Technische Universität Berlin
Fakultät I - Fachgebiet Audiokommunikation

Master's Thesis

Automatic Chord Recognition with Fully Convolutional Neural Networks

Hamed Habibi Fard

Supervision:

1. Prof. Dr. Stefan Weinzierl
2. Dr. Athanasios Lykartsis

Initial Issue: 25th September 2020

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt gegenüber der Fakultät I der Technischen Universität Berlin, dass die vorliegende, dieser Erklärung angefügte Arbeit selbstständig und nur unter Zuhilfenahme der im Literaturverzeichnis genannten Quellen und Hilfsmittel angefertigt wurde. Alle Stellen der Arbeit, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen wurden, sind kenntlich gemacht. Ich reiche die Arbeit erstmals als Prüfungsleistung ein. Ich versichere, dass diese Arbeit oder wesentliche Teile dieser Arbeit nicht bereits dem Leistungserwerb in einer anderen Lehrveranstaltung zugrunde lagen.

Titel der schriftlichen Arbeit

Automatic Chord Recognition with Fully Convolutional Neural Networks

Verfasser

Habibi Fard, Hamed, [redacted]

Betreunde Dozenten

Prof. Dr. Stefan Weinzierl,
Dr. Athanasios Lykartsis

Mit meiner Unterschrift bestätige ich, dass ich über fachübliche Zitierregeln unter-richtet worden bin und verstanden habe. Die im betroffenen Fachgebiet üblichenZitiervorschriften sind eingehalten worden. Eine Überprüfung der Arbeit auf Plagiatemithilfe elektronischer Hilfsmittel darf vorgenommen werden.

Berlin, den

.....

Abstract

This thesis proposes a new, fully convolutional neural network architecture for frame-wise automatic chord recognition and, by doing so, expands the deep learning vocabulary of the field. The architecture is based on the MultiResUnet, which is then extended with spatial dropout, image pyramids, and a structured training approach. In the first step, three data sets are merged to obtain a data collection of 361 songs with a total of up to 20 hours, 27 minutes, and 16 seconds of real-world music. Furthermore, pitch-shifting data augmentation is applied where each song is shifted up or down by one to six discrete semitones resulting in 12 deformations for each audio file. Moreover, all original and pitch-shifted audio files are transformed into a three-dimensional representation via the harmonic constant-Q transform. This representation is used as an input to the model mentioned above. The model is then trained in a supervised manner and predicts for each frame, the root pitch class, the active pitch classes, and the bass pitch class from the spatial dropout's output layer. Afterward, results are evaluated using 15 different comparison methods implemented by the mir_eval package. In addition, it is also shown that the proposed model exceeds the baseline over every single chord recognition metric, while training time per epoch is reduced by half, and fewer training parameters are required. The thesis continues with a detailed root confusion and strict quality-wise error analysis. Finally, it concludes with a two-track analysis where the reference annotations and the models' estimation are visualized and compared against each other.

Zusammenfassung

Diese Arbeit schlägt eine neue, vollständig konvolutionale neuronale Netzwerkarchitektur für automatisierte und Frame-basierte Akkord-Erkennung vor und erweitert damit die Deep Learning Ansätze in diesem Feld. Die Architektur basiert auf dem MultiResUnet, das dann um spatial dropout, Bildpyramiden und einen strukturierten Trainingsansatz erweitert wird. In einem ersten Schritt werden drei Datensätze zusammengeführt, um eine Datensammlung von 361 Liedern mit insgesamt bis zu 20 Stunden, 27 Minuten und 16 Sekunden realer Musik zu erhalten. Darüber hinaus wird durch eine Tonhöhenverschiebung eine Datenvergrößerung vorgenommen, bei der jedes Lied um einen bis sechs diskrete Halbtöne nach oben oder unten verschoben wird, was zu 12 Deformationen für jede Audiodatei führt. Weiter werden alle originalen und tonhöhen-verschobenen Audiodateien über die harmonisch Konstante Q-Transformation in eine dreidimensionale Darstellung umgewandelt. Diese Darstellung wird als Eingag für das oben erwähnte Modell verwendet. Das Modell wird dann via überwachtes Lernen trainiert und sagt für jedes Frame die Grundtonklasse, die aktiven Tonklassen und die Basstonklasse aus der Ausgabeschicht des spatial dropout voraus. Anschließend werden die Ergebnisse mit Hilfe von 15 verschiedenen Vergleichsmethoden, die durch das Paket mir_eval implementiert wurden, ausgewertet. Außerdem wird gezeigt, dass das vorgeschlagene Modell für jede einzelne Akkorderkennungsmetrik das zu Grunde liegende Vergleichsmodell übertrifft, während die Trainingszeit pro Epoche um die Hälfte reduziert wird und weniger Trainingsparameter erforderlich sind. Die These wird mit einer detaillierten Grundakkord- und einer strengen qualitativen Fehleranalyse fortgesetzt. Schließlich schließt sie mit einer Analyse von zwei Liedern, in denen die Referenzannotationen und die Schätzung der Modelle visualisiert und miteinander verglichen werden, ab.

Contents

List of Figures

List of Tables

1 Introduction

After two decades of active research, automatic chord recognition (ACR) has become a classic yet still open MIR challenge. The interest comes from the importance of harmony in **Western tonal music** and the demand the general public places on chord-based music representations of popular music. However, manually identifying chords from recorded audio is an arduous and time-consuming task and requires professional musical training. To save time and help less experienced musicians, web-services such as Chordify use ACR techniques to display chords of any audio file to their users [11]. Alexa [3], a popular web analytics service, ranks Chordify among the top 6000 websites in global internet engagement, i.e., 200 million unique users, at the time of writing [20]. Therefore there is ample motivation to develop efficient, high-performance automated chord recognition systems. Furthermore, since chords and chord progressions are highly abstract mid-level representations of polyphonic music, they also play a role in other high-level MIR tasks such as genre classification [5], cover song identification [7, 53] and music emotion recognition [15, 18], music structure segmentation [76] and even melody transcription [52].

Historically speaking, typical chord recognition systems followed a three-stage pipeline[1]: feature extraction, pattern matching, and chord sequence decoding (post-filtering), where all three stages were originally hand-crafted. However, with the emergence of deep neural networks, focus shifted to data-driven methods. Most modern deep learning-based ACR systems follow a two-stage architecture that is quite common in speech recognition. The first stage, also called *acoustic model*, processes the acoustic features in audio signals and extracts discriminative features from them [47, p.10] while the second stage, called *temporal* or *language model*, takes the estimates of the acoustic model and models the temporal relationship and structure in the sequence of chord symbols and decodes them into human-readable chord segments [47, p.10-11]. Almost all ACR systems transform the raw audio files via a short term Fourier transform (STFT) [4] or a constant-Q transform [12, 13] into a more suitable time-frequency representation. This time-frequency representation serves as an input to the acoustic model. Various acoustic models have been explored over time such as convolutional neural networks (CNNs) [37, 62, 95], deep belief network (DBN) [10, 23], feed-forward deep neural network (DNN) [84, 48, 69], and hidden Markov models (HMMs) [17]. Later on, with the advancements in temporal modeling, HMMs were replaced by RNNs [10]. Most ACR systems that followed were either variants or extensions of the system above. For example, [23, 34, 93, 62, 69, 95] used either bidirectional long-short term memory (BLSTM) recurrent neural networks [82, 29] or long-short term memory (LSTM) [33] or bidirectional gated recurrent units (Bi-GRUs) [16] to avoid the vanishing gradient problem that the vanilla RNNs had [75].

The general assumption in ACR is that RNNs have a better ability to model long-range dependencies and learn and apply musical knowledge instead of just smoothing the acoustic model's output. Nevertheless, this belief is challenged by two studies: first, [6] developed a broad temporal convolutional neural network (TCN) and conducted a series of experiments in sequence prediction tasks from various domains in which generic RNNs (namely, vanilla RNNs, LSTMs, and GRUs) were known to perform rather well. Their results showed that a general TCN, with its components all coming from standard modern practices in computer vision, could indeed outperform LSTMs/GRUs (with the same model size as the TCN) in

[1]For a thorough review of chord recognition before the advent of deep learning techniques, see [64]

most of their considered tasks.

Second, [50] stated that incorporating and training sophisticated temporal models on a time-frame basis is pointless in practice. They carried out two experiments to support their claim. Firstly, two temporal models, namely, a first-order Markov chain and an RNN with LSTM units, were compared according to their capacity to model chord sequences. The results showed that RNN, despite its higher modeling capability, performed only marginally better. Secondly, the performance of two temporal models was investigated by integrating them into a full chord recognition framework. The results of this experiment suggested that when complex RNN models were deployed within a complete chord recognition framework, they could not outperform the simpler first-order HMM. In addition, [58] compared two different deep learning approaches for melody extraction: the first framework was based on a BLSTM-RNN model that approached melody extraction as a sequence prediction problem. In contrast, the other framework considered it as a segmentation problem. Their results indicated that the segmentation based system could deliver comparable and, in one case, even better results than the well established BLSTM-RNN approach.

Meanwhile, more modern design principles and architectures in computer vision found their way into the MIR domain. For example, [95] used DeeplabV3+ [14] for polyphonic music transcription, whereas [25, 44, 35] applied the U-Net architecture [79] for dominant melody estimation, singing voice separation and melody extraction, respectively. While other fields in the MIR domain are leveraging state-of-the-art computer vision deep learning techniques with great success, ACR has not caught up to the latest trends in computer vision and, as discussed before, considering ACR as a sequence prediction task and applying RNNs and its variants as a temporal model led most ACR systems to converge to the same architecture. Thus, it would be intriguing to see if rather new deep learning architectures in computer vision can improve chord recognition accuracy.

Therefore, in this master's thesis, the typical conventions of using RNNs for temporal modeling shall be abandoned, and a new perspective of the problem shall be adopted. The following chapters will demonstrate how by exploiting intrinsic structural similarities between chord classes and using a well known fully convolutional architecture in the field of biomedical image segmentation, namely the U-Net architecture, and applying specific modification to it that are motivated by the best practices in computer vision, a chord recognition framework is established that not only delivers comparable results with the state-of-the-art approaches in ACR but also has fewer parameters and is faster to train.

1.1 Chronological summary of deep learning-based approaches in ACR to the current state of the art

The following state of the art chapter refers to the developments in ACR in the last eight years and specifically to data-driven/deep learning approaches, which are summarized in Table 1.

Humphrey and Bello [37] were the first who proposed a convolutional neural network (CNN) for the task of automatic chord recognition (ACR). Their system comprised three convolutional layers, an optional pooling layer, and two fully connected layers. They found out that data augmentation reduced the extent to which the network could over-fit the training data. Moreover, they observed that over-fitting occurred in the network's fully connected layers rather than the convolutional ones. Korzeniowski and Widmer [49] dropped the fully connected layers in their system in favor of general average pooling (GAP) [55] and made the network deeper. They used the hidden representation computed by their CNN as features and fed them into a conditional random field (CRF). The primary purpose of the CRF in such a setting is to smooth the sequence.

Nakayama and Shuichi [70] argued that the network architecture of most chord recognition systems using deep learning was shallow, so they used a deep residual neural network [31] to make the network architecture deeper (15 layers). However, the improvement that residual learning brought in comparison to shallower architectures was minimal. It is no secret that many audio processing tasks adapt network architectures that initially came from computer vision. Whereas in computer vision, deep architectures such as ResNet [31] or Inception [88] do indeed outperform shallower ones, they do not necessarily perform better in audio processing tasks. Koutini et al. [51] analyzed the receptive field of CNNs within deeper architectures over input spectrograms and their generalization on unseen samples to understand why these deeper architectures perform worse than shallower ones (for the task of audio sound classification). They showed that for relatively small datasets, a large receptive field over the frequency dimension pushes the CNNs to overfit on the training samples while a smaller than necessary receptive field causes underfitting and hinders the CNN's ability to learn discriminative features. Moreover, by tuning the receptive field of two modern deep architectures (namely, ResNet and DenseNet [39]), they managed to match and even outperform VGG-based [85] models. Note that almost all convolutional neural network architectures used in ACR are still VGG-based, leaving room for improvement.

Wu and Li [94] used a fully convolutional neural network architecture with seven layers to extract harmonically relevant features. They also introduced the concept of residual learning via two skip connections to their network. However, their CNN was trained not only with audio, but also with MIDI-audio pairs. The feature sequence calculated by CNN was fed into a BLSTM for pattern matching, and finally, a CRF inferred the output label sequence. Note that both the classifier (BLSTM) and the decoder (CRF) were trained individually.

They concluded that their system's bottleneck was the BLSTM-CRF sequence classification/decoding model rather than the acoustic model. However, their conclusion regarding the decoder was already mentioned in another study by [50].

Park et al. [74] reasoned that most machine learning approaches proposed for ACR such as CNNs and RNNs, have limitations in capturing long-range dependencies and require an additional model that needs to be trained individually to achieve better performance. They

developed an ACR system based on a Bi-directional Transformer (BTC) [24], which is an attention-based network architecture that is not based on any recurrence or convolution operation. They showed that their system was able to utilize its receptive field and identified sections of chords that were important for the task of chord recognition and captured long-term dependencies while benefiting from a more straightforward training procedure. However, [91] showed that softmax-normalized depthwise-separable convolutions that share weight over the channel dimension (also called lightweight convolutions) and dynamic convolutions could perform competitively or even better than the best-reported self-attention models for large-scale machine translation and language modeling tasks. Also, the argument that CNNs are limited in their ability to capture long-range dependencies can be disputed by the already discussed TCN and the meanwhile famous Wavenet [72] model, an architecture based on dilated convolutions which could not only utilize a very large receptive field and deal with long-range temporal dependencies but could also be efficiently trained on data with ten thousands of samples per second of audio.

Jiang and Chen [45] addressed the challenges of large-vocabulary chord recognition such as class imbalance and the frequency of chord qualities present in the data sets and proposed a model that decomposes the chord labels into smaller sub-components, i.e., root and triad, bass, seventh and ninth, eleventh and thirteenth, each one having a smaller chord vocabulary. Then, a multi-task classifier was trained to detect all the sub-components (given the audio features). Finally, the results were assembled to form the final chord label. Their model used CRNN (convolutional neural network plus a BLSTM) for feature processing and a CRF for sequence decoding. Their decomposition algorithm extends the structured representation used in this thesis by including 9^{th}, 11^{th}, and 13^{th} chords.

Wu and Carsault [92] designed a variational autoencoder with continuous and discrete latent variables that corresponded to chroma textures and chord labels and trained a deep classification model in a semi-supervised manner. Furthermore, they claimed that their approach effectively unifies the discriminative and generative methods. Moreover, error analysis revealed that their semi-supervised learning method still confused rare chord types for more popular ones. In addition, it was also affected by the ambiguity between several chord types in the chroma representation and faced difficulties with seventh chords and inversions in a large-vocabulary chord setting.

Odekerken, Koops, and Volk [71] used several symbolic representations in addition to audio to overcome the problem of limited data sets with which most ACR systems are trained. Their system's input consisted of audio, MIDI, and tab files that were collected through web scraping. Tab and MIDI files were then manually matched to the audio files. For training the audio files, ten recent ACR architectures were used. All estimated chord sequences from different representations were integrated into one final output sequence via a data fusion method. Their method improved on average, the weighted chord symbol recall scores of all ten architectures by 3.05%.

Automatic Chord Recognition (2012 - 2020)			
Year	Author(s)	Title	Approach / Key Contribution(s)
2012	[37]	Rethinking Automatic Chord Recognition with Convolutional Neural Networks	CQT, CNN
2013	[10]	Audio Chord Recognition with Recurrent Neural Networks	STFT,DBN,RNN
2015	[84]	Audio Chord Recognition with Hybrid Recurrent Neural Networks	various acoustic models, RNN
2015	[97]	Chord Detection Using Deep Learning	CQT,DNN,SVM, HMM
2016	[48]	The Deep Chroma Extractor	STFT,DNN
2016	[23]	A Hybrid GMM-HMM-Deep Learning Approach for Automatic Chord Estimation with very Large Vocabulary	DBN,MLP, BLSTM-RNN
2016	[49]	A fully convolutional auditory model for musical chord recognition	STFT,CNN,CRF
2017	[34]	Music chord recognition from audio data using bidirectional encoder-decoder LSTMs	CQT,BLSTM
2017	[70]	Residual DNN-CRF Model for Audio Chord Recognition	Mel, DRN,CRF
2017	[93]	Music Chord Recognition Based On MIDI-Trained Deep Feature And BLSTM-CRF Hybrid Decoding	DRN,BLSTM,CRF
2017	[62]	Structured Training for Large-Vocabulary Chord Recognition	CQT,CNN,Bi-GRU
2018	[69]	DNN-LSTM-CRF Model for Audio Chord Recognition	STFT,DNN,CRF, GRU, LSTM, Bi-LSTM
2019	[94]	Automatic Chord Recognition With MIDI-Trained Deep Feature and BLSTM-CRF Sequence Decoding Model	HCQT, CNN, BLTSM,CRF
2019	[74]	A Bi-Directional Transformer For Music Chord Recognition	CQT, BTC
2019	[45]	Large-vocabulary Chord Transcription Via Chord Structure Decomposition	CQT,CRNN,new CSD
2020	[71]	DECIBEL: Improving Audio Chord Estimation for Popular Music by Alignment and Integration of Crowd-Sourced Symbolic Representations	various Audio, MIDI and Tabs ACR techniques + Data Fusion
2020	[92]	Semi-supervised Neural Chord Estimation Based on a Variational Autoencoder with Discrete Labels and Continuous Textures of Chords	semi-supervised learning based on VAE

Table 1: Chronological summary of deep learning based advances in ACR from 2012 - 2020.

1.2 Aim and Goals of the Thesis

In the introduction section, the convergence of ACR systems towards a recurrent convolutional model was discussed. Scholz, Ramalho, and Cabral [81] shed further light on the diminishing performance of ACR systems in the MIREX competition and gave two suggestions for development: ACR systems should either incorporate more musical expert knowledge (especially in the post-processing phase) or explore new deep learning techniques. The latter lend themselves particularly well for this thesis since there are myriad deep learning models that have not been used in chord recognition yet. *Therefore, this thesis's primary goal is to design a fully convolutional encoder-decoder model that is computationally more efficient than the recurrent convolutional models and yet attains comparable results to that of the state-of-the-art approaches without relying on any recurrence*. For this purpose, the architecture of choice is a modified MultiResUnet [40], an extended, and improved version of the U-Net architecture designed to work with very few training samples. To the author's best knowledge, MultiResUnet has not been used in ACR research yet. The proposed model's performance is compared to a robust convolutional recurrent architecture proposed by [62], which is equally well suited for limited- and large-vocabulary chord recognition and shall serve as a baseline throughout the experiments.

Furthermore, since only a small percentage of published studies have reported inversion prediction results, this thesis will address the problem by evaluating model performance across all bass-agnostic as well as bass-sensitive mir_eval chord recognition metrics. Moreover, by extending the proposed model with the structured training approach of the baseline and evaluating their results in a large-vocabulary setting, further insights into both systems' behavior shall be gained.

2 Methods

This section outlines the data sets, data augmentation, proposed architecture, training strategy, and deep learning methods used for the experiments.

2.1 Reference Annotations

One of the first significant efforts to gather reference chord annotations led to the development of the **Isophonics** dataset [42], covering bands such as The Beatles (12 albums, 180 songs), adding up to approximately 8 hours, 8 minutes, and 53 seconds of music, Carole King (one album, 14 songs), and Zweieck (one album, 18 songs) as well as Queen (Greatest Hits I-II, 20 songs, totaling approximately 1 hour, 13 minutes and 51 seconds of music). Due to content access, only 180 songs from The Beatles and 20 songs from the Queen repertoire are used from this data set. Later on, in 2011, the Music and Audio Research Lab (MARL) [67] provided 100 chord annotations from the **RWC-Pop** collection, totaling approximately 6 hours, 46 minutes, and 45 seconds of music. In 2013, [28] provided the chord annotations for the first five albums of **Robbie Williams**, which contain 61 songs with a total of about 4 hours, 17 minutes, and 47 seconds of music. Figure 1 shows the reference annotation for the first 21 seconds of the song-'Good Night' by *The Beatles*. From the figure, it can be seen that the first 0.471 seconds of the song does not contain any chords, hence the no-chord label 'N'. Then, for the next 2.062 seconds, a G: `maj7` chord is played, followed by a `C/5` chord, et cetera.

time	duration	value
0.000	0.471	N
0.471	1.591	G:maj7
2.061	1.881	C/5
3.942	1.800	G:maj7
5.742	1.834	C/5
7.576	1.683	G:maj7
9.260	1.707	C/5
10.966	1.823	G:maj7
12.789	1.939	C/5
14.728	1.950	G
16.678	1.823	B:min7
18.501	1.881	A:min7
...	...	...

Figure 1: Reference annotation for the first 21 seconds of the song- 'Good Night' by *The Beatles*.

A chord in the reference annotation can be any valid string that complies with the formal language defined by [30, 93-106]:

$root : (interval1,\ interval2,\ ...)/bass$

or

$root : shorthand\ \ (extra - intervals)/bass$

The colon ':' separates the root pitch name from a comma-delimited list of intervals written in parenthesis. The forward slash '/' character denotes the bass note if it differs from the root, i.e., the chord type is inverted; *shorthand* is an abbreviation for a given chord type, e.g., the shorthand for major is `maj`; *extra-intervals* is an optional list of added or suppressed intervals. Table 2 lists the 14 chord qualities and corresponding interval list and semitones under consideration for this thesis. The table allows decomposing any chord string to its intervals and semitones. For example, a C major triad with a fifth in the bass can be written as C: `maj`/ 5, which is the third chord symbol[2] shown in Figure 1. The fifth in the bass of C changes the order of its interval list and semitones to (5, 1, 3) and {7, 0, 4}, respectively.

Chord Type		Shorthand	Interval List	Semitones
Triad Chords	Major	*maj*	(1, 3, 5)	{0, 4, 7}
	Minor	*min*	(1, b3, 5)	{0, 3, 7}
	Augmented	*aug*	(1, 3, #5)	{0, 4, 8}
	Diminished	*dim*	(1, b3, b5)	{0, 3, 6}
Sixth Chords	Major Sixth	*maj6*	(1, 3, 5, 6)	{0, 4, 7, 9}
	Minor Sixth	*min6*	(1, b3, 5, 6)	{0, 3, 7, 9}
Seventh Chords	Major Seventh	*maj7*	(1, 3, 5 ,7)	{0, 4, 7, 11}
	Minor Seventh	*min7*	(1, b3, 5, b7)	{0, 3, 7, 10}
	Dominant Seventh	*7*	(1, 3, 5, b7)	{0, 4, 7, 10}
	Diminished Seventh	*dim7*	(1, b3, b5, bb7)	{0, 3, 6, 9}
	Half-Diminished Seventh	*hdim7*	(1, b3, b5, b7)	{0, 3, 6, 10}
	Minor Major Seventh	*minmaj7*	(1, b3, 5, 7)	{0, 3, 7, 11}
Suspended Chords	Suspended Fourth	*sus4*	(1, 2, 5)	{0, 5, 7}
	Suspended Second	*sus2*	(1, 4, 5)	{0, 2, 7}

Table 2: Chord types and corresponding interval list and semitones.

2.2 Data Augmentation

The data augmentation technique used in this thesis follows the observation that not every chord quality in the data set appears in every root position. For example, there are only six root position `minmaj7` chords in the entire data collection, making it impossible for this quality to appear in every root position. Humphrey and Bello [37] and Humphrey [36, pp. 81-82] exploited the linearity of pitch in the constant Q transform and shifted the pitch of their audio signals up or down by one to six semitones so that all chord qualities could be observed in all root positions, and all root, bass, and pitch values would occur. However, this operation does not retain the labels, so they must be adjusted accordingly. Later on , this

[2]The Beatles data set does not use the `maj` shorthand for major triads. In this data set, a major triad without a shorthand implies `maj`.

data manipulation technique was also explored by [49], where it proved necessary to prevent overfitting.

2.3 Pre-Processing

In the input pre-processing step, to better capture harmonic relationships, each audio signal is transformed into a log-frequency magnitude spectrogram via a harmonic constant-Q transform . Constant-Q transform (CQT) is a technique that transforms a time-domain signal $x(n)$ into a time-frequency representation where the center frequencies of the frequency bins are geometrically spaced, and their Q-factor (the ratios of center frequencies to band-widths) of all bins are equal [46].

Brown [12] showed that the linear representation given by the discrete Fourier transform (DFT) yields to components that do not map to musical frequencies and for musical applications, its use is inefficient. In other words, the Fourier transform faces two problems [83]: 1) it does not account for a proper resolution regarding the frequency of different musical intervals. Since octaves are closer in lower and sparser at higher frequencies, a higher resolution would be needed at lower, and a lower resolution would be required at higher frequencies.
2) Implementing a variable window size in the Fourier analysis is a computation heavy and tedious process. The CQT transform, on the other hand, not only accounts for a variable window size but also sets the window size for each frequency.

Consider the geometrically spaced center frequencies f_k as : [83]:

$$f_k = (2^{\frac{k}{b}}) \, . \, f_{min} \;\; (k = 0, ...) \tag{1}$$

where k represents the k-th frequency bin, and b denotes the number of filters per octave. If semitone spacing is desired, b = 12, else if a quarter-tone spacing is required, b = 24; f_{min} designates the minimum frequency of the CQT. Choosing the bandwidth of the k-th filter as :

$$\Delta_k^{cqt} = f_{k+1} - f_k = f_k \, . \, (2^{\frac{1}{b}} - 1) \tag{2}$$

results in a constant Q factor which can be written as follows [12]:

$$Q = \frac{f_k}{\Delta_k^{cqt}} = \frac{1}{(2^{\frac{1}{b}} - 1)} \tag{3}$$

Based on (2) and (3) the CQT of a discrete time-domain signal $x(n)$ is defined by [12]:

$$X(k) = \frac{1}{N[k]} \sum_{n=0}^{N[k]-1} W[k,n] x(n) \exp -j \frac{2\pi Q}{N[k]} n \tag{4}$$

$N[k]$ is the variable window length given by :

$$N[k] = \frac{f_s}{f_n} Q \tag{5}$$

where f_s is the sampling rate. $W[k, n]$ is a window function (usually Hanning). Comparing (4) to the discrete short time Fourier transform [4]:

$$X(k) = \frac{1}{N[k]} \sum_{n=0}^{N-1} W[n] x(n) \exp -j \frac{2\pi k}{N} n \tag{6}$$

shows that the window function in the constant-Q transform is a function of k as well as n. Another difference between (4) and (6) is that in the CQT, since the number of terms varies with k the sum has to be normalized by $N[k]$. However, equation 1 is limited in its coverage of semitones as precise coverage can only be granted for the power of 2 harmonics. Since k and B are integers, f_k can not output the integers 3, 5, 6, 7, 9, etc., making it challenging to capture particularly odd harmonics. The problem was addressed by [9], and led to the formulation of the *harmonic* constant-Q transform (HCQT):

$$f_k = h \ . \ f_{min} \ . \ 2^{\ \frac{k}{B}} \tag{7}$$

As evident by the equation mentioned above, the calculation of the HCQT does not differ much from a standard CQT; the only difference is the scaling of the minimum frequency by the harmonic: $(h.f_{min})$. In addition, the same number of octaves and frequency resolution are maintained across all harmonics. Stacking these scaled CQTs results in a 3-dimensional representation indexed by time, frequency, and **harmonic** ($\mathcal{H}[t, f, h]$), as such, the HCQTs' third axis will contain the audio signal's harmonic components, now localized in the time-frequency domain across the first and second dimensions [25]. This property of the HCQT allows the efficient use of two-dimensional CNNs' 3-D filters, which can now be trained to localize the harmonic components in the time and frequency plane [26].

Figure 2 illustrates the HCQT of the song- 'Good Night' by *The Beatles*, computed at a sampling rate of 44.1 kHz with a hop length of 4096 samples ($\approx$ 93 ms in time), for $h \in \{1, 2, 3\}$: the fundamental (1), and up to two harmonics above the fundamental. The minimum frequency (f_{min}) of the transform is 32.70 Hz at h = 1, with 36 bins per octave spanning over 6 octaves in frequency. Shifting focus towards the figure, a visible frequency shift from $h = 1$ to $h = 3$ can be observed, but the energy across the harmonic axis does not seem to vary significantly.

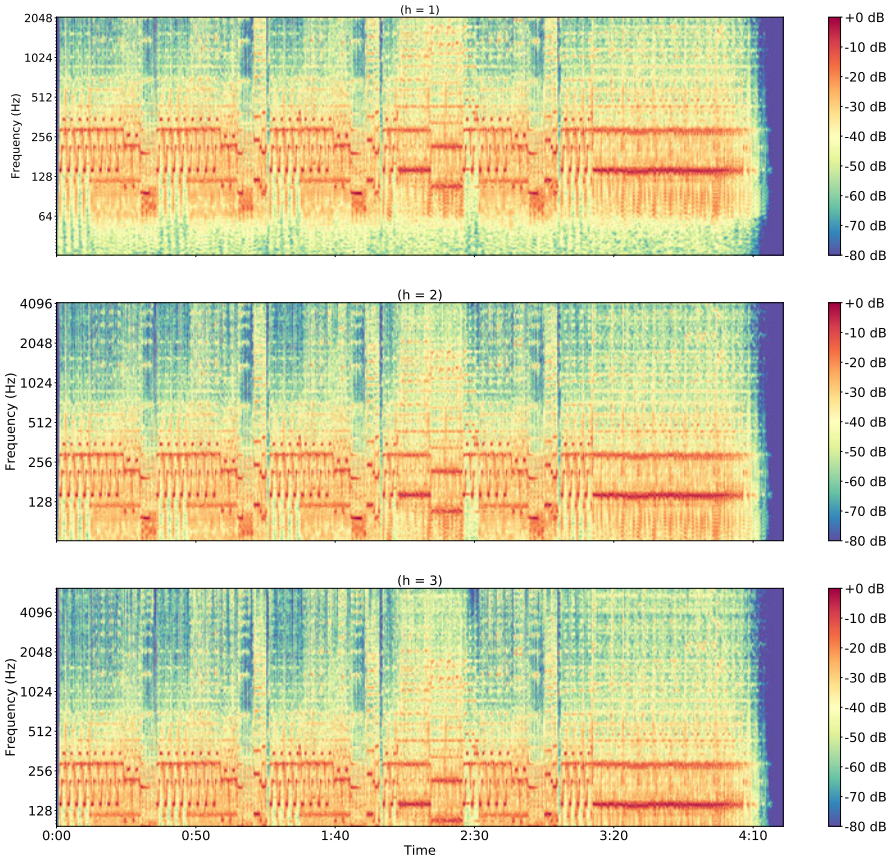

Figure 2: Harmonic constant-Q transform of the song- 'Good Night' by *The Beatles* computed for $h \in \{1, 2, 3\}$ with the minimum frequency of 32.70 Hz at $h = 1$. Sampling rate and hop length of the transform were set to 44.1 kHz and 4096 samples. The transform spans over 6 octaves, with 36 bins per octave.

2.4 Network Architecture

This section briefly introduces the concept of fully convolutional neural networks. It then provides a detailed summary of U-Net and MultiResUnet. Afterward, spatial dropout, structured representation, and the complete set of mir_eval chord recognition metrics will be explained. The section concludes with data set statistics.

2.4.1 FCN

.

Both U-Net and MultiResUnet fall under the umbrella of fully convolutional encoder-decoder architecture. The fully convolutional neural network (FCN) [57] uses, as the name suggests, only convolution, pooling and upsampling layers, i.e., no dense or recursive layers are used in this architecture. However, the use of strided convolutions or upsampling layers produces a final feature map that lacks the necessary spatial resolution for detecting object boundaries.

The removal of max-pooling layers, on the other hand, leads to an underperformance, since the receptive field only increases linearly and not exponentially with the number of layers [43].

2.4.2 U-Net

The U-network architecture [79] shown in Figure 3, is a modified and extended fully convolutional network architecture designed to work with very few training samples. The network architecture comprises a contracting/encoding (downsampling) path and an expansive/decoder (upsampling) path. The contracting path has four blocks; at each block, two unpadded three by three convolutions are applied where each convolution is succeeded by a rectified linear unit (ReLU) [68]. The second convolution layer is followed by a two by two max-pooling operator for downsampling. The stride of all max-pooling layers is set to two. After each max-pooling operation, the number of feature channels is doubled. The bottom-most block is the bottleneck layer, which contains two consecutive three by three convolution layers, each succeeded by a ReLU operation. The expansive path mirrors the contracting path and consists of an upsampling of the expansive path's feature map, concatenated with the corresponding contracting path's feature map, followed by two three by three convolutions, each using a ReLu activation function. The convolutional layer's output in the encoding path is transferred to the decoder before each block's pooling operation. These connections between the encoder and the decoder are called "*skip connections*" and are the novelty of the U-Net architecture and allow the network to retrieve spatial information lost due to downsampling [27]. Concatenating the upsampled feature map with the corresponding contracting path's feature map results in a better localization. The upsampling operator is a two by two transposed convolution [96] that halves the number of feature channels. Finally, the classification layer uses a one by one convolution operation with a sigmoid activation function to generate the output segmentation map.

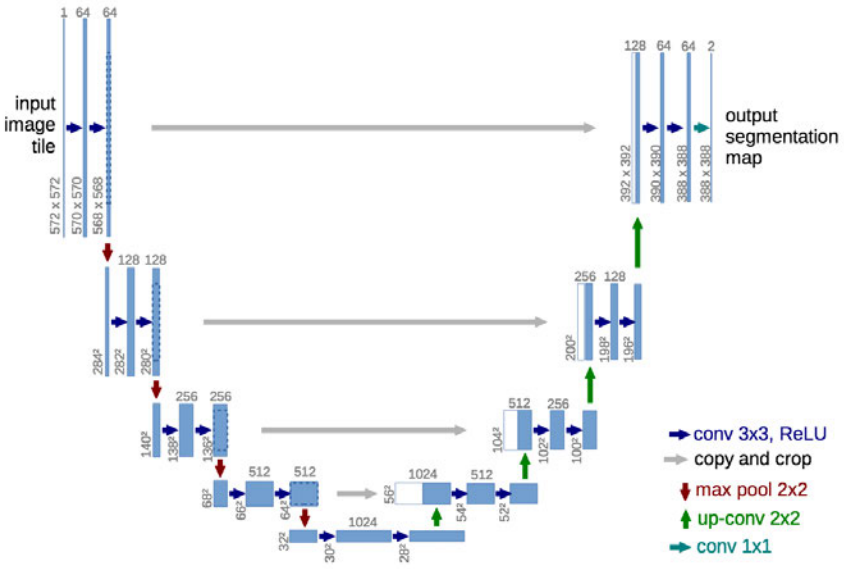

Figure 3: Illustration of the U-Net [79].

2.4.3 MultiResUNet

The MultiResUNet illustrated in Figure 6 is essentially an extended and improved U-Net architecture first proposed by [40]. As mentioned before, the U-Net applied sequence of **two** three by three convolution layers after each pooling and upsampling layer. MultiResUNet, however, applies **three** three by three convolution layers after each pooling and upsampling layer while gradually increasing the number of filters of the convolution layers (from the first to the third convolutional layer). The reason for doing so is the observation that if two convolutional layers follow each other in deep networks, the number of filters in the first one has a quadratic effect over the memory footprint [89]. Therefore, each convolutional layer's number of filters is gradually increased to prevent the earlier layers' memory load from inordinately propagating to the network's deeper layers [40]. Furthermore, the concatenation of all three convolution layers' output allows the extraction of spatial features from different scales hence the name MultiResUnet. Also, a residual connection between the input of the first convolution layer and the three concatenated layers' output is added [31]. The steps described above are visualized in Figure 4. To keep the number of parameters between MultiResUnet and U-Net roughly the same, a W parameter is defined which controls the number of filters inside the MultiRes block and is computed as follows:

$$W = U * \alpha \tag{8}$$

Where U is the number of filters in the U-Net's corresponding layer ([32, 64, 128, 256, 512]), and α is the scalar coefficient set to 1.67. In each MultiRes block the three successive convolution layers have $[\frac{W}{6}]$, $[\frac{W}{3}]$ and $[\frac{W}{2}]$ filters, e.g. starting with the first block and setting U = 32, results in $W = 53.44$. Therefore, for this block, the first, second, and third convolution layers will have 8, 17, and 26 filters. Similar to the U-Net model, W's value is doubled after each max pooling or upsampling layer.

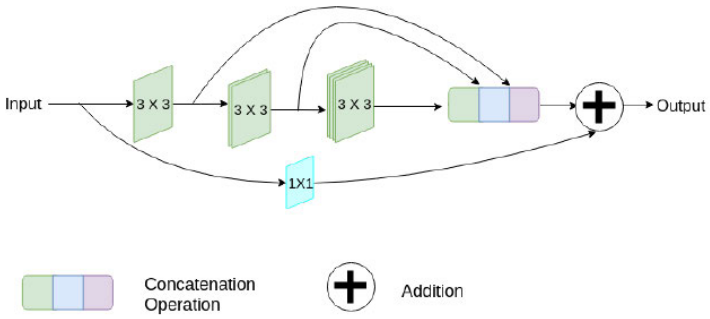

Figure 4: Illustration of a MultiRes block [40] where the number of filters in the following layers is increased, and a residual connection between the block's input and output is added.

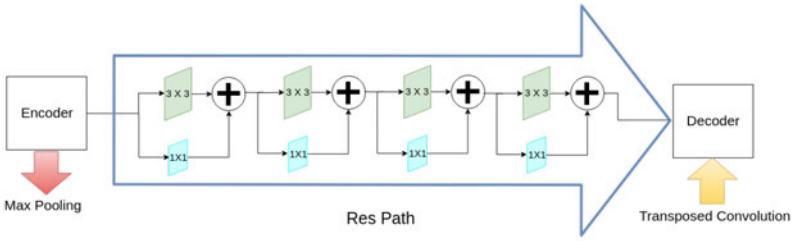

Figure 5: Illustration of a Res path . The encoder features are passed through a sequence of convolutional layers [40]. As with the MultiRes block, residual connections are introduced between the convolutional layers' input and output.

Moreover, in the U-Net architecture, the encoded features preceding the first max-pooling layer are concatenated with the decoder's top-most upsampled features. The features coming from the encoder are computed at an earlier stage of the network and are low-level features. In contrast, the features computed at the last decoder level are high-level features since they go through much more processing before arriving at the decoder's last layer. Hence, there is a semantic gap between the two sets of features being concatenated. Ibtehaz and Rahman [40] state that the fusion of two sets of features coming from two different modalities may cause some discrepancy throughout the learning and can negatively affect the prediction procedure. Nevertheless, the discrepancy is likely to decrease the more downwards the features go in the encoding path. The reason is that the encoder features go through more processing as they approach the bottleneck layer, while at the same time, they are merged with the decoder features that have not yet gone through much more processing. Therefore, the semantic gap between the encoder and the decoder features at the bottom-most skip connection is the smallest while being the largest for the up-most skip connection [40]. Hence, in the MultiResUnet architecture, instead of directly propagating the encoder features to the decoder, each skip connection (here called, Res path) applies a different number of convolution layers to the encoder features. For example, the bottom-most Res path applies one, and the most upward Res path includes four convolution layers. Moreover, additional residual connections are added to the convolution layers in the 'Res path' to facilitate the learning process. Figure 5 illustrates the upmost Res path in MultiResUnet with four convolution layers and added residual connections. Looking at the figure reveals one oddity: the residual path includes a one by one convolution layer, which is missing in the original ResNet [31]. The authors mention that the one by one convolution layer is left there to "maybe" allow for extra spatial information extraction. However, such one by one convolution layers are usually applied where there is a need to reduce or increase the depth dimension of a network (for example, when there are dimension mismatch problems). Nonetheless, this is not the case because the number of input features does not change during the convolution operations, so it is practically left there for experimental purposes.

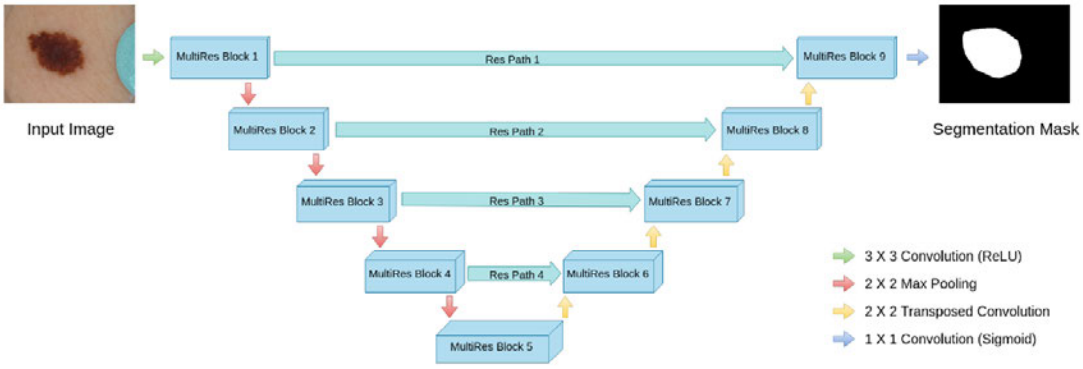

Figure 6: Illustration of the original MultiResUnet [40], which was first proposed for the task of medical image segmentation.

2.5 Preventing overfitting with spatial dropout

The regular dropout [32, 86] is a regularization technique first used to avoid overfitting in fully connected neural networks. Nevertheless, for rather recent CNN models, dropout is partially dismissed, due to its minor performance improvements [54] and is replaced by the batch normalization technique [41]. For a given convolution feature map of size $n_{feat} * time * frequency$, spatial dropout only performs n_{feat} dropout trials and the dropout value is extended across the entire feature maps [90]. Now, suppose adjacent time frames or frequency bins within feature maps are highly correlated (as is normally the case in early convolution layers). In that case, regular dropout will not regularize the activations and otherwise decrease the effective learning rate (over-training) [19]. Hence, using spatial dropout instead of the regular dropout should help promote independence between feature maps. Recently, [54] conducted a series of experiments on various CNN architectures by either adding spatial dropout layers to existing models or replacing standard dropout layers with spatial dropout layers and observed performance improvements for the CNN models. Motivated by the studies mentioned above, a set of experiments was carried out on the MultiResUnet model by either adding dropout or spatial dropout layers and observing the changes in performance. To this end, using a single spatial dropout layer in conjunction with batch normalization layers (which the MultiResUnet already incorporated after all convolutional layers except the classification layer) has led to the best results and is kept for the main experiments.

2.6 Structured Representation

Most ACR systems map all chords to the classic 12 major, 12 minor triads, and one 'no-chord' (N) formulation (overall 25 classes). Sixth , seventh, ninth, eleventh, thirteenth chords and chord inversions are quantized into root triads, e.g. $\mathrm{G:sus2} \rightarrow \mathrm{G:maj}$, $\mathrm{G:dim7} \rightarrow$ G: min or $\mathrm{G:maj(9)/5} \rightarrow \mathrm{G:maj}$. Ignoring information such as suppressed or additional notes introduces label noise and may negatively affect accuracy [38, 62]. The assumption here is that all classes are unrelated (mutually independent). Therefore through this quantization process, all observations are assigned to the corresponding one-of-K classification. The flip side of the coin is that this formulation ignores the chords' inherent structure and weights all errors equally. For example, if the model predicts C: maj7 instead of $\mathrm{A:min7}$ the penalty is just as bad as if it had predicted $C\sharp\mathrm{:maj7}$. However, some mistakes are more severe than others, e.g. C: maj7 and $\mathrm{A:min7}$ share the three notes C, G and E while C: maj7 and $C\sharp\mathrm{:maj7}$ share none, which is not reflected in the one-of-K classification problem. Some studies [94, 93, 62, 23, 45,] focus on large vocabulary chord recognition to avoid the noise introduced by mapping all chords to the classic major-minor formulation, and match the set of active pitches against a larger template. Omitting larger chord templates will map a compound chord such as $A\flat\mathrm{:min(9)/5}$ (A flat minor add 9 with a fifth in the bass) to $G\sharp\mathrm{:min}$. However, $G\sharp\mathrm{:min}$ ($G\sharp$, B and $D\sharp$) neither implies the presence of the added 9 ($A\sharp$) in the chord nor the fifth ($D\sharp$) in the bass, although it is explicitly contained in the reference annotation. The structured representation , on the other hand, is a label decomposition algorithm consisting of two parts: a) *simplification*, which is common to all chord classification models; and b) *encoding*. The following steps can best describe the simplification [62]:

1. Define a mapping of all valid chord [3] types to a finite chord vocabulary.
2. Discard inversions and suppressed or additional notes, e.g.,

   $$A\flat\mathrm{:min(9)/5} \rightarrow A\flat\mathrm{:min}$$

3. Decompose labels into root and pitch classes[4] (w.r.t the roots):

   $$A\flat : min \rightarrow \begin{cases} 8 & \text{root} \\ (0,3,7) & \text{pitch classes} \end{cases}$$

4. Match the set of active pitches against the desired templates [5]
5. Resolve enharmonic equivalencies by mapping the root and matched templates to a canonical form:
   $(8,(0,3,7)) \rightarrow G\sharp : min$
6. If the set of active pitches does not match one of the templates [6], map it to X (the unknown chord symbol).
7. Define a special root/bass category for the non-chord symbol N and map X to an all-zeros pitch vector.

[3] The syntax of valid chord types is defined in Section 2.1.

[4] List of semitones is presented in Table 2.

[5] List of chord types under consideration for this thesis is presented in Table 2.

[6] Power chords, i.e., chords with only the root and the fifth or one-chords (G (1)) are examples of chords that will map to X.

Encoding comprises the decomposition of each chord into a set of musically meaningful sub-components: root pitch class, bass pitch class, and active pitch classes. Root and the bass pitch classes are sparsely encoded as integers in the range of (0, 12). [7] The set of active pitch classes is binary encoded. Figure 7 visualizes the steps described above.

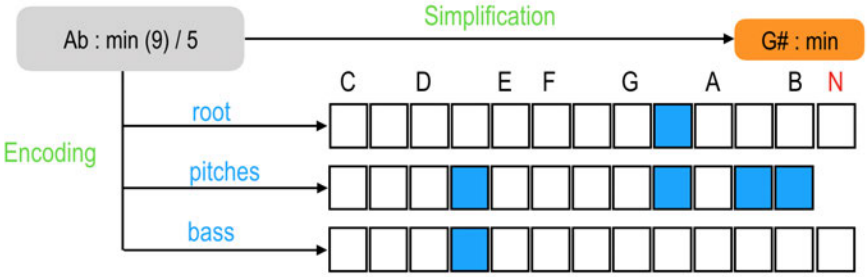

Figure 7: An illustration of the label decomposition algorithm, where target chords are represented as both binary vectors encoding the root, bass and pitch classes, and in simplified canonical form. N, X map to a special root/bass class N, and the all-zeros pitch vector [62].

2.7 Chord recognition evaluation methods

The quality of an automatic chord recognition algorithm is evaluated by comparing a ground truth to a predicted annotation. The MIREX ACE contest uses **chord symbol recall** (**CSR**) for this task [66]. Up until 2013, MIREX used an approximate CSR calculated by sampling both the predicted annotations and the ground-truth every 10ms and dividing the number of correctly annotated samples by the total number of samples [65]. Since 2013, MIREX assumes the ground-truth annotations and the predicted annotations as continuous segmentation of the audio and considers the cumulative length of the correctly overlapping segments for CSR calculation [65]. The segment-based approach is a more precise and computationally more efficient method to calculate the CSR [30, p. 220]. The formula for the segment-based CSR is given by [30, p. 209] :

$$CSR_T(S_E, S_A) = \frac{\sum_{S_A^j} \sum_{S_E^i} |S_E^i \cap S_A^j| \cdot \mathbb{M}_T(S_A^j, S_E^i)}{\sum_{S_A^j} |S_A^j|} \tag{9}$$

The ground-truth annotation A and predicted annotation E are a sequence of the segments S_A and S_E respectively. $|\cdot|$ denotes the duration of a segment. $\mathbb{M}$ is the matching function defined by :

$$\mathbb{M}_T = \begin{cases} 1 & \text{if } X \text{ matches } Y \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

T specifies the comparison method for evaluating the matching function's results and is dependent on the chord vocabulary used.

[7] The last integer is reserved for the non-chord symbol N

Chord Comparison Functions			
Name	Equal	Unequal	Ignored
root	F♯:aug , G♭:min	D:maj/3 , B:maj	——
thirds	G:aug , G:maj	F:maj7 , F:min	——
triads	G:hdim , G:dim	C:aug , C:maj	——
tetrads	A:7 , A:9	D:7, D:maj7	——
mirex	D:maj6 , B:min	D:maj , B:min	——
majmin	C:min7 , C:min	C:maj , C:sus2	dim, sus2, sus4
sevenths	C♯:maj9 , C♯:maj7	C♯:maj7 , C♯:7	dim, sus2, sus4

Table 3: Common chord comparison functions and examples implemented in *mir_eval*

Table 3 lists seven of the most common comparison functions used in ACE. The 'root' comparison function only considers the root enharmonic of a chord spelling, i.e., both F♯: `aug` and G♭: `min` will be mapped to the root F♯ and their qualities, `aug`, and `min` will be discarded. Under this constraint, both will be equal. The 'thirds' function compares chords at the level of their major or minor thirds (root and thirds), e.g., 'G: `aug` has a major third interval and hence is mapped to major and since its root is G, it is equal to G: `maj`. Diminished seventh (`dim7`) chords have a minor third and are mapped to minor, `sus2` and `sus4` chords replace the third with a second and fourth, respectively, which means that they do not have a major or minor quality. However, the 'thirds' function maps such chords to major. The 'triads' function compares chords at the level of major or minor triads (minor, diminished, major, augmented, suspended), e.g., C: `maj7` and C: `7` both share the same root and major triad and are equivalent, while C: `min` and C: `dim` are not. The 'mirex' function compares chords at the pitch class level; two chords are considered equal when they have at least three notes in common. The 'mirex' function is a more relaxed evaluation since it allows for misidentified roots and related chords, e.g., D: `maj6` (D, F♯, A and B) and B: `min` (B, D and F♯) share three pitches and even though do not share the same root are considered equivalent. The 'maj-min' function compares only major, minor, and no-chords (N), and ignores other chords for the evaluation. The 'sevenths' function considers only the major, major seventh, minor, minor seventh, and seventh (dominant seventh) for evaluation. Chords outside of this set are ignored. The 'tetrads' function extends the 'sevenths' rule to all chords within an octave, including sixth and half-diminished chords (minor seventh flat five). Extended chords such as 9's, 11's, and 13's are rolled into an octave where upper voicings are included as extensions [78]. Therefore, comparison at 'tetrads' will not distinguish between A: `7` and A: `13`, i.e., both are equal. The open-source evaluation toolbox, *mir_eval* [78] implements each of the seven comparison functions and their inversions (except for the root and 'mirex', which have no inversions) and is used in this thesis to compare the performance of both architectures under consideration.

If the CSR is to be calculated for the entire data set, it is also weighted according to the duration of each song contained in it. The resulting value is called the **weighted chord symbol recall** (**WCSR**) [65]. The WCSR measures how accurate a chord recognition algorithm is at predicting the right chord for an instant in the audio file [30, p. 214] . However, a human listener might not perceive the annotation with the highest WCSR to be the best. For example, Figure 8 shows the ground-truth annotation and two predicted chord annotation sequences.

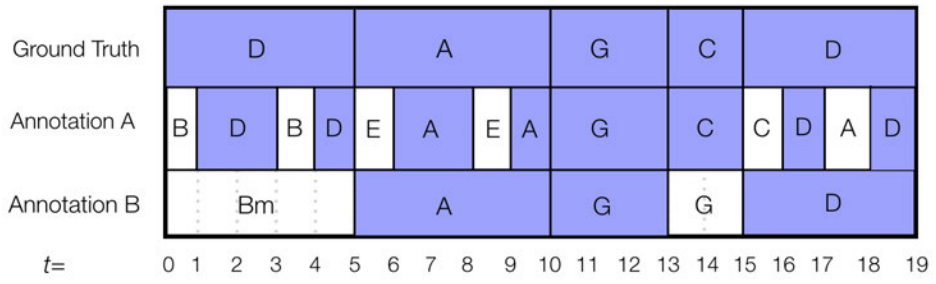

Figure 8: CSR comparison between annotation A, B, and the ground-truth. Annotation A has a better CSR score than annotation B. However annotation A is more fragmented than annotation B. Therefore, a human listener may prefer Annotation B over Annotation A as it is easier to play and more natural to listen to.

Annotation A has a CSR value of $13/19 = 0.685$ because it has 13 correct matching frames out of 19 frames. Annotation B has a CSR value of $12/19 = 0.632$ because it has 12 correct matching frames out of 19 frames. However, for annotation A the chord recognizer has produced a rather fragmented chord sequence. For annotation B, even though the chord recognizer has failed to recognize the right chord label for the first five seconds, it still has managed to predict the correct chord label for the three chord segments A, G, and D. A human listener may prefer the second annotation as it provides a more coherent structure of the underlying harmony, is easier to play and more natural to listen to. Hence, relying on (weighted) CSR alone is not sufficient to illustrate how consistent the algorithm is [30, p. 214]. Therefore, besides weighted CSR, to properly overview different algorithms' performance, another metric is required, which measures the segmentation's quality, i.e., how over- or under-segmented the predictions are. Mauch [59] adopted *the directional Hamming divergence* from image processing as a measure of segmentation quality in chord recognition. Let S_i= [starttime$_i$,endtime$_i$] be an element of a contiguous segmentation S, the directional Hamming divergence measures how much of it is **not** overlapped by the maximally overlapping segment of the other segmentation [p. 51][59]. Afterward, the resulting values are summed over all intervals. Given two segmentations $S^0 = (S_i^0)$, $S = S_i$ the directional Hamming divergence is computed as [30, 59, p. 217, p. 52] :

$$h(S||S^0) = \sum_{i=1}^{N_{S^0}} (\, |S_i^0| - \max_j |S_i^0 \cap S_j| \,) \tag{11}$$

where $|\cdot|$ is the duration of a segment. The directional Hamming divergence describes how fragmented S is to S^0. Let $S^0 = S^G$ be the chord sequence of the ground-truth, and $S = S^A$ the annotation generated by a recognition algorithm, then $h(S^A||S^G) \in [0,T]$ is measure of over-segmentation of annotation A w.r.t to the ground-truth annotation. Conversely, $h(S^G||S^A)$ measures an under-segmentation of A w.r.t the ground-truth annotation. In both cases, the closer the value to zero, the better the transcription. Continuing with the chord sequences in Figure 8:

$$h(S^A||S^G) = \sum_{i=1}^{N_{SG}} (\, |S_i^G| - \max_j |S_i^G \cap S_j^A| \,) = (5-2)+(5-2)+(3-3)+(2-2)+(2-1)+(4-1) = 10 \tag{12}$$

This value indicates that annotation A is over-segmented. However, it is not under-segmented since:

$$h(S^G||S^A) = \sum_{i=1}^{N_{SA}} (\, |S_i^A| - \max_j |S_i^A \cap S_j^G| \,) = 0 \tag{13}$$

Annotation B has an identical segmentation to the ground-truth, therefore: $h(S^B||S^G) = h(S^G||S^B) = 0$. Equation 14 and 15 transform the directional Hamming divergence into a quality measure for over-segmentation and under-segmentation where near-zero values for Over-Segmentation or Under-segmentation correspond to highly over-segmented or under-segmented annotations [59, p. 52].

$$\text{Over-Segmentation}(S||S^0) = 1 - \frac{h(S||S^0)}{\sum_{i=1}^{N_{S^0}} |S_i^0|} \in [0, 1] \tag{14}$$

$$\text{Under-Segmentation}(S||S^0) = 1 - \frac{h(S^0||S)}{\sum_{i=1}^{N_{S^0}} |S_i^0|} \in [0, 1] \tag{15}$$

Equations 16 combines both segmentation measures into a single *Segmentation* measure [71]:

$$Segmentation(S, S^0) = \min \begin{cases} \text{Over-Segmentation}(S, S^0) \\ \text{Under-Segmentation}(S, S^0) \end{cases} \tag{16}$$

Taking the minimum has the advantage that an annotation only gets a high segmentation value if neither over- nor under-segmentation is dominant [59, p. 52] . The *Segmentation* measure can be evaluated without taking the chord symbols in a segment into account.

2.8 Combined Chord recognition F-measure

The previous subsection introduced the WCSR and the Segmentation score as way to evaluate the quality of a chord recognition algorithm . The two scores can be combined into an single chord recognition F-measure:

$$F(G, E) = 2 * \frac{CSR(G, E) * Segmentation(G, E)}{CSR(G, E) + Segmentation(G, E)} \tag{17}$$

Where G and E represent the ground-truth and estimation of the chord recognition algorithm, respectively. The combined chord recognition F-measure is a complementary measure to CSR and provides more information about the performance of the model than using CSR on its own [30, p. 221].

2.9 Data Collection Statistics

This subsection presents some of the statistics for the reference annotations. The complete data set contains 37007 non-unique chord symbols (including the non-chord symbol 'N') with an overall duration of 20 hours, 27 minutes, and 16 seconds, which matches all audio files' aggregate time. The non-chord symbol 'N' is neither a root pitch class nor a chord quality. However, it is still included in the statistics as its own category to put its ratio to other root pitchclasses or chord qualities into perspective. Table 4 shows the frequency and duration statistics for root pitch classes in the data collection. The values in the table account for absolute frequency counts, duration totals, and their percentages. Furthermore, values are also graphically shown in Figure 9. Note that the statistics consider root pitch classes and not root pitch names contained in the data collection. For example, B (2275 chord symbols, 1 hour, 9 minutes and 22 seconds) and Cb (37 chord symbols, 51 seconds) have different pitch names but belong to the same pitch class B/Cb. Therefore, the frequency and duration of these two pitch names are aggregated to form the pitchclass B/Cb (2312 chord symbols, 1 hour, 10 minutes, and 13 seconds). Furthermore, it can be observed that frequency distributions (% frequency in the table) and total time percentages (% time in the table) are quite similar. Howver, the latter is more important than the former since, as mentioned in Section 2.7, it is the cumulative length of the correctly overlapping segments and **not** the frequency of chord symbols that is used for CSR calculation.

Pitch Class	Frequency	% frequency	Aggregate time	% time
A	4986	13.47	10119.047	13.74
G	5164	13.95	9671.23	13.13
D	4923	13.30	9561.910	2.98
C	4078	11.02	8193.20	11.13
E	3643	9.84	8161.95	11.08
F	3160	8.54	5900.82	8.02
B/Cb	2312	6.25	4213.06	5.73
A#/Bb	2069	5.60	3947.43	5.36
F#/Gb	1600	4.32	2962.30	4.02
D#/Eb	1529	4.13	2830.80	3.84
N	871	2.35	2726.65	3.70
G#/Ab	1393	3.77	2679.54	3.64
C#/Db	1279	3.46	2668.35	3.62

Table 4: Statistics for the root pitch classes contained in the merged data collection.

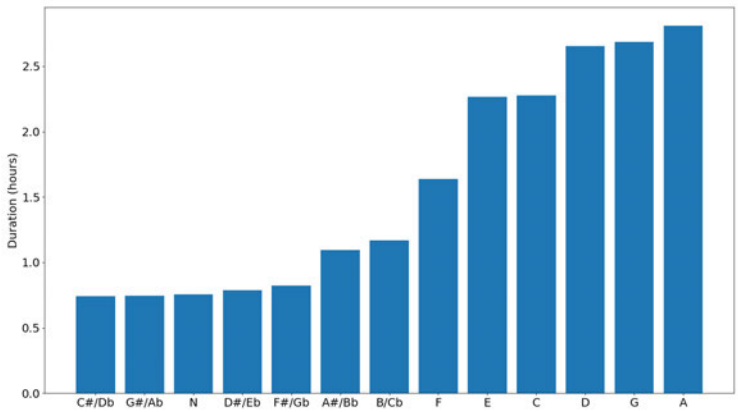

Figure 9: A histogram of the pitch classes contained in the merged data collection.

2.9.1 Root-invariant, bass-blind chord quality statistics

Table 10 presents the root-invariant, bass-blind statistics for 14 chord qualities under consideration. Root-invariant means that for each quality, its total duration is summed over all root pitch classes. Bass-blind accounts for the inclusion of inversions in the statistics, e.g., the aggregate time of maj is the total sum of the duration of all root position and inverted major chord qualities present in the data set. The aggregate time values are also graphically shown by the histogram in Figure 10[8]. Immediately visible from the figure is the extreme class imbalance in the data set. To better understand this class imbalance, chord qualities are grouped into majority classes (`maj, min, min7, 7, N, and maj7`), and minority classes (`maj6, sus4, dim, aug, min6, hdim7, sus2, dim7, minmaj7`). Firstly, it can be observed that there is a relative class imbalance between these two groups. The ratio of aggregate time, for example, between the most and least common chord qualities, `maj` and `minmaj7` is nearly three orders of magnitude ($\frac{42239.64\ sec}{47.95\ sec} \approx 900$). Secondly, there is an overall visible lack of data for minority classes (absolute class imbalance). For example, the total duration of all minority classes accounts for only 4.03 % of total time of the entire data set. Further investigation into the data collection reveals the occurrence of some chord qualities in only the same small number of tracks, e.g., the chord qualities `majmin7`, `sus2`, and `hdim7` appear in 3, 22, and 23 tracks, thus limiting these chord qualities' variability.

Quality	Frequeny	% frequency	Aggregate time	% time
maj	21221	57.34	42239.64	57.36
min	6235	16.85	12264.81	16.66
min7	2415	6.53	4426.24	6.01
7	1981	5.35	4042.83	5.49
N	871	2.35	2726.64	3.70
maj7	1086	2.93	2334.16	3.17
maj6	508	1.37	877.30	1.19
sus4	412	1.11	678.46	0.92
dim	263	0.71	383.67	0.52
aug	192	0.52	322.07	0.44
min6	112	0.30	210.13	0.29
hdim7	91	0.25	162.31	0.22
sus2	104	0.28	154.48	0.21
dim7	70	0.19	128.41	0.17
minmaj7	30	0.80	47.95	0.07
Others	1416	3.12	2637.19	3.58

Table 5: Root-invariant, bass-blind statistics for the 14 chord qualities in the merged data collection.

[8]The entity 'Others' in the table includes all chord qualities that are not examined for this thesis.

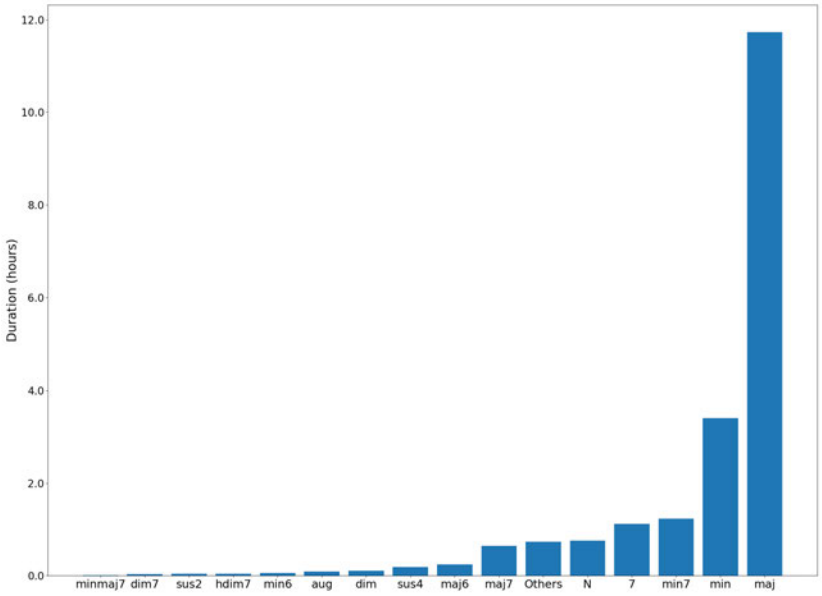

Figure 10: A root-invariant histogram of the qualities contained in the merged data collection.

2.9.2 Root-invariant, bass-sensitive chord quality statistics

Table 6 presents the root-invariant, bass-sensitive statistics for 14 inverted chord qualities under consideration. The aggregate time values in the table are also graphically shown by the histogram in Figure 11. Summing all time-percentage values in the table (% time) reveals that total duration of all inverted chord qualities account for approximately 8.054 % of total duration of the whole data set. Visible from Figure 11, is the presence of relative, and absolute class imbalance discussed in the previous section.

Quality	Frequency	% frequency	Aggregate time	% time
maj/	2705	7.31	3991.91	5.42
N	871	2.35	2726.64	3.70
min/	725	1.96	971.04	1.32
7/	154	0.42	212.21	0.29
maj7/	110	0.30	210.09	0.29
min7/	141	0.38	197.26	0.27
maj6/	65	0.18	91.07	0.12
min6/	36	0.10	54.00	0.07
sus4/	33	0.09	43.07	0.06
minmaj7/	24	0.06	38.88	0.05
dim/	24	0.06	38.64	0.05
hdim7/	19	0.05	28.22	0.04
dim7/	18	0.05	26.28	0.04
aug/	11	0.03	20.46	0.03
sus2/	2	0.01	3.262	0.004

Table 6: Root-invariant, bass-sensitive statistics for triads and tetrads contained in the merged data collection. '/' denotes the inversion symbol, i.e., chord types presented here are in their inverted form.

.

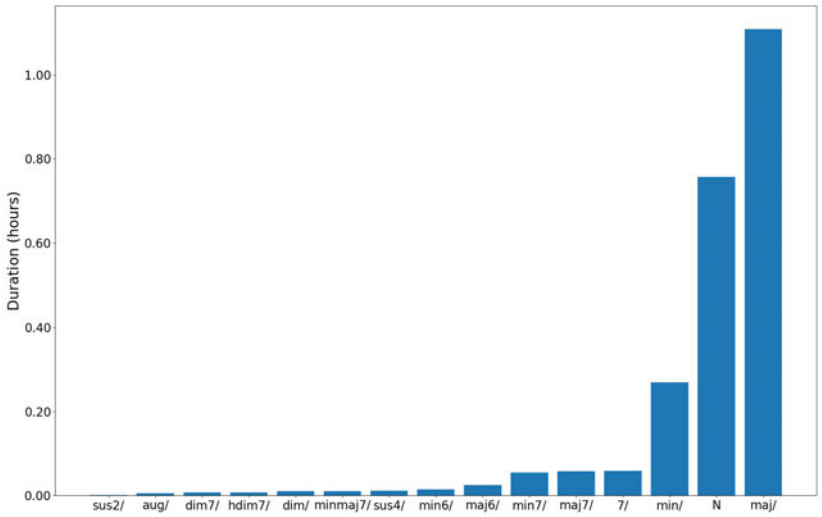

Figure 11: A root-invariant, bass-sensitive histogram of inverted triads and tetrads contained in the merged data collection. '/' denotes the inversion symbol, i.e., chord types presented here are in their inverted form.

The relative class imbalance can be best described by the ratio between the most and least common inverted chord qualities: `maj/` and `sus/2` , which is more than three orders of magnitude ($\frac{3991.91\ sec}{3.262\ sec} \approx 1224$). Indicative of the absolute class imbalance is the overall lack of data for inverted chord-types: `sus2/, aug/, dim7/, hdim7/, dim/, minmaj7/, sus4/, min6/, and maj6/`. For example, the total duration of all rare inverted chord types mentioned above accounts for about 0.46 % of the entire data set's duration.

3 Experimental Setup

3.1 Input Pre-Processing Setup

Table 7 lists the input pre-processing setup used during the experiments. Each audio file and its corresponding chord annotation are first pitch-shifted up or down by one to six semitones. Furthermore, all pitch-shifted and original audio files are transformed into a log-power harmonic constant-Q spectrogram (phase information is discarded) with a minimum frequency of 32.7 Hz, 36 bins per octave, spanning over 6 octaves, and clipped at 80 dB (the magnitude is scaled to dB) below the peak. All audio signals are analyzed at a sampling rate of 44.1 kHz with a hop length (the number of samples between CQT frames) of 4096 samples. The resulting frame rate is approximately $\frac{44100}{4096} \approx 10.77$ Hz. The amount of frequency over-sampling, i.e., bins per semitone (over_sample) and the number of harmonics (h) used for the HCQT calculation, are both set to 3. The number of frequency bins (n_{bins}) used as an input to the two models is given by equation 18 where 12 is the number of semitones in an octave. Pitch-shifting data augmentation is performed with the muda package [60]. HCQT calculation is facilitated using Librosa [63] .

$$nbins = n_{octaves} * 12 * over\_sample = 6 * 12 * 3 = 216 \tag{18}$$

	Setup
input feature	log-magnitude
f_s	44100 Hz
hop length	4096 samples
f_{ps}	10.77 Hz
f_{min}	32.7 HZ
$n_{octaves}$	6
bins per octave	36
over_sample	3
h	3
n_{bins}	216

Table 7: Input Pre-Processing Setup

3.2 Modified MultiResUnet

The configuration of the original MultiResUnet depicted in Figure 6 with its four max-pooling layers in the down-sampling path and four transposed convolution layers in the up-sampling path and $U = 32$ for the first MultiRes block results in a network that has far too many parameters compared to the baseline. Therefore, to achieve a fair comparison between the two models (in terms of the parameter count), the number of max-pooling layers in the down-sampling path, and the number of transposed convolution layers in the up-sampling path is reduced to three. The stride for each max-pooling layer is set to (1, 2), i.e., pooling is only performed on the frequency and not the time domain as pooling on the time domain slightly degraded the results. The stride for the transposed convolution layers is set to (1, 2) accordingly. The number of filters in the first MultiRes block is set to 16, instead of 32. In addition, the activation function of all three by three convolution layers is changed to

ELU [22]. The steps describe above are summarized in Table 8 [9]. Furthermore, an image pyramid is applied to MultiRes block1 and MultiRes block 2 [2]. The output of the last MultiRes block enters a **spatial dropout** layer with a rate of 0.2 (20 %). Figure 12 illustrates the spatial dropout and the structured representation for the modified MultiResUnet. The color-coded header for each box specifies the operation's type. Each box's body describes the configuration of that operation, e.g., the 'axis' parameter inside the batch normalization boxes defines on which axis batch normalization is performed on. The structured training approach predicts for each frame t, the root pitch class, the bass pitch class, and the set of active pitch classes from the spatial dropout layer's output. Root and bass prediction are modeled as a multi-class prediction, each using a soft-max non-linearity (*conv2d_44*, *conv2d_45*). Pitch class estimation is modeled as a multi-label prediction, and therefore uses a sigmoid non-linearity (*conv2d_43*). The output of the three convolutional layers are concatenated with the spatial dropout layer's output to construct the structured representation from which the final chord label is predicted (*conv2d_46*) [62]. Finally, the average pooling layers perform downsampling on the frequency domain (reduce the dimension from 216 to 1), and the lambda layers squeeze the output of the average pooling layers, i.e., they reshape the output of the average pooling layers, so it matches the shape of the encoded labels.

Modified MultiResUnet					
Block	Layer (filter size)	#filters	Path	Layer (filter size)	#filters
MultiRes Block 1 MultiRes Block 7	Conv2D (3,3) Conv2D (3,3) Conv2D (3,3) Conv2D (1,1)	4 8 13 25	ResPath 1	Conv2D (3,3) Conv2D (1,1) Conv2D (3,3) Conv2D (1,1) Conv2D (3,3) Conv2D (1,1)	16 16 16 16 16 16
MultiRes Block 2 MultiRes Block 6	Conv2D (3,3) Conv2D (3,3) Conv2D (3,3) Conv2D (1,1)	8 17 26 51	ResPath 2	Conv2D (3,3) Conv2D (1,1) Conv2D (3,3) Conv2D (1,1)	32 32 32 32
MultiRes Block 3 MultiRes Block 5	Conv2D (3,3) Conv2D (3,3) Conv2D (3,3) Conv2D (1,1)	17 35 53 105	ResPath 3	Conv2D (3,3) Conv2D (1,1)	64 64
MultiRes Block 4	Conv2D (3,3) Conv2D (3,3) Conv2D (3,3) Conv2D (1,1)	35 71 106 212			

Table 8: Modified MultiResUnet Architecture Details

[9] For a comparison between the original and modified MultiResUnet see, [40]

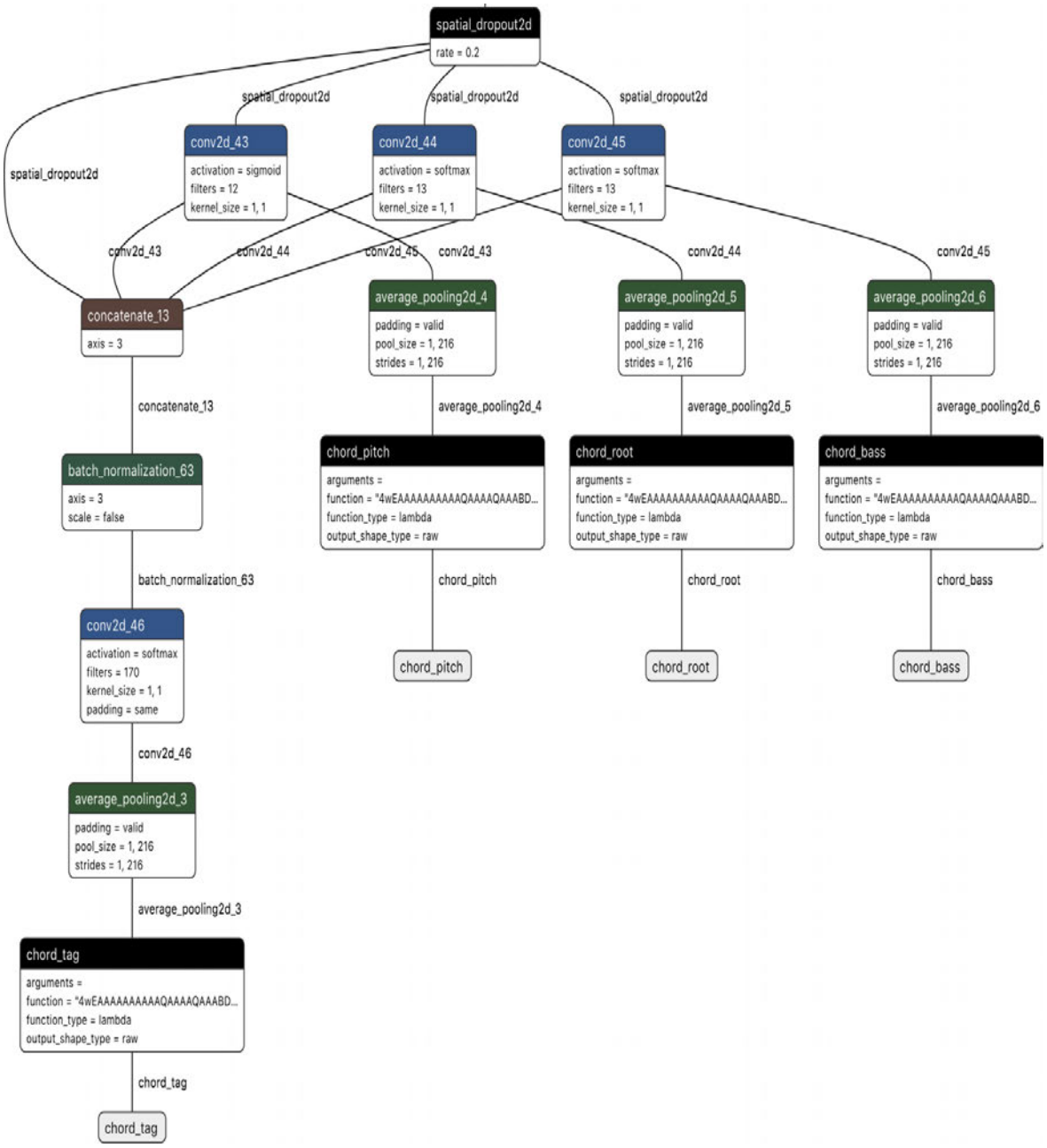

Figure 12: Spatial dropout and classification heads of the modified MultiResUnet.

3.3 Baseline

The re-implemented baseline used for the experiments is the convolutional and recurrent estimators for music analysis model (Crema) [61], which is the updated, and optimized model first proposed by [62]. Figure 13 provides a detailed summary of the model. The model has nine layers (not counting reshaping/merging and batch norm). The first convolution layer uses only one filter with a kernel size of 5 * 5, and the second convolution layer includes 72 filters with a kernel size of 1 * 72. Both convolutional layers use a relu activation function [68] and construct the encoding part of the model. The decoding part comprises two bi-directional gated recurrent units (Bi-GRU) [21]. Crema's structured training approach is similar to that of the MultiResUnet described above but relies on fully connected layers for root, bass, pitch, and final chord label prediction.

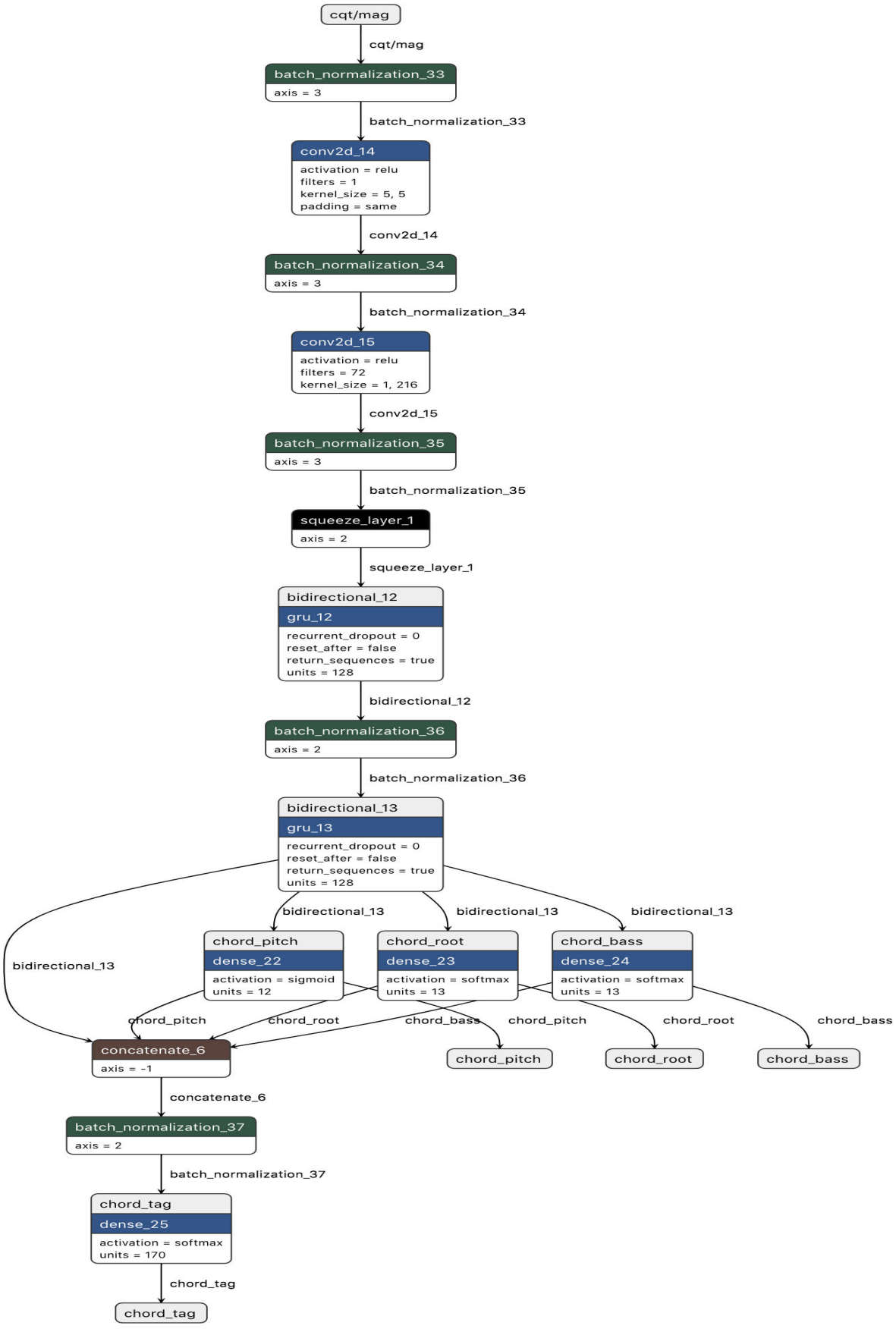

Figure 13: Summary of the convolutional and recurrent estimators for music analysis model (Crema)

3.4 Training

Both models are trained on 6 second patches equaling 64 frames:

$$floor(6 * f_s * \frac{1}{\text{hop length}}) = floor(6 * 44100 * \frac{1}{4096}) = 64. \tag{19}$$

Furthermore, both models are trained using a batch size of 16 with each epoch having 512 batches. Hence, both models' input has the shape: (16, 64, 216, 3). 16 is the batch size, 64 is the number of time frames (6 seconds), 216 is the number of frequency bins, and 3 stands for the number of channels. Models are trained in a supervised manner using the Adam optimization method with default parameters (learning rate at start = 10^{-3}). The learning rate is reduced if there is no improvement in the validation score after 10 epochs. The models are trained for a maximum of 100 epochs. Early stopping is activated if the validation accuracy does not improve for 20 epochs. During training, each model learns to jointly estimate the structured representation and chord labels where the former learns to minimize the sum of all losses, i.e., root, pitch, bass, and decoder losses across all outputs [62]. The loss function used for root pitch class, bass pitch class, and final chord label prediction is the sparse categorical cross-entropy which is given by [8, p. 209] :

$$\ell(\theta) = \frac{-1}{n} \sum_{t}^{n} \sum_{j}^{c} y_{tj} log(p_{tj}) \tag{20}$$

where the double sum is over the time frames t, with n denoting the number of frames, j represents the categories with c denoting the number of the categories (classes), and y is the label and $p_{tj} \in (0, 1) : \sum_j p_{tj} 1 \forall t, j$ is the prediction of the model for a frame. 'Sparse' indicates that the labels are integer encoded rather than one-hot encoded.

The loss function used for pitch class prediction is the binary cross-entropy, which is a special case of equation 20 for $c = 2$ [8, p. 206]:

$$\ell(\theta) = \frac{-1}{n} \sum_{t}^{n} [y_t \log(p_t) + (1 - y_t) \log(1 - p_i)] \tag{21}$$

For c = 12, the binary cross-entropy sets 12 separate binary classification problems where each binary classifier is trained independently. Then the loss is summed over the different binary problems, producing a multi-label for each frame. The models are implemented in Tensorflow 1.9 [1] (tf.keras 2.1).

3.5 Evaluation

The proposed system and the re-implemented baseline are evaluated using 5-fold cross-validation [87] on the compound data set described in Section 2.1. $\frac{1}{4}$ of each training set is hold out for validation where there is no overlap between the training and test set with each fold having roughly the same number of tracks. In addition, for the Beatles and Robbie Williams data set, each split has the same album distribution. Both systems are compared using the methods described in Section 2.7. The average training time per epoch for the MultiResUnet and Crema is roughly six and 16 minutes.

4 Results & Discussion

4.1 Main Results

Figure 14 presents the main results of the evaluation. Except for the last three subplots in the third row, each subplot reports the *median* [10] weighted recall scores of both models for a specific comparison function computed by the mir_eval library. The last three subplots, namely, *underseg*, *overseg*, and *seg* show the median segmentation results. Immediately visible from the figure is the better performance of the proposed model across all mir_eval metrics.

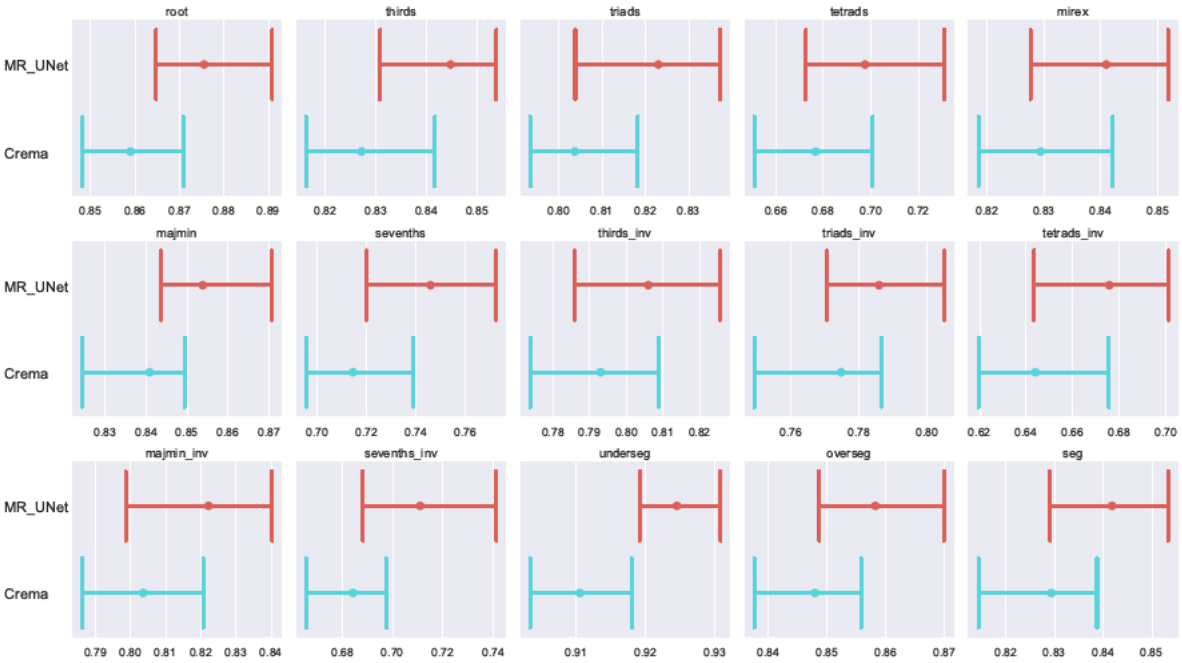

Figure 14: Median weighted symbol recall and median segmentation scores for the MultiResUnet and Crema. Each dot designates the *median* score over all test points. The Error bars cover the 95% confidence interval estimated by bootstrap sampling [62]. *MR_Unet* represents the modified MultiResUnet proposed for this thesis; *Crema* denotes the convolutional recurrent chord estimator for music analysis (the baseline).

Method	root	thirds	triads	majmin	mirex	sevenths	tetrads
MultiResUnet	**0.875**	**0.845**	**0.823**	**0.854**	**0.841**	**0.745**	**0.697**
Crema	0.859	0.828	0.804	0.841	0.83	0.714	0.677

Table 9: Median weighted recall scores of the bass-blind comparison methods for models under consideration.

Table 9 lists the results shown in Figure 14. There are several observations that may be drawn from this table:

[10]The median is chosen to reduce the influence of erroneous and problematic reference annotations reported by [38]. Moreover, similar median weighted recall scores have been reported by [62, 71]. For the sake of completeness, mean recall and segmentation scores are graphically shown in appendix A.

i) both models' highest WCSR achieved is for the 'root' comparison with MultiResUnet yielding a 1.6 % improvement over the baseline.

ii) WCSR scores for the 'thirds' and 'triads' drop compared to the 'root' with MultiResUnet, still gaining a 1.7 % and 1.9 % improvement over the baseline.

iii) WCSR for the 'triads' is lower than the 'thirds', which is expected because the 'thirds' comparison is solely based on the root and the presence or absence of the minor third scale degree. Thus, diminished chords which do have a minor third scale degree will be grouped with minor chords (`min`) and all other chords, for example, augmented chords (`aug`) with major (`maj`). Therefore, 263 `dim`, and 192 `aug` chords, which make up 0.96 % of the total duration of the entire data set, are now considered `min`, and `maj`, respectively. As a result, neither model will be penalized for predicting, for example, a `dim` instead of a `min` chord. However, the 'triads' comparison distinguishes between `maj`, `min`, `aug`, and `dim` chords, and will include the previously excluded 'dim', and 'aug' chords for evaluation. As a result, both models can be penalized for predicting the wrong triad, hence the drop in WCSR.

iv) WCSR scores for the 'majmin' comparison is higher than the 'triads' with the MultiResUnet yielding a 1.3 % improvement over Crema. However, this is not surprising since `maj`, and `min` chords alone make up 74.02 % of the entire data set's total duration.

v) WCSR scores for the 'sevenths' drop significantly compared to the other methods discussed so far, with the MultiResUnet showing a 3.1 % improvement over the baseline. The 'sevenths' extends the 'majmin' comparison by including the `maj7`, `min7`, and `7` chords for evaluation. The total duration of these seventh chords makes up to 14.67 % of the duration of the entire data set for which the models can now be penalized for.

vi) WCSR scores for the 'tetrads' comparison follow the same trend as the 'sevenths' with MultiResUnet yielding a 2 % improvement over Crema. The 'tetrads' comparison extends the 'sevenths' rule by considering chords at the level of their entire quality within a single octave. Including new chord types such as `min6`, `maj6`, `hdim7`, `dim7`, and `minmaj7` result in lower WCSR scores than the 'sevenths' comparison. Overall, it can be concluded that both models are:

- quite robust at estimating major and minor triads
- better at estimating major and minor than augmented and diminished triads
- better at estimating triads than tetrads

Comparison methods discussed so far are all bass-blind, i.e., only root position chords are considered for evaluation. Algorithms that can not generate inversions always estimate root position chords by default. For the inversion-sensitive comparisons, both models first have to find the most probable bass note, and if that bass note is within the detected quality, it will be predicted as an inversion. One small implementation detail worth mentioning is that inversion prediction does not require creating new vocabulary entries as the bass prediction of the structured training approach can be used to predict bass notes, which are then added to the models' predicted annotations. WCSR scores for the bass-sensitive comparison methods are listed in Table 10. From the table, it can be observed that the results for inversion-sensitive comparisons are lower than the inversion-agnostic ones. One possible explanation for this

Method	thirds_inv	triads_inv	majmin_inv	sevenths_inv	tetrads_inv
MultiResUnet	**0.806**	**0.785**	**0.822**	**0.711**	**0.676**
Crema	0.796	0.744	0.803	0.684	0.644

Table 10: Median weighted recall scores of the bass-sensitive comparison methods for MultiResUnet and Crema.

drop is the small ratio of chords in their inverted form to root position chords. For example, the total duration of all inverted chord types considered for evaluation makes up less than 9 % of the entire data set's total duration.

Therefore the possibility of wrongfully estimating a bass on a root position outweighs the benefit of possibly estimating the right inversion [77]. Furthermore, the problematic inter-, and more importantly, absolute class imbalance among these inverted chord types only adds to the problem hindering models from learning the subtle differences between some of these inverted chord types, for example, 'dim/' vs. 'hdim7/' vs. 'dim7/'. From the results shown in Tables 9 and 10, it can be concluded that extending the chord vocabulary towards inversions will not necessarily result in better recognition but provides a much more detailed view of the algorithm's performance under consideration.

Method	underseg	overseg	seg
MultiResUnet	**0.924**	**0.859**	**0.841**
Crema	0.910	0.848	0.830

Table 11: Segmentation scores for models under consideration.

In Section 2.7 it was stated that WCSR alone is not sufficient by itself to show how consistent chord recognition algorithms are. Therefore, three metrics were introduced to measure an algorithm's under-, over-, and overall segmentation quality. Table 11 reports the results of this evaluation, which suggest that chord estimates generated by both models tend to be over- rather than under-segmented, with the MultiResUnet achieving better scores across all three metrics. The final segmentation score of the proposed model shows a 1.1 % improvement over the baseline, i.e., chord estimates produced by MultiResUnet have a better segmentation quality. This is an important finding as it showcases the potential of segmentation architectures in sequence prediction tasks where RNNs are ought to perform better. Also, MultiResUnet requires 51971 fewer training parameters than the baseline, while training time per epoch is reduced by **half**.

Section 2.8 introduced the combined chord recognition F-measure as a means to provide more information about the performance of algorithms. Tables 12 and 13 present the F-measure scores of both models under consideration. Results are in accordance with WCSR scores listed in Tables 9 and 10. Both models are better at predicting 'root', 'majmin', and 'triads' than 'sevenths and 'tetrads'. Furthermore, both models show a better performance for bass-blind rather than bass-sensitive methods with MultiResUnet yielding better F-measure scores across all mir_eval metrics. However, MultiResUnet's higher F-measure scores were

expected as the combined chord recognition F-measure is the harmonic mean of chord symbol recall and segmentation, where the model had already achieved better results in both cases.

Method	root	thirds	triads	majmin	mirex	sevenths	tetrads
MultiResUnet	**0.854**	**0.834**	**0.823**	**0.840**	**0.834**	**0.777**	**0.755**
Crema	0.839	0.823	0.801	0.825	0.823	0.762	0.731

Table 12: F-measure scores of the bass-blind methods for models under consideration.

Method	thirds_inv	triads_inv	majmin_inv	sevenths_inv	tetrads_inv
MultiResUnet	**0.8101**	**0.802**	**0.817**	**0.759**	**0.734**
Crema	0.799	0.784	0.801	0.740	0.716

Table 13: F-measure scores of the bass-sensitive methods for models under consideration.

4.2 Error Analysis

The error analysis presented in this section is motivated by the huge class imbalance in the data set, which makes WCSR, by definition (see Section 2.7), put greater emphasis on the majority classes. Therefore, performing **strict** root and quality-wise error analysis is crucial to gain further insights into model performance on rare chord types. As a result, the normalized root and within-root quality confusion matrices [62] are presented in this section, leading to a better understanding of the models' errors during prediction. The root confusion matrix for MultiResUnet is shown in Figure 15. The x- and y-axis represent the estimated and actual root pitch classes, respectively. The main diagonal shows how accurately the model could estimate the actual root pitch classes. Except for D♯, the higher accuracy for A, G, C, D, and F is in agreement with the occurrence of these pitchclasses in the data collection (see Table 4). One visible trend from the figure is the confusion of most roots towards their perfect fourths (P4) and perfect fifths (P5). For the root C, for example, 2% and 3% of the errors are towards E, and G, i.e., its P4 and P5. One possible explanation for this is the strong correlation of the fundamental frequency of the perfect fifth f^5 to that of the root f^1 which is given by [73]:

$$f^5 \approx \frac{3}{2} f^1 \tag{22}$$

The result is an overlap of every even harmonic of the fifth with the root note's harmonic. This overlap may confuse the model towards estimating a P5 instead of the actual root note. The normalized root confusion matrix for the baseline is depicted in Figure 16. Similar to Figure 15, the confusion of roots here are mostly towards their P4 and P5, and the higher accuracy for A, G, C, D, and F is following their natural bias in the data set.

The within-root quality confusion matrices for MultiResUnet, and Crema are depicted in Figures 15 and 16, and are computed as follows: for each frame of a test track, its reference label is first simplified according to the simplification procedure described in Section 2.6, and then the simplified reference label is compared to the label predicted by the model if the roots match [62] . Mathematically, this can be expressed as $P[estimated\_quality = reference\_quality | estimated\_root = reference\_root]$. Finally, the results are summed

over all tracks and normalized by the frequency of the qualities in the reference annotations [62]. The first and second columns in MultiResUnet's within-root quality confusion matrix reveal a natural bias towards predicting more major and minor chords. This is expected as `maj`, and `min` chords alone make up to 74.02 % of the entire data set duration. The confusion of minor chords (`min`) is mostly towards major (`maj`). Furthermore, the performance for dominant seventh chords (`7`) seems to be relatively low, and 81 % of actual `7` frames are predicted as `maj`, even though `7` chords are the fourth most frequent chord quality in the data set. The confusion of `7` to `maj` is likely the result of V vs. V7 confusion. For example, the triad built on the fifth scale degree of a major scale is a major triad (V); however, the tetrad built on the fifth scale degree of a major scale is a dominant seventh (V7) chord. As a result, the model might have confused these two chord types and thus mapped the `7` to `maj`. Note that some confusions can be understood as simplifications (see Section 2.6): e.g., (`maj7`, `7`) to `maj` or (`min7`, `minmaj7`) to `min`. Nevertheless, the model still struggles with rare classes such as `min6`, `aug`, `sus2`, and `sus4` chords. Moreover, the model resolves `hdim7`, and `dim7` chords towards `min`. One possible explanation is that the frequency of these rare chord qualities in the data set is far too low for the fully convolutional model to learn the relevant hierarchical features necessary for their classification, and, therefore, resolves them towards the most frequent chord classes. Likewise, 99 % of the frames classified as 'X' (unknown chords) map to the most frequent chord quality, namely the `maj` chord. The within-root confusion matrix of the Crema model depicted in Figure 18 shows a similar bias towards `min`, and `maj` chord classes. However, the Crema model shows improvements for the rare chord classes: `dim`, `hdim7`, `dim7`, `sus2`, and `sus4`. Moreover, the entries in the first two columns of Crema's within-root quality confusion matrix reveal a reduction of confusion to `maj` and `min`, which indicates that Crema is more robust towards the inherent quality bias in the data set. The comparison of within-root confusion matrices, and WCSR scores listed in Table 9 indicates that strict quality-wise recall may have an inverse relationship with weighted average recall. Acknowledging this relationship, model selection may conclusively be a function of the application. For example, what should the model under consideration excel at? Classifying simpler chords, i.e., major or minor chords? Or should the model be able to classify rather rarer chord types, i.e., `minmaj7`, and `dim7`?

Moreover, it should be mentioned that the number of MultiRes blocks and Res paths and the number of filters within each one has been reduced in the original MultiResUnet to achieve a fair comparison between the models (in terms of parameter count), i.e., the original architecture has not necessarily been optimized, but instead shrunk to meet the criteria, which may have had a negative impact on the final result.

Finally, in the next section, the thesis is concluded with a two-track analysis since the predictions of two computationally different models allow to investigate to what extent they coincide.

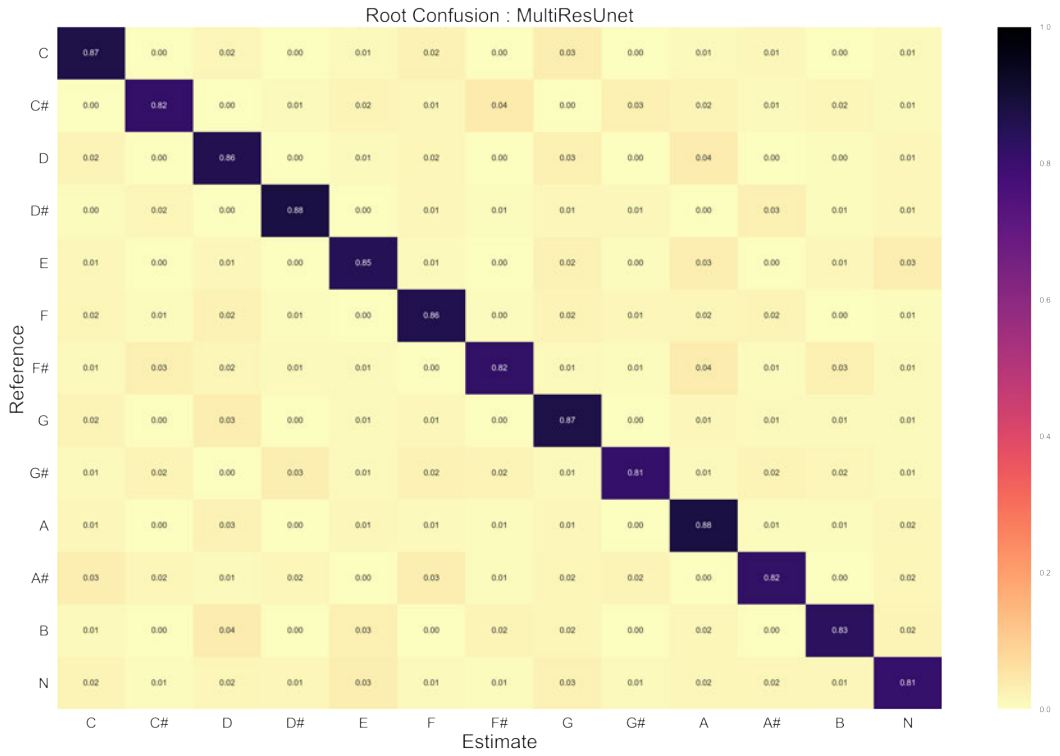

Figure 15: Frame-wise root confusion matrix for Crema. Confusion of most roots is towards their perfect fourths and perfect fifths.

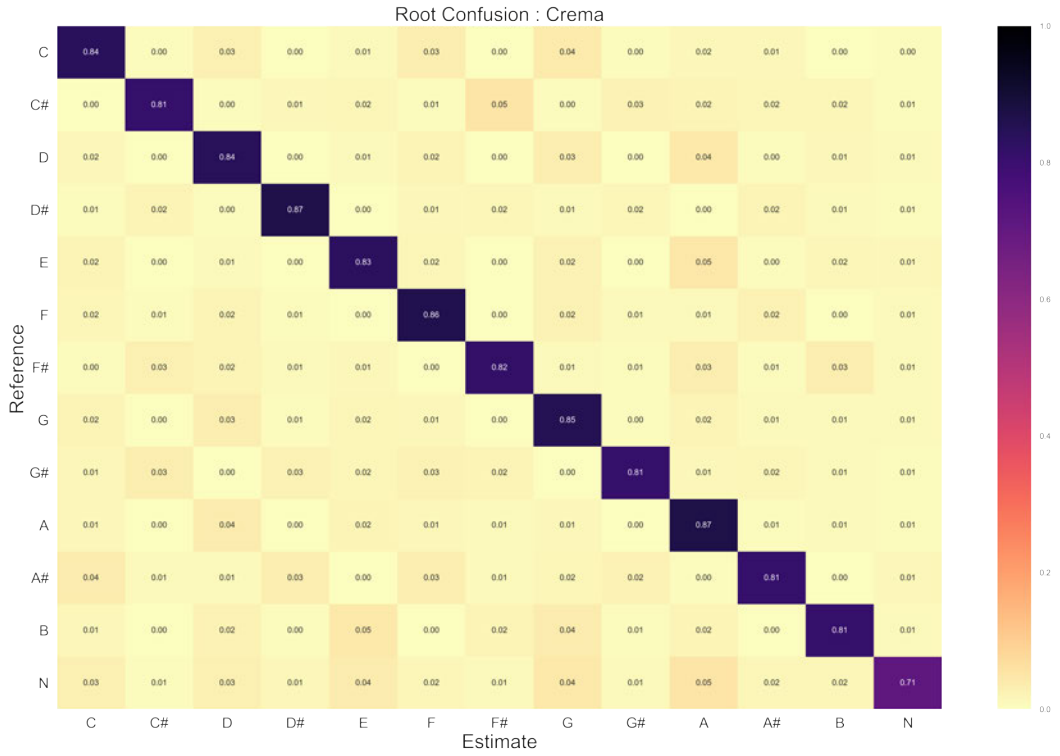

Figure 16: Frame-wise root confusion matrix for the Crema model. Confusion of most roots is towards their perfect fourths and perfect fifths.

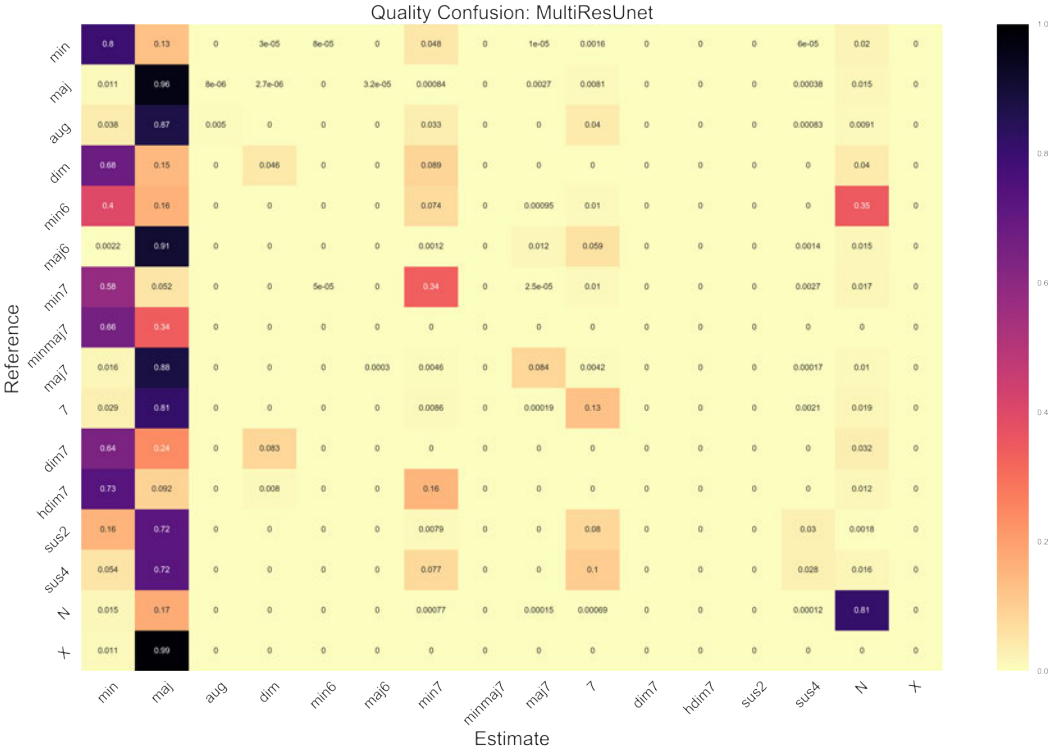

Figure 17: Within-root, frame-wise quality confusion for MultiResUnet.

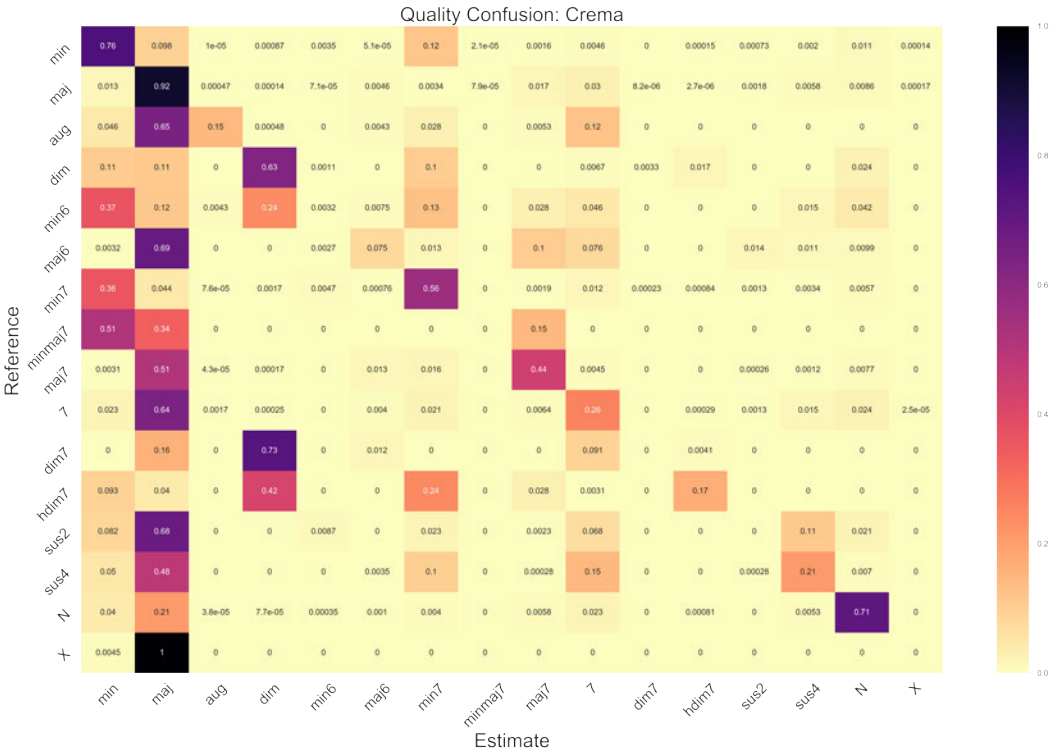

Figure 18: Within-root, frame-wise quality confusion for Crema.

4.3 Track Analysis

As a final investigation into model performance, focus now shifts towards track analysis where w.r.t reference annotations, the models' estimated chord sequences for two songs will be compared against each other. Furthermore, each track is accompanied by a table listing its results for several mir_eval metrics.

4.3.1 'Wild Honey Pie'

Figure 19 shows the reference and estimated chord sequences for the song- 'Wild Honey Pie' by *The Beatles*. Noteworthy of this song is its tuning frequency deviation from concert pitch by roughly a quarter tone [30, p. 249] . However, the reference annotation indicates that the key of the song is in G major. This property raises the question of how robust the models are against tuning deviations. In addition, the reference annotation of this song includes only 7 chords where the first 39 seconds follow a repetitive G:7→ F:7→ E:7→ Eb:7→ D:7 progression. For the remainder, the reference annotation alternates between G:7 and F:7. However, results of strict quality-wise confusion matrices indicated that the models face difficulties estimating 7 chords with MultiResUnet more so than Crema which makes this track particularly interesting as it only contains 7 chords.

To the question above, Crema's estimated chord sequences clearly illustrate that the model is not following the song's harmonic progression at all and stays for the majority of the time on 'N'. One possible explanation for this is the tuning issue described above, which prevents the model from estimating the correct root of chords and, therefore, stays on the 'N'. Crema's poor root and 'sevenths' comparison scores listed in Table 14 follow the above assumption. In contrast, MultiResUnet's estimated chord sequences follow the harmonic progression of the song to a certain degree (up to triads), e.g., the last 11.33 seconds of the song where the reference annotation takes turns between G:7 and F:7. From the figure, it can be observed that the proposed model alternates mostly between G:maj and F:maj and achieves a much higher CSR for the root comparison method than Crema. However, the predictions of MultiResUnet are severely over-segmented, which leads to a considerable number of wrong predictions, resulting in a worse 'sevenths' score than Crema. Nevertheless, the higher 'sevenths' score of Crema here is not the result of the model being better at predicting sevenths but rather a consequence of its under-segmentation as neither of the three A♯:7 nor two A♯:maj6 chords predicted by it match the chord sequences of the reference annotation. Moreover, the poor CRS scores of the 'sevenths' comparison coupled with the graphically displayed chord estimations of Figure 19 illustrate both models' deficient performance regarding 7 chords which are in agreement with their confusion matrices.

Method	root	sevenths	underseg	overseg	seg
MR_Unet	**0.3351**	0.0058	**0.9504**	0.42317	**0.42317**
Crema	0.0102	**0.0102**	0.3170	**0.94115**	0.3170

Table 14: CSR scores of models under consideration for the song- 'Wild Honey Pie' by *The Beatles*.

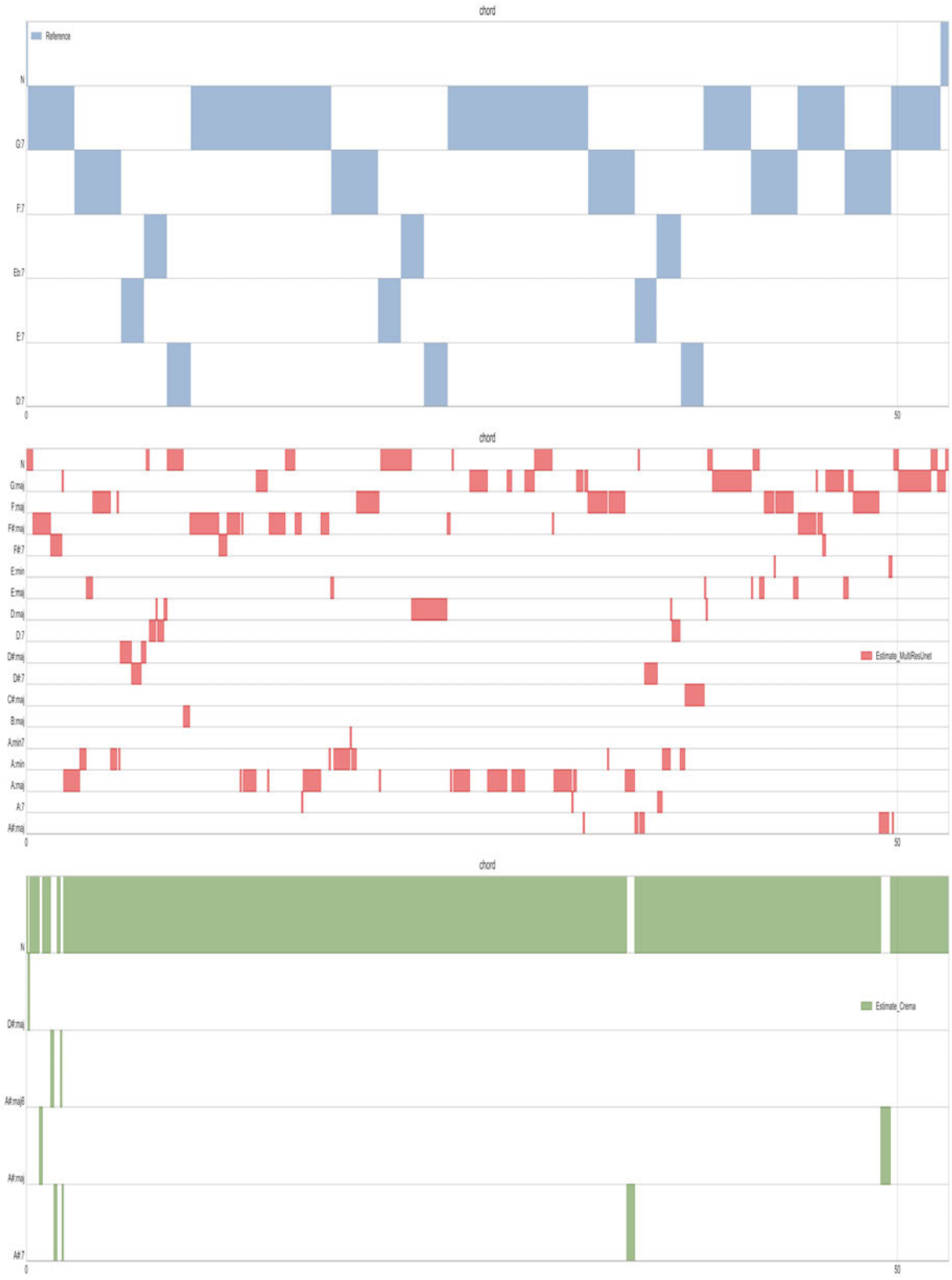

Figure 19: Reference and estimated chord sequences for the song 'Wild Honey Pie'.

4.3.2 'Dear Prudence'

The song- 'Dear Prudence' by 'The Beatles' is one of the only three songs in the entire data set containing `minmaj7` chords. As listed in Table 5, there are 30 such chords in the merged data collection, of which 24 are in their inverted form. Surprisingly, all 24 inverted `minmaj7` chords are contained in this particular song, making it a good candidate for track analysis. Reference and estimated chord sequences for 'Dear Prudence' are illustrated in Figure 20. Out of the two models, Crema is the only one holding three B: `minmaj7 / b3` chords in its estimated chord sequences. However, they all overlap with a single G: `maj7 / 3` chord in the reference annotation and are considered a mismatch for inversion-agnostic and inversion-sensitive methods. Another noteworthy characteristic of this song is a 20 seconds long D: `maj` chord in the middle of the reference annotation where the MultiResUnet has correctly estimated this chord's entire duration, except for a spurious jump. In contrast, Crema's estimated chord sequences for this chord are visibly over-segmented. Table 15 presents the CSR scores of several mir_eval comparison methods for the models under consideration.

Method	root	majmin	majmin_inv	tetrads	tetrads_inv	underseg	overseg	seg
MR_Unet	**0.6286**	**0.65774**	**0.3540**	**0.4018**	**0.3001**	0.5304	**0.8646**	0.5304
Crema	0.5814	0.5710	0.3000	0.3585	0.2564	**0.5858**	0.7962	**0.5858**

Table 15: CSR scores of models under consideration for the song- 'Dear Prudence' by the *The Beatles*.

Noticeable from the table is the sharp drop in CSR scores while changing the comparison method from 'majmin' to 'majmin_inv'. The same applies to 'tetrads' and 'tetrads_inv'. The cause of the decline in CSR scores is the relatively large number of incorrect bass notes predicted for the inverted chord types as the 'majmin_inv' and 'tetrads_inv' comparisons penalize the models for predicting the wrong bass notes. Furthermore, since CSR scores of MultiResUnet are higher but its segmentation result lower than Crema's, it offers the opportunity to compare model performance by the F-measure, e.g., the F-measure of MultiResUnet and Crema for 'majmin_inv' is 0.4246 and 0.3968. However, while the CSR scores for 'majmin_inv' showed a performance difference of 5.4 % between the models, the F-measure now indicates a difference of 2.78 % between the two.

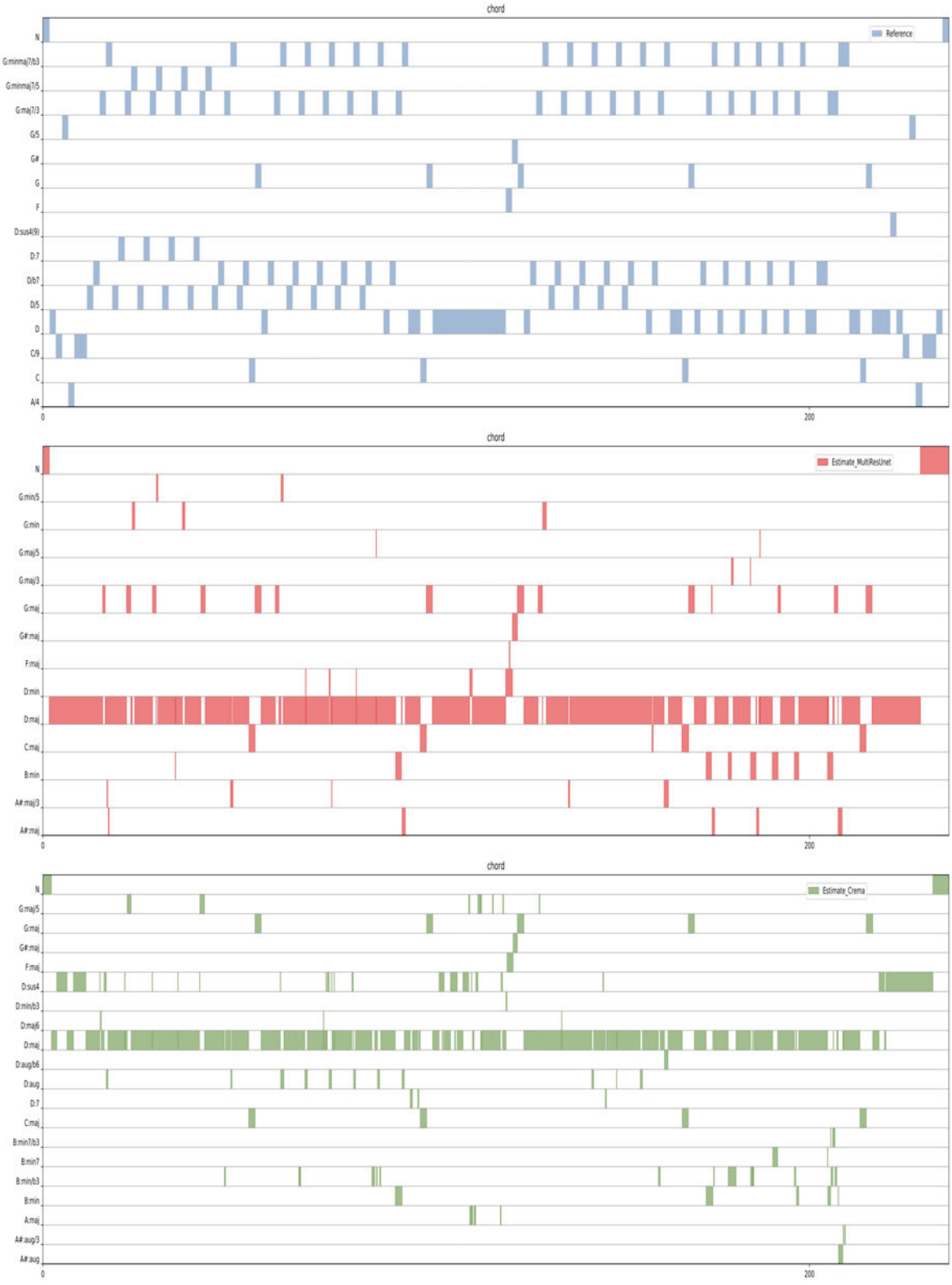

Figure 20: Reference and estimated chord sequences for the song 'Dear Prudence'.

5 Conclusion and Outlook

This thesis comprises the design and development of a new, fully convolutional neural network architecture for the task of frame-wise automatic chord recognition. The design's relevance is evaluated by comparing the proposed model's performance against a recurrent convolutional baseline using 15 mir_eval chord recognition metrics. Results of the bass-agnostic comparison functions show that the proposed model is quite robust at predicting major and minor triads (WCSR_'maj_min' = 0.854). However, the performance drops slightly for the 'triads' ($\approx$ 3.1 %) and significantly for the 'tetrads'($\approx$ 15.7 %) comparison functions. Results of the bass-sensitive comparison functions are up to a 6 % lower than the bass-agnostics. Nevertheless, the proposed model shows improvements across all bass-agnostic and bass-sensitive comparison functions.

Shifting focus towards segmentation scores, MultiResUnet achieves better results across all three segmentation metrics, i.e., the chord sequences predicted by the model have a better segmentation quality. This is an important finding as it highlights the potential of fully convolutional encoder-decoder segmentation architectures in sequence prediction tasks where RNNs are ought to perform better.

Furthermore, within-root quality confusion matrices illustrate the poor performance of MultiResUnet on the minority chord classes. The recurrent convolutional baseline, however, shows improvements for all minority classes. One possible explanation is that the frequency of these rare chord qualities in the data set is far too low for the fully convolutional model to learn the relevant hierarchical features necessary for their classification. The seemingly conflicting results of weighted average chord symbol recall and within-root quality confusion allude to the notation that model selection may conclusively be a function of the application. However, MultiResUnet requires 51971 fewer training parameters while training time is reduced by half, making it an appropriate candidate for real-time applications downstream.

Although the proposed model showcases the many potentials of fully convolutional encoder-decoder architectures, there are clear directions forward in extending the concepts presented in this thesis: first, to achieve a fair comparison between the proposed model and the baseline (in terms of parameter count), the number of MultiRes blocks and Res paths was reduced. This might have negatively affected the final performance of the model. Second, given the limited data sets available for automatic chord recognition, the problematic chord distributions in these data sets, and the difficulties of curating such corpora, there is an opportunity to move away from supervised learning and explore the potential of semi-supervised or unsupervised methods.

A Box Plots for mean and median WCSR scores and segmentation results.

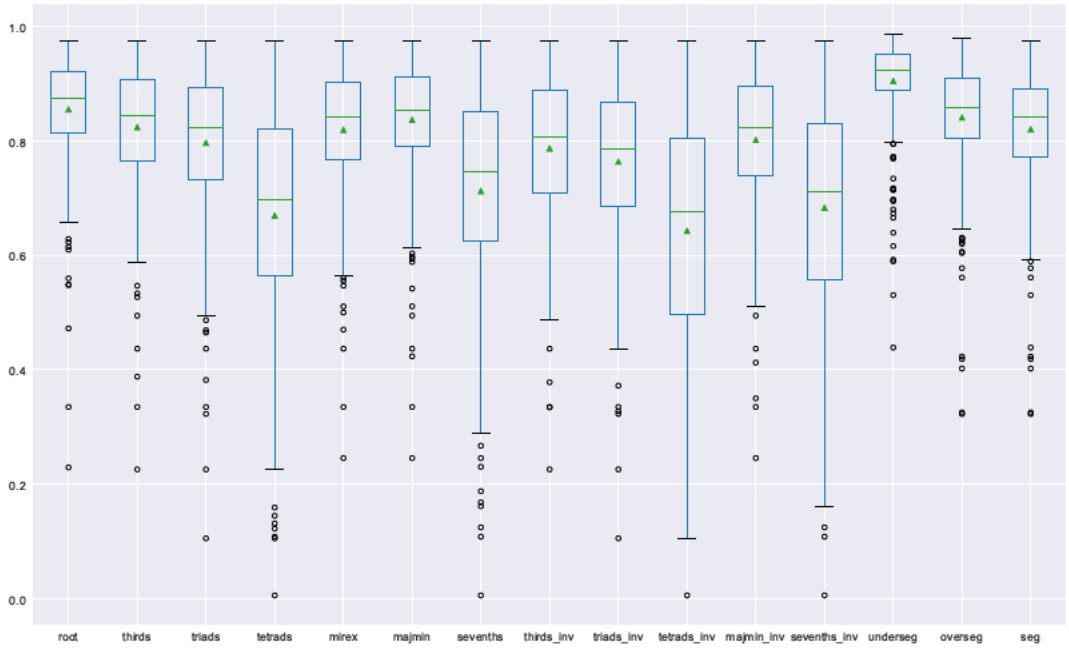

Figure 21: Mean and Median recall and segmentation scores for MultiResUnet. Green triangles mark the mean. Horizontal lines inside boxes mark the median. The ends of boxes represent the upper and lower quartiles. The two horizontal lines outside of boxes are whiskers. Outliers are shown by plotted dots.

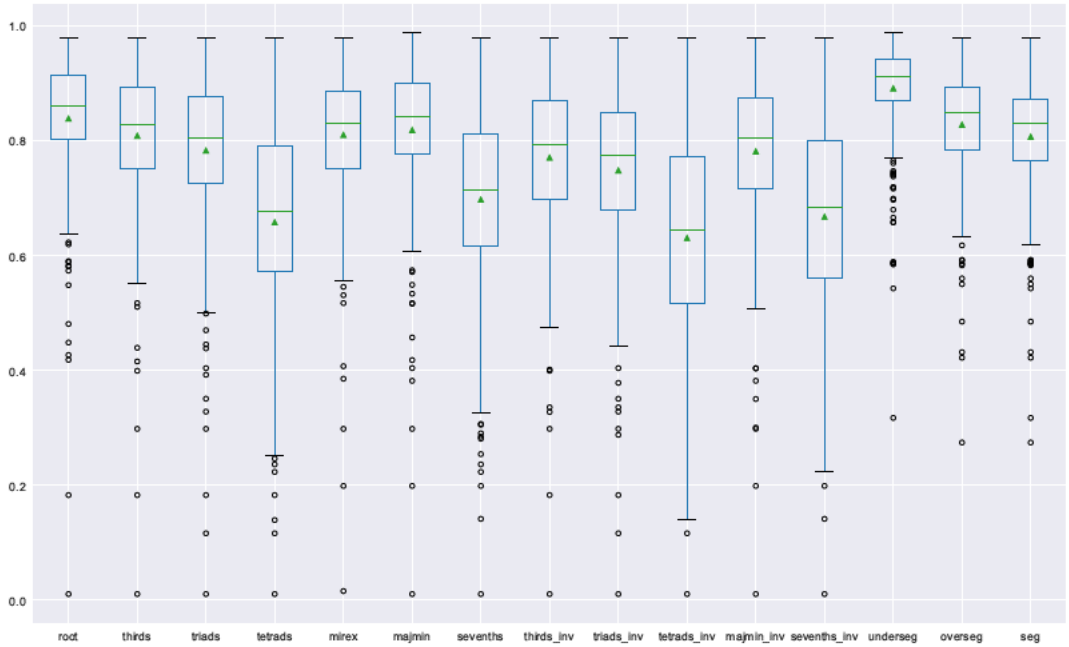

Figure 22: Mean and Median recall and segmentation scores for Crema. Green triangles mark the mean. Horizontal lines inside boxes mark the median. The ends of boxes represent the upper and lower quartiles. The two horizontal lines outside of boxes are whiskers. Outliers are shown by plotted dots.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/ 29

[2] N. Abraham and N. M. Khan, "A novel focal tversky loss function with improved attention u-net for lesion segmentation," in *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. IEEE, 2019, pp. 683–687. 26

[3] Alexa, "Alexa," https://www.alexa.com/siteinfo/chordify.net#section_traffic, [Online; accessed 25th September 2020]. 1

[4] J. Allen, "Short term spectral analysis, synthesis, and modification by discrete fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 3, pp. 235–238, 1977. 1, 10

[5] A. Anglade, R. Ramirez, S. Dixon *et al.*, "Genre classification using harmony rules induced from automatic chord transcriptions." in *ISMIR*, 2009, pp. 669–674. 1

[6] S. Bai, J. Z. Kolter, and V. Koltun, "Convolutional sequence modeling revisited," *ICLR*, 2018. 1

[7] J. P. Bello, "Audio-based cover song retrieval using approximate chord sequences: Testing shifts, gaps, swaps and beats." in *ISMIR*, vol. 7. Citeseer, 2007, pp. 239–244. 1

[8] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. 29

[9] R. M. Bittner, B. McFee, J. Salamon, P. Li, and J. P. Bello, "Deep salience representations for f0 estimation in polyphonic music." in *ISMIR*, 2017, pp. 63–70. 10

[10] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Audio chord recognition with recurrent neural networks." in *ISMIR*. Citeseer, 2013, pp. 335–340. 1, 5

[11] D. Bountouridis, K. Hendrik Vincent, F. , Wiering, and V. Remco C, "A data-driven approach to chord similarity and chord mutability," in *Second International Conference on Multimedia BigData, IEEE*, 2016, p. 275–278. 1

[12] J. C. Brown, "Calculation of a constant q spectral transform," *The Journal of the Acoustical Society of America*, vol. 89, no. 1, pp. 425–434, 1991. 1, 9

[13] J. C. Brown and M. S. Puckette, "An efficient algorithm for the calculation of a constant q transform," *The Journal of the Acoustical Society of America*, vol. 92, no. 5, pp. 2698–2701, 1992. 1

[14] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818. 2

[15] H.-T. Cheng, Y.-H. Yang, Y.-C. Lin, I.-B. Liao, and H. H. Chen, "Automatic chord recognition for music classification and retrieval," in *2008 IEEE International Conference on Multimedia and Expo*. IEEE, 2008, pp. 1505–1508. 1

[16] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014. 1

[17] T. Cho, "Improved techniques for automatic chord recognition from music audio signals," Ph.D. dissertation, New York University, 2014. 1

[18] Y.-H. Cho, H. Lim, D.-W. Kim, and I.-K. Lee, "Music emotion recognition using chord progressions," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2016, pp. 002 588–002 593. 1

[19] F. Chollet *et al.*, "Keras," https://keras.io/api/layers/regularization_layers/spatial_dropout2d/, 2015. 15

[20] Chordify, "Chordify," https://chordify.net/pages/chordify-top-5000-biggest-websites-worldwide/, [Online; accessed 25th September 2020]. 1

[21] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014. 27

[22] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015. 26

[23] J.-q. Deng and Y.-K. Kwok, "A hybrid gaussian-hmm-deep learning approach for automatic chord estimation with very large vocabulary." in *ISMIR*, 2016, pp. 812–818. 1, 5, 16

[24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018. 4

[25] G. Doras, P. Esling, and G. Peeters, "On the use of u-net for dominant melody estimation in polyphonic music," in *2019 International Workshop on Multilayer Music Representation and Processing (MMRP)*. IEEE, 2019, pp. 66–70. 2, 10

[26] G. Doras and G. Peeters, "Cover detection using dominant melody embeddings," *arXiv preprint arXiv:1907.01824*, 2019. 10

[27] M. Drozdzal, E. Vorontsov, G. Chartrand, S. Kadoury, and C. Pal, "The importance of skip connections in biomedical image segmentation," in *Deep Learning and Data Labeling for Medical Applications*. Springer, 2016, pp. 179–187. 12

[28] B. D. Giorgi, M. Zanoni, A. Sarti, and S. Tubaro, "Automatic chord recognition based on the probabilistic modeling of diatonic modal harmony," in *nDS '13; Proceedings of the 8th International Workshop on Multidimensional Systems*, 2013, pp. 1–6. 7

[29] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional lstm networks for improved phoneme classification and recognition," in *International Conference on Artificial Neural Networks*. Springer, 2005, pp. 799–804. 1

[30] C. Harte, "Towards automatic extraction of harmony information from music signals," Ph.D. dissertation, Department of Electronic Engineering,Queen Mary, University of London, 2010. 7, 17, 18, 19, 20, 37

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. 3, 13, 14

[32] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012. 15

[33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. 1

[34] T. Hori, K. Nakamura, and S. Sagayama, "Music chord recognition from audio data using bidirectional encoder-decoder lstms," in *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2017, pp. 1312–1315. 1, 5

[35] T.-H. Hsieh, L. Su, and Y.-H. Yang, "A streamlined encoder/decoder architecture for melody extraction," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 156–160. 2

[36] E. J. Humphrey, *An exploration of deep learning in content-based music informatics*. New York University, 2015. 8

[37] E. J. Humphrey and J. P. Bello, "Rethinking automatic chord recognition with convolutional neural networks," in *2012 11th International Conference on Machine Learning and Applications*, vol. 2. IEEE, 2012, pp. 357–362. 1, 3, 5, 8

[38] ——, "Four timely insights on automatic chord estimation." in *ISMIR*, vol. 10, 2015, pp. 673–679. 16, 30

[39] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, "Densenet: Implementing efficient convnet descriptor pyramids," *arXiv preprint arXiv:1404.1869*, 2014. 3

[40] N. Ibtehaz and M. S. Rahman, "Multiresunet: Rethinking the u-net architecture for multimodal biomedical image segmentation," *Neural Networks*, vol. 121, pp. 74–87, 2020. 6, 13, 14, 15, 26

[41] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015. 15

[42] Isophonics, "Center for Digital Music,Queen Mary, University of London." http://isophonics.net/datasets/, [Online; accessed 25th September 2020]. 7

[43] S. Jafarlou, S. Khorram, V. Kothapally, and J. H. Hansen, "Analyzing large receptive field convolutional networks for distant speech recognition," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 252–259. 12

[44] A. Jansson, E. Humphrey, N. Montecchio, R. Bittner, A. Kumar, and T. Weyde, "Singing voice separation with deep u-net convolutional networks," *ISMIR 2017*, 2017. 2

[45] J. Jiang, K. Chen, W. Li, and G. Xia, "Large-vocabulary chord transcription via chord structure decomposition." in *ISMIR*, 2019, pp. 644–651. 4, 5, 16

[46] H. V. Koops, W. B. de Haas, J. Bransen, and A. Volk, "Chord label personalization through deep learning of integrated harmonic interval-based representations," *arXiv preprint arXiv:1706.09552*, 2017. 9

[47] F. Korzeniowski, *Harmonic Analysis of Musical Audio using Deep Neural Networks*. Linz: Johannes Kepler Universität Linz, 2018, urn:nbn:at:at-ubl:1-25092. 1

[48] F. Korzeniowski and G. Widmer, "Feature learning for chord recognition: The deep chroma extractor," *arXiv preprint arXiv:1612.05065*, 2016. 1, 5

[49] ——, "A fully convolutional deep auditory model for musical chord recognition," in *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2016, pp. 1–6. 3, 5, 9

[50] ——, "On the futility of learning complex frame-level language models for chord recognition," *arXiv preprint arXiv:1702.00178*, 2017. 2, 3

[51] K. Koutini, H. Eghbal-zadeh, M. Dorfer, and G. Widmer, "The receptive field as a regularizer in deep convolutional neural networks for acoustic scene classification," *CoRR*, vol. abs/1907.01803, 2019. [Online]. Available: http://arxiv.org/abs/1907.01803 3

[52] A. Laaksonen *et al.*, "Automatic melody transcription based on chord transcription." in *ISMIR*, 2014, pp. 119–124. 1

[53] K. Lee, "Identifying cover songs from audio using harmonic representation," *MIREX 2006*, p. 36, 2006. 1

[54] S. Lee and C. Lee, "Revisiting spatial dropout for regularizing convolutional neural networks," *Multimedia Tools and Applications*, pp. 1–13, 2020. 15

[55] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013. 3

[56] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," *arXiv preprint arXiv:1908.03265*, 2019.

[57] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440. 11

[58] W.-T. Lu and L. Su, "Deep learning models for melody perception: An investigation on symbolic music data," in *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2018, pp. 1620–1625. 2

[59] M. Mauch, "Automatic chord transcription from audio using computational models of musical context," Ph.D. dissertation, School of Electronic Engineering and Computer ScienceQueen Mary, University of London, 2010. 19, 20

[60] B. McFee, E. Humphrey, and J. Bello, "A software framework for musical data augmentation," in *16th International Society for Music Information Retrieval Conference*, ser. ISMIR, 2015. 25

[61] B. McFee, "Convolutional and Recurrent Estimators for Music Analysis," Oct. 2017. [Online]. Available: https://doi.org/10.5281/zenodo.1010486 27

[62] B. McFee and J. P. Bello, "Structured training for large-vocabulary chord recognition." in *ISMIR*, 2017, pp. 188–194. 1, 5, 6, 16, 17, 26, 27, 29, 30, 33, 34

[63] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, vol. 8, 2015, pp. 18–25. 25

[64] M. McVicar, R. Santos-Rodríguez, Y. Ni, and T. De Bie, "Automatic chord estimation from audio: A review of the state of the art," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 22, no. 2, pp. 556–575, 2014. 1

[65] Mirex, "Music Information Retrieval Evaluation exchnage," https://www.music-ir.org/mirex/wiki/2013:Audio_Chord_Estimation_Results_MIREX_2009, 2013, [Online; accessed 25th September 2020]. 17, 18

[66] ——, "Music Information Retrieval Evaluation exchnage," https://www.music-ir.org/mirex/wiki/MIREX_HOME/, 2020, [Online; accessed 25th September 2020]. 17

[67] T. Music and A. R. Lab, "The McGill Billboard Project," https://ddmal.music.mcgill.ca/research/The_McGill_Billboard_Project_(Chord_Analysis_Dataset)/, [Online; accessed 25th September 2020]. 7

[68] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010. 12, 27

[69] S. Nakayama and S. Arai, "Dnn-lstm-crf model for automatic audio chord recognition," in *Proceedings of the International Conference on Pattern Recognition and Artificial Intelligence*, ser. PRAI 2018. New York, NY, USA: ACM, 2018, pp. 82–88. [Online]. Available: http://doi.acm.org/10.1145/3243250.3243270 1, 5

[70] S. Nakayama and A. Shuichi, "Residual dnn-crf model for audio chord recognition," in *International Conference on Intelligent System and Image Processing (ICISIP)*, 2017, pp. 92–98. 3, 5

[71] D. Odekerken, H. V. Koops, and A. Volk, "Decibel: Improving audio chord estimation for popular music by alignment and integration of crowd-sourced symbolic representations," *arXiv preprint arXiv:2002.09748*, 2020. 4, 5, 20, 30

[72] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016. 4

[73] K. O'Hanlon and M. B. Sandler, "The fifthnet chroma extractor," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 3752–3756. 33

[74] J. Park, K. Choi, S. Jeon, D. K. Kim, and J. Park, "A bi-directional transformer for musical chord recognition," *ArXiv*, vol. abs/1907.02698, 2019. 3, 5

[75] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, 2013, pp. 1310–1318. 1

[76] J. Pauwels, F. Kaiser, and G. Peeters, "Combining harmony-based and novelty-based approaches for structural segmentation." in *ISMIR*, 2013, pp. 601–606. 1

[77] J. Pauwels and G. Peeters, "Evaluating automatically estimated chord sequences," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 749–753. 32

[78] C. Raffel, B. McFee, E. J. Humphrey, J. Salamon, O. Nieto, D. Liang, D. P. Ellis, and C. C. Raffel, "mir_eval: A transparent implementation of common mir metrics," in *In Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR*. Citeseer, 2014. 18

[79] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241. 2, 12

[80] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[81] R. E. Scholz, G. L. Ramalho, and G. Cabral, "Cross task study on mirex recent results: An index for evolution measurement and some stagnation hypotheses." in *ISMIR*, 2016, pp. 372–378. 6

[82] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997. 1

[83] Z. Shi, H. Lin, L. Liu, R. Liu, and J. Han, "Is cqt more suitable for monaural speech separation than stft? an empirical study," *arXiv preprint arXiv:1902.00631*, 2019. 9

[84] S. Sigtia, N. Boulanger-Lewandowski, and S. Dixon, "Audio chord recognition with a hybrid recurrent neural network." in *ISMIR*, 2015, pp. 127–133. 1, 5

[85] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 3

[86] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014. 15

[87] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 36, no. 2, pp. 111–133, 1974. 29

[88] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 3

[89] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826. 13

[90] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 648–656. 15

[91] F. Wu, A. Fan, A. Baevski, Y. N. Dauphin, and M. Auli, "Pay less attention with lightweight and dynamic convolutions," *arXiv preprint arXiv:1901.10430*, 2019. 4

[92] Y. Wu, T. Carsault, E. Nakamura, and K. Yoshii, "Semi-supervised neural chord estimation based on a variational autoencoder with discrete labels and continuous textures of chords," 2020. 4, 5

[93] Y. Wu and W. Li, "Music chord recognition based on midi-trained deep feature and blstm-crf hybird decoding," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 376–380. 1, 5, 16

[94] ——, "Automatic audio chord recognition with midi-trained deep feature and blstm-crf sequence decoding model," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 27, no. 2, pp. 355–366, Feb. 2019. [Online]. Available: https://doi.org/10.1109/TASLP.2018.2879399 3, 5, 16

[95] Y.-T. Wu, B. Chen, and L. Su, "Polyphonic music transcription with semantic segmentation," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 166–170. 1, 2

[96] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *2010 IEEE Computer Society Conference on computer vision and pattern recognition*. IEEE, 2010, pp. 2528–2535. 12

[97] X. Zhou and A. Lerch, "Chord detection using deep learning," in *Proceedings of the 16th ISMIR Conference*, vol. 53, 2015. 5