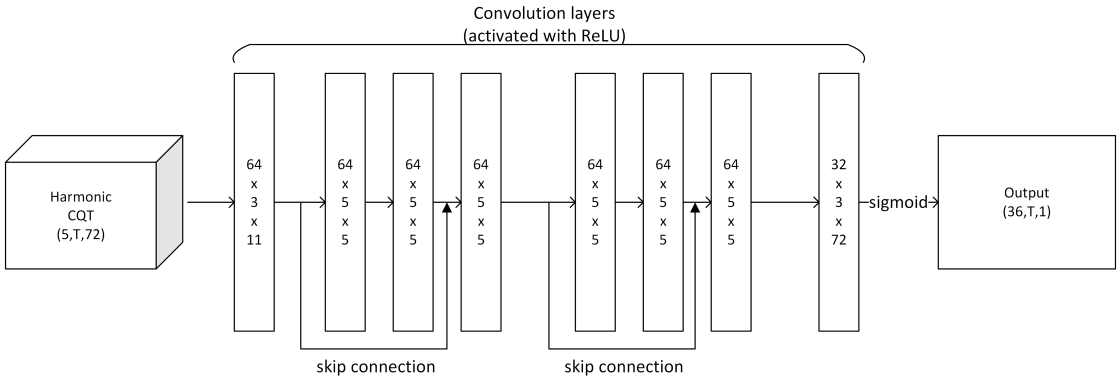


Figure 3.2: The architecture of the CNN feature extraction model.

interval of 11 frames (about 1 second). The following layers have 64 filters of size 5×5. The output of each convolutional layer is batch-normalized, and then activated with a rectifier linear unit (ReLU).

Two skip connections are introduced in the hidden layers, which means the input values of the layers are added to the output. Skip connection is a network design technique introduced in Resnet [55], which makes network training easier by letting the nonlinear transform learn the residual mapping between input and output, instead of direct mapping. The output layer has 36 filters, each filter covers the whole range across the second axis of input, and maps the covered area to an output channel which represents a single dimension of the feature. Finally, the output layer, which is activated with sigmoid function, outputs an array of size $36 \times N$, where each value is ranged from 0.0 to 1.0.

3.2.4 Network Training

The DNN is trained to minimize the cross-entropy error between the network output and the binary target vectors. The optimal parameters of the neural network is estimated using stochastic gradient descent algorithm. In light of our experience, the network is optimzed using AdaDelta optimization algorithm [56] in our experiments. As is described in Fig. 3.3, the music audio and the ground-truth feature representation for supervised training are both generated from the MIDI files.

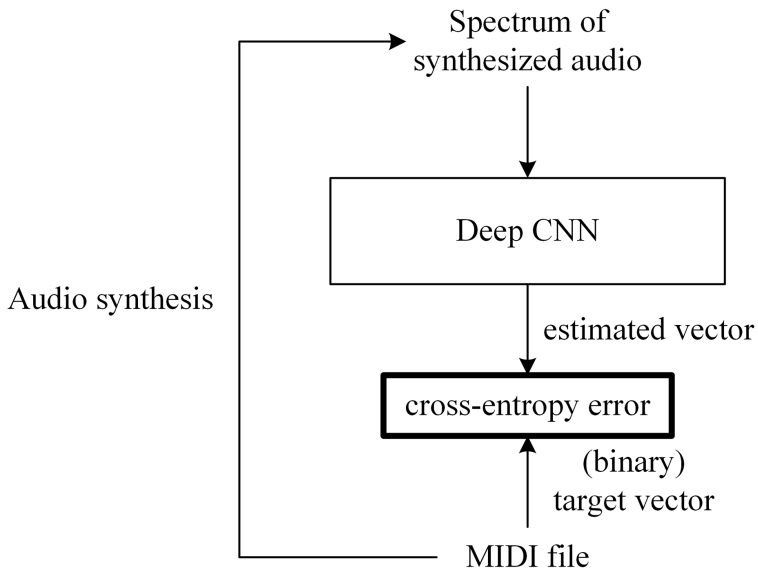


Figure 3.3: The calculation pipeline of the CNN feature extractior trained with the synthesized MIDI data.

For network training, we collected 6000 Standard MIDI files (SMFs) from *RWC Classical, Jazz and Genres dataset* [49] and Lakh MIDI dataset [53]. When synthesizing music audio, we used Chorium soundfont as the sound source of the instruments. An example of the chromagram extracted from song *Modoranai Natsu* in *RWC-Popular* dataset, in light of different extraction methods, is shown in Fig. 3.5. The three subplots are (a) logarithmic-compressed normal chromagram calculated from CQT spectrogram (b) chromagram calculated using the *deep chroma extractor* [8], and (c) the one calculated using the CNN trained with the proposed method (middle part), respectively. We also plot the "ground-truth" chroma according to the time-synchromized MIDI file of the song for reference.

Although the feature extraction model is trained on the synthesized music audio, Fig. 3.5 shows that it is capable of dealing with real-world music recordings. The advantages of the feature extracted with data-driven methods can be clearly observed from the examples: in (a), the percussion sounds often introduce disturbing energy peak across a wide frequency range, but in (b) and (c) they are effectively discarded. The feature (c) is obviously closer to the ground-

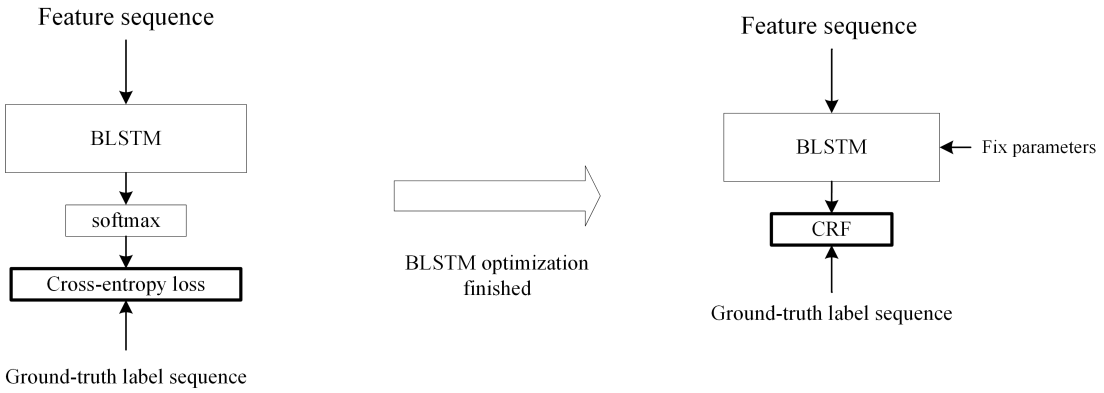


Figure 3.4: Overview of the training procedure for the BLSTM-CRF decoder. The BLSTM network is first trained as a classifier, then the CRF is trained with the same dataset. On each forward computation the input feature sequences are rolled along the pitch axis for a random number of times, and the ground truth label sequences are also correspondingly adjusted.

truth feature than (b), showing that the MIDI-trained feature can more precisely extract the actual pitch class information.

3.2.5 Chord Sequence Decoder

To complete the automatic chord estimation system, we design a decoding model for pattern matching and decoding chord sequences when the proposed feature sequence is given. The decoding model is consisted of a Bidirectional Long-Short Term Memory (BLSTM) model that performs pattern matching, and a linear chain Conditional Random Field (CRF) The overall architecture of the proposed ACE method is illustrated in Fig. 3.6.

3.2.6 Decoder Training

The ordinary audio-chord annotation pair dataset is used for training the classifier and the decoder. More specifically, the training dataset is composed of pairs of chroma vector sequences (obtained from above feature extraction stage) and time-synchronized chord annotation data. As there is no need for the decoder to learn the dependency across the whole music, the BLSTM-CRF model is trained by randomly taking a sequence of a fixed length (128 frames, or about

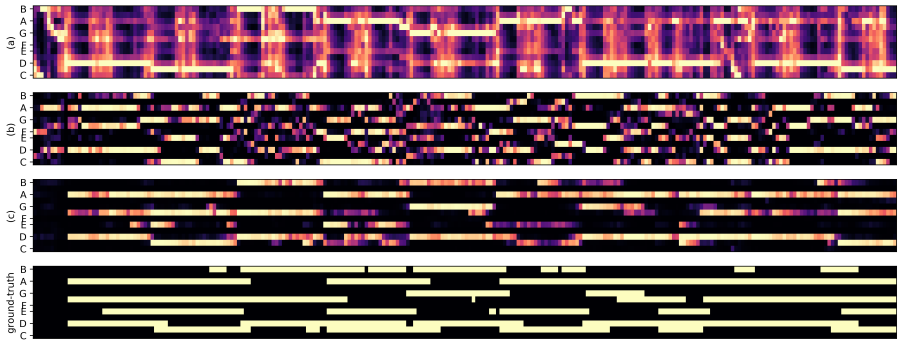


Figure 3.5: Illustration of the chroma vector sequence calculated from an audio snippet of song *Modoranai natsu* in RWC-Popular dataset, with different feature extraction methods: (a) logarithmic-compressed normal chromagram computed from the CQT spectrogram; (b) chromagram extracted with the *deep chroma extractor*; (c) chromagram(the middle note part) extracted with the proposed CNN extractor trained on the MIDI dataset. The bottom plot is the ground-truth chromagram calculated from the time-synchronized MIDI file of the song.

10 seconds) each time.

As illustrated in Fig. 3.4, the classifier (BLSTM) and the decoder (CRF) component are trained individually. First, we train the BLSTM network with the output layer activated with softmax function, to classify the feature sequence by itself. Then we fix the well-trained parameters of the BLSTM network and use the same dataset to train the parameters of the CRF. To compensate for the chord class imbalance within the dataset, on each forward computation in the training stage, the bass, middle, and top note part of the input chroma vectors are rolled along the pitch axis for a random number of time (from 0 to 11). The target ground-truth sequences are also adjusted according to the shift amount.

3.3 Towards Large Vocabulary Chord Estimation

As is discussed in Chapter 1, major and minor triads are just a small part of the chord vocabulary defined by the musical theory. Recognizing the chord types that are less frequently used than major and minor triads is a challenging problem in ACE researches. Most chord estimation methods, including the pro-

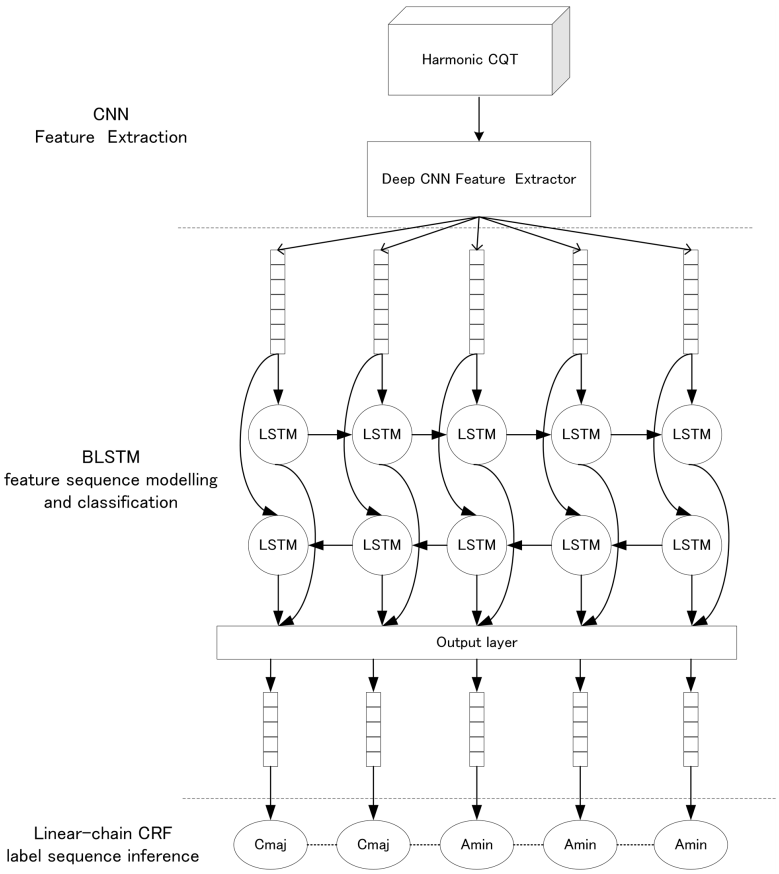


Figure 3.6: Overview of the presented chord classification model. The model is composed of a deep CNN feature extraction model trained with MIDI dataset, a BLSTM sequence classifier and a CRF sequence decoder.

posed neural network architecture, are focused on recognizing the 24 major and minor triads. The complex chords are generally quantized into the root triads for simplicity. Typically, the estimation process is regarded as a quantization process that assigns all observations to corresponding one-of-K representations, built on the assumption that the 24 classes are mutually independent.

However, if we add the above complex chords into the vocabulary, this assumption no longer holds, because there exist chords related hierarchically [57]. In addition, as discussed in Chapter 1, there is a large imbalance between the amount of normal triad chords and other chord types in the annotated datasets. In the flat classification scheme, the chord vocabulary is essentially not extendable: it is impossible to train the classifier to recognize some rare chord

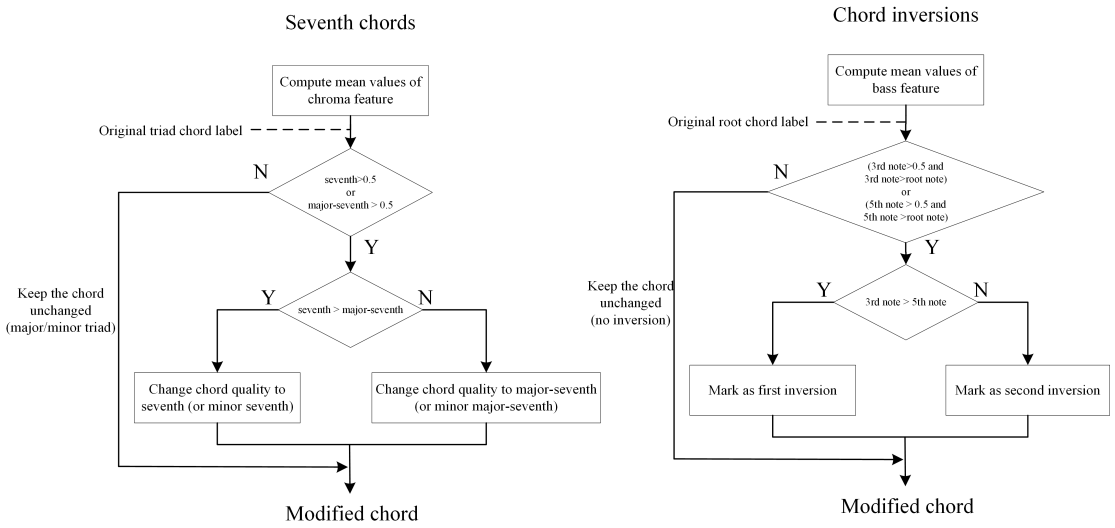


Figure 3.7: The flow chart of chord type decision process for seventh and chord inversions. Given the estimated triads and the corresponding feature sequence, the actual chord types are determined with simple thresholding rules and comparisons of the average values of the feature.

labels which definitely have explicit definitions but not used in the training dataset. To make large vocabulary chord estimation systems extendable, it is necessary that we go back to the *explicit definitions* of those complex chords at an appropriate point, rather than following a flat classification scheme that regards sevenths and inversions as new independent chords.

We experimentally design a two-stage complex chord estimation method to verify the effectiveness of our approach. Specifically, we introduce an additional post-processing step that modifies the quality and inversion of each recognized chord signature. Given a chord signature (major or minor triad) and the feature sequence of the corresponding time region, the mathematical means of the feature values are calculated along the dimension of its third, fifth, seventh, and major-seventh notes, and bass feature values of its root, third, and fifth notes. Then the true quality and inversion are determined with an explicit thresholding metric as described in Fig. 3.7. In this way, the chord estimation system is able to support 60 types of chords if considering the seventh chords, and 180 types of chords if taking first and second chord inversions into account. At the same

time, the estimation accuracy in triads metric is not affected.

3.4 Evaluation

In this section, we first describe the training and testing datasets and give the definitions of different evaluation metrics. Then we present the results of the quantitative experiments to investigate the ACE performance of our proposed system under various conditions.

3.4.1 Datasets and Evaluation Metrics

The proposed system and other variations is evaluated on a compound dataset comprising the **Isophonics** and **RWC-Popular** dataset. We performed a 6-fold cross-validations on the dataset, and computed the per-song weighted accuracy of the estimated labels using *mir_eval* evaluation toolbox [44]. The accuracy is calculated using the three metrics implemented in the library:

- **Majmin**, considers only major and minor triads, which is the most conventional comparison metric;
- **Sevenths**, considers seventh, minor-seventh, and major-seventh qualities in addition;
- **Sevenths-inv**, considers triads, sevenths and chord inversions.

The three metrics correspond to "majmin", "sevenths", and "sevenths-inv" metrics in MIREX evaluation.

During the cross-validation process, the same feature extraction model that is pre-trained on 6000 pieces of MIDI music data from Lakh MIDI dataset [53] is used. Note that the two evaluation datasets do not contain any songs in the training set used for the feature extraction model. We also compare our system with the estimation results of MIREX2017 submissions for *Audio Chord Estimation* task. The submitted systems are trained on different datasets, but are evaluated on the same **Isophonics** and **RWC-Popular datasets**, using the estimation results of the submissions which are available online.

3.4.2 Experimental Conditions

To evaluate the deep harmonic feature extracted with the proposed CNN, we do the same 6-fold evaluation on our proposed systems (denoted as **CNN-BLSTM-CRF**) and its two alternatives:

- **BLSTM-CRF**: The CNN feature extraction model is removed and the BLSTM-CRF decoder takes the raw CQT spectrum sequences as input;
- **DC-BLSTM-CRF**: The feature extraction model is replaced with a re-implementation of the *deep chroma extractor* [8];
- **DRN-BLSTM-CRF**: The feature extraction model is replaced with the frame-wise DRN model proposed in [58];
- **CNN-CRF**: The baseline ACE model proposed by Korzeniowski [29].

The compared MIREX submissions are:

- **MIREX2017-KBK1/2**: The pre-trained implementations of chord classifier using the *deep chroma extractor* (KBK1) [8] and the CNN-CRF classifier (KBK2) [29], designed specifically for (25-classes) triad chord estimation. According to the abstract by the authors, the two submissions are pre-trained with about 1000 tracks of annotated music audio, including the Isophonics and RWC dataset, so the scores are not entirely comparable with other methods, namely they might be higher than they should be;
- **MIREX2017-JLW2**: The chord estimation system on the basis of a random forest classifier and integrated estimation of beat and chord sequence with CRF. This model is trained on NNLS-Chroma sequences and their annotations released by McGill Billboard Project [52];
- **MIREX2017-CM2** The Chordino [11] system which is oriented to chord estimation for large chord vocabulary.

3.4.3 Results and Discussions

Below we compare the proposed system with its variances and other related systems, in terms of major/minor triads and complex chords under the two public

Table 3.1: Cross-validated WAOR in majmin metric

	Isophonics	**RWC-Popular**
CNN-CRF	84.3%	82.0%
BLSTM-CRF	82.0%	77.2%
DC-BLSTM-CRF	82.9%	79.3%
DRN-BLSTM-CRF [58]	84.1%	80.8%
CNN-BLSTM-CRF	**85.3%**	**84.3%**

Table 3.2: Majmin metric WAOR comparison with MIREX submissions

	Isophonics	**RWC-Popular**
MIREX2017-KBK1	82.7%	81.3%
MIREX2017-KBK2	82.0%	87.2%
MIREX2017-JLW2	80.9%	82.2%
CNN-BLSTM-CRF	**85.3%**	**84.3%**

datasets. Furthermore, we investigate the influence of deep feature extraction and the size of the MIDI dataset on the chord estimation performance.

Major and Minor Traiads Estimation

The cross-validated scores in *majmin* metric are listed in Table 3.1. The last three rows of Table 3.1 indicate that **CNN-BLSTM-CRF** achieved better performance than its two variants **BLSTM-CRF** and **DC-BLSTM-CRF** on both the Isophonics and RWC-Popular dataset. Concretely, when using the MIDI-trained feature extraction model, the cross-validated WAOR improved by about $4-7\%$ compared with the case where raw CQT was used, and about $2-5\%$ compared with the case where *deep chroma extractor* was used. Since the three systems used the same decoding modules (the **BLSTM-CRF** architecture), this result verifies the effectiveness of the proposed feature. Moreover, the proposed method achieves higher performance than the frame-wise DRN [58], which shows the advantage of employing the CNN architecture.

Compared with the **CNN-CRF** baseline system, **CNN-BLSTM-CRF** system also showed the advantage. The cross-validation score improved by about $1-2\%$ compared with the **CNN-CRF** system, which verifies the effect of introducing MIDI data for feature learning from another perspective.

Table 3.3: Larger vocabulary metric WAOR comparison with MIREX submissions

	Sevenths		Sevenths-inv	
	Isophonics	RWC-Popular	Isophonics	RWC-Popular
MIREX2017-CM2	54.8%	63.3%	52.4%	59.8%
MIREX2017-JLW2	67.3%	70.1%	65.8%	67.9%
DC-BLSTM-CRF	68.2%	64.2%	64.4%	61.5%
DRN-BLSTM-CRF []	71.9%	66.5%	67.3%	63.7%
CNN-BLSTM-CRF	**71.2**%	**67.9**%	**67.3**%	**65.1**%

In Table 3.2, the cross-validated scores of *CNN-BLSTM-CRF* are compared with the evaluation scores of the three MIREX2017 submissions. According to Table 3.2, the proposed system, trained with large MIDI dataset and comparatively small annotated data, achieved competitive results compared with the scores of the MIREX submissions. The scores of major/minor triads classification were higher than the score of **MIREX2017-KBK1** and **MIREX2017-JLW2** system, but lower than the scores of **MIREX2017-KBK2** system by about 2% for each dataset. This comparison indicates that the amount of manually-annotated training data is still a critical factor for deep-learning based chord estimation system, but introducing MIDI data for feature training can help the chord estimation system to get much closer to those models trained with the larger fully-annotated datasets.

Large Vocabulary Chord Estimation

We show the estimation scores of larger chord vocabulary in Table 3.3. In this table, we again take the results from MIREX. The two MIREX submissions represent the typical one-of-K learning and classification approach for large-vocabulary chord estimation. The JLW2 submission is the state-of-the-art system in large vocabulary chord estimation evaluation metric in MIREX chord estimation task.

In this comparison, **CNN-BLSTM-CRF** achieved better overall evaluation score than the CM2 submission and the DC-BLSTM-CRF algorithm. It outperformed JLW2 submission on Isophonics dataset, but achieved a lower score on

RWC. This result indicates that the machine-learning based classification still has advantages over the naive thresholding method, even though we are working on a clearer harmonic representation.

Since the overall WAOR in the seventh or seventh-inv metric tends to be correlated with WAOR in majmin metric, we need to further look into the estimation results to know the actual performance of complex chord estimation. In Fig. 3.8 we present the confusion matrices of sevenths and chord inversion estimation. The ratios in the matrices are computed on all intervals that are correctly recognized in the triad chord level in each system.

For seventh chord estimation, it can be observed that the proposed method tended to misclassify triads as seventh chords more frequently than **JLW2**. In addition, **JLW2** recognized major seventh chords more accurately than the proposed method. This can be the main reason why our method ' s WAOR of complex chord estimation was lower than **JLW2** in RWC-Popular dataset, where complex chord types appeared more frequently. On the other hand, the **CM2** system resulted in better confusion matrix than our proposed method, even though the WAOR score was lower. The difference in the accuracy of triad-level classification was responsible for this result. For chord inversion estimation, the proposed approach showed better performance than both MIREX submissions. This result can be attributed to the high accuracy of bass note estimation with the feature extraction model.

In summary, we tried to recognize complex chord types based on explicit definition. The proposed method was able to achieve competitive performance in complex chord estimation using the naive and extendable decision rules, though there was still much room for improvement. We believe that definition-based complex chord estimation is a promising approach for future research.

3.4.4 Influence of the Training Dataset Size

To extract the harmonic feature, the deep convolutional network was trained on a large set (6000 pieces of music for the above experiments) of MIDI files and their synthesized audio signals. The performance of a deep learning-based

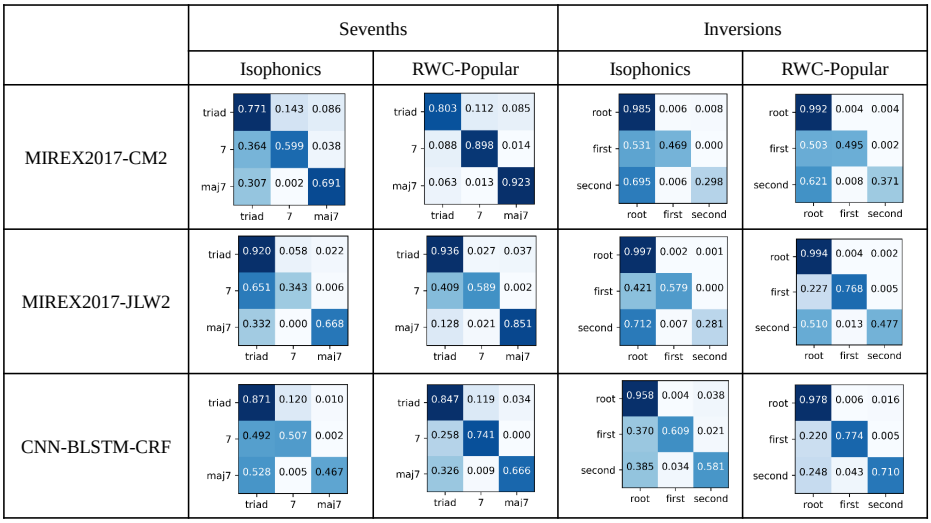


Figure 3.8: The confusion matrices of large vocabulary chord estimation by the proposed method and the MIREX submissions.

system heavily depends on the amount of data amount used for training. To examine the influence of data amount on ACE performance, we varied the MIDI dataset size from 100 to 6000 tracks, and examined the relationship between the estimation accuracy and different amount of training data. Fig. 3.9 shows the relationship curve, where evaluation scores in *majmin* metric were calculated again on *Isopnonics* and *RWC-Popular* dataset, respectively. It can be apparently observed that with the increase of the training data, the estimation performance simultaneously ascended. Especially, when the dataset size was below 1000, the curve went up rapidly. When the dataset size increased from 1000 to 2000 and 3000, the curve tended to be converged, and the overall chord estimation performance would no longer improve significantly.

From this analysis, we see that the performance came to an inflection point when the training dataset size is around 2000. Since the chord estimation accuracy reflects a whole behavior of the front-end feature extraction model trained on the MIDI dataset and the backend BLSTM-CRF decoder, it would be interesting to investigate how the performance of the model itself changes with the MIDI training dataset size. To this end, we designed another simple “note

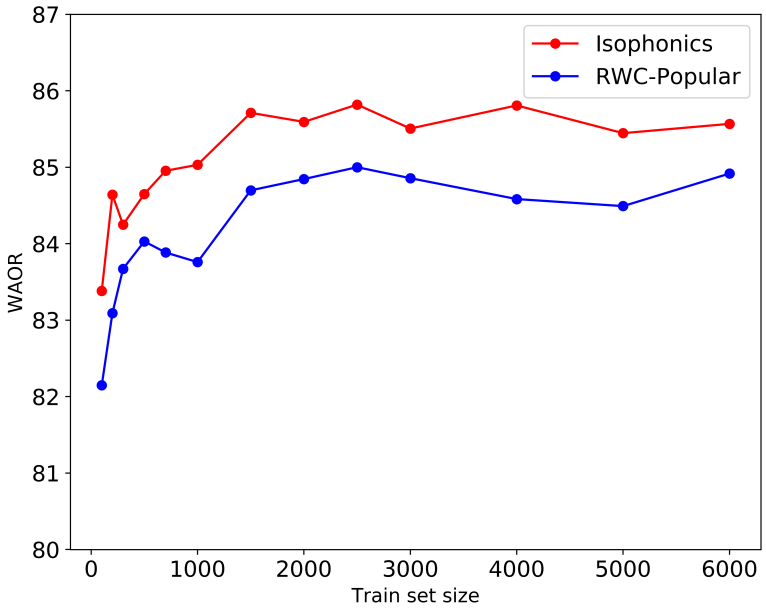


Figure 3.9: The evaluation scores in *majmin* metric, with the feature extraction model trained with the train set of different size.

transcription " evaluation on the MIDI dataset. More specifically, we randomly chose 500 MIDI files as the test set, and observed how well the feature extraction model (trained with various train set sizes) could reveal the pitch class activation states of each frame. The trained feature extraction model transformed the spectrogram of each synthesized audio into the feature vector sequence. The feature vector values were then quantized into binary values; If the vector value was bigger than/less than 0.5, the corresponding pitch class is marked as " active/inactive " , and vice versa. We compared the quantized feature vector sequence with the target vector sequence calculated from MIDI note information and calculated the F-measures for note activation detection.

The estimation performances of the estimator under different conditions are summarized in Fig. 3.10. It can be observed that the bigger the training set size was, the higher the F-measure became. Unlike the overall performance, this monotonically increasing tendency holds even when the training data amount sent up to 6000 songs (though with regard to the top note vector, the estimation

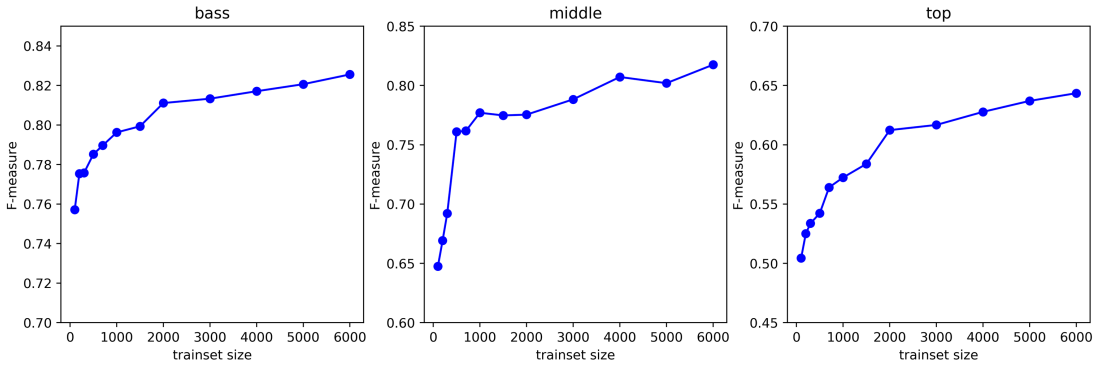


Figure 3.10: F-measures for bass note, middle notes and top note salience estimation, with the feature extraction model trained with train sets in different size.

accuracy is at a relatively low level). This result indicates that with the increase of MIDI training dataset, the extracted deep feature is able to describe the pitch class salience more precisely.

To sum up the experiments on the feature extraction model, the overall chord estimation accuracy quickly reached an upper bound when we keep the MIDI train set increasing, even though the feature extraction model was providing better extracting results. It reveals the fact that the bottleneck of our proposed system comes from the sequence classification model. Essentially, this result shows the limitation of the general approach in machine learning-based music content analysis researches, which align the discrete symbol domain with the continuous signal domain by cutting the symbols into short-time frame sequence. There has been research work that reported the incompatibility of contemporary sequence models to properly model frame-level chord label sequences, and our experiments resulted in supporting this conclusion from another perspective.

3.5 Conclusion

In this section, we mainly propose a novel feature learning procedure for ACE task, which is based on the idea of training a CNN-based feature extraction model using MIDI music data. We showed that the MIDI-trained feature ex-

traction model is fully capable of extracting harmonic information from real-world music audio recordings, and significantly improve the ACE performance. Through the experiments, we show the flexibility of feature representation learnt by the CNN, thanks to the complete and perfectly precise note information that MIDI files provide. We believe that there are still potentials in this method to further improve the performance with more carefully designed representations. We also aim at improving the complex chord estimation process with machine-learning based strategies, instead of the explicit thresholding metric. Moreover, similar to the feature obtained by the *deep chroma extractor*, we believe that the proposed feature (or the feature learning procedure) is compatible with other MIR tasks that use chroma vectors.

Chapter 4

VAE-Based Semi-supervised Chord Estimation

This chapter describes a statitically-principled semi-supervised method of ACE that can make effective use of deep generative models [59]. Different from conventional DNN-based ACE methods that model the discriminative process only, we propose a unified *generative* and *discriminative* approach in the framework of amortized variational inference.

4.1 Introduction

As discussed in Chapter 2, conventional ACE methods only formulate the discriminative process using DNNs, which oversimplifies the real-world music transcription process. Music transcription is essentially an activity to *guess* the composer's intension (which is notated as the musical score) from the observed music audio. From this point of view, a generative model that formulates the probabilistic process of music composition is essential when formulating a music transcription model: a good transcription is the one that maximizes the likelihood of the observation (music audio) with respect to the generative model conditioned by the transcription. In earlier AMT researches, the usage of temporal generative models on the basis of hidden Markov models (HMMs) [18] have been extensively discussed. HMM-based formulations assume that the latent variable sequences are Markov chains, and that the observed feature sequences

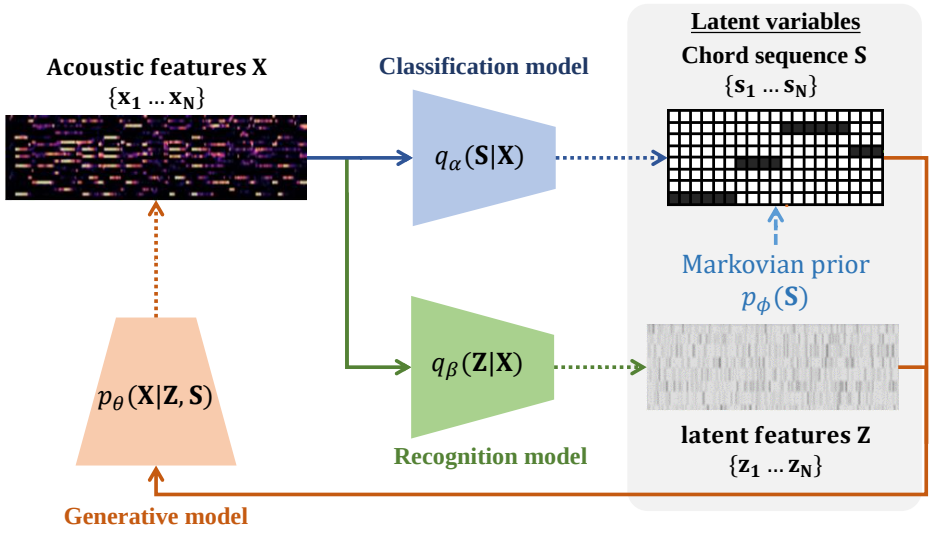


Figure 4.1: The variational autoencoder consisting of a deep generative model of chroma vectors, a deep classification model of chord labels, and a deep recognition model of latent features. Dashed arrows indicate stochastic relations. These three models are trained jointly in a semi-supervised manner by using annotated and non-annotated music signals.

are conditionally independent *w.r.t*each time step, so the optimal latent variable sequence, which maximizes the posteriors given an observed feature sequence can be efficiently inferred using Viterbi algorithms.

In conventional methods, the generative and discriminative models are integrated at the post processing stage: the optimal label sequence that maximizes the product of label posteriors *w.r.t*the discriminative model and the likelihood *w.r.t*the generative model is determined, using efficient path-finding method like Viterbi algorithm or beam-search algorithm [32]. In contrast to these methods, we propose a unified training methodology which jointly optimizes the deep generative and discriminative models to effectively improve chord estimation performance. More specifically, we formulate a deep generative model that represents the generative process of chroma vectors (observed variables) from discrete chord labels and continuous features (latent variables). To complete the Bayesian formulation, we further introduce the prior distributions on the latent variables: the chord labels are assumed to follow a first-order Markov model favoring self-transition, and the latent features, which abstractly represent the

fine structure of the chroma vectors, are assumed to follow a standard Gaussian distribution. In the framework of amortized variational inference (AVI) [60], DNN-based discriminative models are then introduced as variational posterior distributions to approximate the posterior distribution of the latent variables from an observed chroma sequence. The generative and discriminative models can be trained jointly in a semi-supervised manner by using music signals with or without chord annotations.

The VAE has two different latent variables corresponding to chord labels (*categorical* variables) and latent features (*continuous* variables). This model is similar to JointVAE [61] in a sense that both discrete and continuous representations are learned jointly. In our model, a Markov prior favoring self-transition is put on chord labels for regularizing the classification model in the VAE training. This is different from conventional studies where the Markov prior is used only for smoothing chord labels estimated by a classification model.

The main contribution of the proposed method is to draw the potential of the powerful deep discriminative model for ACE, by integrating it into the principled statistical inference formalism of the generative approach. This is the first attempt to use a VAE for semi-supervised ACE. We experimentally show the effectiveness of regularization by the deep generative model and the Markov prior on semi-supervised training.

4.2 Method

4.2.1 Problem Specification

For a simplified explanation, suppose that we have one musical piece. Let $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ be the sequence of observed variable of length N, representing the audio feature of a piece of music. In our method, we use the MIDI-trained feature extraction model described in Chapter 3 to obtain the observed variables from music audio. The advantage of the proposed chroma feature is that it is a highly abstracted representation of the music audio, and its generative model is easy to learn, compared with other representations such as raw audio

spectrogram or traditional chroma vectors.

Let $\mathbf{S} = \{\mathbf{s}_1, \cdots, \mathbf{s}_N\}$ be a sequence of chord labels corresponding to $\mathbf{X}$, where $\mathbf{s}_n \in \{0, 1\}^K$ is a categorical variable (K-dimensional one-hot vector) indicating the chord label of frame n. Here, the chord vocabulary consists of all possible combinations of the twelve root notes with six types of triad chords (with shorthands *maj*, *min*, *dim*, *aug*, *sus2*, and *sus4*) and two types of power chords (with shorthands *1* and *5*), and a no-chord label (with shorthand *N*), *i.e.*, $K = 12 \times 8 + 1 = 97$.

Let $\mathbf{Z} = \{\mathbf{z}_1, \cdots, \mathbf{z}_N\}$ be a sequence of latent features, where $\mathbf{z}_n \in \mathbb{R}^L$ is a continuous variable that abstractly represents how $\mathbf{x}_n$ is deviated from a basic chroma pattern specified by $\mathbf{s}_n$ ($L = 64$). *i.e.*, not only $\mathbf{s}_n$ but also $\mathbf{z}_n$ are needed to fully describe $\mathbf{x}_n$. If a region of C:maj includes some passing notes with different volumes, for example, the chroma vectors are deviated from a basic pattern that take the value 1 at the dimensions corresponding to C, E, and G.

Our goal is to train a classification model $p(\mathbf{S}|\mathbf{X})$ and use it for estimating chord labels behind unseen music signals. In an unsupervised condition, the classification model should be trained from $\mathbf{X}$ without using the ground-truth data of $\mathbf{S}$. In a supervised condition, the classification model can be trained by using paired data of $\mathbf{X}$ and $\mathbf{S}$. In a semi-supervised condition, some part of $\mathbf{S}$ is given as ground-truth data.

4.2.2 Generative Model

We formulate a probabilistic hierarchical generative model (joint probability distribution) of chord labels $\mathbf{S}$, latent features $\mathbf{Z}$, and chroma vectors $\mathbf{X}$ as follows:

$$p_{\theta,\phi}(\mathbf{X}, \mathbf{S}, \mathbf{Z}) = p_\theta(\mathbf{X}|\mathbf{S}, \mathbf{Z}) p_\phi(\mathbf{S}) p(\mathbf{Z}), \tag{4.1}$$

where $p_\theta(\mathbf{X}|\mathbf{S}, \mathbf{Z})$ is a likelihood function of $\mathbf{S}$ and $\mathbf{Z}$ for $\mathbf{X}$, $p_\phi(\mathbf{S})$ is a prior distribution of $\mathbf{S}$, $p(\mathbf{Z})$ is the prior of $\mathbf{Z}$, and θ and ϕ are model parameters. While the standard HMMs used for ACE are represented as $p_{\theta,\phi}(\mathbf{X}, \mathbf{S}) = p_\theta(\mathbf{X}|\mathbf{S}) p_\phi(\mathbf{S})$ [62], we use both $\mathbf{S}$ and $\mathbf{Z}$ for precisely representing $\mathbf{X}$, *i.e.*, $p_\theta(\mathbf{X}|\mathbf{S}, \mathbf{Z})$ is a deep *gen-*

erative model represented as follows:

$$p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z}) = \prod_{n=1}^{N}\prod_{d=1}^{D} \mathrm{Bernoulli}(x_{nd}|[\boldsymbol{\omega}_\theta(\mathbf{S},\mathbf{Z})]_{nd}), \tag{4.2}$$

where $\boldsymbol{\omega}_\theta(\mathbf{S},\mathbf{Z})$ is the ND-dimensional output of a DNN with parameters θ that takes $\mathbf{S}$ and $\mathbf{Z}$ as input and the notation $[\mathbf{A}]_{ij}$ indicates the ij-th element of $\mathbf{A}$. Although x_{nd} should take a binary value in theory, we allow it to take a real value between 0 and 1 from a practical point of view.

If we have no prior knowledge on chord labels $\mathbf{S}$, a natural choice of $p_\phi(\mathbf{S})$ is a uniform distribution as follows:

$$p_\phi(\mathbf{S}) = \prod_{n=1}^{N} \mathrm{Categorical}\left(\mathbf{s}_n|\left[\tfrac{1}{K},\cdots,\tfrac{1}{K}\right]\right). \tag{4.3}$$

Since chord labels $\mathbf{S}$ have temporal continuity, *i.e.*, change infrequently at the frame level, we follow the common assumption on chord labels [63] and use a first-order Markov model favoring self-transitions of chord labels as the chord language model $p_\phi(\mathbf{S})$ as follows:

$$\begin{aligned} p_\phi(\mathbf{S}) &= p(\mathbf{s}_1)\prod_{n=2}^{N} p(\mathbf{s}_n|\mathbf{s}_{n-1}) \\ &= \prod_{k=1}^{K} \phi_k^{s_{1k}} \prod_{n=2}^{N}\prod_{k'=1}^{K}\prod_{k=1}^{K} \phi_{k'k}^{s_{n-1,k'}s_{nk}}, \end{aligned} \tag{4.4}$$

where ϕ_k is the initial probability of chord k and $\phi_{k'k}$ is the transition probability from chord k' to chord k. Since ϕ_k and $\phi_{k'k}$ represent probabilities, $\sum_{k=1}^{K}\phi_k = 1$, $\sum_{k=1}^{K}\phi_{k'k} = 1$. In the ACE researches that use HMMs, the transition probability matrices typically have high self-transition probability, which reflects the continuity of chord label sequences [24]. In this paper, we set $\phi_{kk} = 0.9$.

Since we have no strong belief about the abstract latent features $\mathbf{Z}$, the prior $p(\mathbf{Z})$ is set to a standard Gaussian distribution as follows:

$$p(\mathbf{Z}) = \prod_{n=1}^{N} \mathcal{N}(\mathbf{z}_n|\mathbf{0}_L,\mathbf{I}_L), \tag{4.5}$$

where $\mathbf{0}_L$ is the all-zero vector of size L and $\mathbf{I}_L$ is the identity matrix of size $L\times L$.

4.2.3 Classification and Recognition Models

Given chroma vectors $\mathbf{X}$ as observed data, we aim to infer the chord labels $\mathbf{S}$ and the latent features $\mathbf{Z}$ from $\mathbf{X}$ and estimate the model parameters θ and ϕ in the maximum-likelihood framework. Because the DNN-based formulation makes the posterior distribution $p_{\theta,\phi}(\mathbf{S},\mathbf{Z}|\mathbf{X}) \propto p_{\theta,\phi}(\mathbf{X},\mathbf{S},\mathbf{Z})$ analytically intractable, we compute it approximately with an AVI technique. More specifically, we introduce a sufficiently expressive variational distribution $q_{\alpha,\beta}(\mathbf{S},\mathbf{Z}|\mathbf{X})$ parametrized by α and β and optimize it such that the Kullback-Leibler (KL) divergence from $q_{\alpha,\beta}(\mathbf{S},\mathbf{Z}|\mathbf{X})$ to $p_{\theta,\phi}(\mathbf{S},\mathbf{Z}|\mathbf{X})$ is minimized. Considering that both $\mathbf{S}$ and $\mathbf{Z}$ make an effect on the generative model $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$, they are assumed to be conditionally independent in the inference model $q_{\alpha,\beta}(\mathbf{S},\mathbf{Z}|\mathbf{X})$ as follows:

$$q_{\alpha,\beta}(\mathbf{S},\mathbf{Z}|\mathbf{X}) = q_\alpha(\mathbf{S}|\mathbf{X})q_\beta(\mathbf{Z}|\mathbf{X}), \tag{4.6}$$

where $q_\alpha(\mathbf{S}|\mathbf{X})$ and $q_\beta(\mathbf{Z}|\mathbf{X})$ are *classification* and *recognition* models that infer $\mathbf{S}$ and $\mathbf{Z}$, respectively. In our study, (4.6) was found to work better than $q_{\alpha,\beta}(\mathbf{S},\mathbf{Z}|\mathbf{X}) = q_\alpha(\mathbf{S}|\mathbf{X})q_\beta(\mathbf{Z}|\mathbf{X},\mathbf{S})$ respecting the chain rule of probability as in [37]. In this paper, these models are implemented with DNNs parameterized by α and β as follows:

$$q_\alpha(\mathbf{S}|\mathbf{X}) = \prod_{n=1}^{N} \mathrm{Categorical}(\mathbf{s}_n|[\boldsymbol{\pi}_\alpha(\mathbf{X})]_n), \tag{4.7}$$

$$q_\beta(\mathbf{Z}|\mathbf{X}) = \prod_{n=1}^{N} \mathcal{N}(\mathbf{z}_n|[\boldsymbol{\mu}_\beta(\mathbf{X})]_n, [\boldsymbol{\sigma}^2_\beta(\mathbf{X})]_n), \tag{4.8}$$

where $\boldsymbol{\pi}_\alpha(\mathbf{X})$ is the NK-dimensional output of the DNN with parameters α, and $\boldsymbol{\mu}_\beta(\mathbf{X})$ and $\boldsymbol{\sigma}^2_\beta(\mathbf{X})$ are the NL-dimensional outputs of the DNN with parameters β. Similar to the deep generative model, the outputs of the DNNs represent the parameters of probabilistic distributions.

4.2.4 Unsupervised Learning with Non-Annotated Data

Instead of directly maximizing the log-marginal likelihood $\log p_{\theta,\phi}(\mathbf{X})$ with respect to the model parameters θ and ϕ, we maximize its variational lower bound

$\mathcal{L}_{\mathbf{X}}(\theta, \phi, \alpha, \beta)$ derived by introducing $q_{\alpha,\beta}(\mathbf{S}, \mathbf{Z}|\mathbf{X})$ as follows:

$$\begin{aligned}
\log p_{\theta,\phi}(\mathbf{X}) &= \log \iint p_{\theta,\phi}(\mathbf{X}, \mathbf{S}, \mathbf{Z}) d\mathbf{S} d\mathbf{Z} \\
&= \log \iint \frac{q_{\alpha,\beta}(\mathbf{S}, \mathbf{Z}|\mathbf{X})}{q_{\alpha,\beta}(\mathbf{S}, \mathbf{Z}|\mathbf{X})} p_{\theta,\phi}(\mathbf{X}, \mathbf{S}, \mathbf{Z}) d\mathbf{S} d\mathbf{Z} \\
&\geq \iint q_{\alpha,\beta}(\mathbf{S}, \mathbf{Z}|\mathbf{X}) \log \frac{p_{\theta,\phi}(\mathbf{X}, \mathbf{S}, \mathbf{Z})}{q_{\alpha,\beta}(\mathbf{S}, \mathbf{Z}|\mathbf{X})} d\mathbf{S} d\mathbf{Z} \\
&\stackrel{\text{def}}{=} \mathcal{L}_{\mathbf{X}}(\theta, \phi, \alpha, \beta),
\end{aligned} \tag{4.9}$$

where the equality holds, *i.e.*, $\mathcal{L}_{\mathbf{X}}(\theta, \phi, \alpha, \beta)$ is maximized, if and only if $q_{\alpha,\beta}(\mathbf{S}, \mathbf{Z}|\mathbf{X}) = p_{\theta,\phi}(\mathbf{S}, \mathbf{Z}|\mathbf{X})$. Note that this condition cannot be satisfied because $p_{\theta,\phi}(\mathbf{S}, \mathbf{Z}|\mathbf{X})$ is hard to compute. Because the gap between $\log p_{\theta,\phi}(\mathbf{X})$ and $\mathcal{L}_{\mathbf{X}}(\theta, \phi, \alpha, \beta)$ in (4.9) is equal to the KL divergence from $q_{\alpha,\beta}(\mathbf{S}, \mathbf{Z}|\mathbf{X})$ to $p_{\theta,\phi}(\mathbf{S}, \mathbf{Z}|\mathbf{X})$, the minimization of the KL divergence is equivalent to the maximization of $\mathcal{L}_{\mathbf{X}}(\theta, \phi, \alpha, \beta)$.

To approximately compute $\mathcal{L}_{\mathbf{X}}(\theta, \phi, \alpha, \beta)$, we use Monte Carlo integration as follows:

$$\begin{aligned}
&\mathcal{L}_{\mathbf{X}}(\theta, \phi, \alpha, \beta) \\
&= \mathbb{E}_{q_\alpha(\mathbf{S}|\mathbf{X}) q_\beta(\mathbf{Z}|\mathbf{X})}[\log p_\theta(\mathbf{X}|\mathbf{S}, \mathbf{Z})] \\
&\quad + \mathbb{E}_{q_\alpha(\mathbf{S}|\mathbf{X}) q_\beta(\mathbf{Z}|\mathbf{X})}[\log p(\mathbf{Z}) - \log q_\beta(\mathbf{Z}|\mathbf{X})] \\
&\quad + \mathbb{E}_{q_\alpha(\mathbf{S}|\mathbf{X})}[\log p_\phi(\mathbf{S}) - \log q_\alpha(\mathbf{S}|\mathbf{X})] \\
&\approx \frac{1}{I} \sum_{i=1}^{I} \log p_\theta(\mathbf{X}|\mathbf{S}_i, \mathbf{Z}_i) - \mathrm{KL}(q_\beta(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})) \\
&\quad + \mathrm{Entropy}[q_\alpha(\mathbf{S}|\mathbf{X})] + \mathbb{E}_{q_\alpha(\mathbf{S}|\mathbf{X})}[\log p_\phi(\mathbf{S})],
\end{aligned} \tag{4.10}$$

where $\{\mathbf{S}_i, \mathbf{Z}_i\}_{i=1}^{I}$ are I samples drawn from $q_\alpha(\mathbf{S}|\mathbf{X}) q_\beta(\mathbf{Z}|\mathbf{X})$. Following the typical VAE implementation, we set $I = 1$ and hence the index i can be omitted. To make (4.10) partially differentiable with respect to the model parameters θ, ϕ, α, and β, we use reparametrization tricks for deterministically representing the categorical variables $\mathbf{S}$ [64] and the Gaussian variables $\mathbf{Z}$ [10] as follows:

$$\boldsymbol{\epsilon}_n^{\mathbf{s}} \sim \mathrm{Gumbel}(\mathbf{0}_K, \mathbf{1}_K), \tag{4.11}$$

$$\mathbf{s}_n = \mathrm{softmax}((\log[\boldsymbol{\pi}_\alpha(\mathbf{X})]_n + \boldsymbol{\epsilon}_n^{\mathbf{s}})/\tau), \tag{4.12}$$

$$\boldsymbol{\epsilon}_n^{\mathbf{z}} \sim \mathcal{N}(\mathbf{0}_L, \mathbf{I}_L), \tag{4.13}$$

$$\mathbf{z}_n = [\boldsymbol{\mu}_\beta(\mathbf{X})]_n + \boldsymbol{\epsilon}_n^{\mathbf{z}} \odot [\boldsymbol{\sigma}_\beta(\mathbf{X})]_n, \tag{4.14}$$

where (4.11) indicates the standard Gumbel distribution, $\mathbf{1}_K$ is the all-one vector of size K, $\odot$ means the element-wise product, and $\tau > 0$ is a temperature parameter that controls the uniformity of $\mathbf{s}_n$ ($\tau = 0.1$ in this paper).

We can now compute the four terms of (4.10) evaluating the fitness of $\mathbf{S}$ and $\mathbf{Z}$: a reconstruction term indicating the likelihood of $\mathbf{S}$ and $\mathbf{Z}$ for $\mathbf{X}$ and three regularization terms making the posterior $q_\beta(\mathbf{Z}|\mathbf{X})$ close to the prior $p(\mathbf{Z})$, increasing the entropy of $q_\alpha(\mathbf{S}|\mathbf{X})$, and making $\mathbf{S}$ temporally coherent, respectively (Fig. 4.2). More specifically, the first term is given by (4.2) and the second and third terms are given by

$$\begin{aligned}&\mathrm{KL}(q_\beta(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z}))\\ &=\sum_{n=1}^{N}\sum_{k=1}^{K}\left(\frac{[\boldsymbol{\sigma}_\beta(\mathbf{X})]_{nk}^2+[\boldsymbol{\mu}_\beta(\mathbf{X})]_{nk}^2-1}{2}-\log([\boldsymbol{\sigma}_\beta(\mathbf{X})]_{nk})\right),\end{aligned} \tag{4.15}$$

$$\begin{aligned}&\mathrm{Entropy}[q_\alpha(\mathbf{S}|\mathbf{X})]\\ &=-\sum_{n=1}^{N}\sum_{k=1}^{K}[\pi_\alpha(\mathbf{X})]_{nk}\log([\pi_\alpha(\mathbf{X})]_{nk}).\end{aligned} \tag{4.16}$$

The last term $\mathbb{E}_{q_\alpha(\mathbf{S}|\mathbf{X})}[\log p_\theta(\mathbf{S})]$ can be calculated analytically. If $p_\theta(\mathbf{S})$ is the uniform distribution given by (4.3), we have

$$\mathbb{E}_{q_\alpha(\mathbf{S}|\mathbf{X})}[\log p_\phi(\mathbf{S})] = -\sum_{n=1}^{N}\sum_{k=1}^{K}[\boldsymbol{\pi}_\alpha(\mathbf{X})]_{nk}\log K. \tag{4.17}$$

If $p_\theta(\mathbf{S})$ is the Markov model given by (4.4), we use a dynamic programming technique similar to the forward algorithm of the HMM. Let $\gamma(\mathbf{s}_n) = \mathbb{E}_{q_\alpha(\mathbf{S}|\mathbf{X})}[\log p(\mathbf{s}_{1:n})]$ be a forward message at frame n, which can be calculated recursively as follows:

$$\gamma(\mathbf{s}_1) = \log p_\phi(\mathbf{s}_1), \tag{4.18}$$

$$\gamma(\mathbf{s}_n) = \sum_{\mathbf{s}_{n-1}} q_\alpha(\mathbf{s}_{n-1}|\mathbf{X})\big(\gamma(\mathbf{s}_{n-1}) + \log p_\phi(\mathbf{s}_n|\mathbf{s}_{n-1})\big), \tag{4.19}$$

$$\mathbb{E}_{q_\alpha(\mathbf{S}|\mathbf{X})}[\log p_\phi(\mathbf{S})] = \sum_{\mathbf{s}_N} q_\alpha(\mathbf{s}_N|\mathbf{X})\gamma(\mathbf{s}_N). \tag{4.20}$$

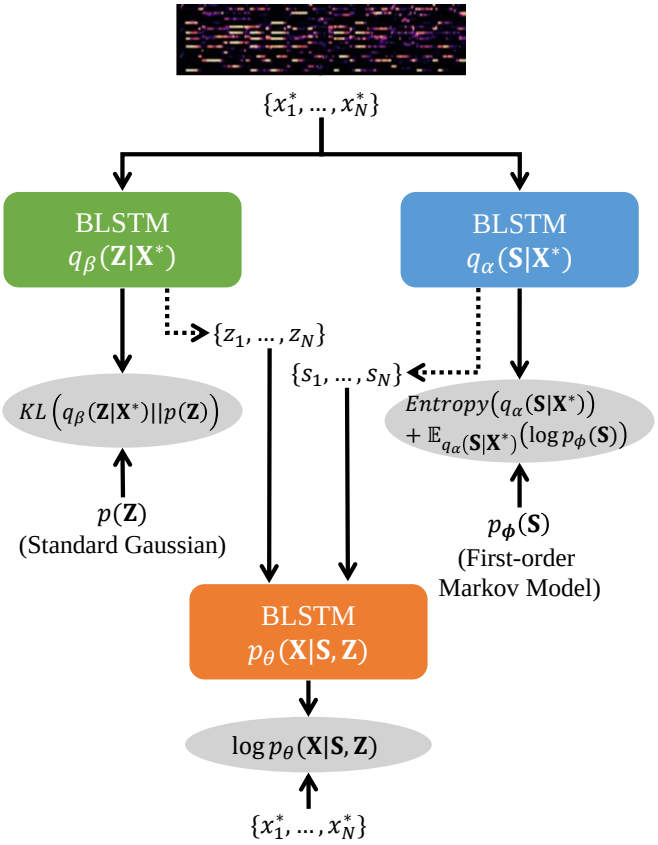


Figure 4.2: The computation flow of the unsupervised learning. The gray areas correspond to the four terms in the target function $\mathcal{L}_\mathbf{X}$ given by (4.10)

We now have a VAE (Fig. 4.2) that consists of:

- The deep classification model $q_\alpha(\mathbf{S}|\mathbf{X})$ given by (4.7) with the reparametrization trick given by (4.11) and (4.12),
- The deep recognition model $q_\beta(\mathbf{Z}|\mathbf{X})$ given by (4.8) with the reparametrization trick given by (4.13) and (4.14), and
- The deep generative model $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$ given by (4.2) with the prior distributions $p_\phi(\mathbf{S})$ and $p(\mathbf{Z})$ given by (4.4) and (4.5), respectively.

In the unsupervised condition, all models are jointly optimized in the framework of the VAE by using a variant of stochastic gradient descent such that (4.10) is maximized with respect to θ, α, and β.

4.2.5 Supervised Learning with Annotated Data

Under the supervised condition that chroma vectors $\mathbf{X}$ and the corresponding chord labels $\mathbf{S}$ are given as observed data, we aim to maximize the variational lower bound $\mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta,\beta)$ of the log-likelihood $\log p_\theta(\mathbf{X}|\mathbf{S})$, which can be derived in a way similar to (4.10) as follows:

$$\begin{aligned}\log p_\theta(\mathbf{X}|\mathbf{S}) &= \log \int \frac{q_\beta(\mathbf{Z}|\mathbf{X})}{q_\beta(\mathbf{Z}|\mathbf{X})} p_\theta(\mathbf{X},\mathbf{Z}|\mathbf{S}) d\mathbf{Z} \\ &\geq \int q_\beta(\mathbf{Z}|\mathbf{X}) \log \frac{p_\theta(\mathbf{X}|\mathbf{Z},\mathbf{S})p(\mathbf{Z})}{q_\beta(\mathbf{Z}|\mathbf{X})} d\mathbf{Z} \\ &\stackrel{\text{def}}{=} \mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta,\beta). \end{aligned} \tag{4.21}$$

Using Monte Carlo integration, $\mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta,\beta)$ can be approximately computed as follows:

$$\begin{aligned}&\mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta,\beta) \\ &\stackrel{\text{def}}{=} \mathbb{E}_{q_\beta(\mathbf{Z}|\mathbf{X})}(\log p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z}) + \log p(\mathbf{Z}) - \log q_\beta(\mathbf{Z}|\mathbf{X})) \\ &\approx \frac{1}{I}\sum_{i=1}^{I} \log p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z}_i) + \mathrm{KL}(q_\beta(\mathbf{Z}|\mathbf{X})\|p(\mathbf{Z})), \end{aligned} \tag{4.22}$$

where $\{\mathbf{Z}_i\}_{i=1}^I$ are I samples drawn from $q_\beta(\mathbf{Z}|\mathbf{X},\mathbf{S})$ using the reparametrization trick ($I=1$ in this paper). Because the chord estimator $q_\alpha(\mathbf{S}|\mathbf{X})$, which plays a central role in ACE, does not appear in (4.22), $q_\alpha(\mathbf{S}|\mathbf{X})$ cannot be trained only by maximizing (4.22). As suggested in the semi-supervised learning of a VAE [37], one could thus define an alternative objective function $\mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta,\alpha,\beta)$ including α by adding a classification performance term to (4.22) as follows:

$$\mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta,\alpha,\beta) \stackrel{\text{def}}{=} \mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta,\beta) + \log q_\alpha(\mathbf{S}|\mathbf{X}). \tag{4.23}$$

In this approach, the regularization terms (4.16) and (4.20) enhancing the entropy of $q_\alpha(\mathbf{S}|\mathbf{X})$ (preventing the overfitting) and smoothing the output of $q_\alpha(\mathbf{S}|\mathbf{X})$, respectively, are not taken into account. Therefore, we propose a new objective function $\mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta,\phi,\alpha,\beta)$ by summing (4.10) and (4.23) as follows:

$$\mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta,\phi,\alpha,\beta) = \mathcal{L}_{\mathbf{X}}(\theta,\phi,\alpha,\beta) + \mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta,\alpha,\beta). \tag{4.24}$$

In this function, the chroma vectors $\mathbf{X}$ with the annotations $\mathbf{S}$ are used twice as unsupervised and supervised training data as if they were *not* annotated (first term) and as they are (second term), respectively.

4.2.6 Semi-supervised Learning

Under the semi-supervised condition that partially-annotated chroma vectors are available, we define an objective function by summing the objective functions (4.10) and (4.24) corresponding to the unsupervised and supervised conditions, respectively, as follows:

$$\begin{aligned}
&\mathcal{L}'_{\mathbf{X},\mathbf{S}}(\theta, \phi, \alpha, \beta) \\
&\stackrel{\text{def}}{=} \sum_{\mathbf{X}\ \text{w/o}\ \mathbf{S}} \mathcal{L}_{\mathbf{X}}(\theta, \phi, \alpha, \beta) + \sum_{\mathbf{X}\ \text{with}\ \mathbf{S}} \mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta, \phi, \alpha, \beta) \\
&= \sum_{\mathbf{X}} \mathcal{L}_{\mathbf{X}}(\theta, \phi, \alpha, \beta) + \sum_{\mathbf{X}\ \text{with}\ \mathbf{S}} \mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta, \alpha, \beta).
\end{aligned} \tag{4.25}$$

Note that $q_\alpha(\mathbf{S}|\mathbf{X})$ can always be regularized by $p_\theta(\mathbf{X}|\mathbf{S}, \mathbf{Z})$ and $p_\phi(\mathbf{S})$ regardless of the availability of annotations.

4.2.7 Training and Prediction

The model parameters θ, ϕ, α, and β can be optimized with a stochastic gradient descent method (*e.g.*, Adam [65]) such that the objective function (4.10), (4.24), or (4.25) is maximized. Nonetheless, in this paper, ϕ is fixed (not optimized) because of its wide acceptable range (see Section 4.3.2). Under the semi-supervised condition, each mini-batch consists of non-annotated and annotated chroma vectors (50%–50% in this paper) randomly selected from the training dataset.

In the test phase, using the neural chord estimator $q_\alpha(\mathbf{S}|\mathbf{X})$, a sequence of the posterior probabilities of chord labels $\mathbf{S}$ are calculated from a sequence of chroma vectors $\mathbf{X}$ extracted from a target music signal. The optimal temporally coherent path of chord labels is then estimated via the Viterbi algorithm with the transition probabilities $\boldsymbol{\phi}$.

4.3 Evaluation

This section reports comparative experiments conducted for evaluating the effectiveness of the proposed method. Specifically, we investigate the effectiveness of the VAE-based regularized training, that of using the Markov prior on chord labels, and, that of using external non-annotated data. The experiments are implemented using PyTorch, and the source code is made public at GitHub [1].

4.3.1 Experimental Conditions

We here explain the compared ACE methods, network configurations, datasets, and evaluation procedures and measures.

Compared Methods

As listed in Table 4.1, we trained a chord estimator $q_\alpha(\mathbf{S}|\mathbf{X})$ in five different ways:

- **ACE-SL (baseline)**: As in most ACE methods, $q_\alpha(\mathbf{S}|\mathbf{X})$ was trained in a *supervised* manner by minimizing the cross-entropy loss for the ground-truth labels $\mathbf{S}$, *i.e.*, maximizing the following objective function:

$$\mathcal{L}_{\mathbf{X},\mathbf{S}}(\alpha) = \log q_\alpha(\mathbf{S}|\mathbf{X}). \tag{4.26}$$

- **VAE-UN-SL**: $q_\alpha(\mathbf{S}|\mathbf{X})$ was trained in a *supervised* manner by maximizing (4.24) with the uniform prior (4.3).
- **VAE-MR-SL**: $q_\alpha(\mathbf{S}|\mathbf{X})$ was trained in a *supervised* manner by maximizing (4.24) with the Markov prior (4.4).
- **VAE-UN-SSL**: $q_\alpha(\mathbf{S}|\mathbf{X})$ was trained in a *semi-supervised* manner by maximizing (4.25) with the uniform prior (4.3).
- **VAE-MR-SSL**: $q_\alpha(\mathbf{S}|\mathbf{X})$ was trained in a *semi-supervised* manner by maximizing (4.25) with the Markov prior (4.4).

For convenience, let '*' denote the wild-card character, *e.g.*, **VAE-*-SL** means **VAE-MR-SL** or **VAE-UN-SL**.

[1] https://github.com/Xiao-Ming/VAEChordEstimation

Table 4.1: Experimental conditions

	Training	Regularization	Prior
ACE-SL (baseline)	Supervised	NA	NA
VAE-UN-SL	Supervised	VAE-based	Uniform
VAE-MR-SL	Supervised	VAE-based	Markov
VAE-UN-SSL	Semi-supervised	VAE-based	Uniform
VAE-MR-SSL	Semi-supervised	VAE-based	Markov

Table 4.2: Durations of chord types in datasets used for evaluation

Chord type	Duration [h]
maj	53.09
min	16.63
aug	0.15
dim	0.36
sus4	1.63
sus2	0.25
1	0.76
5	0.84

Comparing **ACE-SL** with **VAE-*-SL**, we evaluated the effectiveness of the VAE architecture in regularizing $q_\alpha(\mathbf{S}|\mathbf{X})$. Comparing **VAE-*-SSL** with **VAE-*-SL**, we evaluated the effectiveness of the semi-supervised learning. Comparing **VAE-MR-*** with **VAE-UN-***, we evaluated the effectiveness of the Markov prior on $\mathbf{S}$.

Network Configurations

Each of the classification model $q_\alpha(\mathbf{S}|\mathbf{X})$, the recognition model $q_\beta(\mathbf{Z}|\mathbf{X})$, and the generative model $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$ was implemented with a three-layered BLSTM network [66] followed by layer normalization [67]. The final layer of $q_\alpha(\mathbf{S}|\mathbf{X})$ consisted of softmax functions that output the frame-level posterior probabilities of K chord labels. The final layer of $q_\beta(\mathbf{Z}|\mathbf{X})$ consisted of linear units that output $\boldsymbol{\mu}_\beta(\mathbf{X})$ and $\log \boldsymbol{\sigma}^2_\beta(\mathbf{X})$. The final layer of $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$ consisted of sigmoid functions that output $\boldsymbol{\omega}_\theta(\mathbf{S},\mathbf{Z})$. Each hidden layer of the BLSTM had 256 units. Because the architecture of $q_\alpha(\mathbf{S}|\mathbf{X})$ was similar to that of a state-of-the-art chord estimator [30], ACE-SL was considered a reasonable baseline method.

We used Adam optimizer [65], where the learning rate was first set to 0.001

and then decreased exponentially by a scaling factor of 0.99 per epoch. Gradient clipping with norm 5 was additionally applied to the optimization process. Each minibatch consisted of 16 sequences, each of which contained 645 frames (1 min). The parameter of the Markov prior ϕ was fixed as stated in 4.2.2, and θ, α and β were iteratively updated. In all methods and configurations, the number of training epochs was set to 300, which was sufficiently large to reach convergence (early stopping was not used).

Datasets

We collected 1210 *annotated* popular songs consisting of 198 songs from Isophonics [45], 100 songs from RWC-MDB-P-2001 [49], 186 songs from uspop2002 [51][2], and 726 songs from McGill Billboard dataset [52]. As shown in Table 4.2, the six types of triad chords and the two types of power chords are heavily imbalanced in the chord annotations, where most of the annotated chord labels belong to the *major* and *minor* triads. We also collected 700 *non-annotated* popular songs composed by Japanese and American artists. To compensate for the imbalance of the ratios of the 12 chord roots in the training data, chroma vectors and chord labels were jointly pitch-rotated by a random number of semitones in each training iteration. Pitch shifting of the multi-channel chroma vectors can be done by performing vector rotation on each channel, and pitch shifting of chord annotations can be done by changing the root notes.

Evaluation Procedures

To conduct five-fold cross validation, we divided the 1210 annotated songs into five subsets (242 songs each). In each fold, one of the subsets was kept as test data and the remaining four subsets were used as training data, in which I and $4-I$ subsets were treated as annotated and non-annotated songs, respectively ($I \in \{1,2,3,4\}$). Not only the classification model $q_\alpha(\mathbf{S}|\mathbf{X})$ but also the recognition model $q_\beta(\mathbf{Z}|\mathbf{X})$ and the generative model $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$ were trained jointly by

[2]The annotations for RWC-MDB-P-2001 and uspop2002 are provided by the Music and Audio Research Lab at NYU.

using all the training data (**VAE-*-SSL**) or only the annotated data (**VAE-*-SL**). In **ACE-SL**, on the other hand, only the classification model $q_\alpha(\mathbf{S}|\mathbf{X})$ was trained by using the annotated data.

VAE-*-SSL was additionally tested under *extended* semi-supervised conditions that annotated 976 songs (four subsets) and M non-annotated songs ($M \in \{250, 500, 700\}$) were used as training data in each fold. Note that the performance was measured on the remaining 242 annotated songs, which did not overlap with the 700 non-annotated songs.

Evaluation Measures

The chord estimation performance (accuracy) of each method was measured in terms of the frame-level match rate between the estimated and ground-truth chord sequences. The weighted accuracy for each song was measured with *mir_eval* library [44] in terms of the *majmin* criterion considering only the *major* and *minor* triads plus the *no-chord* label ($K = 25$) and the *triads* criterion with the vocabulary defined in Section 4.2.1 ($K = 97$). The overall accuracy was given as the average of the song-wise accuracies weighed by the song lengths.

4.3.2 Experimental Results

The overall accuracies of **ACE-SL** and **VAE-*-*** with respect to the numbers of annotated and non-annotated songs (denoted as A+B) used for training are shown in Fig. 4.3. To investigate the temporal continuity of $\mathbf{S}$ induced by the Markov prior, we tested **VAE-MR-SSL** with a self-transition probability $\phi_{kk} \in \{1/K, 0.3, 0.5, 0.7, 0.9\}$. The song-wise accuracies and average chord durations at 976+0 and 976+700 are compared in Fig. 4.4. Examples of estimated chord sequences estimated by **ACE-SL**, **VAE-UN-SSL**, and **VAE-MR-SSL** with $\phi_{kk} = 0.9$ are shown in Fig. 4.5.

Evaluation of VAE-Based Regularized Training

We confirmed the effectiveness of the VAE-based regularized training of $q_\alpha(\mathbf{S}|\mathbf{X})$ under all conditions. As shown in Fig. 4.3 and Fig. 4.4(a), **VAE-*-SL** clearly

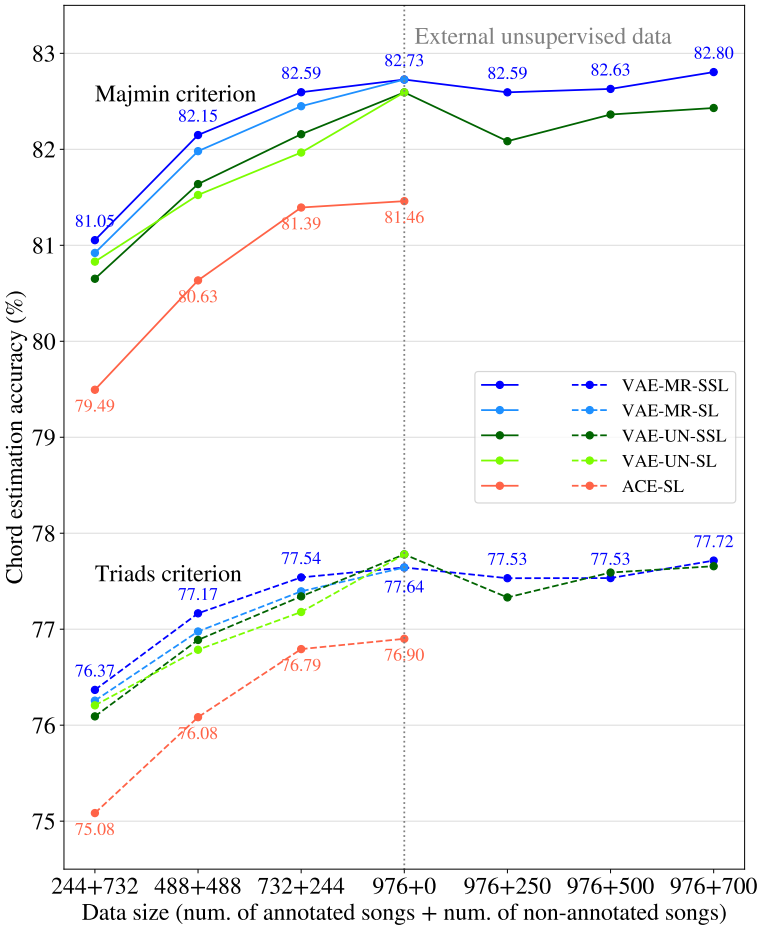


Figure 4.3: The experimental results of the five-fold cross validation using the 1210 annotated songs and the 700 external non-annotated songs.

outperformed **ACE-SL** by a large margin around 1 pts. $q_\alpha(\mathbf{S}|\mathbf{X})$ was regularized effectively by considering its entropy, the Markov or uniform prior of $\mathbf{S}$, and the reconstruction of $\mathbf{X}$ from $\mathbf{S}$ and $\mathbf{Z}$.

Evaluation of Semi-supervised Learning

We confirmed the effectiveness of the semi-supervised learning of $q_\alpha(\mathbf{S}|\mathbf{X})$. Under the semi-supervised conditions at $244{+}732$, $488{+}488$, and $732{+}244$ in the left half of Fig. 4.3, **VAE-MR-SSL** and **VAE-UN-SSL** outperformed **VAE-MR-SL** and **VAE-UN-SL**, respectively, where the proposed **VAE-MR-SSL** worked best. The success of the semi-supervised learning was attributed to the fact that the musical and acoustic characteristics of the 976 songs used for training were consistent

with those of the 244 songs used for testing in the five-fold cross-validation with the 1210 songs. Under the supervised condition at $976{+}0$, **VAE-MR-SSL** and **VAE-UN-SSL** were equivalent to **VAE-MR-SL** and **VAE-UN-SL**, respectively.

Under the *extended* semi-supervised conditions at $976{+}250$, $976{+}500$, and $976{+}700$ in the right half of Fig. 4.3, the performance dropped once and then barely recovered as the number of non-annotated songs used for training increased. The large difference in the musical and acoustic characteristics of the annotated and non-annotated songs is considered to have hindered the semi-supervised learning. More specifically, because the latent feature estimated by the classification model is not *disentangled* from the chord labels, the regularized training by the unsupervised training has little effect on improving the chord classification model.

Evaluation of Markov Prior

We confirmed the effectiveness of the Markov prior on $\mathbf{S}$ under all the conditions. Under the semi-supervised conditions at $244{+}732$, $488{+}488$, and $732{+}244$ and the supervised condition at $976{+}0$ in the left half of Fig. 4.3, **VAE-MR-SL** and **VAE-MR-SSL** outperformed **VAE-UN-SL** and **VAE-UN-SSL**, respectively. The Markov prior played a vital role under the *extended* semi-supervised conditions at $976{+}250$, $976{+}500$, and $976{+}700$ in the right half of Fig. 4.3. While **VAE-UN-SSL** at $976{+}250$ significantly underperformed **VAE-UN-SSL** at $976{+}0$, **VAE-MR-SSL** did not experience large performance drop and achieved slightly better performance at $976{+}700$ by encouraging $q_\alpha(\mathbf{S}|\mathbf{X})$ to yield a temporally-smooth estimate of $\mathbf{S}$.

Fig. 4.4(b) shows that the Markov prior mitigated the negative effect of the semi-supervised learning. **VAE-UN-SSL** (**VAE-MR-SSL** with $\phi_{kk} = 1/K$) performed better, but yielded shorter duration chord segments than **ACE-SL** because $q_\alpha(\mathbf{S}|\mathbf{X})$ was trained under the condition that chord labels were allowed to change frequently for non-annotated songs such that the unsupervised learning objective (4.10) was maximized. By contrast, **VAE-MR-SSL** with a higher self-transition probability yielded longer chord segments and the chord dura-

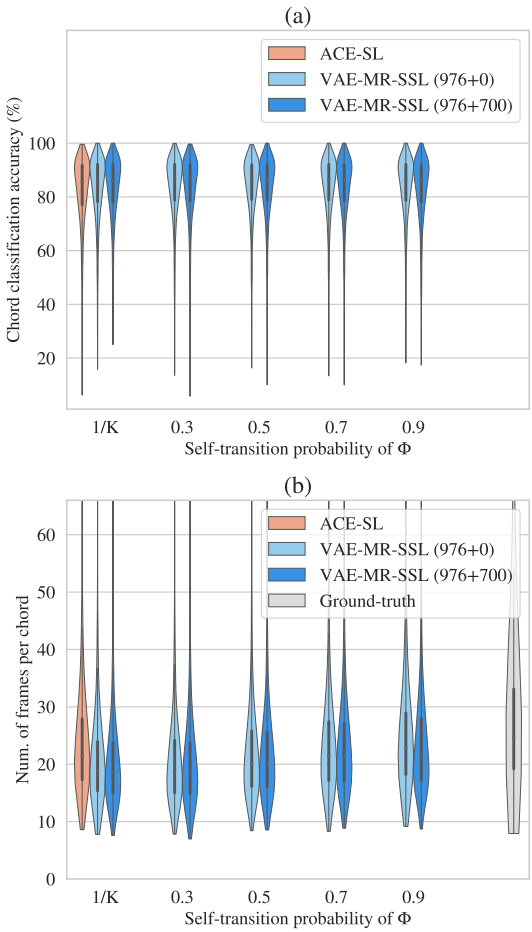


Figure 4.4: The song-wise accuracies (in *majmin* criterion) and average durations of chord labels estimated by **ACE-SL** and **VAE-MR-SSL** with different self-transition probabilities ($\phi_{kk} \in \{1/K, 0.3, 0.5, 0.7, 0.9\}$), where **VAE-MR-SSL** with $\phi_{kk} = 1/K$ is equivalent to **VAE-UN-SSL**. The chord durations were measured before the Viterbi post-filtering.

tions were distributed in a way similar to the ground-truth data. As shown in Fig. 4.4(a), the choice of the self-transition probability between 0.3 and 0.9 had a small impact on the overall chord estimation accuracy.

In Fig. 4.5, some errors made by **ACE-SL** (light-green segments) were corrected by the VAE-based method. The chord label sequence obtained by **VAE-UN-SSL**, however, included a number of incorrect short fragments caused by the local fluctuations of the chroma vectors. As discussed above, $q_\alpha(\mathbf{S}|\mathbf{X})$ was encouraged to frequently vary over time such that the fine structure of chroma

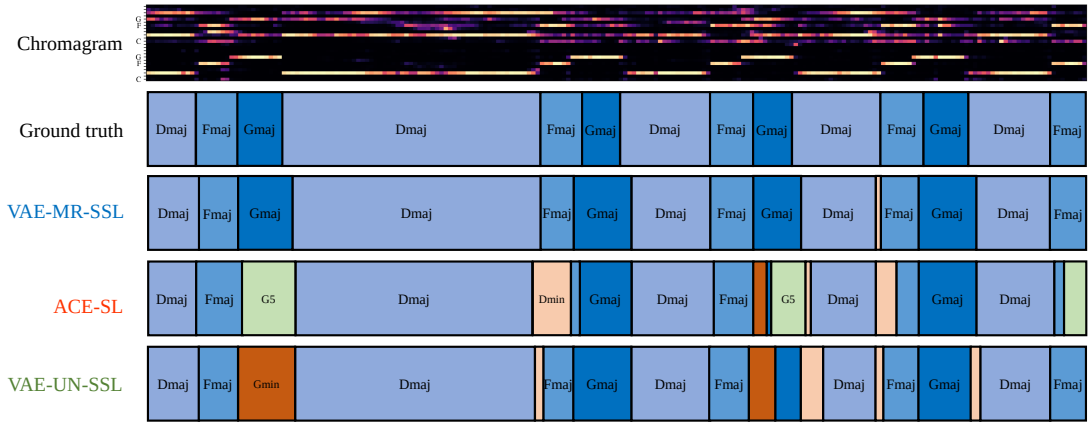


Figure 4.5: An example of chord label sequences estimated by the supervised and semi-supervised methods without the Viterbi post-filtering. For readability, only the first 24 dimensions (bass and middle channels) of the chroma vectors are displayed.

vectors could be reconstructed precisely in the VAE framework, while the chord label sequence obtained by **VAE-MR-SSL** included fewer transitions and was much closer to the ground-truth sequence.

4.3.3 Further Observations

Fig. 4.6 shows the confusion matrices obtained by **ACE-SL** and **VAE-MR-SSL**. While the accuracies on the *maj*, *min*, *aug* and *dim* types were improved by **VAE-MR-SSL**, the accuracies on other uncommon chord types were significantly degraded. Rare chords tended to be wrongly classified to the *maj* and *min* triads. Interestingly, the accuracies on the eight chord types were not necessarily correlated to their ratios in the training data (Table 4.2), where the *sus4* triads were much more frequently used than the *aug* and *dim* triads. The same problem occurred in **VAE-UN-SSL** because the unsupervised learning objective (4.10) has no mechanism to prevent $q_\alpha(\mathbf{S}|\mathbf{X})$ from excessively yielding popular chord types.

As shown in Fig. 4.7, the generative model $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$ successfully reconstructed chroma vectors, conditioned by the *maj*, *min*, *aug*, and *dim* triads, *i.e.*, the reconstructed chroma vectors had high probability on the pitch classes of

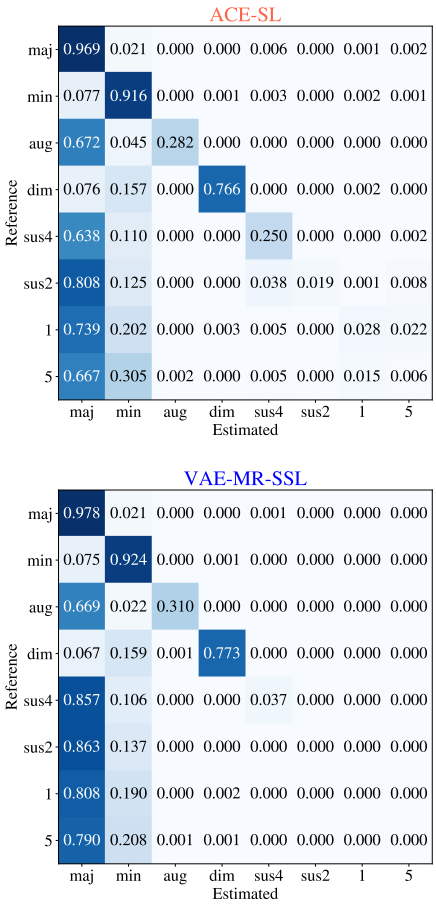


Figure 4.6: Confusion matrices with respect to chord types. Estimated chords with wrong root notes are not taken into account to calculate the correct rates.

the chord notes. By contrast, $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$ failed to reconstruct chroma vectors, when conditioned by *sus2* and *sus4* triads and power chords. Comparing Fig. 4.6 with Fig. 4.7, we investigate the relationships between the estimation accuracy of chord labels and the reconstruction quality of the chroma vectors. The generative model $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$ failed to learn the pitch-class distributions of several chord classes (*e.g.*, *sus2* and *sus4*) whose chroma vectors often had no clear peaks on the chord notes. For such chroma vectors, $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$ always gave a lower probability even if $\mathbf{S}$ was the ground-truth chord classes corresponding to $\mathbf{X}$. Note that $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$ was used for regularizing the classification model $q_\alpha(\mathbf{S}|\mathbf{X})$ in the unsupervised learning objective (4.10), where $q_\alpha(\mathbf{S}|\mathbf{X})$ was trained

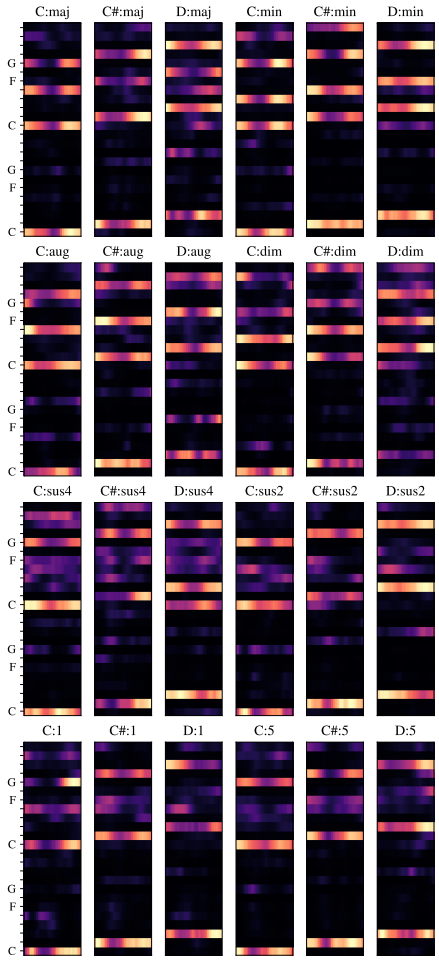


Figure 4.7: The probability distributions obtained by $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$ conditioned by different chord labels $\mathbf{S}$. Only the first 24 dimensions (bass and middle channels) of the chroma vectors are displayed.

to avoid *sus2* and *sus4* triads and favor *maj* triads to maximize $p_\theta(\mathbf{X}|\mathbf{S},\mathbf{Z})$.

4.4 Conclusion

This paper describes a statistical method that trains a neural chord estimator in a semi-supervised manner by constructing a VAE with latent chord labels and features. This is a new approach to ACE that unifies the generative and discriminative methods. Our method incorporates a Markov prior on chord labels to encourage the temporal continuity of chord labels estimated with the chord

estimator. The comparative experiment clearly showed the effectiveness of using the generative model of chroma vectors and the Markov prior on chord labels in regularizing the chord estimator for performance improvement. The limitation of the proposed semi-supervised learning is that the trained chord classification model tends to mistakenly classify some specific chord types into popular types. This may become more problematic when a larger chord vocabulary including seventh chords and chord inversions is used.

The success of the semi-supervised VAE for ACE indicates the effectiveness of unifying the deep generative and discriminative methods in automatic music transcription (AMT). Using the AVI framework, we can explicitly introduce prior knowledge on musical symbol sequences as a regularization term. Such knowledge is hard to automatically extract from training data in supervised discriminative methods. One way to improve the performance of ACE is to replace the frame-level Markov prior of chord labels with a beat- or symbol-level language model, which has been considered to be effective for solving the ambiguity in audio features [26, 33]. We also plan to develop a comprehensive AMT system on the basis of a unified VAE that can treat mutually dependent musical elements such as keys, beats, and notes as latent variables.

Chapter 5

VAE-Based Joint Chord and Key Estimation

This chapter uses the unified deep generative and discriminative approach for joint estimation of chords and keys from music signals.

5.1 Introduction

As described in Chapter 2, the chord sequence is merely one of the symbolic representations of music. The musical key, which specifies a set of musical notes used in the music, is also an important property of music, and thus key estimation from music audio has also been a well-explored research theme. According to western musical theories, different musical concepts that describe the same piece of music are semantically related. For example, the usage of chord labels is highly dependent to the key of the music [5]; The chord and key transitions almost always occur at (down)beat positions [39]. In the context of DNN-based AMT research, joint estimation of multiple kinds of musical elements has scarcely been investigated so far.

The main practical problem of multiple music label estimation lies in the limited amount of music data with complementary annotations [68]. DNNs are scalable to large-scale training data due to their deep, complex structure, and their performance highly depends on the amount of training data. However, producing time-synchronized annotations of musical labels for music audio is

a labor-consuming work, and thus the amount of music data with annotations of multiple musical labels is even fewer. Therefore, a multi-task classification model often underperforms a fully-trained single-task model.

To obtain a high-performance classification model for multiple music labels, we propose to combine the VAE-based learning methodology described in Chapter 4 with a multi-task learning technique [69]. More specifically, we extend the deep *hierarchical* latent variable model to formulate the generative process of chroma vectors (observed variables) from key classes, chord classes, and continuous latent features, where the key and chord labels are assumed to follow a certain language model. In the VAE framework, we introduce a deep classification model that jointly estimates chords and keys from chroma vectors, and another deep recognition model that infers latent features from chroma vectors. All models are then trained jointly where the generative model acts as a regularizer for the classification model.

In this chapter, we examine the effect of the extended VAE-based learning method for improving the accuracy of joint key and chord estimation from music audio. By comparing the key and chord estimation performances with different models, we try to discover the optimal approach for applying the VAE-based method to multiple musical labels.

5.2 Method

We explain the proposed method of joint chord and key estimation. There are various choices in the classification and language models. To formulate the classification model that predicts chord and key labels from chroma vectors, we can use two **separated** single-task models, or a **shared** model that jointly predicts the posteriors of the two labels. Furthermore, we can also consider a **hierarchical** model that predicts the labels from chroma vectors to chords, and chords to keys. To formulate the language model that defines the consistency of the chord and key labels, we use an deep **autoregressive** model implemented with a recurrent neural network, or a Markov language model proposed in

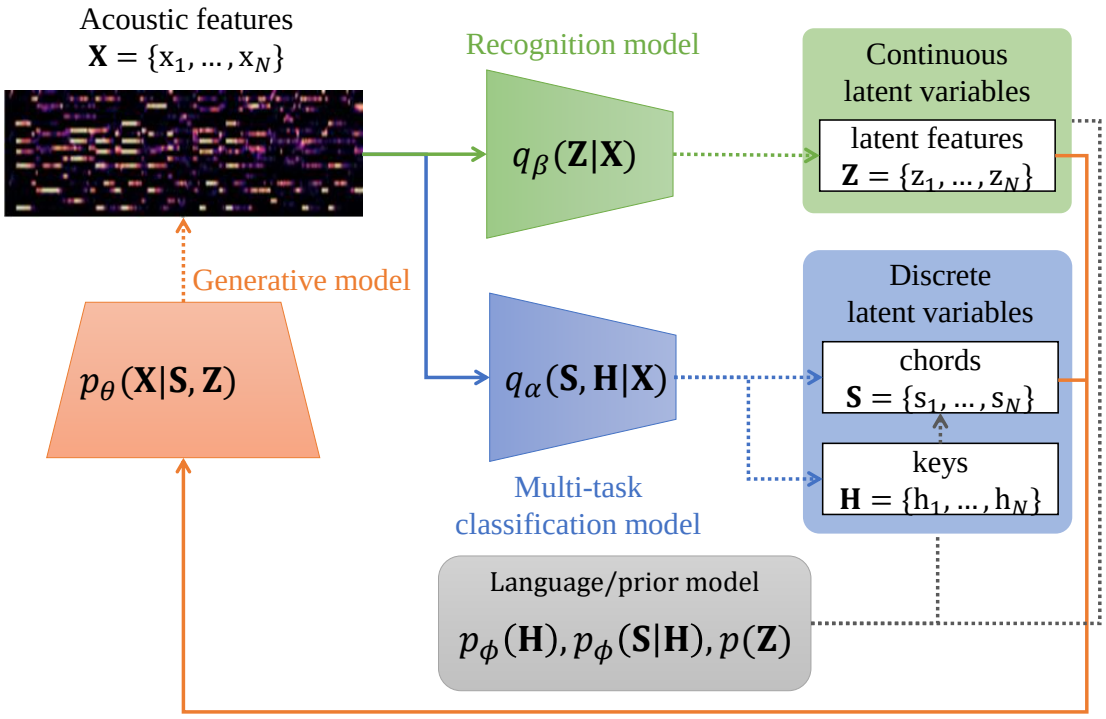


Figure 5.1: The overview of the variational autoencoding framework for joint key and chord estimation, consisting of multi-task classification model, recognition model, language model, and generative model. Solid arrows indicate data input, and dashed arrows indicate stochastic relations.

Chapter 4.

5.2.1 Generative Model

In addition to the variables defined in Chapter 4, we introduce a sequence of key classes $\mathbf{H} = \{\mathbf{h}_1, ..., \mathbf{h}_n\}$ as the latent variable. $\mathbf{h}_n \in \{0, 1\}^{K_H}$ is a discrete variable represented by a one-hot vector of K_H dimensions. The generative model is formulated by decomposing the joint probability $p(\mathbf{X}, \mathbf{S}, \mathbf{H}, \mathbf{Z})$. Here we assume that the observation $\mathbf{X}$ is generated by the following procedure:

1. The musical key of the music is first determined under the prior distribution $p(\mathbf{H})$.
2. Given the musical key, the chord progression is generated under the distribution $p(\mathbf{S}|\mathbf{H})$.

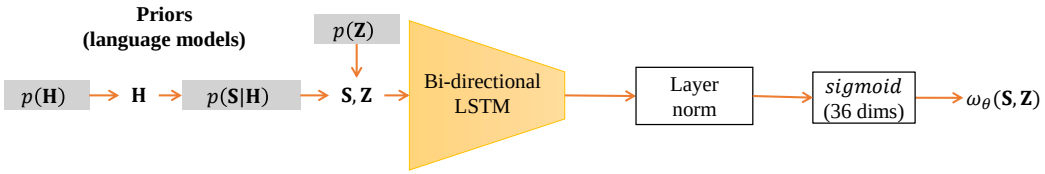


Figure 5.2: Calculation flow of the generative model.

3. The latent feature is generated under the prior distribution $p(\mathbf{Z})$.
4. Given the chord progression sequence and the latent feature, the actual pitch sequence is generated under the distribution $p(\mathbf{X}|\mathbf{S}, \mathbf{Z})$.

To formulate such a process, the joint probability is decomposed as follows:

$$p_\theta(\mathbf{X}, \mathbf{S}, \mathbf{H}, \mathbf{Z}) = p_\theta(\mathbf{X}|\mathbf{S}, \mathbf{Z})p(\mathbf{S}|\mathbf{H})p(\mathbf{H})p(\mathbf{Z}), \tag{5.1}$$

where $p_\theta(\mathbf{X}|\mathbf{S}, \mathbf{Z})$ is a DNN that takes $\mathbf{S}$ and $\mathbf{Z}$ as input and output the distribution of the chroma vectors $\mathbf{X}$:

$$p(\mathbf{X}|\mathbf{S}, \mathbf{H}, \mathbf{Z})_\theta = \prod_{n=1}^{N}\prod_{d=1}^{D} \text{Bernoulli}(x_{nd}|[\omega_\theta(\mathbf{S}, \mathbf{Z})_{nd}]), \tag{5.2}$$

where $\omega_\theta(\mathbf{S}, \mathbf{Z})_{nd}$ represents the output of the DNN. The DNN takes an $N(K_S + K_H + L)$-dimensional vector as input and outputs a $36N$-dimensional vector (Fig. 5.2).

5.2.2 Language Model

We consider three approaches to formulate the prior distributions of the discrete variables $\mathbf{S}$ and $\mathbf{H}$:

Deep Autoregressive Model

Instead of modelling $p(\mathbf{H})$ and $p(\mathbf{S}|\mathbf{H})$ separately, we jointly model $p(\mathbf{H}, \mathbf{S})$ using a deep temporal model. Specifically, we define an autoregressive temporal model of $\mathbf{S}$ and $\mathbf{H}$:

$$p_\phi(\mathbf{S}, \mathbf{X}) = p_\phi(\mathbf{s}_1, \mathbf{h}_1)\prod_{n=2}^{N} p_\phi(\mathbf{s}_n|\mathbf{s}_{1..n-1}, \mathbf{h}_{1..n-1})p_\phi(\mathbf{h}_n|\mathbf{s}_{1..n-1}, \mathbf{h}_{1..n-1}), \tag{5.3}$$

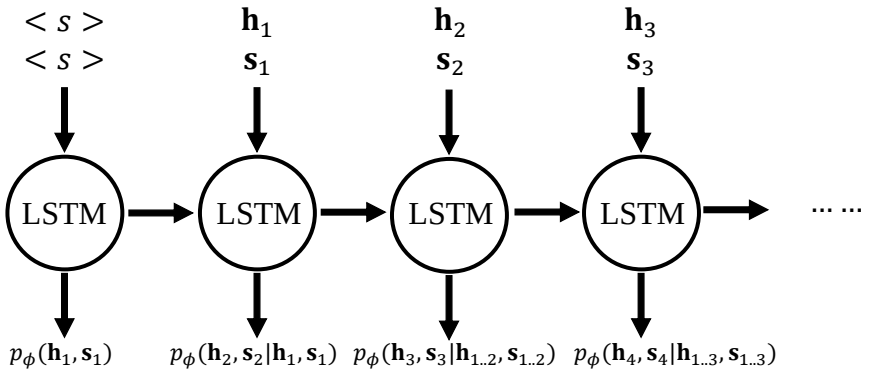


Figure 5.3: Calculation flow of the language model $p_\phi(\mathbf{S}, \mathbf{H})$ implemented as a deep autoregressive model. $< s >$ represents the *start of sentence* label.

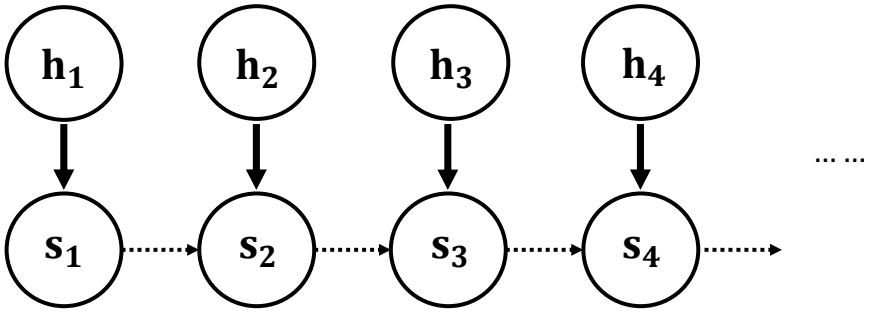


Figure 5.4: Calculation flow of language model implemented as a Markov model. The solid arrows represent the emission probability $p_\phi(\mathbf{s}_n|\mathbf{h}_n)$, and the dashed arrows represent the label transition probability $p_\phi(\mathbf{s}_n|\mathbf{s}_{n-1})$.

$$p_\phi(\mathbf{s}_n|\mathbf{s}_{n-1}, \mathbf{h}_{n-1}) = \mathrm{Categorical}(s_n|[\omega_\phi(\mathbf{s}_{1..n-1}, \mathbf{h}_{1..n-1})]), \tag{5.4}$$

$$p_\phi(\mathbf{h}_n|\mathbf{s}_{n-1}, \mathbf{h}_{n-1}) = \mathrm{Categorical}(h_n|[\omega_\phi(\mathbf{s}_{1..n-1}, \mathbf{h}_{1..n-1})]), \tag{5.5}$$

where $\omega_\phi(\mathbf{S}, \mathbf{H})$ is the DNN-based temporal model that takes a pair of chord and key labels on each time step, and outputs the distributions of the next time step (Fig.5.3). The model parameter ϕ is estimated through supervised and unsupervised learning, which will be described in the later sections.

Markov Model

Similar to Chapter 4, we can use the Markov model to formulate the language model. Specifically, we assume that the probability of a chord label $\mathbf{s}_n$ depends on the previous chord label $\mathbf{s}_{n-1}$ and the current key label $\mathbf{h}_n$ (Fig. 5.4). The key labels are generated from a uniform prior. Therefore, the priors of $\mathbf{S}$ and $\mathbf{H}$ are defined as follows:

$$p(\mathbf{H}) = \prod_{n=1}^{N} \mathrm{Categorical}\big(\mathbf{h}_n | \tfrac{1}{K_H}\mathbf{1}_{K_H}\big), \tag{5.6}$$

$$\begin{aligned} p_\phi(\mathbf{S}|\mathbf{H}) &= p_\phi(\mathbf{s}_1|\mathbf{h}_1) \prod_{n=2}^{N} p_\phi(\mathbf{s}_n|\mathbf{h}_n, \mathbf{s}_{n-1}) \\ &= \prod_{k=1}^{K_s} \phi_k^{\mathbf{h}_1,\mathbf{s}_1,k} \prod_{n=2}^{N} \prod_{k'=1}^{K_s-1} \prod_{k=2}^{K_s} \phi_{k'k}^{\mathbf{h}_n,\mathbf{s}_{n-1,k},\mathbf{s}_{n,k}}, \end{aligned} \tag{5.7}$$

where ϕ is the parameter of the Markov models representing the state transition probabilities and the label emission probabilities conditioned by the hidden states. As is described later, the parameters ϕ are specified by the statistics of the key and chord annotation dataset.

Uniform Distribution

Another simple approach to define the prior as uniform categorical distributions as follows:

$$p(\mathbf{S}|\mathbf{H}) = \prod_{n=1}^{N} \mathrm{Categorical}\big(\mathbf{s}_n | \tfrac{1}{K_S}\mathbf{1}_{K_S}\big), \tag{5.8}$$

$$p(\mathbf{H}) = \prod_{n=1}^{N} \mathrm{Categorical}\big(\mathbf{h}_n | \tfrac{1}{K_H}\mathbf{1}_{K_H}\big), \tag{5.9}$$

where $\mathbf{1}_K$ is the all-one vector of size L. In this case, the likelihoods of $\mathbf{H}$ and $\mathbf{S}$ are always constant, regardless of the values.

5.2.3 Classification and Recognition Models

Given the chroma vectors $\mathbf{X}$ as observed data, we aim to infer the latent variables $\mathbf{S}$, $\mathbf{H}$ and $\mathbf{Z}$, and estimate the model parameters θ. Similar to Chapter 4, we

use the amortized variational inference technique that introduces a variational distribution $q_{\alpha,\beta}(\mathbf{S}, \mathbf{H}, \mathbf{Z}|\mathbf{X})$ to approximate the true posteriors of the latent variables: $p_\theta(\mathbf{S}, \mathbf{H}, \mathbf{Z}|\mathbf{X})$. Concretely, we consider three implementations for the classification model of chords $\mathbf{S}$ and keys $\mathbf{H}$.

Single-task Model

We assume that the posteriors of musical labels $\mathbf{S}$, $\mathbf{H}$ and the latent features $\mathbf{Z}$ are all inferred from $\mathbf{X}$, and are conditionally independent. Therefore, the variational posterior is decomposed as follows:

$$q_{\alpha,\beta}(\mathbf{S}, \mathbf{H}, \mathbf{Z}|\mathbf{X}) = q_{\alpha_s}(\mathbf{S}|\mathbf{X}) q_{\alpha_h}(\mathbf{H}|\mathbf{X}) q_\beta(\mathbf{Z}|\mathbf{X}). \tag{5.10}$$

The three terms represent three independent classification models for the latent variables. They are implemented with DNNs parametrized by α_s, α_h and β, respectively:

$$q_{\alpha_s}(\mathbf{S}|\mathbf{X}) = \prod_{n=1}^{N} \mathrm{Categorical}(\mathbf{s}_n|[\boldsymbol{\pi}_{\alpha_s}(\mathbf{X})]_n), \tag{5.11}$$

$$q_{\alpha_h}(\mathbf{H}|\mathbf{X}) = \prod_{n=1}^{N} \mathrm{Categorical}(\mathbf{h}_n|[\boldsymbol{\pi}_{\alpha_h}(\mathbf{X})]_n), \tag{5.12}$$

$$q_\beta(\mathbf{Z}|\mathbf{X}) = \prod_{n=1}^{N} \mathcal{N}(\mathbf{z}_n|[\boldsymbol{\mu}_\beta(\mathbf{X})]_n, [\boldsymbol{\sigma}^2_\beta(\mathbf{X})]_n), \tag{5.13}$$

where $\pi_{\alpha_s}(\mathbf{X})$, $\pi_{\alpha_s}(\mathbf{X})$ are the outputs of the DNN with parameters α_s and α_h, $\boldsymbol{\mu}_\beta(\mathbf{X})$ and $\boldsymbol{\sigma}^2_\beta(\mathbf{X})$ are the NL-dimensional outputs of the DNN with parameters β.

Shared-parameter Model

We assume that the musical classes $\mathbf{S}$ and $\mathbf{H}$ are dependent, and can be jointly inferred from $\mathbf{X}$ using a single model with shared parameters. In this case, we decompose the variational posterior as follows:

$$q_{\alpha,\beta}(\mathbf{S}, \mathbf{H}, \mathbf{Z}|\mathbf{X}) = q_\alpha(\mathbf{S}, \mathbf{H}|\mathbf{X}) q_\beta(\mathbf{Z}|\mathbf{X}). \tag{5.14}$$

The first term is implemented with a multi-task DNN parameterized by α:

$$q_\alpha(\mathbf{S}, \mathbf{H}|\mathbf{X}) = \prod_{n=1}^{N} \mathrm{Categorical}(\mathbf{s}_n|[\boldsymbol{\pi}_\alpha(\mathbf{X})_{1..73}]_n) \prod_{n=1}^{N} \mathrm{Categorical}(\mathbf{h}_n|[\boldsymbol{\pi}_\alpha(\mathbf{X})_{74..97}]_n), \tag{5.15}$$

where $\pi_\alpha(\mathbf{X})$ is the $N(K_S + K_H)$-dimensional output of the multi-task DNN with parameters α.

Hierarchical Model

We consider a hierarchical estimation process, where the chords are estimated given the chroma vectors, and the keys are estimated given the chord labels. Under this assumption, the variational posterior is decomposed as follows:

$$q_{\alpha,\beta}(\mathbf{S}, \mathbf{H}, \mathbf{Z}|\mathbf{X}) = q_{\alpha_s}(\mathbf{S}|\mathbf{X}) q_{\alpha_h}(\mathbf{H}|\mathbf{S}) q_\beta(\mathbf{Z}|\mathbf{X}). \tag{5.16}$$

The first two terms are implemented with two DNNs:

$$q_{\alpha_s}(\mathbf{S}|\mathbf{X}) = \prod_{n=1}^{N} \mathrm{Categorical}(\mathbf{s}_n|[\boldsymbol{\pi}_{\alpha_s}(\mathbf{X})]_n), \tag{5.17}$$

$$q_{\alpha_h}(\mathbf{H}|\mathbf{S}) = \prod_{n=1}^{N} \mathrm{Categorical}(\mathbf{h}_n|[\boldsymbol{\pi}_{\alpha_h}(\mathbf{S})]_n). \tag{5.18}$$

The implementations of the three classification models are illustrated in Fig.5.5

5.2.4 Unsupervised Training

Under an unsupervised condition that only chroma vectors $\mathbf{X}$ are given as observed data, we aim to jointly train the deep generative model and the classification model such that the marginal log-likelihood $\log p_\theta(\mathbf{X})$ is maximized. Similar to Chapter 4, this is done by maximizing the variational lower bound $\mathcal{L}_\mathbf{X}(\theta, \alpha, \beta)$. By introducing the variational prior $q_{\alpha,\beta}(\mathbf{Z}, \mathbf{S}, \mathbf{H}|\mathbf{X})$, the lower bound is derived as follows:

$$\log p_\theta(\mathbf{X}) = \log \iiint p_\theta(\mathbf{X}, \mathbf{Z}, \mathbf{S}, \mathbf{H}) d\mathbf{Z} d\mathbf{S} d\mathbf{H}$$

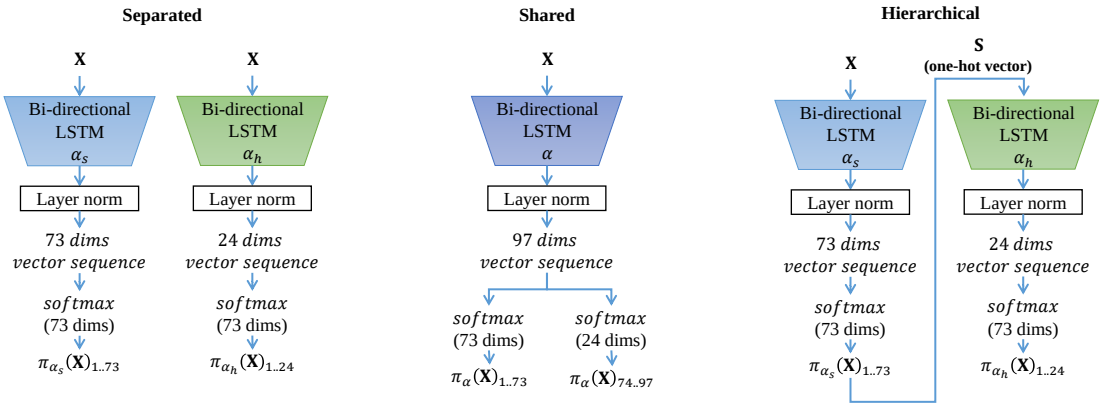


Figure 5.5: Calculation flows of the three types of classification models.

$$
\begin{aligned}
&= \log \iiint \frac{q_{\alpha,\beta}(\mathbf{Z},\mathbf{S},\mathbf{H}|\mathbf{X})}{q_{\alpha,\beta}(\mathbf{Z},\mathbf{S},\mathbf{H}|\mathbf{X})} p_\theta(\mathbf{X},\mathbf{Z},\mathbf{S},\mathbf{H}) d\mathbf{Z} d\mathbf{S} d\mathbf{H} \\
&\geq \iiint q_{\alpha,\beta}(\mathbf{Z},\mathbf{S},\mathbf{H}|\mathbf{X}) \log \frac{p_\theta(\mathbf{X},\mathbf{Z},\mathbf{S},\mathbf{H})}{q_{\alpha,\beta}(\mathbf{Z},\mathbf{S},\mathbf{H}|\mathbf{X})} d\mathbf{Z} d\mathbf{S} d\mathbf{H} \\
&\stackrel{\text{def}}{=} \mathcal{L}_{\mathbf{X}}(\theta,\alpha,\beta),
\end{aligned} \tag{5.19}
$$

where the gap between $\log p_\theta(\mathbf{X})$ and $\mathcal{L}_{\mathbf{X}}(\theta,\alpha,\beta)$ is equal to the KL divergence from $q_{\alpha,\beta}(\mathbf{S},\mathbf{H},\mathbf{Z}|\mathbf{X})$ to $p_\theta(\mathbf{S},\mathbf{H},\mathbf{Z}|\mathbf{X})$, and thus maximizing $\mathcal{L}_{\mathbf{X}}(\theta,\alpha,\beta)$ is equivalent to minimizing the KL divergence [10]. Using the generative model defined in (5.14), the lower bound is approximately computed as follows:

$$
\begin{aligned}
&\mathcal{L}_{\mathbf{X}}(\theta,\alpha,\beta) \\
&= \mathbb{E}_{q(\mathbf{Z}|\mathbf{X})q(\mathbf{S}|\mathbf{X})}[\log p(\mathbf{X}|\mathbf{Z},\mathbf{S})] \\
&\quad + \mathbb{E}_{q(\mathbf{Z}|\mathbf{X})}[\log p(\mathbf{Z}) - \log q(\mathbf{Z}|\mathbf{X})] \\
&\quad + \mathbb{E}_{q(\mathbf{H}|\mathbf{S})q(\mathbf{S}|\mathbf{X})}[\log p(\mathbf{S}|\mathbf{H}) - \log q(\mathbf{S}|\mathbf{X})] \\
&\quad + \mathbb{E}_{q(\mathbf{H}|\mathbf{S})q(\mathbf{S}|\mathbf{X})}[\log p(\mathbf{H}) - \log q(\mathbf{H}|\mathbf{S})] \\
&\approx \frac{1}{I}\sum_{i=1}^{I}[\log p_\theta(\mathbf{X}|\mathbf{Z}_i,\mathbf{S}_i) + \log p(\mathbf{S}_i|\mathbf{H}_i) + \log p(\mathbf{H}_i)] \\
&\quad - KL(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})) \\
&\quad + \text{Entropy}[q(\mathbf{S}|\mathbf{X})] + \text{Entropy}[q(\mathbf{H}|\mathbf{S})],
\end{aligned} \tag{5.20}
$$

where$\{\mathbf{Z}_i,\mathbf{S}_i,\mathbf{H}_i\}_{i=1}^{I}$ are I samples drawn from $q(\mathbf{Z}|\mathbf{X})$, $q(\mathbf{H}|\mathbf{S})$, and $q(\mathbf{S}|\mathbf{X})$. Following the standard VAE implementation, we let $I = 1$ when calculating

the expectation terms. Different from the method in Chapter 4, we do not obtain the discrete variable samples using the gumbel-softmax method, for the sampling method may provide label sequences that significantly differ from the actual estimations by the classification model, especially when the entropy of the predicted label posteriors is high. Instead, we take a simple approach that takes the label sequence with maximum posterior probability as the sampled labels:

$$\mathbf{S}_1 = \mathrm{argmax}_{\mathbf{S}} q_\alpha(\mathbf{S}, \mathbf{H}|\mathbf{X}), \tag{5.21}$$

$$\mathbf{H}_1 = \mathrm{argmax}_{\mathbf{H}} q_\alpha(\mathbf{S}, \mathbf{H}|\mathbf{X}). \tag{5.22}$$

To obtain the one-hot vectors $\mathbf{S}_1$ and $\mathbf{H}_1$ while making the target function differentiable with respect to the model parameters, an implementation trick is used to transform the posterior distributions into one-hot vectors $\mathbf{s}_n$ and $\mathbf{h}_n$:

$$\mathbf{s}_n^{soft} = \mathrm{softmax}(\log[\pi_{\alpha_s}(\mathbf{X})]_n), \tag{5.23}$$

$$\mathbf{s}_n^{max} = \mathrm{onehot}(\log[\pi_{\alpha_s}(\mathbf{X})]_n), \tag{5.24}$$

$$\mathbf{s}_n = \mathbf{s}_n^{soft} + \mathbf{s}_n^{max} - \mathrm{detach}(\mathbf{s}_n^{soft}), \tag{5.25}$$

$$\mathbf{h}_n^{soft} = \mathrm{softmax}(\log[\pi_{\alpha_h}(\mathbf{X})]_n), \tag{5.26}$$

$$\mathbf{h}_n^{max} = \mathrm{onehot}(\log[\pi_{\alpha_h}(\mathbf{X})]_n), \tag{5.27}$$

$$\mathbf{h}_n = \mathbf{h}_n^{soft} + \mathbf{h}_n^{max} - \mathrm{detach}(\mathbf{h}_n^{soft}), \tag{5.28}$$

where $\mathbf{s}_n^{soft}$, $\mathbf{h}_n^{soft}$ are the posterior vectors calculated from the output of π_α, and $\mathbf{s}_n^{max}$,$\mathbf{h}_n^{max}$ are the one-hot vector of the same value with $\mathbf{s}_n$ and $\mathbf{h}_n$. $\mathrm{onehot}(\mathbf{x})$ represents the non-differentiable operation that transforms the vector $\mathbf{x}$ into one-hot vector, and thus $\mathbf{s}_n^{max}$ and $\mathbf{h}_n^{max}$ is detached from the computation graph. $\mathrm{detach}(\mathbf{x})$ represents the operation that detaches the vector $\mathbf{x}$ from the computation graph.

The method of calculating the expectation terms for the language models $p_\phi(H)$ and $p_\phi(\mathbf{S}|\mathbf{H})$ varies by their definitions. When they are jointly defined by the deep regressive model defined by (5.3), the expectation is approximated by substituting the samples $\mathbf{S}_1$ and $\mathbf{H}_1$ into (5.3). When they are defined using

the Markov prior defined by (5.6) and (5.7), the expectation for $p(\mathbf{H})$ and $p(\mathbf{S}|\mathbf{H})$ can be analytically calculated using the dynamic programming technique. The expectation for $p(\mathbf{H})$ is calculated as follows:

$$\gamma(\mathbf{h}_1) = \log p_\phi(\mathbf{h}_1), \tag{5.29}$$

$$\gamma(\mathbf{h}_n) = \sum_{\mathbf{h}_{n-1}} q_\alpha(\mathbf{h}_{n-1}|\mathbf{X})\big(\gamma(\mathbf{h}_{n-1}) + \log p_\phi(\mathbf{h}_n|\mathbf{s}_{n-1})\big), \tag{5.30}$$

$$\mathbb{E}_{q_\alpha(\mathbf{H}|\mathbf{X})}[\log p_\phi(\mathbf{H})] = \sum_{\mathbf{h}_N} q_\alpha(\mathbf{h}_N|\mathbf{X})\gamma(\mathbf{h}_N). \tag{5.31}$$

Similarly, the expectation term for $p(\mathbf{S}|\mathbf{H})$ is calculated as follows:

$$\gamma(\mathbf{s}_1) = \log p_\phi(\mathbf{s}_1), \tag{5.32}$$

$$\gamma(\mathbf{s}_n) = \sum_{\mathbf{s}_{n-1}} \sum_{vh_{n-1}} q_\alpha(\mathbf{h}_{n-1}|\mathbf{X}) q_\alpha(\mathbf{s}_{n-1}|\mathbf{X})\big(\gamma(\mathbf{s}_{n-1}) + \log p_\phi(\mathbf{s}_n|\mathbf{s}_{n-1}, \mathbf{h}_{n-1})\big), \tag{5.33}$$

$$\mathbb{E}_{q_\alpha(\mathbf{S}|\mathbf{X})}[\log p_\phi(\mathbf{S})] = \sum_{\mathbf{s}_N} q_\alpha(\mathbf{s}_N|\mathbf{X})\gamma(\mathbf{s}_N). \tag{5.34}$$

When the $p(\mathbf{S})$ and $p(\mathbf{H})$ are uniform distributions, the expectation terms are irrelevant to the maximization of $\mathcal{L}_\mathbf{X}(\theta, \alpha, \beta)$. The regularization by the posteriors on $\mathbf{S}$ and $\mathbf{H}$ thus corresponds to the maximizion of the entropy of the variational posterior $q_\alpha(\mathbf{S}, \mathbf{H}|\mathbf{X})$.

5.2.5 Supervised Training

The target functions for the supervised conditions depend on whether the latent variables $\mathbf{H}$ and $\mathbf{S}$ are *partly* or *both* available. In the *partly-supervised* condition where $\mathbf{S}$ is available, we aim to maximize the variational lower bound of the log-likelihood $\log p(\mathbf{X}, \mathbf{S})$ which is derived as follows:

$$\begin{aligned}
&\log p(\mathbf{X}, \mathbf{S}) \\
&= \log \int p(\mathbf{X}, \mathbf{Z}, \mathbf{S}, \mathbf{H}) d\mathbf{Z} d\mathbf{H} \\
&\geq \int q(\mathbf{Z}, \mathbf{H}|\mathbf{X}, \mathbf{S}) \log \frac{p(\mathbf{X}, \mathbf{Z}, \mathbf{S}, \mathbf{H})}{q(\mathbf{Z}, \mathbf{H}|\mathbf{X}, \mathbf{S})} d\mathbf{Z} d\mathbf{H} \\
&= \mathbb{E}_{q(\mathbf{Z}|\mathbf{X})}[\log p(\mathbf{X}|\mathbf{Z}, \mathbf{S})] \\
&\quad + \mathbb{E}_{q(\mathbf{Z}|\mathbf{X})}[\log p(\mathbf{Z}) - \log q(\mathbf{Z}|\mathbf{X})]
\end{aligned}$$

$$
\begin{aligned}
&+ \mathbb{E}_{q(\mathbf{H}|\mathbf{S})}[\log p(\mathbf{S}|\mathbf{H})] \\
&+ \mathbb{E}_{q(\mathbf{H}|\mathbf{S})}[\log p(\mathbf{H}) - \log q(\mathbf{H}|\mathbf{S})] \\
&\stackrel{\text{def}}{=} \mathcal{L}_{\mathbf{X},\mathbf{S}} \\
&\approx \frac{1}{I}\sum_{i=1}^{I}[\log p_\theta(\mathbf{X}|\mathbf{Z}_i, \mathbf{S}) + \log p(\mathbf{S}|\mathbf{H}_i) + \log p(\mathbf{H}_i)] \\
&- \mathrm{KL}(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})) + \mathrm{Entropy}(\log q(H|S)).
\end{aligned} \tag{5.35}
$$

Similarly, when $\mathbf{H}$ is available, the target function is the variational lower bound of the log-likelihood $\log p(\mathbf{X}, \mathbf{H})$:

$$
\begin{aligned}
&\log p(\mathbf{X}, \mathbf{H}) \\
&= \log \int p(\mathbf{X}, \mathbf{Z}, \mathbf{S}, \mathbf{H}) d\mathbf{Z} d\mathbf{S} \\
&\geq \int q(\mathbf{Z}, \mathbf{S}|\mathbf{X}, \mathbf{H}) \log \frac{p(\mathbf{X}, \mathbf{Z}, \mathbf{S}, \mathbf{H})}{q(\mathbf{Z}, \mathbf{S}|\mathbf{X}, \mathbf{H})} d\mathbf{Z} d\mathbf{S} \\
&= \mathbb{E}_{q(\mathbf{Z}|\mathbf{X})}[\log p(\mathbf{X}|\mathbf{Z}, \mathbf{S})] \\
&\quad + \mathbb{E}_{q(\mathbf{Z}|\mathbf{X})}[\log p(\mathbf{Z}) - \log q(\mathbf{Z}|\mathbf{X})] \\
&\quad + \mathbb{E}_{q(\mathbf{S}|\mathbf{X})}[\log p(\mathbf{S}|\mathbf{H}) - \log q(\mathbf{S}|\mathbf{X})] \\
&\quad + \log p(\mathbf{H}) \\
&\stackrel{\text{def}}{=} \mathcal{L}_{\mathbf{X},\mathbf{H}} \\
&\approx \frac{1}{I}\sum_{i=1}^{I}[\log p_\theta(\mathbf{X}|\mathbf{Z}_i, \mathbf{S}_i) + \log p(\mathbf{S}_i|\mathbf{H})] \\
&\quad - \mathrm{KL}(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})) + \mathrm{Entropy}(q(\mathbf{S}|\mathbf{X})) \\
&\quad + \log p(\mathbf{H}).
\end{aligned} \tag{5.36, 5.37}
$$

In the *fully-supervised* conditions where $\mathbf{H}$ and $\mathbf{S}$ are both available, the target function becomes the variational lower bound of marginal likelihood $\log p(\mathbf{X}, \mathbf{S}, \mathbf{H})$:

$$
\begin{aligned}
&\log p(\mathbf{X}, \mathbf{S}, \mathbf{H}) \\
&= \log \int p(\mathbf{X}, \mathbf{Z}, \mathbf{S}, \mathbf{H}) d\mathbf{Z} \\
&\geq \int q(\mathbf{Z}|\mathbf{X}, \mathbf{S}, \mathbf{H}) \log \frac{p(\mathbf{X}, \mathbf{Z}, \mathbf{S}, \mathbf{H})}{q(\mathbf{Z}|\mathbf{X}, \mathbf{S}, \mathbf{H})} d\mathbf{Z}
\end{aligned}
$$

$$
\begin{aligned}
&= \mathbb{E}_{q(\mathbf{Z}|\mathbf{X})}[\log p(\mathbf{X}|\mathbf{Z}, \mathbf{S})] \\
&\quad + \mathbb{E}_{q(\mathbf{Z}|\mathbf{X})}[\log p(\mathbf{Z}) - \log q(\mathbf{Z}|\mathbf{X})] \\
&\quad + \log p(\mathbf{S}|\mathbf{H}) + \log p(\mathbf{H}) \\
&\stackrel{\text{def}}{=} \mathcal{L}_{\mathbf{X},\mathbf{S},\mathbf{H}} \\
&\approx \frac{1}{I}\sum_{i=1}^{I} \log p_\theta(\mathbf{X}|\mathbf{Z}_i, \mathbf{S}) \\
&\quad - KL(q(\mathbf{Z}|\mathbf{X})||p(\mathbf{Z})) + \log p(\mathbf{S}|\mathbf{H}) + \log p(\mathbf{H}).
\end{aligned} \tag{5.38}
$$

As mentioned in Chapter 4, the classification model cannot be trained by solely maximizing the lower bounds. Therefore, the label likelihood terms are added to the lower bounds training to form the target functions for the (partly-)supervised training:

$$\mathcal{L}'_{\mathbf{X},\mathbf{S}}(\theta, \alpha, \beta) = \mathcal{L}_{\mathbf{X},\mathbf{S}}(\theta, \alpha, \beta) + \log q_\alpha(\mathbf{S}|\mathbf{X}), \tag{5.39}$$

$$\mathcal{L}'_{\mathbf{X},\mathbf{H}}(\theta, \alpha, \beta) = \mathcal{L}_{\mathbf{X},\mathbf{H}}(\theta, \alpha, \beta) + \log q_\alpha(\mathbf{H}|\mathbf{X}), \tag{5.40}$$

$$\mathcal{L}'_{\mathbf{X},\mathbf{S},\mathbf{H}}(\theta, \alpha, \beta) = \mathcal{L}_{\mathbf{X},\mathbf{S},\mathbf{H}}(\theta, \alpha, \beta) + \log q_\alpha(\mathbf{S}|\mathbf{X}) + \log q_\alpha(\mathbf{H}|\mathbf{X}). \tag{5.41}$$

Given a set of annotated chroma vectors with the ground-truth chords and keys, the training objective is to maximize the sum of the four target functions listed above:

$$
\begin{aligned}
\mathcal{L}_{\theta,\alpha,\beta} = &\sum_{\mathbf{X}} \mathcal{L}_{\mathbf{X}}(\theta, \alpha, \beta) + \sum_{\mathbf{X},\mathbf{S}} \mathcal{L}'_{\mathbf{X},\mathbf{S}}(\theta, \alpha, \beta) \\
&+ \sum_{\mathbf{X},\mathbf{H}} \mathcal{L}'_{\mathbf{X},\mathbf{H}}(\theta, \alpha, \beta) + \sum_{\mathbf{X},\mathbf{S},\mathbf{H}} \mathcal{L}'_{\mathbf{X},\mathbf{S},\mathbf{H}}(\theta, \alpha, \beta).
\end{aligned} \tag{5.42}
$$

To stabilize the semi-supervised training, we use a curriculum learning strategy. First, only the supervised term $\sum_{\mathbf{X},\mathbf{S},\mathbf{H}} \mathcal{L}'_{\mathbf{X},\mathbf{S},\mathbf{H}}(\theta, \alpha, \beta)$ is fully optimized in the non-regularized supervised manner. After the supervised term converged, the VAE is jointly trained such that $\mathcal{L}(\alpha, \beta, \theta)$ is maximized.

5.2.6 Prediction

Similar to the method in Chapter 4, given the chroma vectors $\mathbf{X}$, the chord and key label estimations are given by combining the label posteriors and the Viterbi

post-processing technique. Considering the temporal continuity of chords and keys, the self-transition probabilities for post-processing are set to 0.9 for chords and 0.95 for keys.

5.3 Evaluation

This section reports comparative experiments conducted for evaluating the effectiveness of the proposed method.

5.3.1 Experimental Conditions

Model Configurations

Each of the classification model $q_\alpha(\mathbf{S}, \mathbf{H}|\mathbf{X})$, the recognition model $q_\beta(\mathbf{Z}|\mathbf{X})$ and the generative model $p_\theta(\mathbf{X}|\mathbf{S}, \mathbf{H}, \mathbf{Z})$ was implemented with a three-layered bi-directional long short-term memory (BLSTM) network with layer normalization, where each layer had 128 hidden units for each direction. The output vector of the BLSTM layers was transformed into the desired shape using a fully-connected layer, and then normalized with the softmax or sigmoid function. The deep autoregressive language model was implemented with a single-layer uni-directional LSTM network, where each layer had 32 hidden units.

The parameters θ, ϕ, α, and β were optimized with Adam [65] with an initial learning rate of 0.001. Each stage of the curriculum learning consisted of 30 epochs to ensure convergence. Each minibatch contained 32 sequences randomly picked from training data, and each sequence contains 431 frames (20 sec), where the chroma vectors (and the ground-truth chords and keys if available) were jointly rotated by a random number for compensating for the imbalance in key classes.

Compared Methods

Various implementations for $p_\theta(\mathbf{X}|\mathbf{S}, \mathbf{H}, \mathbf{Z})$ and $q_\alpha(\mathbf{S}, \mathbf{H}|\mathbf{X})$ are compared in our experiments. We tested the possible combinations of two types of classification

models and two types of generative models. The compared classification models are:

- **Separated (SEP)**: The chords $\mathbf{S}$ and the keys $\mathbf{H}$ are estimated separately from $\mathbf{X}$ using independent single-task models, as is defined in (5.11) and (5.12);
- **Shared (SH)**: The chords $\mathbf{S}$ and the keys $\mathbf{H}$ are estimated jointly from $\mathbf{X}$ using a shared-parameter multi-task model, as is defined in (5.15);
- **Hierarchical (HIER)**: The chords $\mathbf{S}$ are estimated from $\mathbf{X}$, and the keys are then estimated from the estimated $\mathbf{H}$, as is defined in (5.17) and (5.18).

Each type of classification model is trained in either **supervised** or **regularized** way. The **supervised** way means that the classification model is solely optimized to maximize the supervised objective $\mathcal{L}_{\mathbf{X},\mathbf{S},\mathbf{H}}$ only. The **regularized** way means that the classification model is jointly optimized together with the generative model in the proposed VAE method. In our experiments, **supervised** and **regularized** learning methods are denoted as **SUP** and **VAE** respectively.

The compared language models are:

- **Autoregressive (AR)**: The RNN-based autoregressive model formulated in (5.3);
- **Markov (MK)**: The Markov models formulated in (5.34);
- **Uniform (U)**: The plain uniform distributions defined in (5.8) and (5.9).

In total, we got 6 combinations of classification and generative models to form a VAE. In the following sections, we refer to the combinations of the shorthands (*e.g.*, **SIN-NH-AR**).

Furthermore, to evaluate the effect of the language model, we conducted additional cross-validation experiments on the VAE of chord and key labels. This VAE is the sub-module of the **HIER-H** combination, composed with the classification model that predicts key label posteriors from chord labels, and the generative model that predicts chord label distribution from keys. It is trained on the data pairs of chord and key labels to optimize the terms in $\mathcal{L}_{\theta,\alpha,\beta}$ with

respect to $\mathbf{S}$ and $\mathbf{H}$:

$$\hat{\mathcal{L}}_{\mathbf{S}} = \mathbb{E}_{q_\alpha(\mathbf{H}|\mathbf{S})}[\log p_\phi(\mathbf{S}|\mathbf{H}) + \log p(\mathbf{H})] + \mathrm{Entropy}(q_\alpha(\mathbf{H}|\mathbf{S})), \tag{5.43}$$

$$\hat{\mathcal{L}}_{\mathbf{S},\mathbf{H}} = \mathrm{Entropy}(q_\alpha(\mathbf{H}|\mathbf{S})) + \log p_\phi(\mathbf{S}|\mathbf{H}) + \log p(\mathbf{H}) + \log q_\alpha(\mathbf{H}|\mathbf{S}). \tag{5.44}$$

In the experiment, we compare the two conditions with respect to the model:

- **KC-SUP**: The classification model is trained in the supervised manner without any regularization;
- **KC-AR-VAE**: The classification model is trained with the regularization method using the autoregressive language model.

By comparing the three conditions, we evaluate the effect of the regularization method using different language models.

Datasets

We collected a set of annotated music recordings for evaluation. Specifically, we collected 224 songs from Isophonics dataset [45] and 63 songs from Robbie Williams dataset [50] that have time-synchronized chord and key annotations. Since the key classification model recognizes only major and minor keys, songs that include other scales (*e.g.*, Mixolydian scale) are excluded from the two datasets. As in Chapter 4, the chord labels in the dataset were reduced to the 73 chord classes that are defined in our model. Under each condition, we conducted 5-fold cross validation on the annotated dataset. To compensate the imbalance of key classes, we applied data augmentation to the training dataset by pitch-shifting the chroma vectors and chord labels in the dataset. As a result, the amount of training data was increased by 12 times.

5.3.2 Evaluation Measures

The chord and key estimation accuracies were measured by the weighted overlap rates between the estimated and ground-truth chord and key classes of the annotated music signals. The weighted accuracy of each song is calculated

Table 5.1: Estimation accuracy of chords and keys

			Key errors(%)		
	chord(%)	key(%)	Parallel	Relative	Fifth
SIN-SUP	81.41	76.93	4.79	6.59	4.38
SH-SUP	81.14	81.75	3.04	4.98	4.86
HIER-SUP	81.51	78.53	2.48	4.71	4.69
SIN-U-VAE	81.95	80.43	4.83	5.36	3.78
SIN-MK-VAE	82.55	79.17	5.08	6.17	3.24
SIN-AR-VAE	81.66	80.69	3.93	5.04	4.27
SH-U-VAE	82.28	80.77	3.69	6.13	3.80
SH-MK-VAE	82.72	81.29	3.52	5.25	4.37
SH-AR-VAE	82.01	81.79	3.29	4.30	5.31
HIER-U-VAE	82.83	84.07	2.12	4.89	3.12
HIER-MK-VAE	82.79	82.68	2.01	5.55	3.17
HIER-AR-VAE	82.29	82.77	2.42	5.4	4.28

with *mir_eval* library [44]. The overall accuracy was given by the average of the piece-wise accuracies weighted by the song length.

For key estimation, we also measured the ratios of typical estimation errors on keys:

- *Parallel*: the estimated key has the same tonic note as the reference key, but differs in scale (*e.g.*, C major and C minor);
- *Relative*: the estimated and reference key shares a same set of musical notes (*e.g.*, C major and A minor);
- *Fifth*: the estimated key is a perfect-5th above the reference key.

These types of errors account for the majority of key estimation errors in the existing studies [70], for the estimated and reference keys share (nearly) the same set of musical notes. Therefore, pitch information like chroma vectors is often too ambiguous for distinguishing these closely-related keys.

5.3.3 Experimental Results

The experimental results of joint chord and key estimation are shown in Table 5.1.

Supervised Training

The first three rows in Table 5.1 list the performances of chord and key classification models trained on the annotated dataset in the non-regularized **supervised** way. We found that the **shared** and the **hierarchical** classification models outperformed the **separated** model in the key estimation task. The **shared** classification models achieved the best key estimation accuracy. More specifically, it significantly reduced the *parallel* and *relative* key errors by more than 1%. This result indicates that multi-task learning method is trained to avoid outputting key labels that are incompatible with the estimated chords. The improvement of the **hierarchical** classification model was small compared with the **shared** model. However, we observed that the **hierarchical** model made even fewer *parallel* and *relative* key errors.

The advantages of the two classification models for key estimation can also be observed by comparing their confusion matrices. From the main diagonal of the matrix in Fig. 5.6, we see that the **shared** model had more correct estimation than the **hierarchical** model on most key labels. The **shared** classifier made fewer errors misjudging major keys as other major keys, and minor keys as major keys. Especially, it made much fewer errors misjudging minor keys as its relative keys, which greatly increased the estimation accuracy on C:min and E:min keys. However, the **shared** classifier tended to misjudge major keys as their relative and parallel keys, and minor keys as their parallel keys. These errors became the main reason that decreased the key estimation accuracy on G:maj and A:min keys. Although the *hierarchical* classifier was inferior to the *shared* classifier in the overall accuracy, it was able to reduce relative and parallel key errors effectively.

There was no significant difference between the chord estimation performances of the three models. This was an expected result since all the three models took chroma vectors as the input. Although the **shared** classification model was able to benefit key estimation accuracy by jointly estimating chord labels, it had little benefit for chord estimation.

Regularized Training

The lower rows of Table 5.1 correspond to the classification models trained with the regularized training methods (*-VAE). The regularized **separated** and **shared** models with the **uniform** prior (**SIN-U-VAE** and **SH-U-VAE**) outperformed the classification models with supervised training (**-SUP), and the regularized model with the **Markov** prior (**SIN-MK-VAE** and **SH-MK-VAE**) were even better. The results again showed the advantage of the Markov language model for ACE task, which we we have already shown in Chapter 4. However, for the **hierarchical** model, **HIER-U-VAE** achieved the highest chord estimation accuracy, even outperforming the other classification models regularized by the **Markov** language model. For all classification models, the **autoregressive** language model showed little advantage over the other methods.

The key estimation accuracy by the **hierarchical** classification model significantly improved when trained with the regularizing method (**HIER-**-VAE**). Similar to the chord estimation task, the classification model performed better when regularized with the **uniform** prior (**HIER-U-VAE**), and that with the **Markov** prior (**HIER-MK-VAE**) achieved lower overall performance. More specifically, the regularized **hierarchical** classification model made fewer *parallel* key and *Fifth* key errors, while did not improve the *relative* key errors.

We observed how the regularized classification model improved the key estimation performance by comparing the estimation results between **HIER-U-VAE** and **HIER-SUP**. According to Fig. 5.7, the estimation accuracy of C:min and E:min keys are improved mainly by reducing the mistakes that misjudged them as their relative keys D:#maj and G:maj. Similarly, the estimation accuracy of C:maj, D:maj and F#:maj are improved mainly by reducing the *fifth* key error. However, the regularized training also had a little negative impact on avoiding the *parallel* key errors. We observed that **HIER-U-VAE** made more parallel key errors than **HIER-SUP** for D:min, E:min, A:min and B:min. The accuracy of A:min was decreased for the regularized classification model misjudged A:min as G:maj and A:maj.

The **shared** classification model with regularized training (**SH-**-VAE**) showed

little improvement on key estimation compared with the non-regularized model **SH-SUP**, regardless of which language model is used. Similar to **HIER-U-VAE**, **SH-U-VAE** also had difficulty avoiding the *parallel* and *relative* key errors. Fig. 5.8 shows that the regularized classification model misjudged D:min and E:min keys as their *parallel* keys D:maj and E:maj, which significantly decreased the accuracy of the two keys. In addition, it misjudged E:maj and G:maj keys as their *relative* keys C#:min and E:min.

5.4 Conclusion

This section describes a DNN-based joint chord and key estimation method that integrates the multi-label estimation task into the VAE-based regularized training method. Extending the VAE-based chord estimation method proposed in Chapter 4, we formulated a hierarchical generative model on the basis of the common music composition procedure. We evaluated the advantages of using various types of classification models and language models under the regularized training framework. The experimental result shows that the the combination of multi-task learning and the VAE training method can further improve AMT models. Specifically, the combination of the hierarchical classification and the uniform language model works best for the proposed VAE training method, achieving the best performance for both chord and key estimation. However, although the regularized classification models had higher overall key estimation accuracy than the non-regularized ones, the regularized models tended to increase misjudgements on some specific types of keys. Similar to the VAE in Chapter 4, the VAE proposed in this chapter also had difficulties dealing with the ambiguity in audio features with respect to keys. In addition, the autoregressive model and the Markov model, which are more meaningful language models, had little advantage when using the hierarchical classification model.

We have made a pioneering attempt to formulate a VAE-based semi-supervised music transcription method that jointly estimates multiple musical elements.

Although our experiments provided some promising results, better implementations for the classification, generation, and the language models are required for developing a comprehensive music transcription method.

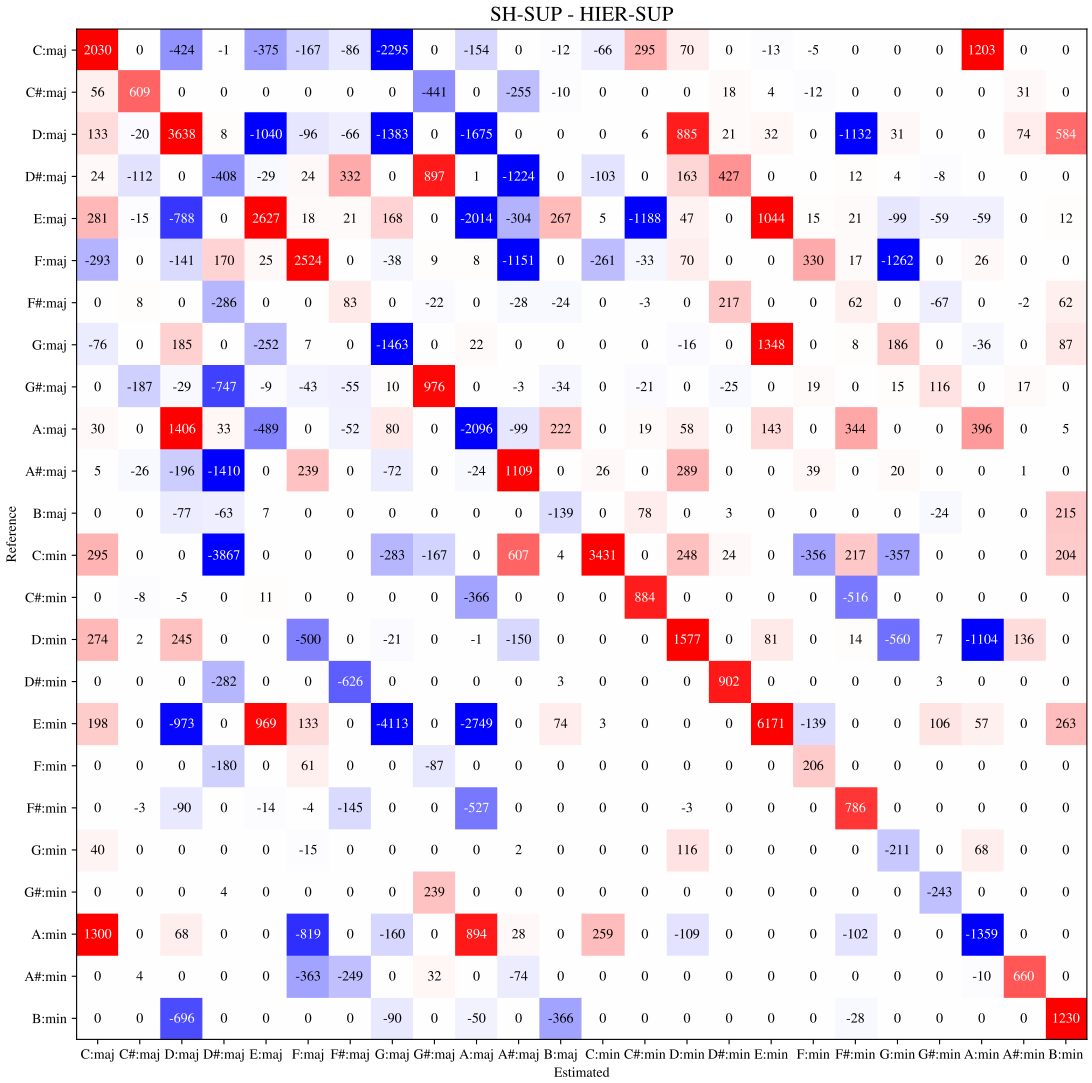


Figure 5.6: The difference between the confusion matrix of key estimation by **SH-SUP** and **HIER-SUP**. The numbers in the matrix represent the number of frames.

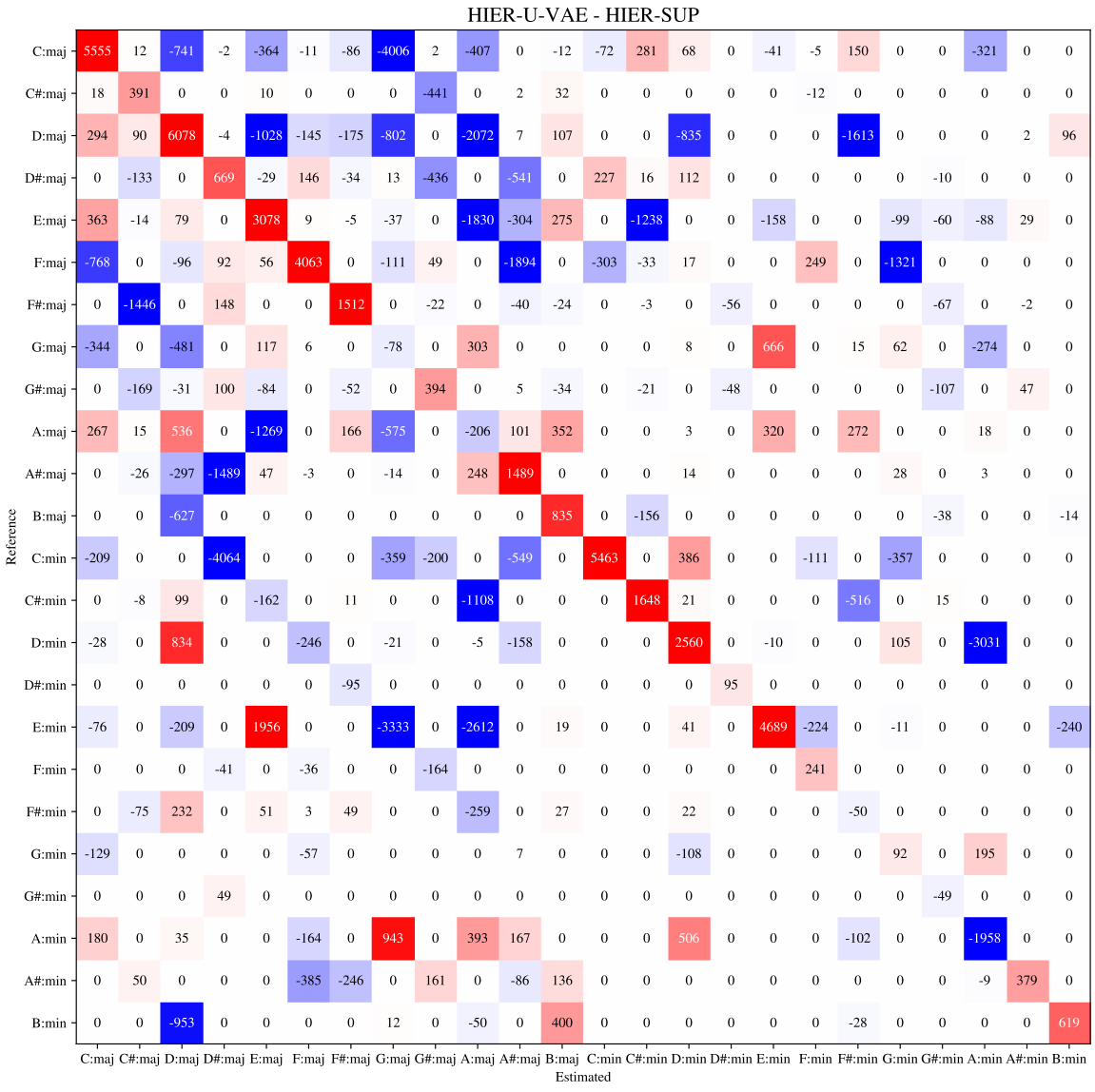


Figure 5.7: The difference between the confusion matrices of key estimation by **HIER-U-VAE** and **HIER-SUP**.

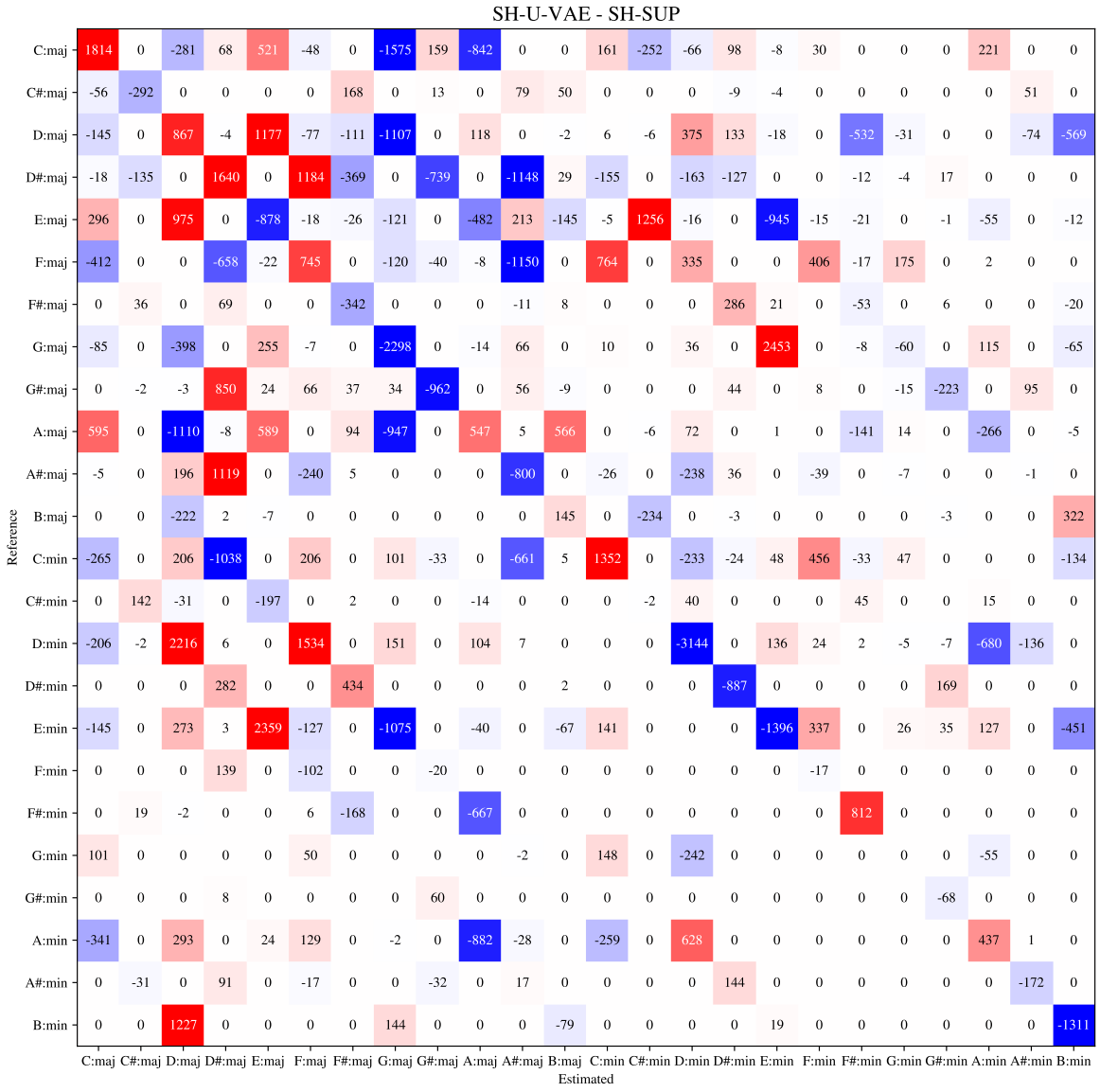


Figure 5.8: The difference between the confusion matrix of key estimation by **SH-U-VAE** and **SH-SUP**.

Chapter 6

Conclusion

We summarize the contributions of this thesis to DNN-based automatic chord estimation (ACE) and discuss the future directions.

6.1 Contributions

In this thesis, we have attempted to make effective use of non-annotated music data (MIDI and audio data) for DNN-based ACE. The amount of annotated data has been the main bottleneck limiting the performance of conventional ACE methods. To address this issue, we improved the feature extraction and estimation model training methodology of a standard ACE framework without paying the additional cost of increasing the annotated training data.

In Chapter 3, we presented a data-driven approach to chroma vector calculation from music signals. We designed a DNN that estimates the pitch class salience from an audio spectrogram at the frame level. For supervised training, we used a sufficient amount of synthesized music signals and the corresponding pitch class salience, both of which were generated from MIDI data. Strong domain knowledge about the effectiveness of the chromatic representation can thus be introduced without collecting additional hand-annotated music data. The experiments in Section 3.4.3 showed that the pitch class salience can be estimated accurately from real-world music recordings and that chroma vectors including little information irrelevant to ACE can be obtained. Using such chroma vectors as the input to a DNN-based chord classifier, the performance of ACE was sig-

nificantly improved. This technique forms the basis of ACE methods described in the following chapters.

In Chapter 4, we proposed an ACE method that unifies *generative* and *discriminative* approaches. We formulated a deep generative model that represents the generative process of chroma vectors from discrete chord labels and continuous latent features. In the framework of amortized variational inference (AVI), deep classification and recognition models were introduced for inferring the chord labels and latent features from the chroma vectors, respectively (Fig. 4.1). By integrating the deep classification model into the principled statistical inference formalism of the generative approach, we drew the potential of the powerful discriminative model. The experiments in section 4.3.2 showed that the regularization mechanisms with the deep generative model and the Markov prior significantly improved the performance of ACE. The chord sequences inferred by the classification model had higher correct rates, and yet were more consistent with respect to the given Markov prior. We found the limitation of the semi-supervised training that the trained classification model tends to mistakenly classify some specific chord types into more common types. Our attempt to use the semi-supervised VAE for ACE opens up a door to unify the deep generative and discriminative methods for AMT.

In Chapter 5, we extended the VAE described in Chapter 4 for joint key and chord estimation on the basis of the multi-task learning approach. To handle the additional latent variable representing key labels, we compared three architectures to implement the multi-task classification model, namely the the *separated*, *shared*, and *hierarchical* architectures (Fig. 5.5). We also compared three language models for regularizing the classification models, namely the *uniform*, *Markov* (Fig. 5.4) and *autoregressive* (Fig. 5.3) architectures. The combination of the *hierarchical* classification model and *uniform* language model showed the best performance in the comparative experiment, outperforming the regularized single-task classification models in both chord and key estimation tasks. We showed the advantage of integrating multi-task learning with the VAE-based training, making a step forward for realizing comprehensive music transcription.

6.2 Future Directions

This section describes the remaining issues of our study on ACE. The chroma feature proposed in Chapter 3 is a low-dimensional, highly abstracted representation, which may possibly bottleneck the chord classification model for further improvement. Especially, such feature representation can be ambiguous when recognizing difficult chord types such as tetrad chords. One straightforward approach is to employ a higher-dimensional representation such as the *piano-roll* representation which tells the salience of actual pitches. Another possible approach is joint fine-tuning of the feature extraction model and the chord classification model, which are trained separately in the proposed framework. Furthermore, we can regard chroma vectors as latent features and formulate a three-layered VAE that incorporates a raw audio spectrogram as an observed variable on top of the proposed two-layered VAE consisting of chroma vectors and chord labels. Using the VAE framework, the chroma feature extraction model can be optimized jointly with the spectrogram generator, the chord classification model, and the chroma feature generator. In this method, the feature extraction model is not only trained to extract pitch information, but also to preserve as much information as needed to reconstruct the original spectrogram.

In the proposed ACE methods, all the variables (chroma vectors, latent features, and musical labels) are defined at the frame level. The chord labels are thus forcibly aligned with frame-level chroma vectors. Although the proposed language models such as the Markov model or the RNN-based autoregressive model can learn the continuity of frame-level label sequences, the experiments showed that they are still incapable of properly modeling the symbol-level long-term dependencies underlying the musical label sequences. Integration of a frame-level classification model and a higher-level (*e.g.*, beat-level or chord symbol-level) musical language model is a promising research direction.

In this thesis, we used only LSTM networks to implement the temporal models (classification, recognition, and generation models) because they have been widely used as solid baseline models in related studies. Therefore, more

advanced temporal models should also be considered. For example, the transformer [71] and its variants have often been used instead of RNNs in a wide variety of research fields including ACE [28]. Since the transformer is capable of learning an arbitrary long-term dependency inherited in a sequence using a self-attention mechanism, it can possibly benefit more from the proposed regularized training framework. Theoretically, advanced temporal models like transformers can be applied to the proposed VAE-based training without major changes, as long as they take a sequence of frame-level acoustic features as input and output a label sequence.

The chord classification models in our work consider only triad chords. The VAE-based method proposed in Chapter 3 does not address the larger chord vocabulary that includes tetrad chords and inversions. Replacing the chroma feature with a higher-dimensional feature representation of concrete pitch information would be necessary to reduce the ambiguity between different types of chords. We are also considering to integrate the structured learning approach [6,7] to the VAE-based learning framework.

The VAE methods proposed in Chapter 4 and 5 are still unable to fully benefit from semi-supervised learning. In other words, although the VAE can theoretically be trained on music signals without annotations as well, the performance of the classification models could not be improved using such unlabelled music data. In the context of semi-supervised VAE [37], *disentanglement* of labels and latent features is considered a key factor to the performance of semi-supervised learning. More specifically, if the latent feature estimated by the recognition model $p(\mathbf{Z}|\mathbf{X})$ contains information about the chord progression of the observation $\mathbf{X}$, then the generative model $p(\mathbf{X}|\mathbf{S},\mathbf{Z})$ may ignore the chord labels $\mathbf{S}$ and try to reconstruct the observed feature from the latent feature $\mathbf{Z}$ only. This phenomenon is known as *posterior collapse*, which has been a critical problem for training a VAE. To prevent posterior collapse and improve the proposed method using semi-supervised learning, it is necessary to introduce disentanglement techniques to ensure that the latent feature is irrelevant to the musical labels.

The AMT research can further be advanced to jointly consider more musical

concepts by resolving these issues. Our ultimate goal is to develop a comprehensive AMT system on the basis of a unified VAE that can treat mutually-dependent musical elements such as keys, beats, melodies, and musical notes as latent variables, without having to collect a large amount of annotations.

Bibliography

[1] M. Goto, K. Yoshii, H. Fujihara, M. Mauch, and T. Nakano, "Songle: A web service for active music listening improved by user contributions," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 311–316, 2011.

[2] A. Anglade, R. Ramirez, and S. Dixon, "Genre classification using harmony rules induced from automatic chord transcriptions.," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 669–674, 2009.

[3] J. P. Bello, "Audio-based cover song retrieval using approximate chord sequences: Testing shifts, gaps, swaps and beats.," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, vol. 7, pp. 239–244, 2007.

[4] M. E. P. Davies, P. Hamel, K. Yoshii, and M. Goto, "Automashupper: Automatic creation of multi-song music mashups," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 12, pp. 1726–1737, 2014.

[5] C. L. Krumhansl, *Cognitive foundations of musical pitch*, vol. 17. Oxford University Press, 2001.

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

[7] J. Jiang, K. Chen, W. Li, and G. Xia, "Large-vocabulary chord transcription via chord structure decomposition," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 644–651, 2019.

[8] F. Korzeniowski and G. Widmer, "Feature learning for chord recognition:

The deep chroma extractor," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 37–43, 2016.

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

[10] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proceedings of the International Conference on Learning Representations (ICLR)*, pp. 1–14, 2014.

[11] 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 (ISMIR)*, pp. 135–140, 2010.

[12] C. Schörkhuber and A. Klapuri, "Constant-Q transform toolbox for music processing," in *Proceedings of Sound and Music Computing Conference*, 2010.

[13] D. Fitzgerald, "Harmonic/percussive separation using median filtering," in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, vol. 13, 2010.

[14] J. Driedger, M. Müller, and S. Disch, "Extending harmonic-percussive separation of audio signals.," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 611–616, 2014.

[15] E. Gómez, "Tonal description of polyphonic audio for music content processing," *INFORMS Journal on Computing, Special Cluster on Computation in Music*, vol. 18, 2006.

[16] T. Cho, *Improved techniques for automatic chord recognition from music audio signals*. PhD thesis, New York University, 2014.

[17] E. J. Humphrey and J. P. Bello, "Rethinking automatic chord recognition with convolutional neural networks," in *2012 11th International Conference on Machine Learning and Applications*, vol. 2, pp. 357–362, 2012.

[18] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[19] K. Lee and M. Slaney, "Acoustic chord transcription and key extraction

from audio using key-dependent hmms trained on synthesized audio," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 2, pp. 291–301, 2008.

[20] K. Lee and M. Slaney, "Acoustic chord transcription and key extraction from audio using key-dependent hmms trained on synthesized audio," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 2, pp. 291–301, 2008.

[21] M. Mauch and S. Dixon, "Simultaneous estimation of chords and musical context from audio," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1280–1289, 2010.

[22] Y. Ni, M. McVicar, R. Santos-Rodriguez, and T. De Bie, "An end-to-end machine learning system for harmonic analysis of music," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 6, pp. 1771–1783, 2012.

[23] M. Khadkevich and M. Omogolo, "Use of hidden Markov models and factored language models for automatic chord recognition," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 561–566, 2009.

[24] R. Chen, W. Shen, A. Srinivasamurthy, and P. Chordia, "Chord recognition using duration-explicit hidden Markov models," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 445–450, 2012.

[25] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Audio chord recognition with recurrent neural networks.," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 335–340, 2013.

[26] F. Korzeniowski and G. Widmer, "Improved chord recognition by combining duration and harmonic language models," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 10–17, 2018.

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

[28] T. P. Chen and L. Su, "Harmony transformer: Incorporating chord segmen-

tation into harmony recognition," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 259–267, 2019.

[29] F. Korzeniowski and G. Widmer, "A fully convolutional deep auditory model for musical chord recognition," in *Proceedings of the IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 13–16, 2016.

[30] 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, and Language Processing*, vol. 27, no. 2, pp. 355–366, 2019.

[31] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[32] S. Sigtia, N. Boulanger-Lewandowski, and S. Dixon, "Audio chord recognition with a hybrid recurrent neural network," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 127–133, 2015.

[33] F. Korzeniowski and G. Widmer, "On the futility of learning complex frame-level language models for chord recognition," in *AES Conference on Semantic Audio*, pp. 2–6, 2017.

[34] F. Korzeniowski, D. R. W. Sears, and G. Widmer, "A large-scale study of language models for chord prediction," in *Proceedings of 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 91–95, 2018.

[35] T. Hori, R. F. Astudillo, T. Hayashi, Y. Zhang, S. Watanabe, and J. L. Roux, "Cycle-consistency training for end-to-end speech recognition," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6271–6275, 2019.

[36] K. Choi and K. Cho, "Deep unsupervised drum transcription," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*,

pp. 183–191, 2019.

[37] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, "Semi-supervised learning with deep generative models," in *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 3581–3589, 2014.

[38] J. Pauwels and J.-P. Martens, "Combining musicological knowledge about chords and keys in a simultaneous chord and local key estimation system," *Journal of New Music Research*, vol. 43, no. 3, pp. 318–330, 2014.

[39] H. Papadopoulos and G. Peeters, "Joint estimation of chords and downbeats from an audio signal," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 1, pp. 138–152, 2011.

[40] S. Böck and M. Davies, "Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 574–582, 2020.

[41] M.-H. Yang, L. Su, and Y.-H. Yang, "Highlighting root notes in chord recognition using cepstral features and multi-task learning," in *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, pp. 1–8, 2016.

[42] S. Böck, M. E. Davies, and P. Knees, "Multi-task learning for tempo and beat: Learning one to improve the other," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 486–493, 2019.

[43] G. X. J. Jiang and D. B. Carlton, "Mirex 2019 submission: Crowd annotation for audio key estimation," *Abstract of MIREX*, 2019.

[44] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. P. W. Ellis, "mir_eval: A transparent implementation of common MIR metrics," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 367–372, 2014.

[45] C. Harte, *Towards automatic extraction of harmony information from music signals*. PhD thesis, Queen Mary University of London, 2010.

[46] 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*, vol. 48, no. 3, pp. 232–252, 2019.

[47] T. Carsault, J. Nika, and P. Esling, "Using musical relationships between chord labels in automatic chord extraction tasks," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 18–25, 2018.

[48] J. Deng and Y. K. Kwok, "Large vocabulary automatic chord estimation with an even chance training scheme," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 531–536, 2017.

[49] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Popular, classical, and jazz music databases," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 287–288, 2002.

[50] B. D. Giorgi, M. Zanoni, A. Sarti, and S. Tubaro, "Automatic chord recognition based on the probabilistic modeling of diatonic modal harmony," in *nDS '13; Proceedings of the 8th International Workshop on Multidimensional Systems*, pp. 1–6, 2013.

[51] A. Berenzweig, B. Logan, D. Ellis, and B. Whitman, "A large-scale evaluation of acoustic and subjective music-similarity measures," *Computer Music Journal*, vol. 28, no. 2, pp. 63–76, 2004.

[52] J. A. Burgoyne, J. Wild, and I. Fujinaga, "An expert ground truth set for audio chord recognition and music analysis," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 633–638, 2011.

[53] C. Raffel, *Learning-Based Methods for Comparing Sequences, with Applications to Audio-to-MIDI Alignment and Matching*. PhD thesis, Columbia University, 2016.

[54] R. Bittner, B. McFee, J. Salamon, P. Li, and J. Bello, "Deep salience representations for f_0 estimation in polyphonic music," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 63–70, 2017.

[55] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[56] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[57] E. J. Humphrey and J. P. Bello, "Four timely insights on automatic chord estimation," in *Proceedings of the International Society for Music Information Retrieval*, pp. 673–679, 10 2015.

[58] Y. Wu and W. Li, "Music chord recognition based on MIDI-trained deep feature and BLSTM-CRF hybird decoding," in *Proceedings of 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 376–380, 2018.

[59] Y. Wu, T. Carsault, E. Nakamura, and K. Yoshii, "Semi-supervised neural chord estimation based on a variational autoencoder with latent chord labels and features," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2956–2966, 2020.

[60] S. Gershman and N. Goodman, "Amortized inference in probabilistic reasoning," in *Proceedings of the annual meeting of the cognitive science society*, vol. 36, pp. 517–522, 2014.

[61] E. Dupont, "Learning disentangled joint continuous and discrete representations," in *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 710–720, 2018.

[62] M. McVicar, R. Santos-Rodriguez, Y. Ni, and T. D. Bie, "Automatic chord estimation from audio: A review of the state of the art," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 2, pp. 556–575, 2014.

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

[64] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with Gumbel-Softmax," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[65] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, pp. 1–15, 2015.

[66] A. Graves, N. Jaitly, and A.-r. Mohamed, "Hybrid speech recognition with

deep bidirectional LSTM," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 273–278, 2013.

[67] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[68] J. Pauwels, K. O'Hanlon, E. Gómez, and M. B. Sandler, "20 years of automatic chord recognition from audio," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 54–63, 2019.

[69] Y. Wu, E. Nakamura, and K. Yoshii, "A variational autoencoder for joint chord and key estimation from audio chromagrams," in *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 500–506, 2020.

[70] F. Korzeniowski and G. Widmer, "Genre-agnostic key classification with convolutional neural networks," in *Proceedings of the International Society for Music Information Retrieval (ISMIR)*, pp. 264–270, 2018.

[71] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Łukasz Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5998–6008, 2017.

List of Publications

Journal Articles

1) Yiming Wu and Wei Li, "Automatic Audio Chord Recognition with MIDI-Trained Deep Feature and BLSTM-CRF Sequence Decoding Model," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 2, pp. 355–366, 2019. → **Chapter 3**.

2) Yiming Wu, Tristan Carsault, Eita Nakamura, and Kazuyoshi Yoshii, "Semi-Supervised Neural Chord Estimation Based on a Variational Autoencoder with Latent Chord Labels and Features," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2956–2966, 2020. → **Chapter 4**.

International Conferences

3) Yiming Wu, Tristan Carsault, and Kazuyoshi Yoshii, "Automatic Chord Estimation Based on a Frame-Wise Convolutional Recurrent Neural Network with Non-Aligned Annotations," in *2019 27th European Signal Processing Conference (EUSIPCO)*, pp. 1–5, 2019.

4) Yiming Wu, Eita Nakamura, and Kazuyoshi Yoshii, "A Variational Autoencoder for Joint Chord and Key Estimation from Audio Chromagrams," in *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APCIPA ASC)*, pp. 500–506, 2020. → **Chapter 5**.

Domestic Conference

5) 呉 益明, Tristan Carsault, 中村 栄太, 吉井 和佳, “音楽音響信号に対するラベル・テクスチャ分離型変分自己符号化器を用いた半教師ありコード推定,” 情報処理学会 第 124 回音楽情報科学研究会, vol. 2019-MUS-124, no. 8, pp. 1–6, 2019.