BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

CONVERSION OF ELECTRIC GUITAR TO MIDI

PŘEVOD ZÁZNAMU ELEKTRICKÉ KYTARY DO MIDI

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR **MICHAL KLČO**
AUTOR PRÁCE

SUPERVISOR **Doc. Dr. Ing. JAN ČERNOCKÝ,**
VEDOUCÍ PRÁCE

BRNO 2018

Master's Thesis Specification/20680/2017/xklcom00

Brno University of Technology - Faculty of Information Technology

Department of Computer Graphics and Multimedia Academic year 2017/2018

Master's Thesis Specification

For:	**Klčo Michal, Bc.**
Branch of study:	Computer Graphics and Multimedia
Title:	**Electric Guitar to MIDI Conversion**
Category:	Signal Processing

Instructions for project work:

1. Get acquainted with the principles of monophonic and polyphonic music transcription, concentrate on the PLCA technique.
2. Get acquainted with MIDI.
3. Survey available standard data, and select a good reference database, eventually, record your own data containing ground-truth.
4. Implement a basic guitar to MIDI converter, evaluate it and identify problematic points.
5. Suggest improvements, implement them and evaluate.
6. Prepare a short video presenting your work.

Basic references:

- Smaragdis, P.; Raj, B.; Shashanka, M.: A Probabilistic Latent Variable Model for Acoustic Modeling. Neural Information Processing Systems Workshop, 2006-12.

Requirements for the semestral defense:
Items 1 to 4.

Detailed formal specifications can be found at http://www.fit.vutbr.cz/info/szz/

The Master's Thesis must define its purpose, describe a current state of the art, introduce the theoretical and technical background relevant to the problems solved, and specify what parts have been used from earlier projects or have been taken over from other sources.

Each student will hand-in printed as well as electronic versions of the technical report, an electronic version of the complete program documentation, program source files, and a functional hardware prototype sample if desired. The information in electronic form will be stored on a standard non-rewritable medium (CD-R, DVD-R, etc.) in formats common at the FIT. In order to allow regular handling, the medium will be securely attached to the printed report.

Supervisor:	**Černocký Jan, doc. Dr. Ing.**, DCGM FIT BUT
Beginning of work:	November 1, 2017
Date of delivery:	May 23, 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

L.S.

Jan Černocký
Associate Professor and Head of Department

Abstract

Automatic music transcription and multi-pitch estimation are still challenging tasks in the field of music information retrieval. The recent state of the art systems incorporate different machine learning techniques to achieve the most accurate transcription of notes. Some of them are also limited to a specific music instrument or a music genre to reduce the diversity of the analyzed sound. In this work, multiple systems for conversion of electric guitar recordings to the MIDI files, based on different machine learning and spectral analysis techniques, are proposed, evaluated and compared.

Abstrakt

Automatický přepis hudby a odhad vícero znějících tónu jsou stále výzvou v oblasti dolování informací z hudby. Moderní systémy jsou založeny na různých technikách strojového učení pro dosažení co nejpřesnějšího přepisu hudby. Některé z nich jsou také omezeny na konkrétní hudební nástroj nebo hudební žánr, aby se snížila rozmanitost analyzovaného zvuku. V této práci je navrženo, vyhodnoceno a porovnáváno několik systémů pro konverzi nahrávek elektrické kytary do MIDI souború, založených na různých technikách strojového učení a technikách spektrální analýzy.

Keywords

automatic music transcription, music information retrieval, machine learning, neural networks, PLCA, electric guitar

Klíčová slova

automatický přepis hudby, dolování informací z hudby, strojové učení, neuronové sítě, PLCA, elektrická kytara

Reference

KLČO, Michal. *Conversion of Electric Guitar to MIDI*. Brno, 2018. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Doc. Dr. Ing. Jan Černocký,

Conversion of Electric Guitar to MIDI

Declaration

Hereby I declare that this master's thesis was prepared as an original author's work under the supervision of Dr. Jan Černocký. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

. . . .
Michal Klčo
May 22, 2018

Acknowledgements

I would like to thank to my supervisor, Dr. Jan Černocký for his professional advices and willingness to help. Also, I would like to thank to Karel Beneš and Michal Hradiš, Ph.D. for their useful advices and to my family and friends for their support.

Contents

List of Acronyms

ACF	Autocorrelation Function
AMDF	Average Magnitude Difference Function
AMT	Automatic Music Transcription
ANN	Artificial Neural Network
BHAD	Blind Harmonic Adaptive Decomposition
BLSTM	Bidirectional Long Short-Term Memory
BPTT	Back-Propagation Through Time
BRNN	Bidirectional Recurrent Neural Network
CNN	Convolution Neural Network
CQT	Constant-Q Transform
DC	Densely Connected
DFT	Discrete Fourier Transform
DNN	Deep Neural Network
ELU	Exponential Linear Unit
EM	Expectation-Maximization
F0	fundamental frequency
FC	Fully Connected
HMM	Hidden Markov Model
HPS	Harmonic Partial Sequence
IDMT	Institute for Digital Media Technology
IF	Instantaneous Frequency
KL	Kullback–Leibler
LSTM	Long Short-Term Memory
MAP	Maximum a Posteriori
MCMC	Markov Chain Monte Carlo
MIREX	Music Information Retrieval Evaluation eXchange
NMF	Non-negative Matrix Factorization
PLCA	Probabilistic Latent Component Analysis
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
STFT	Short-Time Fourier Transform
TDFC	Time-Distributed Fully Connected
TDNN	Time-Delayed Neural Network

Chapter 1

Introduction

In the last years, there has been a great interest in music signal analysis. As the amount of audio content on the internet is increasing, the need to analyze or categorize is still bigger. One of the ways how to analyze music signal is an automatic music transcription (AMT). Most existing systems for automatic music transcription provide transcription of notes played in the given audio recording in the form of piano-roll or MIDI format that can be easily converted to human-readable representation.

However, while the extraction of fundamental frequency (F0) from speech or monophonic record is considered as solved problem, the problem of estimation of multiple fundamental frequencies remains still open. Analysis of music signals compounded of the sounds from multiple sources is difficult because the higher harmonic frequencies overlaps in a frequency spectrum. This makes the estimation of all the fundamental frequencies the challenging task [7][3].

This thesis describes, proposes, evaluates and compares systems for automatic music transcription of electric guitar recordings to MIDI representation. The systems are based on techniques used in the recent automatic music transcription researches: spectrogram factorization using PLCA and deep learning methods using deep neural network are utilized to retrieve pitch, time and duration of played notes.

1.1 Related Work

The multi-pitch estimation is considered to be one of the most difficult music signal analysis. Even the best music transcription systems do not get near the performance of human expert. Despite of that, the field is still very active [7].

In the past, the system for multi-pitch estimation were mainly based on a combination of audio feature extraction and heuristic techniques. These system do not employ specific model, they detect pitches according to features extracted from time-frequency representations of signals either in a joint or an iterative manner. The pitches are estimated using pitch salience function or pitch candidate set score function [29, 36, 49]. The best performing method in Music Information Retrieval eXchange (MIREX) for 2009-2011 was a joint pitch estimation algorithm based on a pitch candidate set score function proposed by Yeh [49]. Another notable recognition system proposed by Pertusa [36] computes a pitch salience function and evaluates combinations of pitch candidates using a measure of distance between a Harmonic Partial Sequence (HPS) and a smoothed HPS.

Systems using statistical model-based multi-pitch detection calculate a Maximum a Posteriori (MAP) of all possible F0 combinations. The *PreFEst* system [22] models each harmonic by a Gaussian centered at its position on the log-frequency axis and calculates MAP using the Expectation-Maximization (EM) algorithm. In similar manner to MAP, a time-domain Bayesian approach for AMT which used a Gabor atomic model was proposed in [16], which used a Markov Chain Monte Carlo (MCMC) method for inference, while the model also supported time-varying amplitudes and inharmonicity.

Another approach for note recognition, used also in this work, is a spectrogram factorization technique. Non-negative Matrix Factorization (NMF) is a technique that decompose an input spectrogram into spectral bases and time activity bases for each pitch. This technique was used in [15] where sparseness constraints were added into the NMF update rules. Bay [3] employed NMF and its alternative formulation called Probabilistic Latent Component Analysis (PLCA) in his work. A model that extended the convolutive PLCA algorithm proposed by Benetos [6] incorporates shifting across log-frequency for supporting frequency modulations.

In the most recent years, AMT system based on Artificial Neural Network (ANN) achieved the best results in the field. In MIREX 2016, the best performing AMT system was SONIC [33] that employed networks of oscillators in a combination with a time-delay neural network. In MIREX 2017, system based on Convolution Neural Network (CNN) [46] outperformed all other participating systems.

1.2 Aims of thesis

The main goal of the thesis is comparison of systems based on different machine learning techniques and spectral analysis methods. Short-Time Fourier Transform (STFT) and Constant-Q Transform (CQT) are analyzed in detail and their suitability in music transcription is compared. For music transcription, a system based on PLCA is created according to the previous works in the field [5, 7] together with systems based on Deep Neural Network (DNN), Recurrent Neural Network (RNN) inspired by [33] and on combination of PLCA and RNN. The systems are evaluated on real guitar recordings obtained from Institute for Digital Media Technology and their performance are compared. Finally, the achieved results are discussed and pros and cons of each technique are described.

1.3 Structure of thesis

In chapter 2, basic theoretical background that this work is based on, is presented. It describes techniques for spectral analysis, multi-pitch estimation, post-processing methods and evaluation metrics used in this work. In chapter 3, data used for training, validation and evaluation of systems are described. Chapter 4 presents the proposed systems, used methods and their specific values of parameters. In chapter 5, results of evaluations of the proposed models are presented. In chapter 6, the whole work is summed up and future directions are suggested.

Chapter 2

Theoretical Background

2.1 Music Signals

Signals that repeat itself at regular time intervals (*periods*) are called *periodic* signals. The *fundamental frequency* (F0) of such signal is defined as the reciprocal of the period. These two attributes are the attributes of periodic signal in the time-domain [7].

A music signal is an audio signal that is typically produced by a combination of one or more sounds generated by music instruments or a singing voice. The music signals produced by electric guitar can be classified as pitched sounds. Unlike unpitched sounds produced by instruments like drums, they have locally stable fundamental periods. Pitched sounds can be described by a series of peaks (harmonics) in the frequency domain that are harmonically related. They appear at integer multiples of fundamental frequency.

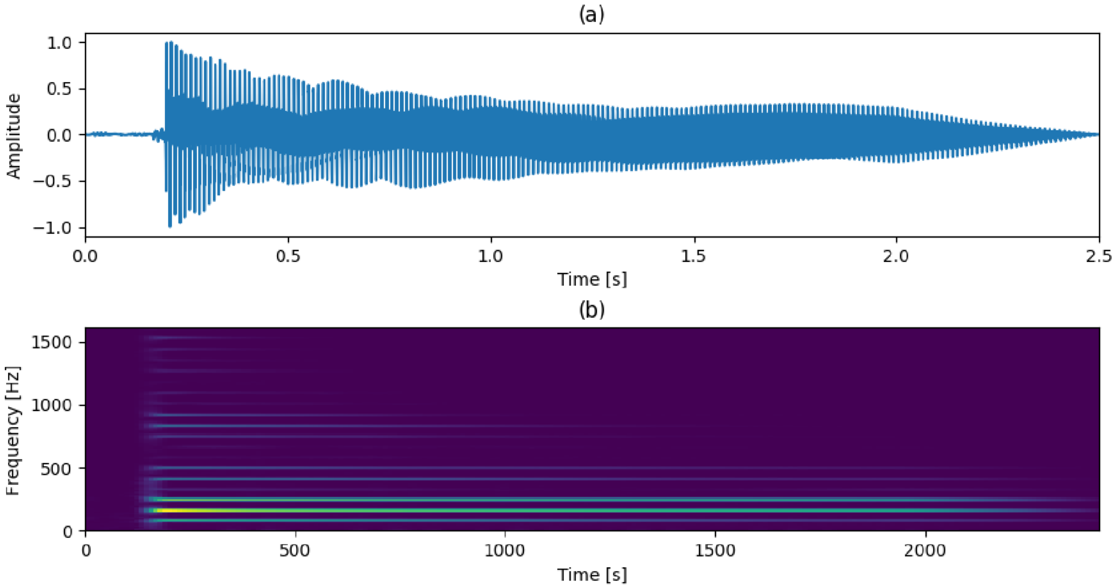


Figure 2.1: E2 guitar note (82.4 Hz). The waveform of the signal (a) and the spectrogram (b).

Figure 2.1 shows the waveform and the spectrogram of the signal of the played E2 guitar note. The harmonics of the tone E2 are located at the integer multiples of fundamental frequency.

When the note is played, its characteristics change overtime. The evolution of played note can be decomposed into several phases. The pitched percussive instruments (guitar, piano) have an attack stage followed by decay, sustain and release stages [24]. The attack and decay stages constitute the part when the string is plucked and set in motion. The sustain stage is that stage for which the characteristics of the sound are relatively constant, while the amplitude is constantly decreasing. The release stage is then defined as the time until the sound fade-away. In Figure 2.2, all the stages of a note can be seen.

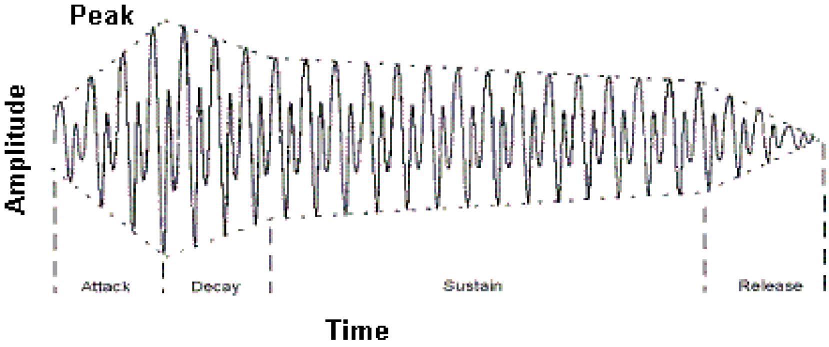


Figure 2.2: The envelope of a waveform with the highlighted stages [24].

2.1.1 Problems of real signals

Real music signals do not meet all the theoretical criteria. They are not periodic in the meaning of exact repeating of the same part of the signal, but they are *pseudo-periodic*. The repeating parts of a pseudo-periodic signal are similar but not the same.

As it was stated in this section, harmonics of the music signal appear on integer multiples of fundamental frequency. In the case of the real music signal, some of the harmonics can be shifted from the integer multiples of F0. This phenomenon is called *inharmonicity* of a signal [5].

Another problem is low or missing peak at fundamental frequency. Human brain perceives the pitch of tone that is physically missing in the signal of tone. This problem is not significant in this work because it occurs in the lowest notes of a piano that cannot be played on an electric guitar with standard tuning.

Figure 2.3 shows the inharmonicity phenomenon of the spectrum of the E2 note where the harmonic peaks are shifted from the expected locations.

The mentioned problems make analysis of a music signal more difficult for methods that take for granted the theoretical structure of a tone spectrum. However, classifiers or a probabilistic models overcome these problems because they do not assume anything about the structure of spectra and work with real spectra used during a training phase.

2.2 MIDI

One of the way how to store music notation is Musical Instrument Digital Interface (MIDI) protocol. MIDI is a technical standard that describes a communication protocol, a digital interface, electrical connectors, and provides the capability of a connection of electronic musical instruments, computers and other related music and audio devices [45]. The parameters like pitch, onset, offset, and intensity can be stored using MIDI protocol, along

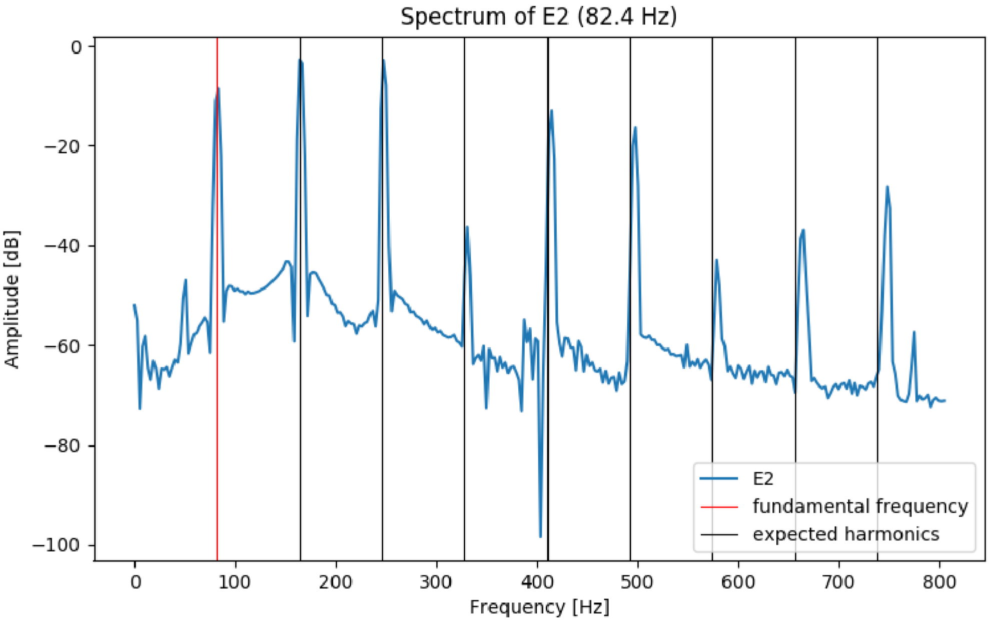


Figure 2.3: The spectrum of guitar note E2 (82.4 Hz). The red vertical line denotes the location of the fundamental frequency and the black vertical lines denote the expected locations of the harmonics.

with additional information such as tempo and key. The MIDI protocol has a lot of advantages, regarding simplicity, but does not provide storage of a proper musical notation or expressive features [7].

In the MIDI protocol, pitches are assigned MIDI numbers which relate the fundamental frequency $F0$. The relation is defined as follows:

$$n_{MIDI} = 12 \cdot \log_2 \frac{F0}{440} + 69 \tag{2.1}$$

$$F0 = 440 \cdot 2^{\frac{n_{MIDI}-69}{12}} \tag{2.2}$$

where n_{MIDI} is the MIDI number of a note with fundamental frequency F0.

Useful visual representation of the MIDI notation is a piano-roll. Figure 2.4 shows the piano-roll for J.S. Bach's prelude in C major, from the Well-tempered Clavier Book I. It depicts pitches in the vertical axis and time in the horizontal axis.

2.3 Spectral Analysis

To process music signals by computer, the signals have to be converted into a digital form with sampling and quantization operations. These time-domain digital forms that are used to store, playback or edit music signal are not applicable to the most of signal processing methods. It is desirable to transform the signals into a more suitable representation. The most popular one is the time-frequency domain representation [25, 28].

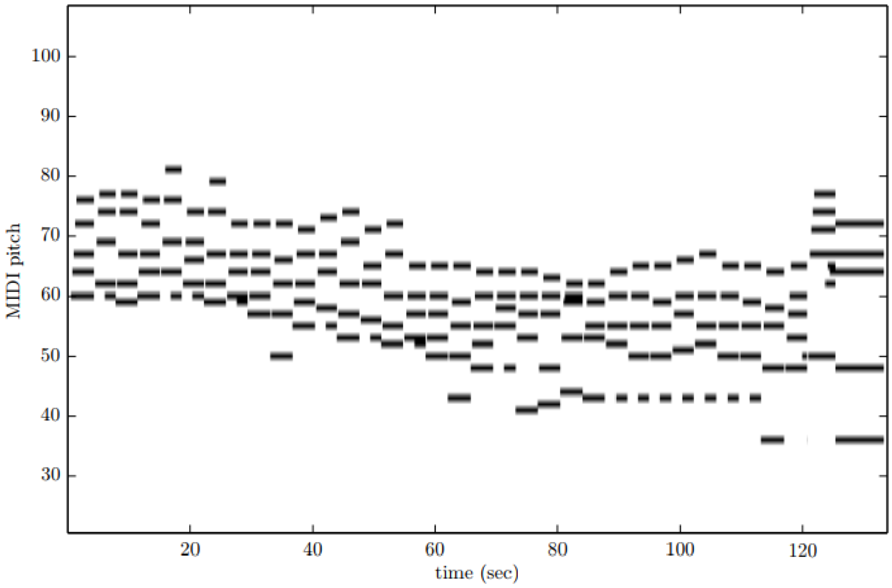


Figure 2.4: The piano roll for J.S. Bach's prelude in C major, from the Well-tempered Clavier Book I [7].

2.3.1 Short-Time Fourier Transform

Short-time Fourier Transform (STFT) is used to get time-varying spectra from non-stationary signals. This is achieved trough a calculation of discrete Fourier Transform (DFT) over short periods of time [3]. The DFT is defined as

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j\omega_k n} \tag{2.3}$$

where $x[n]$ is the discrete time signal and k is the bin-frequency index [2]. Normalized angular frequency is sampled linearly at $\omega_k = 2\pi k/N$ which does not need to be convenient in the case of a music signal processing. This problem is described in the next section.

The part of the signal is selected with an analysis window. Let us define the analysis window function as

$$w[n] = \begin{cases} w[n] > 0, & n = 0, ..., N-1 \\ w[n] = 0, & otherwise. \end{cases} \tag{2.4}$$

where N is the analysis window length. Then, the STFT of the $x[n]$ signal can be written as

$$STFT\left\{x[n]\right\} = X[k,m] = \sum_{n=mH}^{mH+N-1} x[n]w[n-mH]e^{-j\omega_k n} \tag{2.5}$$

where $m = 0, \ldots, M-1$ is the time-frame index and H is the shift of the analysis window in samples. The width of the analysis window has significant impact to how the signal is represented because the STFT has fixed resolution. The window with small length has better time resolution (short notes are easier to detect) but worse frequency resolution (one peak in the frequency domain can be spread over several frequency bins).

The DFT produces a spectrum where all frequency components are on linear scale with a specific frequency resolution that does not change over frequency. This can result in too little resolution for lower music frequencies and better than needed resolution for higher frequencies. For example, consider the note E2 with the frequency equal to 82.4 Hz and the note F2 with the frequency equal to 87.3 Hz (the two lowest notes that can be played on a guitar with a standard tuning). Between these two notes, there is the 4.9 Hz frequency difference. If we consider a sampling frequency (f_s) equal to 22050 Hz and a window size N of 1024 samples used for the DFT, then the frequency resolution is $f_s/N = 21.53$ Hz which is 26% of the frequency of the E2 note. The difference of fundamental frequencies of two adjacent semitones (for example E2 and F2) corresponds to 6% of the frequency of the lower one. Using the DFT, the information about 4 adjacent semitones would be contained in a single frequency component [4][10]. Thus, it might be inefficient in some disciplines of music analysis. This problem can be overcome with multi-resolution analysis like constant-Q transform.

2.3.2 Constant-Q Transform

Constant-Q Transform (CQT) has several advantages over the discrete Fourier Transform (DFT) in analysis of music signals. As it was mentioned in 2.3.1, the DFT produce spectrum where all frequency components are on linear scale with a specific frequency resolution that does not change over frequency. On the other hand, the CQT produces spectrum where frequency components are on logarithmic scale.

Instruments like guitar are tuned in equal temperament with frequencies $f_k \cong f_{min} \cdot (2^{(1/12)k})$ where f_{min} is the frequency of the lowest note and f_k is the frequency of the kth semitone of the guitar. In the same manner, we can write the equation for analysis frequencies with the quartertone spacing:

$$f_j \cong f_{min} \cdot (2^{(1/24)j}) \tag{2.6}$$

where j is the counter of quartertones. The frequency resolution, that is equal to the difference of frequencies of the adjacent frequency components, is defined as:

$$\Delta f_j = f_{j+1} - f_j = 2^{1/24} \cdot f_j - f_j \tag{2.7}$$

Then, the ratio of frequency to resolution or Q for quartertone spacing is constant defined as:

$$f_j/\Delta f_j = 1/(2^{1/24} - 1) \cong 34 \tag{2.8}$$

For the discrete Fourier transform, the frequency resolution is equal to the sampling frequency f_s divided by the windows size N. If the ratio of frequency to resolution is a constant (constant-Q), the window size varies with the frequency component:

$$N[k] = f_s/\Delta f_k = f_s/(f_k/Q) = f_s Q/f_k \tag{2.9}$$

An expression of CQT for the k-th spectral component derived from the expression of short-time DFT is:

$$X^{cq}[k] = \frac{1}{N[k]} \sum_{n=0}^{N[k]-1} w[n,k]x[n]e^{-j2\pi kn/N[k]} \tag{2.10}$$

Figure 2.5 shows frequency resolution (a) and size of the analysis window (b) corresponding to analyzed frequency using the DFT and CQT.

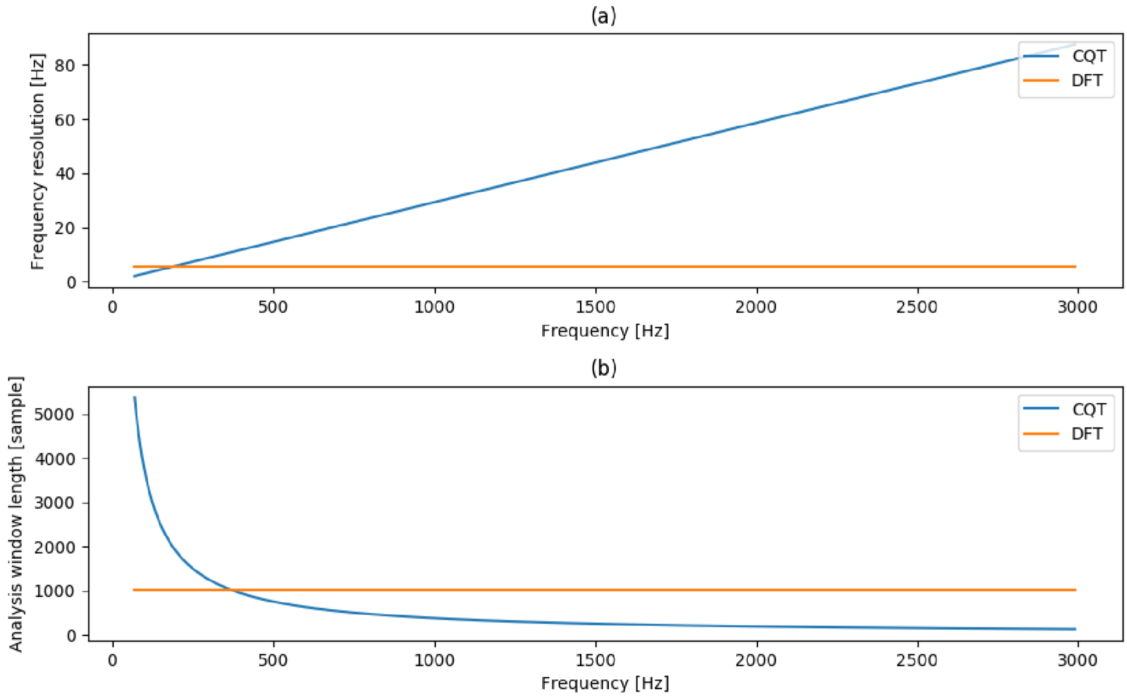


Figure 2.5: Frequency resolution (a) and size of the analysis window corresponding to analyzed frequency using the DFT and CQT.

2.4 Fundamental frequency estimation

Algorithms for estimation of a single fundamental frequency extract the period of the fundamental frequency in the given section of a signal. They assume that there is at most one harmonic source in the observed short-time signal. Methods that estimate F0 from time-domain representations are, for example the extraction of F0 using Autocorrelation Function (ACF) or Average Magnitude Difference Function (AMDF). The techniques like cepstrum based F0 estimation [35], spectral autocorrelation [30] or harmonic matching [21] uses spectral-domain representations of the signal based on Fourier Transform [13]. These are not described in detail because an electric guitar as a music instrument can produce the sound compounded of the sounds from multiple harmonic sources (strings).

For the multi-F0 estimation, two approaches were selected. The first, spectrogram decomposition approach and the method called Probabilistic Latent Component Analysis (PLCA), decomposes the spectrogram into components that can be computed in advance from recordings of individual audio sources. PLCA is very useful for analysis of complex polyphonic sounds [3]. The second approach uses trained models based on artificial neural networks that estimates multi-F0s for given polyphonic records. Both of them are described in the following sections.

2.4.1 Spectrogram Factorization Approach

Probabilistic Latent Component Analysis

PLCA is a spectrogram factorization technique proposed by Smaragdis, Raj, and Shashanka [44]. In general, PLCA decomposes N-dimensional distribution into the specific number

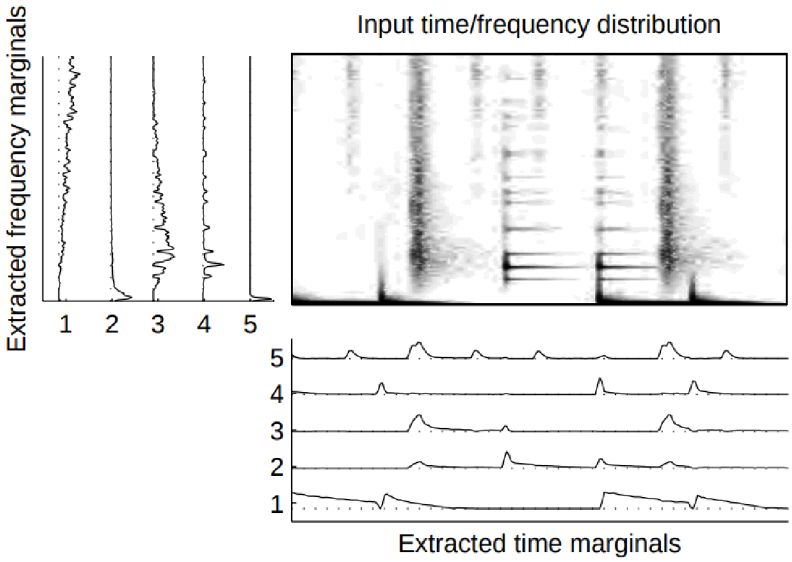


Figure 2.6: Application of PLCA on the shown spectrogram (the middle plot). Smaller plots on the left and bottom represent frequency and time marginals [44].

of basis. Using it on a spectrogram, it can yield a lot of desirable properties. In our case, two dimensional PLCA is used as multi-pitch estimator that decomposes spectrogram into matrices that represent spectral basis (frequency marginals) and their time activations (time marginals). This decomposition is well suited for music transcription because the spectrum of a tone and the time when it is played can be retrieved. It can be also used as a technique for audio source recognition or audio source separation.

The definition of PLCA is

$$P(\mathbf{x}) = \sum_z P(z) \prod_{j=1}^{N} P(x_j|z) \tag{2.11}$$

where $P(\mathbf{x})$ stands for the N-dimensional distribution of the random variable $\mathbf{x} = x_1, x_2, \ldots, x_N$, z is the latent variable or the weight distribution of a component, and $P(x_j|z)$ are one dimensional distributions. To decompose 2-dimensional spectrogram, 2-dimensional PLCA is used. There are two forms of 2-D PLCA: symmetric and asymmetric. The asymmetric form is described later in this chapter. The symmetric form is defined as

$$P(\omega, t) = \sum_z P(z)P(\omega|z)P(t|z) \tag{2.12}$$

where $P(\omega, t)$ is the 2-dimensional distribution (spectrogram), $P(z)$ represents the weight distribution of the latent variable (tone), $P(\omega|z)$ is the frequency marginal for the latent variable z and $P(t|z)$ is the time marginal for the latent variable z. Because PLCA has a statistical form, the input spectrogram has to be approximated as a probability distribution. The time and frequency marginals calculated from this distribution are also probability distributions. This statistical interpretation makes it useful for probabilistic framework applications [3].

Figure 2.6 shows the spectrogram (the large middle plot) as a time-frequency representation of signal that is decomposed into the matrix of frequency marginals (the smaller plot on the left) and the matrix of time marginals (the smaller plot at the bottom). The numbers of time and frequency marginals depend on the number of latent variables z . Each frequency marginal is coupled with one time marginal trough the z variable. In our case, one frequency marginal represents spectral base and time marginal represents a measure of how much the given spectral base contributes to the spectrum of the given time frame. If there is exactly one spectral basis to represent a tone, the z variable corresponds to the tone that is extracted from the spectrogram using the spectral basis $P(\omega|z)$. The form of the spectral basis depends on a learning process.

Asymmetric PLCA

On contrary to the symmetric PLCA, the asymmetric PLCA model decomposes a spectrogram into different components. It is more useful when trying to control the number of the components in a time frame [6]. Shashanka et al. [42] formulate asymmetric PLCA model as:

$$P(\omega, t) = P(t) \sum_z P(\omega|z) P(z|t) \tag{2.13}$$

where $P(\omega|z)$ are the spectral templates of the latent variable z, $P(z|t)$ are the time-varying component activations and $P(t)$ is the energy distribution of the input spectrogram $P(\omega, t)$. The asymmetric form of PLCA is not used in the proposed systems because the output of the symmetric PLCA is considered to be more suitable for further processing with a neural network as it is proposed in chapter 4.

Shift-Invariant PLCA

Shift-invariant PLCA is an extension of the PLCA model proposed by Smaragdis, Raj, and Shashanka [42]. The extended model is able to extract shifted structures from non-negative data. It is useful when the log-frequency representation of signal is used as an input because intervals between adjacent F0s are the same. The shift-invariant model is defined as following:

$$P(\omega, t) = \sum_z P(z) P(\omega|z) *_\omega P(\nu, t|z) \tag{2.14}$$

where ν is the pitch-shifting factor and z is the latent variable. The spectral template $P(\omega|z)$ is shifted across ω, producing the time-varying pitch impulse distribution $P(\nu, t|z)$ [7]. The shift-invariant PLCA model can be also expressed as:

$$P(\omega, t) = \sum_z P(z) \sum_\nu P(\omega - \nu|z) P(\nu, t|z) \tag{2.15}$$

by removing the convolution operator. This extension is not incorporated in the proposed systems because it extends the asymmetric form of PLCA which is also not utilized.

Parameter Estimation

The variant of Expectation-Maximization (EM) algorithm [17] is used to estimate the parameters of the model based on PLCA. It contains the Expectation step and Maximization step. During the estimation, we iteratively alternate between the steps and estimate the parameters until the calculation is stopped [44]. In the Expectation step, the contribution of the latent variable z is calculated as

$$R(\mathbf{x}, z) = \frac{P(z)\prod_{j=1}^{N} P(x_j|z)}{\sum_{z'} P(z')\prod_{j=1}^{N} P(x_j|z')} \tag{2.16}$$

In the Maximization step, this contribution $R(\mathbf{x}, z)$ is used to calculate the parameters $P(z)$ and $P(x_j|z)$:

$$P(z) = \sum_{\mathbf{x}} P(\mathbf{x})R(\mathbf{x}, z) \tag{2.17}$$

$$P(x_j|z) = \frac{\sum_{x_i, x_i \neq x_j} P(\mathbf{x})R(\mathbf{x}, z)}{P(z)} \tag{2.18}$$

If the input spectrogram $P(\mathbf{x})$ is an unnormalized histogram in opposed to density, each component $P(x_j|z)$ needs to be normalized to sum to one in every iteration. Otherwise, no normalization is needed during the parameter estimation.

Piano-roll retrieving

There are multiple ways how to retrieve piano-roll from PLCA's output. One of the simplest method is thresholding. In the of the symmetric PLCA, the output are time activities of all tones. By thresholding, the values of time activations that are higher than selected threshold are estimated to be sounding pitch. The threshold have to be adapted to the length of the input spectrogram because each time activity is normalized in time.

More complex method is note tracking using a Hidden Markov Model (HMM) [11]. Then, the states of the HMM correspond to the stages of a note (attack, decay, sustain, release, silence), which are constrained to occur in a fixed chronological sequence. In some cases, only two states (playing, silence) can be used. Using the Viterbi algorithm, the most likely state sequence is estimated. These sequences then determines parts where the pitch is active. The problem is estimation of fast repeated notes because the HMM attaches a high cost to starting a new note.

This way, more machine learning techniques can be stacked up to process the PLCA's output. In this work, processing with RNN is also used to filter out pitch time activities.

2.4.2 Deep Learning Approach

Another approach to note recognition is recognition with neural networks (NN). In this section, two basic types of neural networks will be described: feed-forward networks with focus on deep neural networks and recurrent neural networks with focus on Long Short-Term Memory method.

Deep Neural Network

Deep neural network (DNN) is a feed-forward network that is based on a multilayer perceptron. In feed forward networks, all outputs of the previous layer are connected to all weights of each neuron in subsequent layer and are often initialized using either an unsupervised or a supervised pretraining technique [18]. These layers are also called Fully Connected (FC) or Densely Connected (DC) layers. DNN's layers are divided into three categories: the first layer is called an input layer, the last layer is called an output layer and all layers

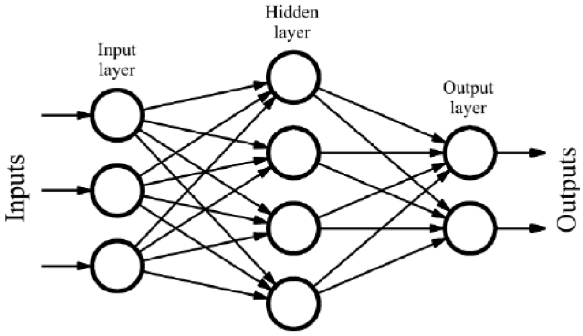


Figure 2.7: A concept of a feedforward architecture with the input, output and hidden layers [37].

between them are called hidden layers. A high number of hidden layers in combination with nonlinear activation functions allow a network can learn complicated relationships between the input and the output, which is not available with a single-layer network [12].

Figure 2.7 illustrates the concept of a feedforward architecture where the input, output and hidden layers are depicted.

Recurrent Neural Network

One of the disadvantages of feedforward networks is that their output is based only on their actual input. Let assume that the input of DNN is a time frame of a musical piece. DNN can learn a note is played in that time frame according to input features but it cannot learn anything about a context of that frame[1]. On contrary, Recurrent Neural Networks (RNN) are able to model temporal contexts due to the use of recurrent connections in the hidden layers [9]. These connections allow to "remember" sequence information in a hidden state of the recurrent neural network. Because of this advantage, RNNs were successfully applied in tasks like speech recognition [39] or piano music transcription [9]. Figure 2.8 illustrates a concept of the RNN.

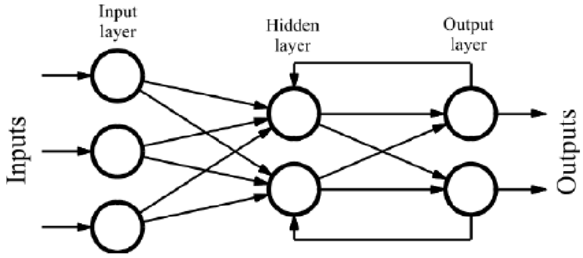


Figure 2.8: A concept of a recurrent network architecture with the input, output and hidden layers [37].

[1] An input of DNN can be multiple time frames, then we can talk about context of frame or frame stacking. But it also can be considered a longer time frame that misses information about its context. In this case the context is meant to be an audio sequence which the time frame is a part of.

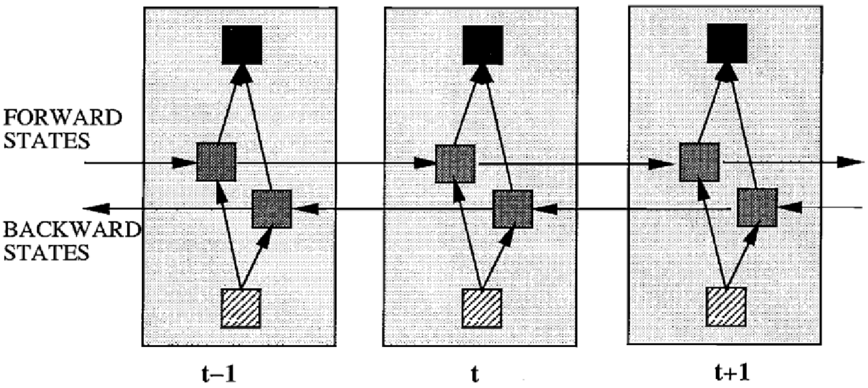


Figure 2.9: General structure of bidirectional recurrent neural network unfolded in three time steps [41].

Bidirectional RNN

Recurrent connections of neurons in the RNN provide a way of dealing with sequential data and allow to remember correlations between data that are close in the sequence [41]. To predict the output value y_{tc}, all the available input information up to the current time frame t_c (i.e. $\{x_t, t = 1, 2, \cdots, t_c\}$) are used. Usually, the input that comes later is also useful for prediction. This can be partially achieved by delaying the output by the certain number of time frames to include future information. However, the optimal delay is task-dependent and has to be found by the „trial and error“ method. Another approach is to use two separate RNN for both time directions and to merge their opinions. Generally, it is not clear how to merge network outputs in an optimal way. Therefore, more elegant solution would be desirable.

Bidirectional Recurrent Neural Network (BRNN) was designed to overcome these limitations [41]. The basic idea is to split the state neurons of a regular RNN in two parts. One part is responsible for the positive time direction and the other for the negative time direction. Input information in the past and the future of the currently evaluated time frame can be used to minimize the objective function without the need for delays. Figure 2.9 shows the general structure of the bidirectional recurrent neural network unfolded in three time steps.

Long Short-Term Memory

In basic recurrent neural networks, it is difficult or impossible to store information over extended time intervals. It is mainly due to blowing up or vanishing error signal during training with Back-Propagation Through Time (BPTT) [48]. Long Short-Term Memory (LSTM) was designed to solve this problem [26]. In combination with BPTT, constant error flow through internal states of the LSTM units is achieved. It can learn to bridge time intervals in high number of steps. Figure 2.10 shows the LSTM unit with the input i, output o and forget f gates. They can be trained like regular neurons. Thus, i_t, o_t and f_t represents activations of the input, output and forget gates in the time step t. The output of the block h_t as well as the activations of the others gates also consider an activation of a memory cell c in the time step $t-1$. The × symbol denotes element-wise multiplication

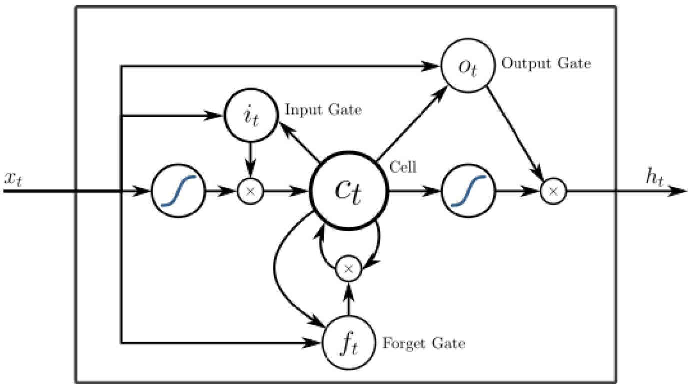


Figure 2.10: The LSTM unit.

between its inputs and a symbol similar to the $\int$ symbol denotes differentiable function (usually *sigmoid* or *tanh*) [23].

Piano-roll retrieving

Output from the ANN has to be processed to determine when pitches are active or not. On contrary to PLCA, values of the output do not depend on the size of the input but they depend on selected activation function of an output layer. The simplest method, used also in this work, is thresholding. A threshold can be estimated globally (one threshold for all tones) or locally (one threshold for each tone). Similarly to PLCA, note tracking with HMM or other machine learning techniques can be applied on the ANN's output.

2.5 Evaluation metrics

To compare performance of the transcription systems, metrics had to be chosen. There are no standardized metrics for evaluation of polyphonic music transcription. The metrics used to evaluate participating transcription systems in MIREX are used. These metrics are divided into two groups: frame-level and note-level.

2.5.1 Frame level

In this type of evaluation, the metrics are calculated frame-wise. Every 10ms, active pitches are observed in the output of the evaluated system and in the ground-truth data. With $N_{tp}[t]$ being the number of correct pitches in the t-th frame, $N_{fp}[t]$ being the number of retrieved pitches that do not occur in the ground-truth and $N_{fn}[t]$ being the number of pitches that occur in the ground-truth but were not retrieved by the system, the metrics that are used to evaluate the systems are the following:

$$Precision = \frac{\sum_t N_{tp}[t]}{\sum_t N_{tp}[t] + N_{fp}[t]} \tag{2.19}$$

$$Recall = \frac{\sum_t N_{tp}[t]}{\sum_t N_{tp}[t] + N_{fn}[t]} \tag{2.20}$$

$$Accuracy = \frac{\sum_t N_{tp}[t]}{\sum_t N_{tp}[t] + N_{fp}[t] + N_{fn}[t]} \tag{2.21}$$

$$F\text{-}Measure = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{2.22}$$

Precision depicts the portion of correct retrieved pitches for all pitches retrieved in each frame, *Recall* is the ratio of correct pitches to all ground-truth pitches, *Accuracy* is then the ratio of correct pitches to all ground-truth pitches and retrieved pitches. A returned pitch is considered to be correct if it is within a half semitone of the ground-truth pitch [19].

2.5.2 Note level

Modified *Precision*, *Recall*, *Accuracy* and *F-Measure* metrics are used. A ground truth note is assumed to be correctly transcribed if the transcription system returns a note that is within a half semitone, the returned note onset and offset is within a 100 ms range(+-50 ms) of the onset and offset of the ground truth note. In this work, offset matching is omitted due to the difficulty of offset detection for electric guitar. The metrics are defined as:

$$Precision = \frac{N_{tp}}{N_{tp} + N_{fp}} \tag{2.23}$$

$$Recall = \frac{N_{tp}}{N_{tp} + N_{fn}} \tag{2.24}$$

$$Accuracy = \frac{N_{tp}}{N_{tp} + N_{fp} + N_{fn}} \tag{2.25}$$

$$F\text{-}Measure = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{2.26}$$

where N_{tp} is the number of correctly transcribed notes, N_{fp} is the number of retrieved notes that do not occur in the ground-truth, and N_{fn} is the number of the notes that occur in the ground-truth but were not retrieved by the system.

2.6 State of the Art

2.6.1 SONIC: Transcription of Polyphonic Piano Music

The transcription system called *SONIC* is a system that participated in Music Information Retrieval Evaluation eXchange[2] (MIREX) 2016 and achieved the best results in multiple fundamental frequency estimation and tracking [33]. MIREX is an event where systems for music analysis are evaluated on the same data and compared with each other. Despite of being designed for piano music transcription, it is evaluated on guitar music data and results are compared with the results of the proposed systems.

SONIC uses a partial tracking technique based on an auditory model and adaptive oscillator networks followed by a Time-Delayed Neural Network (TDNN) for transcription of piano music. It also incorporates an onset detector that is not described in this chapter.

[2] http://www.music-ir.org/mirex/wiki/MIREX_HOME

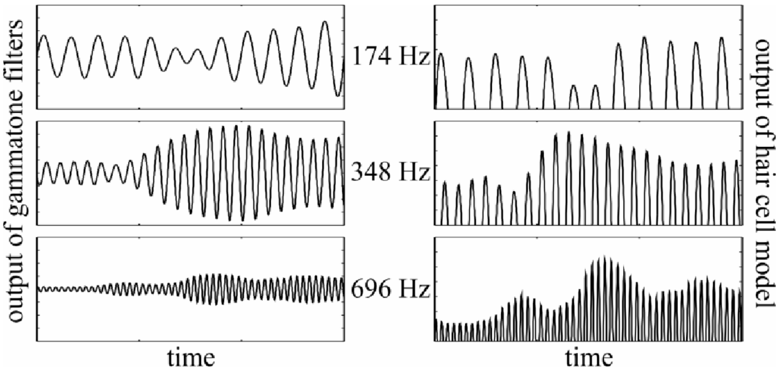


Figure 2.11: Analysis of three partials of the piano tone F3 with the auditory model [33].

Partial tracking

As it was stated in 2.1, pitched sounds produced by music instruments can be described by a series of peaks, also called partials. Partial tracking systems then uses time-frequency representation of musical signal and tracks these partials in time. The authors of the *SONIC* system proposes a new partial tracking technique based on an auditory model which emulates the functionality of human ear and on adaptive oscillators that extract partial tracks from outputs of the auditory model.

The auditory model consists of two stages. A filterbank is first used to split the signal into several frequency channels. These frequency channels model the movement of basilar membrane in the inner ear. The filterbank consists of an array of bandpass IIR filters, called gammatone filters [43]. 200 filters are used with logarithmically spaced center frequencies from 70 to 6000 Hz. In the second stage, the output of each gammatone filter is processed by the Meddis' model of hair cell transduction that produces a probabilistic representation of firing activity in the auditory nerve [34]. Figure 2.11 shows the output of the auditory model for three partials of piano tone F3.

Subsequently, the adaptive Large-Kolen oscillators are used to detect periodicity in frequency channels of the auditory model [31]. After the oscillators are synchronized with the input signal, it indicates that a partial with frequency equal to that of the oscillator is present in the input signal. Each oscillator has 2 parameters: phase and frequency. Synchronization of the oscillator is a process of adjusting its phase and frequency to match that of the input signal. Figure 2.12 shows 3 examples of partial tracking with oscillators. Example A shows case of tracking a 440 Hz sinusoid, example B shows how two oscillators with initial frequencies set to 440 and 445 Hz synchronize to a sum of 440 and 445 Hz sinusoids (5 Hz beating) and example C presents the tracking of a frequency modulated 440 Hz sinusoid. In all cases, the oscillators are synchronized successfully.

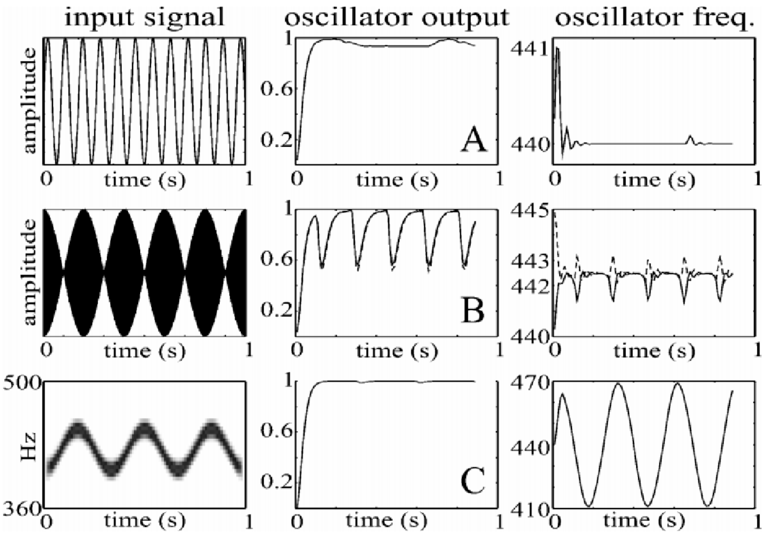


Figure 2.12: Partial tracking with adaptive oscillators [33].

To improve the partial tracking technique, adaptive oscillators are grouped into networks. Networks consist of up to 10 interconnected oscillators with the initial frequencies set to integer multiples of the frequency of the first oscillator in a network. 88 oscillator networks are used in the *SONIC* system and the initial frequency of the first oscillators in all networks are set to the fundamental frequency of each of tone from A0 to C8. Consequently, all other oscillators in networks track partials of tones. This technique produces a clearer representation of the signal than techniques like STFT or oscillator themselves.

Note Recognition

Set of neural networks is used to recognize notes from the partial tracking model. Each of 76 TDNNs are trained to recognize single note [47]. The lowest octave (A0-Ab1 inclusive) was omitted because of poor recognition results. Because each network recognize only one tone, it has only one output that is thresholded to determine if observed tone is or is not played.

2.6.2 Tablature Transcription System (IDMT)

Work of Christian Kehling [27] presents algorithm for automatic transcription and parameter extraction from isolated polyphonic guitar recordings. It retrieves general information such as pitch, onset and offset but also information such as plucked string, plucking and expression styles. Figure 2.13 shows schematic model of the proposed system. Only multipitch estimation and partial tracking will be described in this section because other parts are not related with the previously described techniques.

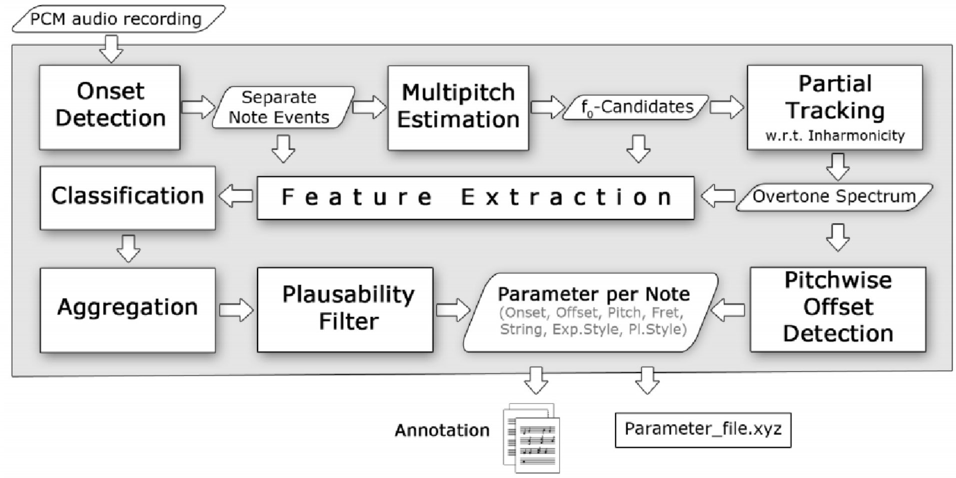


Figure 2.13: Schematic model of the analysis framework [27].

Multipitch Estimation

After an input recording is segmented using onset detection algorithm, reassigned magnitude spectrogram based on the Instantaneous Frequency (IF) [1] representation in addition to the conventional time-frequency transform is computed for each segment. IF magnitude spectrogram with a logarithmically spaced frequency components is used as input for Blind Harmonic Adaptive Decomposition (BHAD) [20]. The framework on which the BHAD relies is the PLCA. Using the note and noise model (described in [20]), the input spectrogram is decomposed in note and noise components.

Partial Tracking

The fundamental frequency and 15 partials of each note event retrieved from the pitch estimation algorithm are tracked over time. Simple peak picking is applied on magnitude spectrum of each time frame. The spectral peaks are then assigned to harmonics of different fundamental frequency candidates by minimizing the distance between the ideal harmonic frequency positions and the detected peak positions. This way, notes played by bending or sliding a string can be easily detected and their pitch can be accurately estimated.

Chapter 3

Data

3.1 Training Dataset

All the systems based on PLCA or neural networks have to be trained before they are used for music transcription. For this task, training data has to be carefully selected, resp. generated.

3.1.1 Data for PLCA

As it was stated in chapter 2, spectral basis for spectrogram decomposition can be learned in advance to improve the pitch estimation. Only one spectral base is used to represent a pitch. To extract the spectral basis for each pitch, recordings of isolated notes are used.

The sound samples from the sound fonts[1] of FS Collection 3[2] were extracted and used for training of the spectral bases. FS Collection 3 is the collection of sound fonts released by FlameStudios under the GNU GPL license. It contains the following sound fonts:

- FS Fender Jaguar Electric Guitar Both Pickups Direct In
- FS Fender Telecaster Electric Guitar Bridge Pickup Direct In
- FS Fender Telecaster Electric Guitar Neck Pickup Direct In
- FS Gibson Les Paul Bridge Pick-Up Electric Guitar Direct In
- FS Gibson Les Paul Neck Pick-Up Electric Guitar Direct In

The names of the sound fonts describe the setting of recordings. First, the name of the guitar model is noted. Three guitars were recorded: Fender Jaguar, Fender Telecaster, and Gibson Les Paul. Next, used pick-ups are denoted (neck, bridge, or both) and the „Direct In“ denotes that the guitar was plugged directly to the recording device during recording (no effects applied).

The audio files were extracted using the sound font decompiler F2Comp[3]. Each sound font consists of audio files for 45 pitches – from note E2 (MIDI number 40) to note C6 (MIDI number 84), inclusive. For each pitch, there are three WAV files with different dynamics of recorded note. They were sampled with 44.1 kHz sampling rate and with a bit depth 16

[1] https://www.digitalsoundfactory.com/what-soundfont

[2] http://www.flamestudios.org/free/Soundfonts

[3] http://www.hammersound.net/mirrors/last_night/sf.htm

Bit. The files are from 5 to 40 seconds long (lower notes sound longer) but only the first 2 seconds from each note recording are used for training. Therefore, the training recording for each note lasts 30 seconds and consists of 15 notes played on different guitars and with different volumes. It improves the ratio of different stages (attack, decay, sustain) of the notes in the training dataset. It also resulted in higher accuracy during early experiments.

3.1.2 Data for neural network

Neural networks are trained on labeled input samples. In this case, the input for DNN are spectra of several time frames and the input for LSTM are 10 seconds long spectra sequences. The most suitable data would be labeled recordings of a real guitar. It is too difficult and time-consuming to create a dataset like this. Instead, synthesized songs were used to train the neural network models.

The training dataset consists of 35 synthesized MIDI songs downloaded from Classical Guitar MIDI Archives[4] and the converted .gp files from the guitarprotabs.org[5] page with the tuxguitar software[6]. The complete list of MIDI songs used for the training dataset can be seen in table 3.1. FluidSynth[7] was used to synthesize MIDI files to WAV files, using all sound fonts from the PLCA training dataset with the same parameters. This results in 525 generated songs (5 guitar, 3 volumes). Due to the nature of the MIDI protocol, the MIDI files served as the labels for synthesized files.

Table 3.1: MIDI songs used for training dataset. The Trans value determines the number of semitones that the song was transposed by.

Artist	Song	Note events	Tones	Song duration [s]	Total notes duration [s]	Trans
ACDC	Hells Bells	1182	29	292	371	0
ACDC	Highway To Hell	1413	13	179	507	18
Dionisio Aguado	Douze Valses No. 1	464	23	71	170	1
Ludwig van Beethoven	Sonata No.8 in C minor	1204	32	87	195	0
Bullet For My Valentine	All these things I hate revolve around me	2217	20	223	484	0
Bullet For My Valentine	End of days	2025	20	253	297	0
John Frusciante	Wind Up Space	855	10	124	368	18
John Butler Trio	Betterman	1339	25	162	334	0
John Mayer	Come Back To Bed	1798	27	306	1035	2
John Mayer	Everyday I Have The Blues	1100	35	228	315	1
Robert Johnson	Crossroad Blues	529	24	78	144	0
Robert Johnson	Drunken Hearted Man	199	23	38	67	0
Led Zeppelin	Gallows Pole	3454	17	317	809	1

[4] http://www.classicalguitarmidi.com/

[5] https://guitarprotabs.org/

[6] http://www.tuxguitar.com.ar/

[7] http://www.fluidsynth.org/

Led Zeppelin	Stairway To Heaven	1935	21	350	772	-1
John Mayer	No Such Thing	2954	24	204	635	1
Johann Kaspar Mertz	Lob Der Tranen	873	34	181	536	2
Pat Metheny	Bright Size Life	653	36	141	124	0
Luis Milan	Two Part Fantasia	484	18	444	777	20
Francesco Molino	Second Nocturne	2683	30	471	677	0
Santiago de Murcia	Paspied Nuovo	364	17	59	127	0
Pearl Jam	Even Flow	1426	19	257	357	17
Pearl Jam	Garden	1462	20	288	416	0
John Petrucci	Hollow Years	1133	29	330	451	1
Django Reinhardt	Blue Drag	249	29	65	54	0
Django Reinhardt	Minor Swing	219	19	70	52	0
Sabicas	Castellana	1382	28	221	414	-1
Julio Salvador Sagreras	Magdalena	720	35	202	632	1
Domenico Scarlatti	Sonate K18/L.416	896	26	247	275	19
Robert Alexander Schumann	Träumerei	248	28	76	113	0
John Scofield	Protocol	955	35	178	129	0
Vincente Emilio Sojo	Niño Lindo	371	22	81	151	0
Alexandre Tansman	In Modo Polonico Kolysanka N.1	617	32	99	268	0
Steve Ray Vaughan	Little Wing	2695	41	388	534	0
Robert De Visée	Overture de la grotte de Versailles	926	23	191	472	1
Silvius Leopold Weiss	Sarabande (from Suite No.2)	463	25	149	215	0

Transposition of training dataset

To create efficient training dataset, it has to be balanced. In this case, it means that the number of samples for each note should be roughly the same. This condition is not satisfied for unmodified MIDI songs. Because of they were selected randomly and most of them are just in a few keys, the differences between the numbers of the note samples for different notes are too high. This problem is partially solved by transposition of some of the training MIDI songs. Several MIDI songs were chosen and pitches in their MIDI events were transposed. The intervals between notes and their durations are preserved, the key and the pitch of the notes are changed. Figure 3.1 shows the histogram of note samples (100ms of note duration) of the original and transposed training dataset.

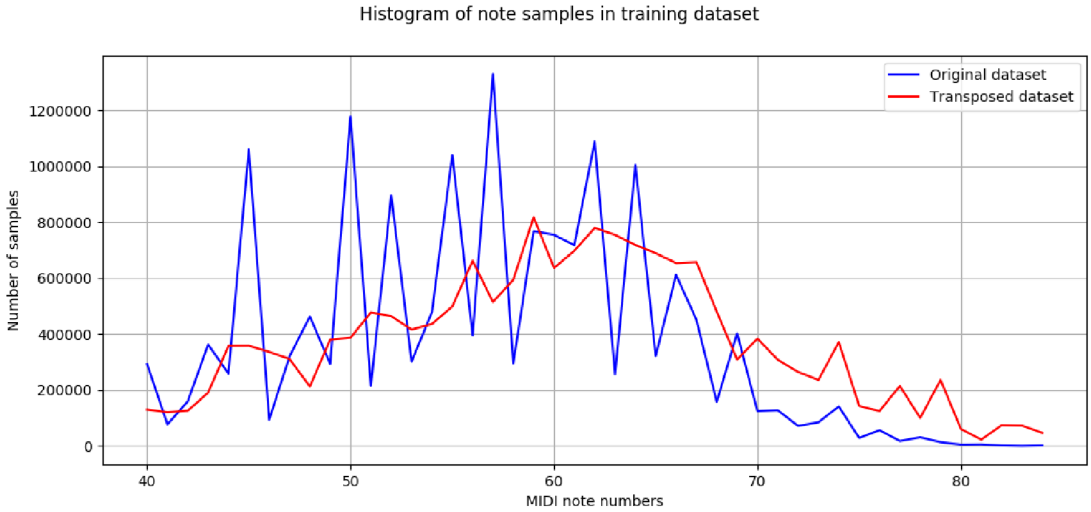


Figure 3.1: The histograms of note samples in the original and transposed training dataset. One sample represents 100ms of a note duration.

3.2 Validation Dataset

The validation dataset serves for evaluation of neural network models after each epoch. The accuracy obtained on this data shows the performance of a model during training session. Also, thresholds for output values of each neural network model are trained on the validation dataset.

Similarly to the training dataset for neural network models, the validation dataset was created by synthesizing 5 MIDI songs with the samples of 5 different guitars and 3 different volumes. Detailed information about the MIDI songs of the validation dataset can be seen in table 3.2.

Table 3.2: MIDI songs used for validation dataset. The Trans value determines the number of tones that the song was transposed by.

Artist	Song	Note events	Tones	Song duration [s]	Total notes duration [s]	Trans
Dionisio Aguado	Morceaux Agréables Pour Guitare Contredanse	379	25	129	177	0
Creedence Clearwater Revival	Fortunate Son	1171	16	128	270	0
Ed Sheeran	Give Me Love	2771	17	318	805	0
Hammerfall	Heeding the Call	186	10	36	51	0
Steve Ray Vaughan	Pride and Joy	1834	34	215	360	0

3.3 Evaluation Dataset

The IDMT-SMT-Guitar database[8] was selected as an evaluation dataset. It is a large database designated for automatic guitar transcription created by Christian Kehling, Andreas Männchen, Arndt Eppler (Fraunhofer IDMT). Seven different guitars and various pick-up settings were used to ensure efficient diversification. The recordings are provided in one channel RIFF WAVE format with 44100 Hz sample rate.

The dataset consists of 4 subsets:

- Dataset1
  - bit depth of 24 bit
  - 2 different guitars
  - single note and chord recordings
- Dataset2
  - bit depth of 16 bit
  - 3 different guitars
  - twelve licks of monophonic and polyphonic parts varied in playing techniques and used guitar
- Dataset3
  - bit depth of 16 bit
  - 1 guitar
  - five short musical pieces
- Dataset4
  - bit depth of 16 bit
  - 3 different guitars
  - 64 short musical pieces grouped into the 8 genres

Dataset4 is not used for evaluation because the labels contain description of chords instead of single notes. For more detailed description of the evaluation dataset, see the IDMT-SMT-Guitar database page. Table 3.3 shows summary of the evaluation dataset.

Table 3.3: Summary of the evaluation dataset.

	Note events	Tones	Total recording duration [s]	Total notes duration [s]
Dataset 1	796	37	973	1778
Dataset 2	4881	48	3238	3198
Dataset 3	90	30	61	100

[8] https://www.idmt.fraunhofer.de/en/business_units/m2d/smt/guitar.html

Chapter 4

Note recognition systems

This chapter proposes and describes 4 systems for automatic music transcription of electric guitar recordings. The first system (*PLCA2MIDI*) is based on spectrogram factorization (PLCA). It decomposes an input spectrogram in time pitch activations using pretrained spectral basis. The second and the third system (*DNN2MIDI* and *LSTM2MIDI*) detect active pitches with trained neural networks. The last system (*HYBRID2MIDI*) combines both approaches.

All systems were implemented in Python 3[1]. Keras[2] was used to create neural networks, LibROSA[3] provided implementation of CQT and MIDO[4] was used to handle MIDI file parsing. Whole development environment with the manual are described in appendix B.

4.1 PLCA2MIDI

4.1.1 Preprocessing

To obtain a spectrogram from an input recording, both methods mentioned in chapter 2 were used to be able to compare the impact on performance of the PLCA2MIDI system.

Before a time-frequency representation is calculated, some modifications of an input music signal are done. The *PlCA2MIDI* system and all other proposed systems are designed to process only the input audio files with 11025 Hz sampling frequency. Therefore, if this condition is not met, the input files have to be resampled before they are processed.

If the input music recording has more channels, only one channel is generated by averaging all channels of the original recording. Then, the recording is divided into 60-second long segments like in [5]. It reduces the time needed for spectrogram factorization with PLCA. Finally, a Gaussian noise is added to prevent zero values in spectrogram because in further calculation, they might appear in denominators of fractions.

4.1.2 Frequency resolution issues

It seems that constant-Q transform (CQT) is more suitable for music signal analysis because it produces the spectrum where the frequency bins are geometrically spaced. Though, it has low time resolution with the parameters that ensure sufficient frequency resolution for

[1] https://www.python.org/

[2] https://keras.io/

[3] https://librosa.github.io/librosa/index.html

[4] https://mido.readthedocs.io/en/latest/#

the lowest analyzed fundamental frequencies. The length of the longest analysis window N_{max} used in CQT is calculated as:

$$N_{max} = \frac{qf_s}{f_{min}(2^{\frac{1}{B}} - 1)} \tag{4.1}$$

where f_s is the sampling rate, f_{min} is the lowest analyzed frequency, B is the number of bins per octave and q is the filter scale factor where the constant $Q = q/(2^{\frac{1}{B}} - 1)$ (derived from eq. 2.9) [40]. Let us assume that f_s is 11025 kHz, f_{min} is 82.4 Hz (fundamental frequency of E2), q is 1 and 24 frequency bins per octave are used. Then, the length of the analysis window is 4566 samples. Such long analysis window causes poor performance of the system in detection of short notes. The time resolution can be increased by lowering the frequency resolution. The first option is lowering the number of frequency bins per octave. In the case we lower it from 24 to 12, the size of the biggest analysis frame would be 2250 samples. The second option is lowering the filter scale factor. If the factor would be lowered from 1 to 0.5, it would have the same effect like using the first option.

In addition to STFT and CQT, we have also experimenting with a third time-frequency representation. Firstly, the STFT magnitude spectrogram is calculated from music signal. Then, this spectrogram is filtered with a semitone filterbank. The same technique was used in [9]. The filterbank is a set of triangular passband filters with the center frequency on fundamental frequencies of semitones. The first passband's center frequency is a frequency of the E2 tone (82.41 Hz). Each next center frequency f is calculated as $f = f_p \cdot 2^{\frac{1}{12}}$ where f_p is the center frequency of the previous passband filter. That many passband filters are calculated until the center frequency of the next one reaches the half of the sampling frequency (5512 Hz). Area of each filter is normalized to 1 to prevent the overemphasis of high frequencies with broad filters.

Figure 4.1 shows the three spectra of note E2. The upper plot shows the spectrum produced by CQT with different values of the filter scale factor. It can be seen that the lower the filter scale factor is, the lower frequency resolution the spectrum has. The middle plot shows the spectrum produced by DFT and the lower plot shows the filtered spectrum (DFT) with the semitone filterbank. The advantage of the first and the third spectra is that their frequency components are logarithmically spaced which corresponds with spacing of fundamental frequencies of tones. Also, a spectrogram produced by filtered STFT has both advantages of the previous spectral analysis - logarithmically spaced frequency components and good time resolution.

According to the results obtained on validation data with the 11025 Hz sampling frequency, the appropriate parameters for STFT and CQT were chosen. The window length for STFT is 1024 samples and the shift between two adjacent frames is 128 samples. The first and the last frequency components of STFT spectrogram are omitted because they do not carry useful information for further note recognition. The minimal frequency for CQT calculation is 70 Hz, the number of bins per octave is 24, the total number of bins of a spectrum is 144, and the filter scale factor is 0.7.

Before the spectrogram is decomposed by PLCA, it has to be normalized to be a probability distribution. This is achieved by dividing the whole spectrogram with the sum of all its values.

4.1.3 PLCA Note Recognition

Active pitches are retrieved from the input spectrogram with PLCA. PLCA, described in section 2.4.1, decomposes the input spectrogram on frequency and time basis. The

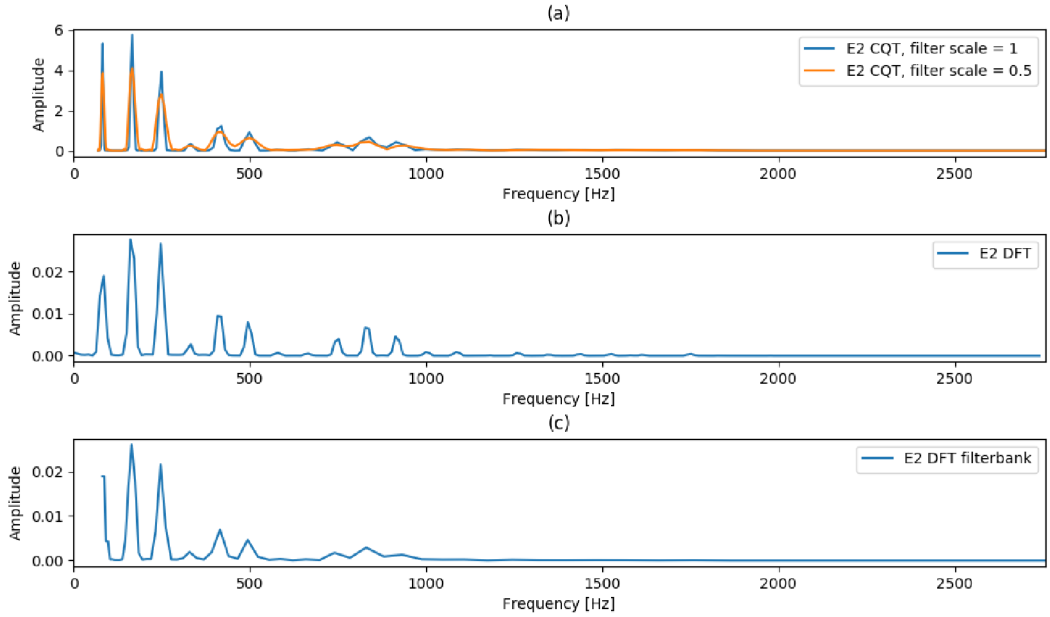


Figure 4.1: Spectrum of note E2. The upper plot (a) shows the two spectra produced by CQT with different filter scale factor. The middle plot (b) shows the spectrum calculated with DFT and the lower plot shows filtered spectrum of DFT with the semitone filterbank. The x axis of all plots are linear but in the case of the plot (a) and (c), it can be seen that the higher frequency components are more coarsely spaced than lower frequency components (their components are on a logarithmic scale).

frequency bases are obtained in advance by supervised learning. The time bases denote pitch activations in time and are used to create a piano-roll.

The learning process consists of decomposition of spectrograms of isolated note recordings. Despite of the recordings contains only the sound of a single note, its spectrum changes overtime according to the note stages (see section 2.1). The biggest differences are between the spectrum of the attack stage and the spectrum of the rest. For this reason, the spectrograms are decomposed in two frequency and time marginals. The final spectral base is calculated as the average of the two frequency marginals. Figure 4.2 shows two trained frequency basis spectra of the tone F2. The spectrum of the tone's attack stage (orange line) has energy more spread over all frequencies while the spectrum of decay stage has a lot more energy in frequency components of harmonics.

When spectral basis of all pitches are known, they are used to decompose an input spectrogram. Expectation-Maximization (EM) algorithm is used to obtain time marginals. Though, the frequency marginals are not updated during the maximization step. The number of iterations of the algorithm was determined by observing the value of Kullback–Leibler (KL) divergence that measures how one probability distribution diverges from another. It is defined as:

$$D_{KL}(P||Q) = \sum_{i} P(i) \log \frac{P(i)}{Q(i)} \tag{4.2}$$

where $P(i)$ is the spectrum of i-th frame of the estimated spectrogram and $Q(i)$ is the spectrum of i-th frame from the input spectrogram [32]. After 50-60 iterations, the values

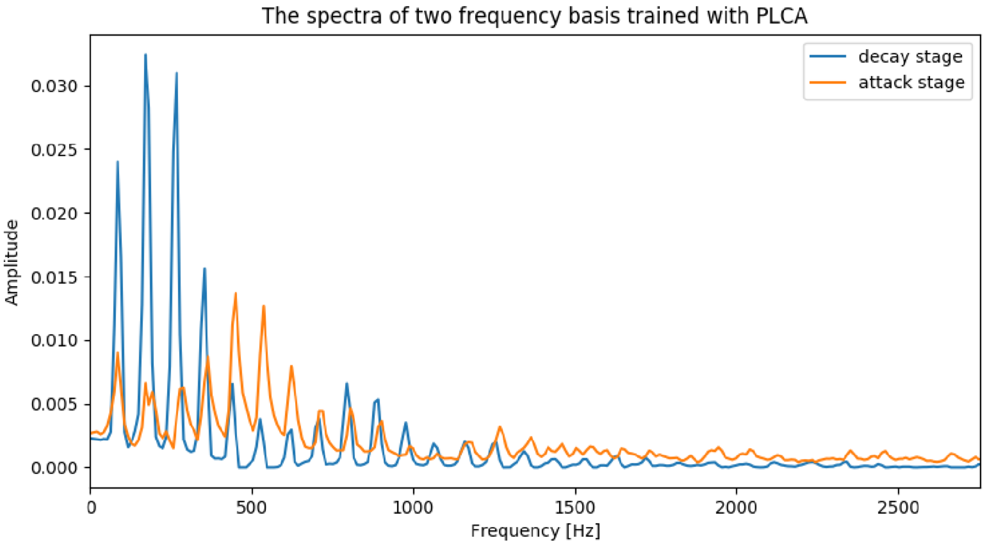


Figure 4.2: Two spectra of trained frequency basis with PLCA.

of KL divergence differ by a negligible value. Therefore, the number of iterations was set to 70 as in [5].

4.1.4 Post-processing

PLCA produces time basis for each tone. Tone's time base consists of values that represent the measure how much the corresponding frequency base contributes to the original spectrum of a specific time frame. As the input spectrogram is always normalized to sum to 1, all produced time basis are normalized. A simple thresholding method is used to recognize a note according to values of the time bases. Because the values of the time bases are dependent on the length of input spectrogram, a value of the threshold has to be adapted to that. The value of the threshold is 15 divided by the size of the output piano-roll (45 notes × number of time frames) according to [5]. Figure 4.3 shows the reference piano-roll of the nocturneNr2.wav recording from the evaluation dataset (a), the piano-roll produced by PLCA on the corresponding recording (b) and the piano-roll after thresholding (c).

After the "soft" piano-roll (piano-roll with real values) is thresholded, notes that are 1 time frame long can appear in the result. Such long notes are considered to be wrongly recognized because it is arguably impossible to play them. Therefore, they are removed from the resulting piano-roll.

4.2 DNN2MIDI

The *DNN2MIDI* system is based on a feedforward Deep Neural Network (DNN) (described in 2.4.2).

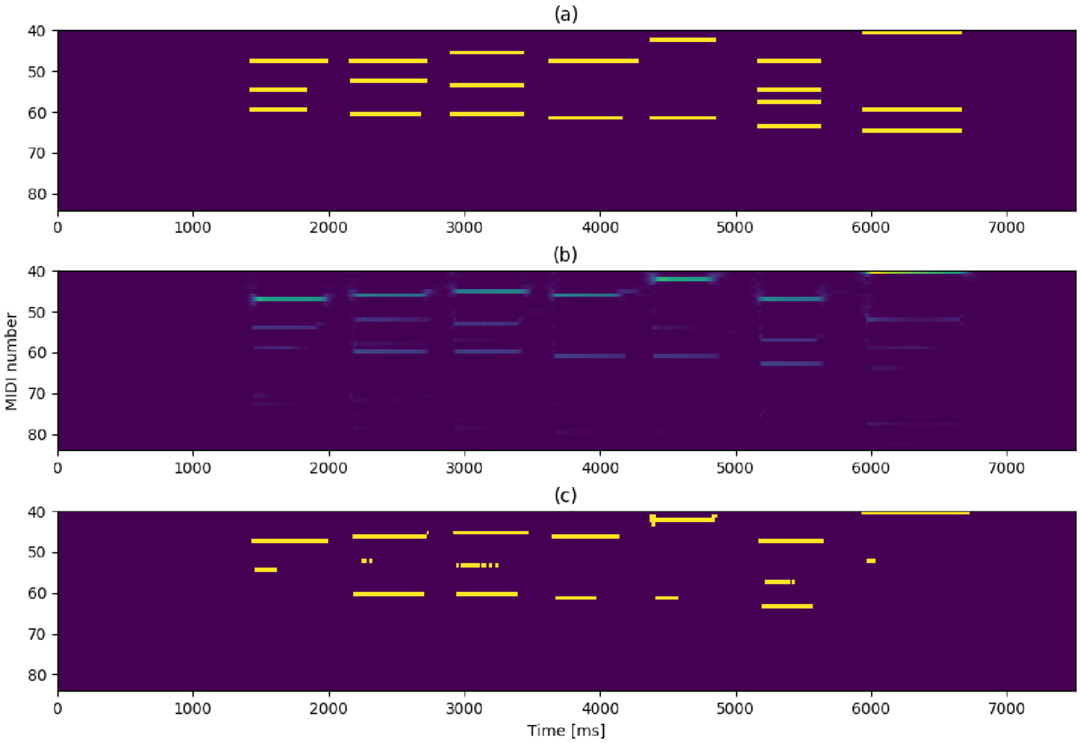


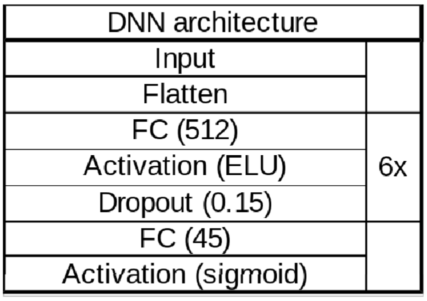
Figure 4.3: The reference piano-roll of the `nocturneNr2.wav` recording from the evaluation dataset (a), the piano-roll produced by PLCA from the same recording before thresholding (b) and the thresholded piano-roll (c).

4.2.1 Preprocessing

The same audio preprocessing and time-frequency representations of input signals that are used for the *PLCA2MIDI* system are used (more details in 4.1.1). The DNN, in opposite to PLCA, does not process whole spectrogram at once. Thus, it has to be further processed.

In early experiments with the DNN, there were problems with training due to incorrect data normalization methods. After numerous methods were tried, the way how to properly scale values of input spectrograms to prevent to weights of the network to blow up was found: the mean and standard deviation are estimated in advance from the training dataset for each frequency component of a spectrum. After that, the means of frequency components are subtracted from each frame of the input spectrogram and these frames are divided by the standard deviations (mean and variance normalization). No batch normalization layers are then needed in the network.

Note recognition with neural networks is considered as a multi-label multi-class classification. There are multiple classes (45 tones) which input data can be classified into and one input sample can be classified into multiple classes (0–6 simultaneously played tones). As the input sample, 19 time frames of the input spectrogram are selected. For each time frame of the spectrogram, 9 adjacent time frames are concatenated from both sides. With the 128 samples shift size and the 11025 Hz sampling frequency, it covers the 220 ms of an input recording. Only notes that are played in the middle time frame are used as a label for a supervised training. This method provides a small temporal context for classification which improves the performance of the system.



DNN architecture	
Input	
Flatten	
FC (512)	6x
Activation (ELU)	
Dropout (0.15)	
FC (45)	
Activation (sigmoid)	

Figure 4.4: Sequence of layers of the DNN network from top to bottom. The 6x symbol denotes that the whole block (FC (512), Activation (ELU) and Dropout (0.15)) is 6 times sequentially repeated.

The last step of data preprocessing for training is shuffling the input samples. Due to the size of the training dataset, only 5 randomly selected files from the dataset are loaded at once, then the context frames are concatenated to each frame and these input samples are shuffled.

4.2.2 DNN Architecture

Deep neural network for note recognition is simply designed as 6 Fully Connected (FC) hidden layers, each containing 512 neurons. Figure 4.4 shows the architecture of the network. “Flatten” layer is used before the fully connected layers to create one-dimensional vector from two-dimensional input sample. There are activation and dropout layers after each of the hidden layers. The activation layers apply the exponential linear unit (ELU) activation function on output of each neuron of the previous FC layer [14]. Higher accuracies were obtained with the ELU function than with the mostly used Rectified Linear Unit (ReLU) function. The dropout layers are used as a regularization to prevent overfitting of the network. Their dropout rate is set to 0.15. The output layer is again FC layer but only with 45 neurons. Each output neuron corresponds to one recognized tone. Activation function of the output layer is set to the *sigmoid* function instead of the mostly used *softmax* function that normalizes all output values to sum to one and is determined to single label classification. The output values are then near zero when tone is not played or up to one if the tone is played in a time frame.

4.2.3 Training

The training dataset described in 3.1.2 was used to train the network. Stochastic gradient descent (SGD) algorithm was used as a training algorithm with the initial learning rate set to 0.0001. The learning rate was continually decreased with a learning scheduler similar to the Newbob scheduler.

Training was divided into several training sessions. Each session was composed of 60 training epochs. After each epoch, the frame-level accuracy (described in 2.5) is calculated on the validation dataset (described in 3.2). The learning rate scheduler decreased the learning rate value when the accuracy on the validation data did not increase for 10 epochs. When the learning rate was decreased, also the network with weights that achieved the best accuracy was loaded. The first training session was executed with the initial learning rate set to 0.0001. The second training session had the same parameters except the initial learning rate that was set to 0.00005. The last training session was executed with the learning rate set to 0.00001.

4.2.4 Calibration

As well as the system based on PLCA, the network produces piano-roll with real values. It is needed determine if the notes are played or not according to these values.

The resulting piano-roll is produced by thresholding the output of the network. A global threshold (one threshold for all tones) is determined trough a process of calibration. The frame-level accuracy is calculated on the validation dataset for each threshold in range from 0 to 1 with step 0.1. The threshold value that provides the highest accuracy is saved. After that, the values of adjacent thresholds are taken and the new threshold is calculated from the range of these adjacent thresholds with step 0.01. The threshold value calculated this way is also used for evaluation. Fig 4.5 shows the reference piano-roll of the nocturneNr2.wav recording from the evaluation dataset (a), the piano-rolls produced by the DNN before (b) and after thresholding (c).

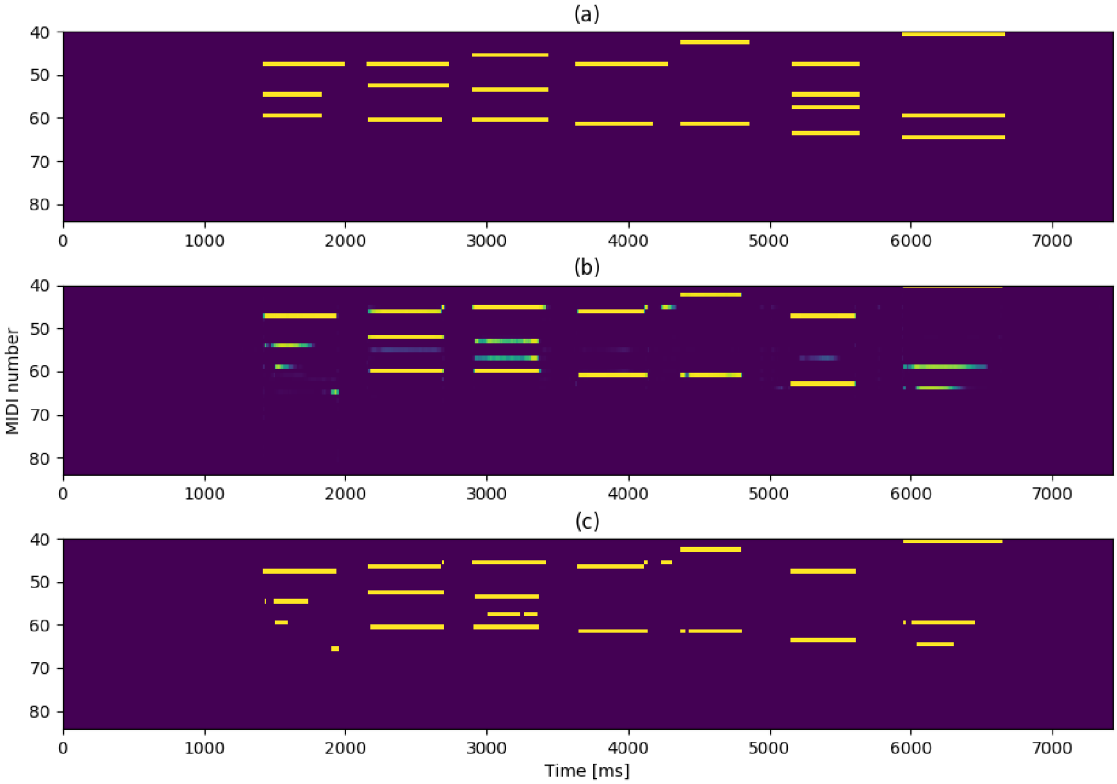


Figure 4.5: The reference piano-roll of the `nocturneNr2.wav` recording from the evaluation dataset (a), the piano-roll produced by the DNN from the same recording before thresholding (b) and the thresholded piano-roll (c).

4.3 LSTM2MIDI

For the *LSTM2MIDI* system, a recurrent neural network is used. As it is described in 2.4.2, recurrent neural networks are able to learn the temporal context from input sequences because of their recurrent connections that can be regarded as memory of the network. This ability is very convenient in case of a music analysis.

4.3.1 Preprocessing

Input samples for the *LSTM2MIDI* system are sequences of time frames of the recordings. On contrary to classification of DNN where one input sample (several stacked time frames are considered as one input sample) is classified as one output label, the recurrent neural network classifies N time frames of input sequence to N output label vectors where N can be different integer for different inputs. For training, an input spectrogram of the training data is divided into 10-second parts that represent the input sequences. Labels for each frame in the training input sequence form the training label sequence. If the last part is shorter than 10 seconds, it is padded with zero frames in both input and output sequences. In case of the validation and evaluation, whole spectrograms of recordings are used as the inputs.

The input spectrograms are preprocessed in the same way as for the *DNN2MIDI* system (see 4.2.1). The same time-frequency representations and normalization techniques (except of frame grouping) are used before the spectrogram is divided into the input sequences.

4.3.2 Architecture

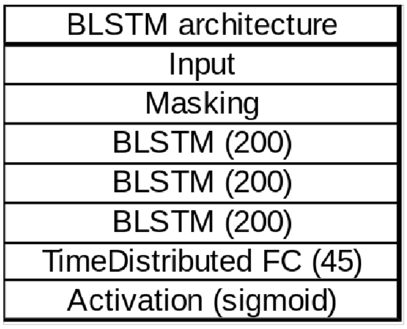
The network, that the *LSTM2MIDI* system is based on, is created according to the network used in [9]. Three Bidirectional Long Short-Term Memory (BLSTM) layers create the core of the network. LSTM layers are used because they solve the problems with vanishing or blowing gradients in the RNN. The Bidirectional LSTM, on contrary to regular LSTM, has the advantage of training simultaneously in the positive and negative time directions. The bidirectional recurrent network and LSTM layers are described in detail in 2.4.2. Each of the BLSTM layers are composed of 200 LSTM units.

The zero frames of the padded input sequences can have negative impact on training of the recurrent neural network. Masking layer is added to overcome this problem. All the frames that consist of only zero values are masked (skipped) during the training process in all downstream layers.

Time-Distributed Fully Connected (TDFC) layer is used as the output layer with the *sigmoid* activation layer. TDFC layer is a standard FC layer that is applied to every time frame of the input sequence. This ensures that the size of the output sequence is the same as the size of the input sequence. 45 neurons are used in the output layer to provide one output value for each of the recognized tone. Figure 4.6 shows the architecture of the network.

4.3.3 Training

Training process of the BLSTM network is similar to the training process of DNN. The same SGD algorithm and the learning rate scheduler was used as it is described in 4.2.3. The initial value of the learning rate was set to 0.001. In the next training sessions it was lowered to 0.0005, then to 0.0001. The number of epochs was set to 30 for each training



BLSTM architecture
Input
Masking
BLSTM (200)
BLSTM (200)
BLSTM (200)
TimeDistributed FC (45)
Activation (sigmoid)

Figure 4.6: Sequence of layers of the BLSTM network from the top to the bottom.

session. Due to the lower number of epochs, the number of training session with the same initial value of learning rate was higher according to the current performance of the network.

4.3.4 Calibration

The result piano-roll is obtained from the output of the BLSTM network by thresholding the output sequence. The same process of calibration is used here as it is used for the *DNN2MIDI* system (see 4.2.4). Figure 4.7 shows the reference piano-roll of the nocturneNr2.wav recording from the evaluation dataset (a), piano-roll with real values produced by the BLSTM network on the same recording (b), and the final thresholded piano-roll (c).

4.4 HYBRID2MIDI

The thresholding of the *PLCA2MIDI* system is a simple postprocessing method. Some existing systems [8, 11] using PLCA process the output with more complex techniques such as Hidden Markov Model (HMM). The BLSTM network used in the *LSTM2MIDI* system is used for processing the output of PLCA to smooth out the result.

4.4.1 Preprocessing

The preprocessing step has two stages. In the first stage, the audio data preprocessing and all the spectral analysis methods are applied on an input data in the same way they are applied in case of the *PLCA2MIDI* system (see 4.1.1 for details). In the second stage, the input is processed by PLCA and the resulting soft piano-roll, instead of being thresholded, is normalized. Values of the soft piano-roll depend on a length of the input because they are normalized in time to sum to 1 during the PLCA processing. Therefore, this soft piano-roll is multiplied by its size (number of time frames × number of tones) and then normalized by subtracting the mean value and dividing with the standard deviation that were calculated in advance from the training data preprocessed with PLCA. Finally, all the training inputs are divided into 10 second long sequences that are used for training.

4.4.2 Architecture, training and calibration

The rest of the system is the same as in the *LSTM2MIDI* system description. The same architecture of the network, the same training method and calibration process are used

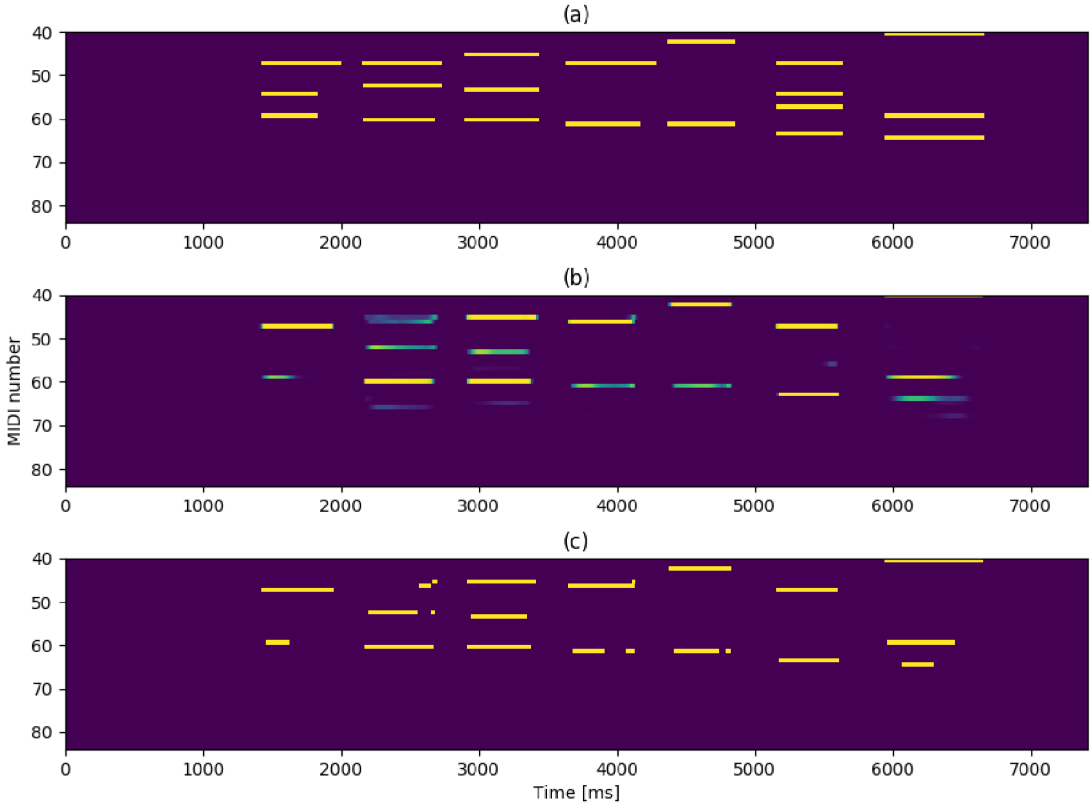


Figure 4.7: The reference piano-roll of the nocturneNr2.wav recording from the evaluation dataset (a), the piano-roll produced by the BLSTM network from the same recording before thresholding (b) and the thresholded piano-roll (c).

(see 4.3). Figure 4.8 shows the reference piano-roll of the nocturneNr2.wav recording from the evaluation dataset (a), the result piano-roll after the same recording was processed by PLCA (b) and subsequently by the BLSTM network (c) and the final thresholded piano-roll (c).

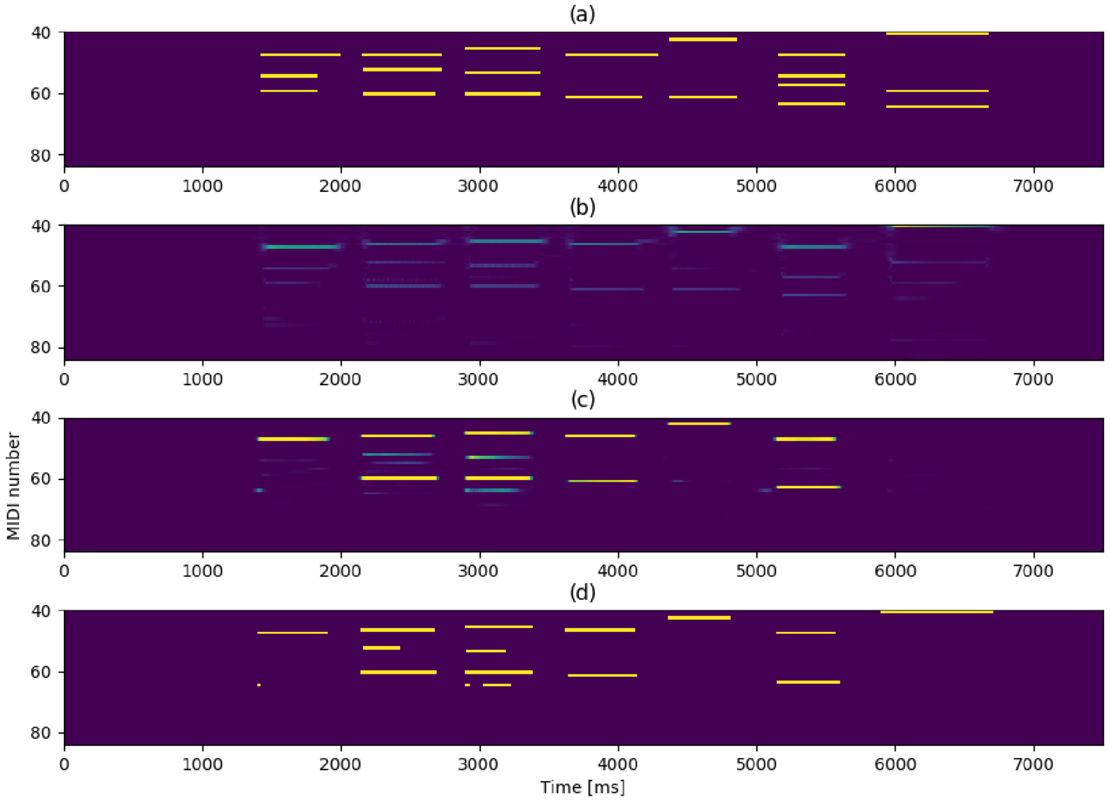


Figure 4.8: The reference piano-roll of the `nocturneNr2.wav` recording from the evaluation dataset (a), the piano-roll produced by the PLCA (b), the piano-roll produced by the BLSTM network before thresholding (c) and the thresholded piano-roll (d).

Chapter 5

Results

In this chapter, the performance of the proposed systems is presented. Evaluation dataset (described in section 3.3) was used and metrics (defined in section 2.5) were evaluated. They are compared in the following column plots. The transcription system *SONIC* [33] (described in 2.6.1) is evaluated, too, to be compared with the proposed systems. Full tables with specific values of metrics for each system can be seen in appendix C.

Table 5.1 shows the results of the system proposed by the authors of the validation dataset - Fraunhofer IDMT, obtained on the dataset 1 and dataset 2 [27] (described in section 2.6.2). They stated that onsets were detected with the 50 ms tolerance, offsets were detected with 200 ms tolerance and the pitch is scored as correct if both annotated and detected frequencies are rounded to the same MIDI pitch numbers.

Detection Function	Precision	Recall	F-Measure
Onset	0.98	0.99	0.99
Offset	0.98	0.98	0.98
Pitch Estimation	0.95	0.98	0.96

Table 5.1: Precision, Recall and F-Measure results of onset detection, offset detection, and pitch estimation of the proposed system by Fraunhofer IDMT group [27].

5.1 Frame level evaluation

5.1.1 Dataset 1

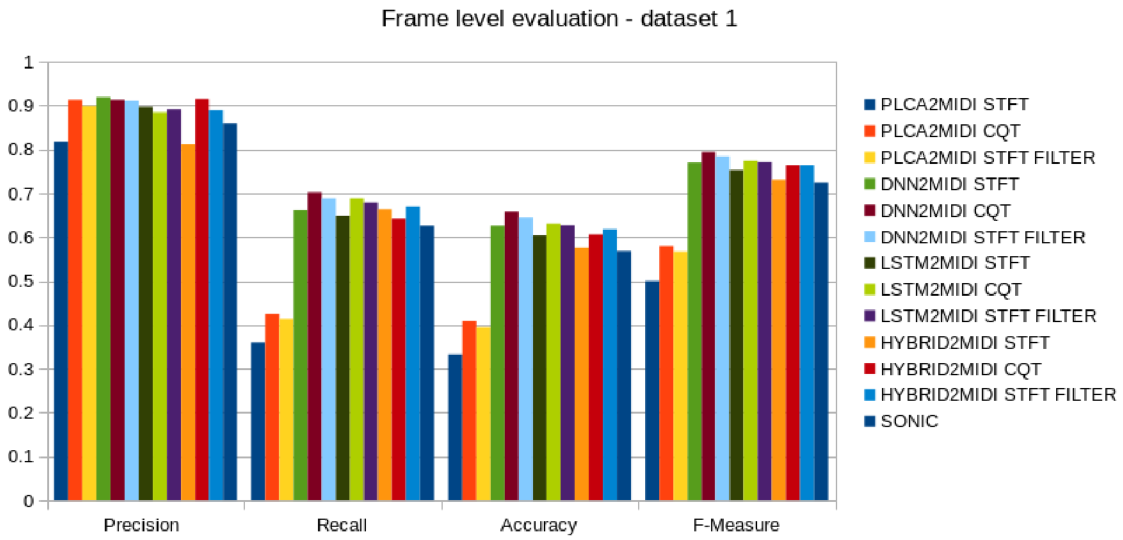


Figure 5.1: Frame level evaluation of the systems on dataset 1.

5.1.2 Dataset 2

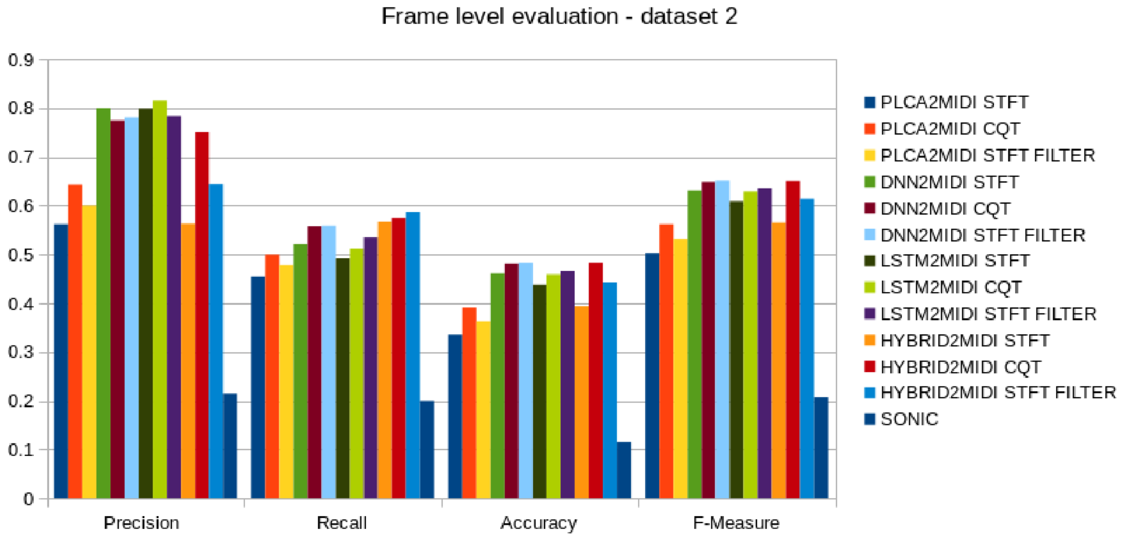


Figure 5.2: Frame level evaluation of the systems on dataset 2.

5.1.3 Dataset 3

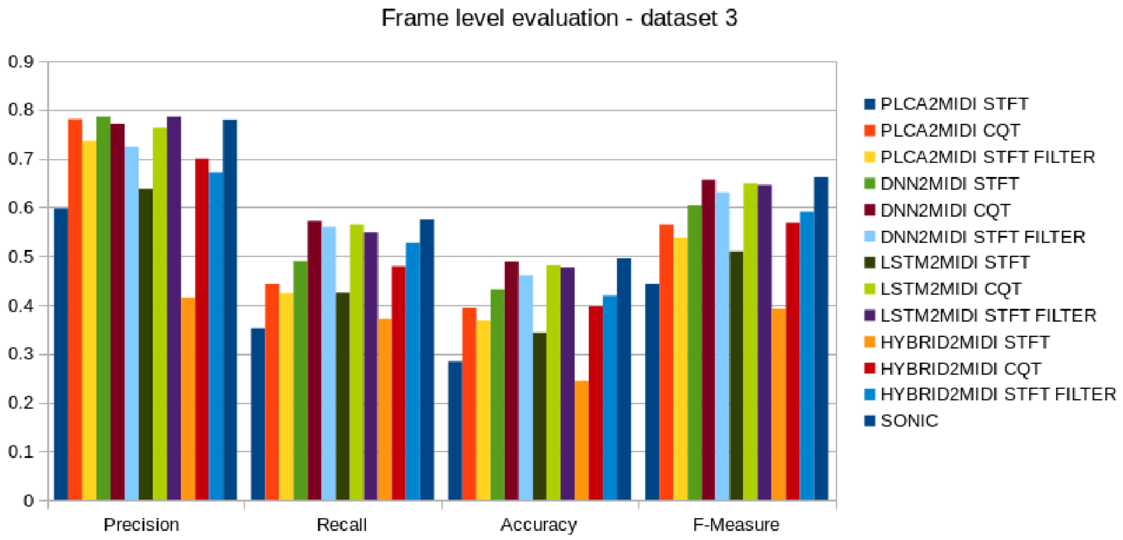


Figure 5.3: Frame level evaluation of the systems on dataset 3.

5.2 Note level evaluation

5.2.1 Dataset 1

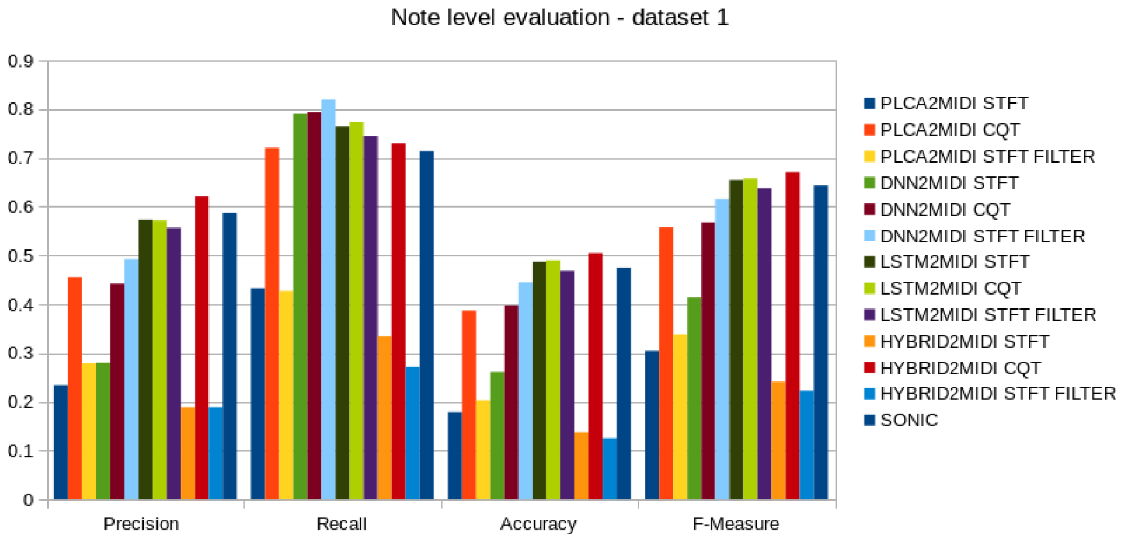


Figure 5.4: Note level evaluation of the systems on dataset 1.

5.2.2 Dataset 2

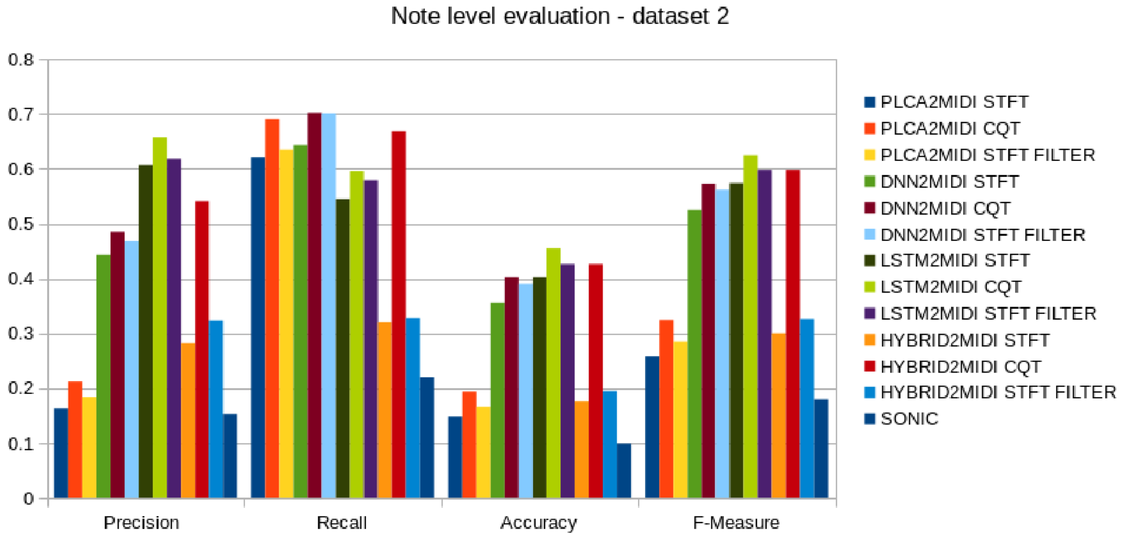


Figure 5.5: Note level evaluation of the systems on dataset 2.

5.2.3 Dataset 3

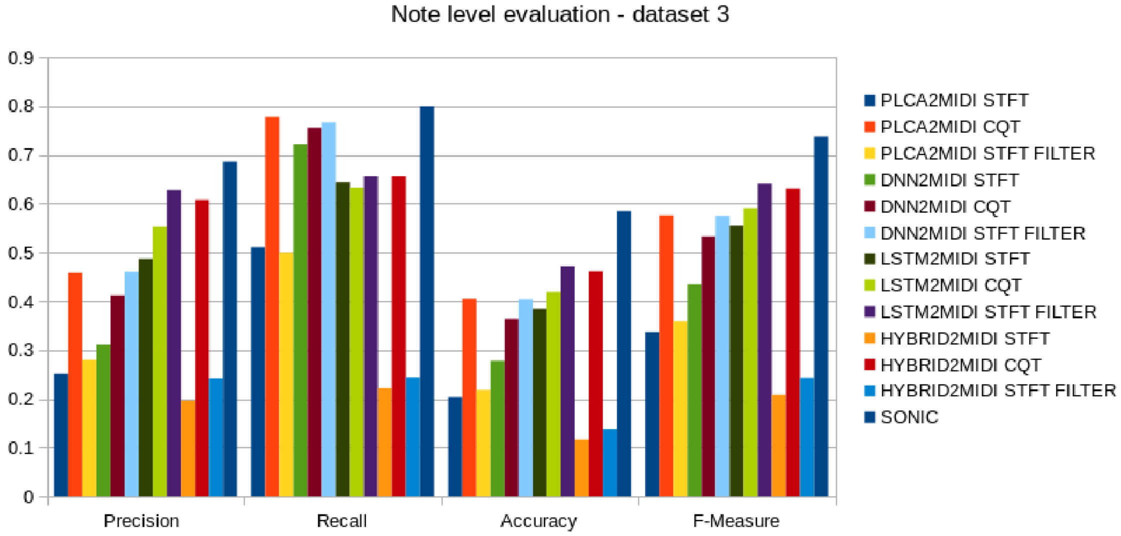


Figure 5.6: Note level evaluation of the systems on dataset 3.

5.3 Discussion

In this section, the achieved results are discussed. The nature of the evaluation data and evaluation metrics have to be considered. Therefore, the section is divided into two parts where frame level and note level evaluations are discussed separately.

5.3.1 Frame level evaluation

Starting with the *SONIC* system, the results significantly vary according to the dataset that the system was evaluated on. As it was described in section 3.3, dataset 1 consists of single notes and chord recordings, dataset 2 consists of licks and polyphonic parts that varied in playing techniques and expression styles, and dataset 3 consists of five short musical pieces played on single guitar. Considering that the *SONIC* system was trained to recognize piano notes, it makes sense that it achieved the best results on dataset 3 where the sound is not so different from the sound produced by piano. The sound of the recordings in dataset 3 is produced by a guitar directly plugged into the recording device and notes are played with a standard playing technique. It also outperformed all the other systems which proves the robustness of the *SONIC* system. On contrary, the results achieved on dataset 2 are the worst due to a variance of playing and expression techniques (e.g. muted strings, slides, natural harmonics) that the *SONIC* system was not trained to recognize, and they are not similar to any piano playing techniques. All the proposed systems were not trained to recognize all these techniques, too, but they recognize more successfully the pitch of these notes due to similarity to notes played on guitar in a standard way.

The *PLCA2MIDI* system has the worst performance of pitch recognition from all the proposed systems. While the *Precision* values (that express how many of recognized pitches occur in the ground truth) are quite high, the low value of *Recall* shows that the system is not able to identify all the pitches from the ground truth in a given time frame. This negatively influences the values of *Accuracy* and $F\text{-}Meassure$.

The *DNN2MIDI* system achieved the highest frame level *Accuracy* and $F\text{-}Meassure$ among the proposed systems. The best results were obtained with the CQT spectral analysis for dataset 1 and 3, and with the filtered STFT for dataset 2. Despite of similar number of trainable parameters of the neural networks used in the *DNN2MIDI* system and the *LSTM2MIDI* system, the *DNN2MIDI* system performs better in the frame level note recognition task. One of the reasons can be that the DNN are more appropriate for mapping features to a more separable space [38]. Therefore, it is easier for the DNN to recognize each pitch in a time frame. However, the *LSTM2MIDI* system achieved similar results.

Postprocessing of the PLCA output with the BLSTM network improved the performance in almost every case. The *HYBRID2MIDI* scored the higher values of *Accuracy* and $F\text{-}Meassure$ except of the case with STFT. The decrease of performance in this case is probably caused by the spurious output of PLCA with the STFT spectral analysis. The *HYBRID2MIDI* system with the filtered STFT spectral analysis even achieved the same highest accuracy for dataset 2 as the *DNN2MIDI* system with filtered STFT.

In general, the *DNN2MIDI CQT* system seems to be the best performing system because acquired the highest value of *Accuracy* and $F\text{-}Meassure$ on almost each dataset. Also, CQT and filtered STFT appear to be the more suitable spectral analysis methods for music transcription than simple STFT.

5.3.2 Note level evaluation

In the case of a note level evaluation, note is assumed to be correctly transcribed if a transcription system returns a note that is within a half semitone and the returned note onset is within the range of 100 ms of the ground truth note. This type of evaluation offers different view on a performance of transcription systems. The *SONIC* system's results look similar to the results from the frame level evaluation. The results achieved on dataset 2 are the lowest because of the large diversity of the sound in the recordings. The results achieved on dataset 3 are the highest among all the systems. The main reason of such high metric values is probably an onset detector that is a part of the *SONIC* system and that ensures that the most of the onsets of the transcribed notes are within the time tolerance of the onsets of the ground truth notes.

The results show very distinctive performance of the *PLCA2MIDI* and *HYBRID2MIDI* systems. These systems with the CQT spectral analysis achieved significantly higher values of the *Accuracy* and *F-Meassure* metrics than the systems with the STFT and filtered STFT spectral analysis. The *HYBRID2MIDI* system with CQT even achieved the highest *Accuracy* on dataset 1. These huge differences in the performance are probably caused by a poor onset detection ability of the PLCA method using the STFT. It turns out that note onsets and offsets in the output piano-roll of PLCA are "blurry" over time and it is difficult to determine where the notes start or end. This phenomenon can be seen in figure 5.7 where the reference piano-roll of the first six seconds of Beethoven's Sonata No. 8 in C minor midi song (a), the soft piano-roll produced by PLCA using filtered STFT of the corresponding recording (b), the soft piano-roll produced by the BLSTM network processing the PLCA output (c) and the thresholded piano-roll (d) are shown. In plot (c), the blurry output of the BLSTM network can be seen which is probably caused by the training process when the PLCA output does not provide the sharp onsets and offset and the values in the PLCA output indicate that notes are probably played but the training labels tell that they are not and vice versa. This does not happen in such scale if CQT is used, and the onsets of transcribed notes lie in the allowed time range of the note onsets of the ground truth. A reason for this can be the different time resolution of CQT and STFT. Another drawback of the *PLCA2MIDI* system is high number of returned spurious notes during the note attack stages when an energy in the spectrum is more spread over the frequencies. This has bigger impact on the *Precision*, *Accuracy* and *F-Meassure* metric values in the case of the note level evaluation. However, it is a failure that can be successfully filtered out by the BLSTM network.

The most successful and stable results are provided by the *LSTM2MIDI* system. It achieved the highest accuracy in a note transcription with CQT on dataset 2 and with filtered STFT on dataset 3. The ability of learning the temporal context from input sequences allows to produce more coherent piano-roll. The lack of this ability in the *DNN2MIDI* and *PLCA2MIDI* systems causes that longer notes are often divided into several shorter notes that do not occur in the ground truth. These redundant notes decrease the overall values of the note level *Precission*, *Accuracy* and *F-Meassure* metrics.

5.3.3 Summary

From the results, we can conclude that systems based on neural networks perform better that system based solely on PLCA. The systems that incorporate the BLSTM network (*LSTM2MIDI* and *HYBRID2MIDI* with CQT) achieved the best note level transcription performance among the proposed systems. On the other, the *DNN2MIDI* system achieved

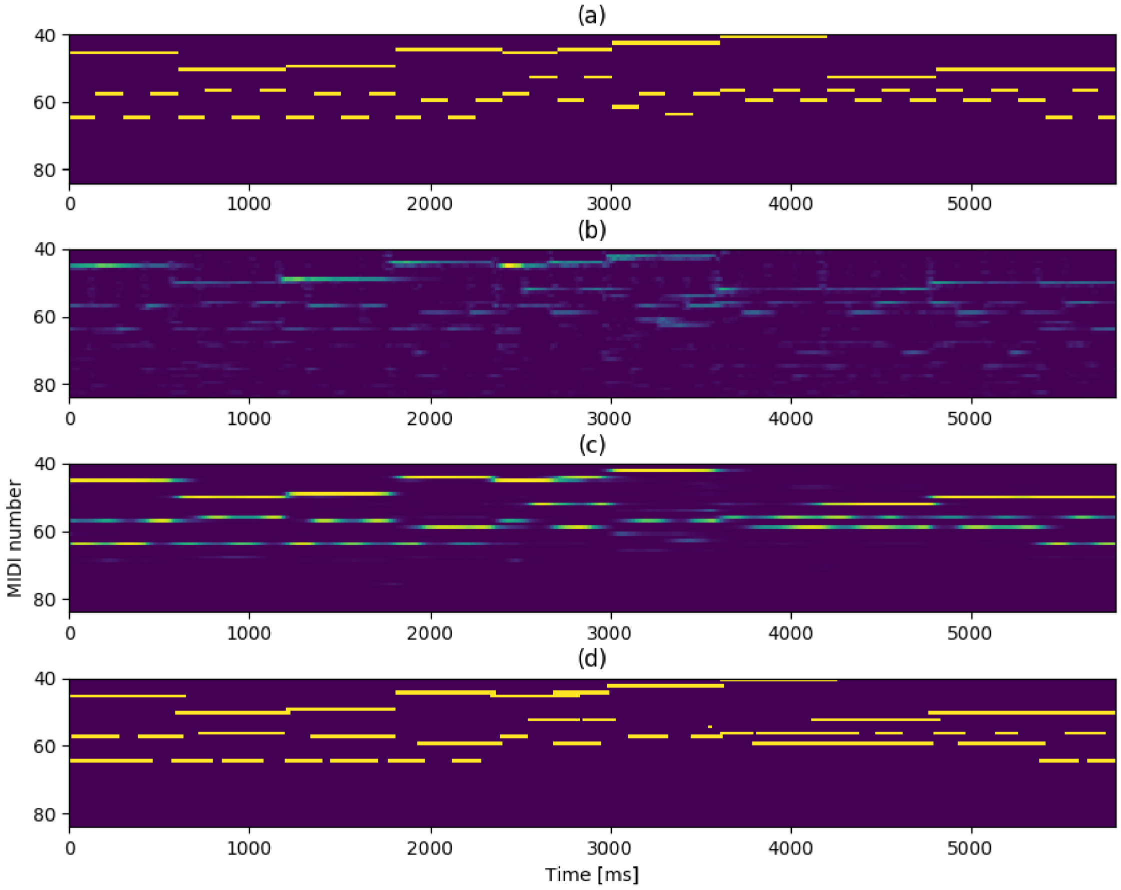


Figure 5.7: The reference piano-roll of the first six seconds of the Beethoven's Sonata No.8 in C minor midi song (a), the soft piano-roll produced by PLCA using filtered STFT of the corresponding recording (b), the soft piano-roll produced by the BLSTM network processing the PLCA output (c) and the final thresholded piano-roll (d).

the best frame level transcription performance. Also, the results show the CQT and filtered STFT spectral analysis are the most suitable spectral analysis for both, the note level and frame level transcription, probably because their components are logarithmically spaced on the frequency axis.

5.4 Examples

We have prepared demonstration files to show the actual output of the systems. They can be generated from data by running the script stored on appended memory medium (see appendix A and B) or can be downloaded from https://drive.google.com/open?id=1ZWpnqD7IQYFz7024lT7nQSpOPsuqJlBV.

Chapter 6

Conclusion

This thesis represents an experimental work where various techniques for automatic music transcription with the specialization on electric guitar were described, evaluated on the same data and compared. First, the potential of machine learning methods and spectral analysis techniques had to be discovered in the scope of music information retrieval. Short-Time Fourier Transform, Constant-Q Transform and Probabilistic Component analysis were described in detail. Also, the basic principles of neural networks were presented and evaluation metrics were introduced. Because all the examined techniques had to be trained in advance on proper data, several representative MIDI songs were obtained, synthesized with the recordings of single notes from various electric guitars and used as training data. The real recordings of electric guitar notes, chords and musical pieces provided by Institute for Digital Media Technology Fraunhofer served as the evaluation dataset. Subsequently, four different note transcription systems with three spectral analysis methods were proposed and described. After the systems were trained, they were evaluated and the results were presented. They show that the systems that employ the neural networks perform better than systems based only on PLCA technique. While the system with the DNN had the best results in frame level pitch estimation, the systems that incorporate the BLSTM network achieved higher accuracy in note level evaluation due to their capability of temporal modeling. The results of these system were compared to the results of the *SONIC* piano transcription system; in some experiments, it still outperformed all the proposed systems. This failure is attributed mainly to the lack of a more complex onset detection algorithm in the proposed systems.

6.1 Future research

As it was mentioned in this chapter, most of the proposed systems suffers from poor onset detection performance. Utilization of an onset detector would bring the improvement especially in the systems that are based on PLCA. It would be possible to detect onset of notes more precisely, identify the attack stages of the notes and withdraw them from analysis to avoid the errors made by decomposition of these parts. Consequently, it would provide more accurate piano-roll produced by PLCA that serves as input features for the BLSTM network in the *HYBRID2MIDI* system.

Because the guitar is able to produce a wide variety of sounds by applying different playing techniques and expression styles (muted strings, slides, bends, natural harmonics), modifications of the proposed systems to recognize these techniques and styles would improve the recognition capabilities, too. The different techniques for partial tracking could help to recognized bends and slides, while the additional detections systems could detect if a string is muted or if natural harmonics are played.

Another aspects that would be worth of further research are multi-resolution spectral analysis and input feature composition for the neural networks. Also, creating more spectrograms with different time and frequency resolution provides more informations about the audio signal. Features like the first derivation of the input spectrogram could be useful for note recognition with the neural networks.

Using augmented data during the training process of the neural networks could bring higher robustness of the note recognition systems, too. Adding noise generally helps, but in the case of the electric guitar recordings, there are other approaches. Expanding the training dataset with the data with slightly shifted pitches of some notes that would represents mistakes of a guitar player or a string that is out of tune, or applying guitar effects on a clear sound of a guitar would simulate more realistic guitar recordings.

In this work, single threshold was used on all tones to get the resulting piano-roll. Another option is to estimate threshold for each tones separately. Thresholding would be adapted to a character of each tone which could lead to a performance improvement.

Bibliography

[1] Abe, T.; Kobayashi, T.; Imai, S.: Harmonics tracking and pitch extraction based on instantaneous frequency. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 1. IEEE. 1995. pp. 756–759.

[2] Alan, V. O.; Ronald, W. S.; John, R.: Discrete-time signal processing. *New Jersey, Printice Hall Inc.* 1989.

[3] Bay, M.: *Methods for multiple pitch tracking and instrument separation from monaural polyphonic recordings*. PhD. Thesis. 2012.

[4] Beauchamp, J. W.: *Analysis, synthesis, and perception of musical sounds*. Springer. 2007. ISBN 038732576X.

[5] Bednařík, J.: Převod záznamu piana z WAV do MIDI. FIT VUT Brno. Bachelor's Thesis. 2013.

[6] Benetos, E.; Dixon, S.: A shift-invariant latent variable model for automatic music transcription. *Computer Music Journal*. vol. 36, no. 4. 2012: pp. 81–94.

[7] Benetos, E.; Dixon, S.; Giannoulis, D.; et al.: Automatic music transcription: challenges and future directions. *Journal of Intelligent Information Systems*. vol. 41, no. 3. 2013: pp. 407–434.

[8] Benetos, E.; Weyde, T.: Explicit duration hidden markov models for multiple-instrument polyphonic music transcription. 2013.

[9] Böck, S.; Schedl, M.: Polyphonic piano note transcription with recurrent neural networks. In *Acoustics, speech and signal processing (ICASSP), 2012 IEEE international conference on*. IEEE. 2012. pp. 121–124.

[10] Brown, J. C.: Calculation of a constant Q spectral transform. *The Journal of the Acoustical Society of America*. vol. 89, no. 1. 1991: pp. 425–434.

[11] Cheng, T.; Dixon, S.; Mauch, M.: Improving piano note tracking by HMM smoothing. In *Signal Processing Conference (EUSIPCO), 2015 23rd European*. IEEE. 2015. pp. 2009–2013.

[12] Choi, K.; Fazekas, G.; Cho, K.; et al.: A tutorial on deep learning for music information retrieval. *arXiv preprint arXiv:1709.04396*. 2017.

[13] Chunghsin, Y.: *Multiple fundamental frequency estimation of polyphonic recordings*. PhD. Thesis. Ph. D. dissertation, University Paris 6. 2008.

[14] Clevert, D.-A.; Unterthiner, T.; Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (ELUs). *arXiv preprint arXiv:1511.07289*. 2015.

[15] Cont, A.: Realtime multiple pitch observation using sparse non-negative constraints. In *International Symposium on Music Information Retrieval (ISMIR)*. 2006. pp. 206–211.

[16] Davy, M.; Godsill, S.; Idier, J.: Bayesian analysis of polyphonic western tonal music. *The Journal of the Acoustical Society of America*. vol. 119, no. 4. 2006: pp. 2498–2517.

[17] Dempster, A. P.; Laird, N. M.; Rubin, D. B.: Maximum likelihood from incomplete data via the EM algorithm. *Journal of the royal statistical society. Series B (methodological)*. 1977: pp. 1–38.

[18] Deng, L.; Yu, D.; et al.: Deep learning: methods and applications. *Foundations and Trends® in Signal Processing*. vol. 7, no. 3–4. 2014: pp. 197–387.

[19] Dressler, K.: Multiple fundamental frequency extraction for MIREX 2012. *Eighth Music Information Retrieval Evaluation eXchange (MIREX)*. 2012.

[20] Fuentes, B.; Badeau, R.; Richard, G.: Blind harmonic adaptive decomposition applied to supervised source separation. In *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*. IEEE. 2012. pp. 2654–2658.

[21] Goldstein, J. L.: An optimum processor theory for the central formation of the pitch of complex tones. *The Journal of the Acoustical Society of America*. vol. 54, no. 6. 1973: pp. 1496–1516.

[22] Goto, M.: A real-time music-scene-description system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication*. vol. 43, no. 4. 2004: pp. 311–329.

[23] Greff, K.; Srivastava, R. K.; Koutník, J.; et al.: LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems*. vol. 28, no. 10. 2017: pp. 2222–2232.

[24] Guzman, J.: “Sustain” Myth and Science. 2013.
Retrieved from: http://www.cycfi.com/2013/11/sustain-myth-science/

[25] Hlawatsch, F.; Auger, F.: *Time-frequency analysis*. John Wiley & Sons. 2013.

[26] Hochreiter, S.; Schmidhuber, J.: Long short-term memory. *Neural computation*. vol. 9, no. 8. 1997: pp. 1735–1780.

[27] Kehling, C.; Abeßer, J.; Dittmar, C.; et al.: Automatic Tablature Transcription of Electric Guitar Recordings by Estimation of Score-and Instrument-Related Parameters. In *DAFx*. 2014. pp. 219–226.

[28] Klapuri, A.; Davy, M.: *Signal processing methods for music transcription*. Springer Science & Business Media. 2007. ISBN 0387328459.

[29] Klapuri, A. P.: Multiple fundamental frequency estimation based on harmonicity and spectral smoothness. *IEEE Transactions on Speech and Audio Processing*. vol. 11, no. 6. 2003: pp. 804–816.

[30] Lahat, M.; Niederjohn, R.; Krubsack, D.: A spectral autocorrelation method for measurement of the fundamental frequency of noise-corrupted speech. *IEEE transactions on acoustics, speech, and signal processing*. vol. 35, no. 6. 1987: pp. 741–750.

[31] Large, E. W.; Kolen, J. F.: Resonance and the perception of musical meter. *Connection science*. vol. 6, no. 2-3. 1994: pp. 177–208.

[32] MacKay, D. J.: *Information theory, inference and learning algorithms*. Cambridge university press. 2003.

[33] Marolt, M.: Multiple fundamental frequency estimation & tracking submission for MIREX 2016. *Music Information Retrieval Evaluation eXchange (MIREX)*. 2016.

[34] Meddis, R.: Simulation of mechanical to neural transduction in the auditory receptor. *The Journal of the Acoustical Society of America*. vol. 79, no. 3. 1986: pp. 702–711.

[35] Noll, A. M.: Cepstrum pitch determination. *The journal of the acoustical society of America*. vol. 41, no. 2. 1967: pp. 293–309.

[36] Pertusa, A.; Inesta, J. M.: Multiple fundamental frequency estimation using Gaussian smoothness. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*. IEEE. 2008. pp. 105–108.

[37] Quiza, R.; Davim, J.: Computational modeling of machining systems. *Intelligent machining: modeling and optimization of the machining processes and systems. London: ISTE*. 2009: pp. 173–213.

[38] Sainath, T. N.; Vinyals, O.; Senior, A.; et al.: Convolutional, long short-term memory, fully connected deep neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE. 2015. pp. 4580–4584.

[39] Sak, H.; Senior, A.; Beaufays, F.: Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth annual conference of the international speech communication association*. 2014.

[40] Schörkhuber, C.; Klapuri, A.: Constant-Q transform toolbox for music processing. In *7th Sound and Music Computing Conference, Barcelona, Spain*. 2010. pp. 3–64.

[41] Schuster, M.; Paliwal, K. K.: Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*. vol. 45, no. 11. 1997: pp. 2673–2681.

[42] Shashanka, M.; Raj, B.; Smaragdis, P.: Probabilistic latent variable models as nonnegative factorizations. *Computational intelligence and neuroscience*. vol. 2008. 2008.

[43] Slaney, M.; et al.: An efficient implementation of the Patterson-Holdsworth auditory filter bank. *Apple Computer, Perception Group, Tech. Rep.* vol. 35, no. 8. 1993.

[44] Smaragdis, P.; Raj, B.; Shashanka, M.: A probabilistic latent variable model for acoustic modeling. *Advances in models for acoustic processing, NIPS*. vol. 148. 2006: pp. 8–1.

[45] Swift, A.: A brief introduction to midi. *URL http://www. doc. ic. ac. uk/˜ nd/surprise_97/journal/vol1/aps2*. vol. 6. 1997.

[46] Thickstun, J.; Harchaoui, Z.; Foster, D.; et al.: MIREX 2017: FREQUENCY DOMAIN CONVOLUTIONS FOR MULTIPLE F0 ESTIMATION.

[47] Waibel, A.; Hanazawa, T.; Hinton, G.; et al.: Phoneme recognition using time-delay neural networks. In *Readings in speech recognition*. Elsevier. 1990. pp. 393–404.

[48] Williams, R. J.: Training recurrent networks using the extended Kalman filter. In *Neural Networks, 1992. IJCNN., International Joint Conference on*, vol. 4. IEEE. 1992. pp. 241–246.

[49] Yeh, C.; Robel, A.; Rodet, X.: Multiple fundamental frequency estimation of polyphonic music signals. In *Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on*, vol. 3. IEEE. 2005. pp. iii–225.

Appendix A

DVD content

The content of DVD consists of the following files:

`Anaconda3-5.1.0-Linux-x86_64.sh`	installation file of Anaconda3
`demo1.wav`, `demo2.wav`, `demo3.wav`	demonstration recordings from the evaluation dataset
`guitar2midi.yaml`	development environment configuration file
`IDMT-SMT-GUITAR_V2`	directory that contains the evaluation dataset
`models`	directory that contains trained neural network models and files with thresholds
`README.txt`	file with a basic description and manual
`resources`	directory that contains pretrained frequency bases for PLCA
`run_demonstration.sh`	script that generates midi files from the demonstration recordings
`sound_fonts`	directory that contains guitar recordings used for generating training dataset
`src`	directory that contains source files
`synthesize.sh`	script that synthesizes midi songs for the training and validation dataset
`train_transposed`	directory that contains transposed midi songs of the training dataset
`valid`	directory that contains midi songs for the validation dataset

Appendix B

Manual

This appendix describes how to prepare environment for executing source files on Linux operating system. The current working directory has to be set to the root folder of the appended memory medium. Before the systems can be executed, Anaconda3 has to be installed. It can be installed using the `Anaconda3-5.1.0-Linux-x86_64.sh` installation file with the following command:

```
$ bash Anaconda3-5.1.0-Linux-x86_64.sh
```

After it is successfully installed, the *guitar2midi* environment has to be imported and activated using the `guitar2midi.yaml` configuration file:

```
$ conda env create -f guitar2midi.yaml
$ source activate guitar2midi
```

Now, demonstration midi files can be generated by running the following command:

```
$ bash run_demonstration.sh
```

and the training and validation dataset can be generated by running the following command:

```
$ bash synthesize.sh
```

Appendix C

Full tables with results

C.1 Frame level evaluation

C.1.1 Dataset 1

System	Spectral analysis	Precision	Recall	Accuracy	F-Measure
PLCA2MIDI	STFT	0.818	0.36	0.333	0.5
	CQT	0.913	0.425	0.409	0.58
	STFT FILTER	0.899	0.414	0.395	0.567
DNN2MIDI	STFT	0.919	0.662	0.626	0.77
	CQT	0.913	0.702	0.658	0.794
	STFT FILTER	0.911	0.688	0.645	0.784
LSTM2MIDI	STFT	0.897	0.649	0.604	0.753
	CQT	0.884	0.688	0.631	0.774
	STFT FILTER	0.891	0.679	0.627	0.771
HYBRID2MIDI	STFT	0.812	0.664	0.576	0.731
	CQT	0.915	0.642	0.606	0.764
	STFT FILTER	0.889	0.669	0.618	0.764
SONIC		0.859	0.626	0.568	0.724

Table C.1: Frame level evaluation on dataset 1.

C.1.2 Dataset 2

System	Spectral analysis	Precision	Recall	Accuracy	F-Measure
PLCA2MIDI	STFT	0.562	0.455	0.336	0.503
	CQT	0.643	0.5	0.391	0.562
	STFT FILTER	0.6	0.478	0.363	0.532
DNN2MIDI	STFT	0.8	0.521	0.461	0.631
	CQT	0.775	0.558	0.481	0.649
	STFT FILTER	0.781	0.559	0.483	0.652
LSTM2MIDI	STFT	0.799	0.492	0.438	0.609
	CQT	0.816	0.512	0.459	0.629
	STFT FILTER	0.784	0.535	0.466	0.636
HYBRID2MIDI	STFT	0.563	0.567	0.394	0.565
	CQT	0.751	0.575	0.483	0.651
	STFT FILTER	0.644	0.587	0.443	0.614
SONIC		0.215	0.2	0.115	0.207

Table C.2: Frame level evaluation on dataset 2.

C.1.3 Dataset 3

System	Spectral analysis	Precission	Recall	Accuracy	F-Measure
PLCA2MIDI	STFT	0.598	0.352	0.284	0.443
	CQT	0.781	0.443	0.394	0.565
	STFT FILTER	0.736	0.424	0.368	0.538
DNN2MIDI	STFT	0.786	0.49	0.432	0.604
	CQT	0.771	0.572	0.489	0.657
	STFT FILTER	0.724	0.56	0.461	0.631
LSTM2MIDI	STFT	0.638	0.425	0.343	0.51
	CQT	0.764	0.565	0.481	0.649
	STFT FILTER	0.786	0.549	0.477	0.646
HYBRID2MIDI	STFT	0.415	0.372	0.244	0.392
	CQT	0.7	0.479	0.397	0.569
	STFT FILTER	0.672	0.527	0.419	0.591
SONIC		0.78	0.575	0.495	0.662

Table C.3: Frame level evaluation on dataset 3.

C.2 Note level evaluation

C.2.1 Dataset 1

System	Spectral analysis	Precission	Recall	Accuracy	F-Measure
PLCA2MIDI	STFT	0.234	0.433	0.179	0.304
	CQT	0.455	0.721	0.387	0.558
	STFT FILTER	0.279	0.427	0.203	0.338
DNN2MIDI	STFT	0.28	0.791	0.261	0.414
	CQT	0.442	0.794	0.397	0.568
	STFT FILTER	0.493	0.82	0.445	0.616
LSTM2MIDI	STFT	0.573	0.765	0.487	0.655
	CQT	0.572	0.774	0.49	0.658
	STFT FILTER	0.557	0.745	0.468	0.638
HYBRID2MIDI	STFT	0.189	0.334	0.137	0.241
	CQT	0.621	0.73	0.505	0.671
	STFT FILTER	0.189	0.271	0.125	0.223
SONIC		0.587	0.714	0.475	0.644

Table C.4: Note level evaluation on dataset 1.

C.2.2 Dataset 2

System	Spectral analysis	Precission	Recall	Accuracy	F-Measure
PLCA2MIDI	STFT	0.163	0.621	0.148	0.258
	CQT	0.212	0.691	0.194	0.324
	STFT FILTER	0.184	0.635	0.166	0.285
DNN2MIDI	STFT	0.443	0.643	0.356	0.525
	CQT	0.485	0.702	0.402	0.573
	STFT FILTER	0.468	0.701	0.39	0.562
LSTM2MIDI	STFT	0.607	0.545	0.402	0.574
	CQT	0.657	0.596	0.455	0.625
	STFT FILTER	0.618	0.579	0.426	0.598
HYBRID2MIDI	STFT	0.282	0.32	0.176	0.3
	CQT	0.541	0.668	0.426	0.598
	STFT FILTER	0.323	0.328	0.195	0.326
SONIC		0.153	0.22	0.099	0.18

Table C.5: Note level evaluation on dataset 2.

C.2.3 Dataset 3

System	Spectral analysis	Precission	Recall	Accuracy	F-Measure
PLCA2MIDI	STFT	0.251	0.511	0.203	0.337
	CQT	0.458	0.778	0.405	0.576
	STFT FILTER	0.28	0.5	0.218	0.359
DNN2MIDI	STFT	0.311	0.722	0.278	0.435
	CQT	0.412	0.756	0.364	0.533
	STFT FILTER	0.46	0.767	0.404	0.575
LSTM2MIDI	STFT	0.487	0.644	0.384	0.555
	CQT	0.553	0.633	0.419	0.591
	STFT FILTER	0.628	0.656	0.472	0.641
HYBRID2MIDI	STFT	0.196	0.222	0.116	0.208
	CQT	0.608	0.656	0.461	0.631
	STFT FILTER	0.242	0.244	0.138	0.243
SONIC		0.686	0.8	0.585	0.738

Table C.6: Note level evaluation on dataset 3.