TU WIEN Informatics

A Comparison of Audio Preprocessing Methods for Music Autotagging using CNN-architectures

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering and Internet Computing

eingereicht von

Maximilian Damböck, BSc.
Matrikelnummer 01526625

an der Fakultät für Informatik
der Technischen Universität Wien
Betreuung: Privatdoz. Dipl.-Ing. Dr.techn. Peter Knees

Wien, 17. Mai 2022

Maximilian Damböck | Peter Knees

TU WIEN Informatics

A Comparison of Audio Preprocessing Methods for Music Autotagging using CNN-architectures

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering and Internet Computing

by

Maximilian Damböck, BSc.
Registration Number 01526625

to the Faculty of Informatics

at the TU Wien

Advisor: Privatdoz. Dipl.-Ing. Dr.techn. Peter Knees

Vienna, 17th May, 2022

Maximilian Damböck

Peter Knees

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Maximilian Damböck, BSc.

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 17. Mai 2022

Maximilian Damböck

Danksagung

Danke an meinen Betreuer, Peter Knees für die Unterstützung und Anregungen bei dieser Arbeit. Weiters möchte ich meiner Familie und allen Freunden danken, die mich hierbei und auch während des gesamten Studiums hindurch begleitet und unterstützt haben.

Acknowledgements

Thanks to my advisor, Peter Knees for the support and all suggestions at this thesis. I also want to thank my family and all friends that accompanied and supported me on this work and also during my entire study.

Kurzfassung

Im letzten Jahrzehnt wurden Convolutional Neural Networks (CNNs) bekannt durch ihre herausragenden Ergebnisse in verschiedenen Teilgebieten von Computer Vision. Mithilfe der Short Time Fourier Transformation (STFT) als Verarbeitungsmethode für Audio-Signale finden CNNs auch in Music Information Retrieval (MIR) unterschiedliche Anwendungen, insbesondere im Bereich Autotagging von Musik wo sie regelmäßig neue Top-Ergebnisse erreichen.

Ziel dieser Arbeit ist es herauszufinden, wie die am weitest verbreiteten Eingabe-Repräsentationen von Audio-Signalen die Klassifikationsperformance von CNNs für das automatische Taggen von Musik beeinflussen, zu diesen gehören: die unverarbeitete Signalform, STFT, Mel Spectrogram, Mel Frequency Cepstral Coefficient (MFCC) und Constant-Q Transformation (CQT). Dafür wird die Performance anhand fünf verschiedener CNN Architekturen an zwei verschiedenen Datensätzen verglichen, letztere sind MagnaTagATune (MTAT) und MTG-Jamendo (MTGJ). Die Two-way ANOVA Analyse der Ergebnisse zeigt, dass sowohl die Wahl des CNN Modells als auch die Wahl der Eingaberepräsentation einen signifikanten Einfluss auf die Klassifikationsergebnisse haben. Die STFT erzielt im Durchschnitt die besten Ergebnisse auf beiden Datensätzen. Dies ist auch bei der Auswertung pro Tag-Kategorie - Genre, Instrument und Stimmung der Fall. Weiters konnten hier auch keine allgemein gültigen Unterschiede der Klassifikationsperformances der einzelnen Eingaberepräsenetationen zwischen den Kategorien ausgemacht werden. MFCCs liefern durchwegs zufriedenstellende Ergebnisse, obwohl diese um vier bis zwanzig mal kleiner als die anderen Eingabeformate sind und daher auch eine, um bis zu vier mal kürzere Trainingszeit aufweisen. CQT hat die zweitschlechteste Performance auf dem MTAT Dataset und die zweitbeste auf MTGJ. Um genauere Aussagen über dessen Anwendbarkeit zu treffen sind daher noch weitere Untersuchungen notwendig.

Darüber hinaus wird in dieser Arbeit untersucht, inwiefern Dilated Convolutions die Klassifikationsperformance von Musik Autotagging Modellen verbessern können. Speziell deren ressourcenschonende Merkmalserkennung ist interessant für die Genre- und Stimmungsklassifikation von Musik. Dazu wird das bestehende CNN Musicnn (vgl. [Pons and Serra, 2019]) mit Stacked Parallel Dilated Convolutions erweitert. Die Ergebnisse für den MTAT Datensatz zeigen eine signifikante Verbesserung des ROC-AUC Wertes von ursprünglich 90.99% auf 91.49% und von 36.74% auf 37.78% für den PR-AUC Wert, bei gleichzeitiger Reduktion der Trainings-Epochenzeit um 59%.

Abstract

CNNs have become famous for their outstanding results in various computer vision tasks in the last decade. Based on the STFT as preprocessing method for audio signals, CNNs are used in Music Information Retrieval (MIR) as well. Especially for autotagging of music, they have become state-of-the-art.

This thesis investigates how the following most commonly used input representations of audio signals affect the classification performance of CNNs on music autotagging: raw waveform, STFT, Mel spectrogram, Mel frequency cepstral coefficients (MFCCs), and constant-Q transformation (CQT). For this purpose, their performance is compared using five different CNN architectures on two datasets, MagnaTagATune (MTAT) and MTG-Jamendo (MTGJ). A Two-way ANOVA analysis shows that both model and input representation significantly impact the classification results. On average, the STFT has the best overall performance on both datasets. It also outperforms all other input representations for all specific tag categories genre, instrument, and mood. No special trends can be observed for the classification performances of the different input representations on the respective tag categories. MFCCs provide good results while having a four to twenty times smaller size than the other input representations and consequently an up to four times shorter epoch-time during training. The CQT transformation shows the worst results on MTAT but performs second-best on MTGJ. Therefore, more research on this preprocessing method is needed for the results to be more conclusive.

Apart from that, this study investigates the applicability of dilated convolutions for music autotagging models. Their ability to capture large receptive fields while keeping the resource consumption low can be interesting for the genre- and mood classification. To prove this conjecture, the existing CNN model Musicnn (cf. [Pons and Serra, 2019]) is extended with stacked parallel dilated convolutions and then compared to the original model on the MTAT dataset. The results show a significant enhancement of the average ROC-AUC score from 90.99% to 91.49% and from 36.74% to 37.78% for the PR-AUC score, while reducing the average training epoch-time by 59%.

Contents

CHAPTER 1

Introduction

This chapter introduces the topic and presents the problems addressed throughout this work. The first Section 1.1 starts with the motivation for music autotagging and why we deal with audio input representations. In the next Section 1.2, the concrete problem statement is formulated, including three main questions answered throughout this work. Section 1.3 presents the outline for this thesis.

1.1 Motivation

Deep Learning has become the state-of-the-art in many tasks of MIR [Purwins et al., 2019]. One of them is autotagging of music which is a multi-label classification task to automatically predict tags for audio samples like ”rock”, ”piano” or ”fast”. Typical use-cases are database management tasks, music recommendation engines, and other music tools. Over the last few years, CNNs have provided the best results at this task, using different audio-signal representations like raw waveform or different types of audio spectrograms as input [Won et al., 2020, Yu et al., 2021]. Most of the current research in this area focuses on finding the best performing models and hyperparameter-tuning. The preprocessing phase is then often decided heuristically and not subject to further optimization [Choi et al., 2018]. However, the choice of the input representation can be crucial for the performance of the deep-learning models, as shown in other tasks like Natural Language Processing, and should therefore be well reasoned [Camacho-Collados and Pilehvar, 2018]. Tags for songs can be grouped into several categories, and for this work, the following three categories are considered: Genre, Instrument, and Mood. Each category has different dependencies on audio features like rhythm, frequency, or amplitude, so the categorization might perform differently for each audio-signal representation.

There are several ways to vary CNN models, like in- or decreasing the layer depth, using different filter-kernel shapes, or applying self-attention [Won et al., 2020]. Another way is to use dilated convolutions to extract broader audio features while keeping the

resource consumption within a certain limit. In the context of image classification, a dilated convolution can be described as "skipping" one pixel between each other in the filter kernel. So the filter range increases while the resolution decreases. This might be beneficial for music autotagging, especially for classifying genre- and mood tags since they usually depend on longer audio sequences. Dilated convolutions have shown promising results in image segmentation- and classification over the last few years [Yu et al., 2017, Tsang, 2018].

1.2 Problem Statement

The main goal of this thesis is to find out how different preprocessing methods of audio signals affect the classification performance of CNNs on music autotagging. Moreover, it will be evaluated if specific input representations are better suited for certain tag categories or if this effect is negligible. For this, the following three will be considered: Genre, Instrument, and Mood. For the preprocessing of the audio signals raw waveform, STFT, Mel spectrogram, MFCCs, and CQT [Huzaifah, 2017, Won et al., 2020, Yu et al., 2021]. Apart from that, I investigate how dilated convolutions can enhance the classification performance or reduce the training time of existing models for this task. Therefore, the following three research questions are formulated, which will be answered in this thesis:

- Which audio input representations generally provide better results for CNNs than others, and to what extent?
- How much difference in classification performance is between different audio input representations for certain tag categories?
- To what extent are dilated convolutions beneficial for music autotagging in terms of classification performance and training time?

I will answer this question by conducting several experiments throughout this work, including enhancing the existing Musicnn (cf. [Pons and Serra, 2019]) with dilated convolutions and a comprehensive comparison of the input representations on various CNNs. The CNNs are selected in a way to cover a broad range of different model-architectures and include VGG-16 [ul Hassan, 2018], ResNet [He et al., 2016], Senet [Hu et al., 2018], and Musicnn [Pons and Serra, 2019].

To sum up, the main contributions of this thesis for future work on music autotagging are:

- Create a basis for the decision of the best input representation for a specific research task in music autotagging that focuses on CNNs.
- Evaluate the effect of the input representations on single tag categories to support fine-tuning of autotagging models on the one hand and research focusing on single categories on the other hand.

- Explore the potential of dilated convolutions for music autotagging.

1.3 Outline

In Chapter 2, recent work on music autotagging is presented to assess the current state-of-the-art and research trends on this topic. In Chapter 3, some fundamental knowledge about audio signals is presented, and the relevant preprocessing methods are described in detail. In Chapter 4, the two datasets MTGJ and MTAT, which are used in the experiments, are presented, and afterward, the other most relevant datasets for music autotagging are summarized. In Chapter 5, the CNN architectures used for the experiments are described in detail, including the dilated CNN developed during the experiments. In Chapter 6, the methodological approach and the structure of the experiments are presented. Afterward, the exact conduction, including setup, preprocessing, hyperparameter tuning, training, and testing of each configuration, is described. The results for each experiment are presented and discussed in Chapter 7. In the last Chapter 8, the whole thesis is summarized, and the research questions will be answered to conclude the work. Afterward, the limitations of this work are shown, and an outlook for future research on this topic is given.

CHAPTER 2

State of the Art

Automated music tagging is useful for music recommendation engines, database management tasks, and various music tools. In the last decade, deep learning has become the state-of-the-art for this task, and especially CNNs have reached outstanding results. This chapter surveys some recent advancements on this and closely related topics, focusing on different aspects like CNN architectures, the effects of preprocessing methods, or data augmentation approaches. Each chapter of this thesis includes a more specific literary review of the respective subject. Therefore, the focus of this chapter lies in broader studies and surveys to give a short insight into music autotagging.

2.1 Pre-DL autotagging methods

Over the last ten years, different types of Deep Learning (DL) have become state-of-the-art in music autotagging. Before that, plenty of different ways and algorithms existed to annotate songs automatically. This section gives a short introduction to methods used before the advent of DL.

In [Mandel et al., 2011], a discriminative Restricted Boltzmann machine (RBM) is presented to autotag music. The RBM is a Machine Learning (ML) method where a neural network consisting of visible and hidden units can learn a specific probability distribution from the input data. "Restricted" refers to the bipartite dependency between the hidden and the visible units, meaning the independence of hidden variables when conditioned on the visible ones and vice versa. The discriminative RBM is trained to predict the respective tag class labels from the audio inputs. For this, timbral features are extracted from the songs MFCCs, and rhythmic features are extracted from the modulation spectra in four large frequency bands. The model reached new state-of-the-art performances during the experiments, outperforming existing methods like the support vector machine and logistic regression. [Barrington et al., 2008] presents a supervised multi-label model to autotag music. The songs are approximated as K-component Gaussian mixture model

(GMM) distributions over the feature space of the audio data. Then these songs are annotated with semantic multinomials, which are normalized vectors of likelihoods for their respective audio features under those GMMs. These semantic multinomials are used to calculate a probability score for each tag GMM equal to the relevance of this tag for the song. The system showed the top overall performance in the MIREX 2008 "Audio Tag Classification" task. In Ellis et al., 2013, the authors propose a Bag of Systems (BoS) representation for audio signals. Generative models capture timbral and temporal characteristics of music to form a dictionary of musical codewords. Each song consists of multiple such codewords that can be used for text document retrieval algorithms to perform the autotagging. The following section discusses current state-of-the-art DL-models.

2.2 Current state-of-the-art models

In Won et al., 2020, the authors evaluate the current state-of-the-art CNN models for music auto-tagging. MTAT, Million Song Dataset (MSD), and MTGJ serve as datasets for training and evaluation. Several different CNN architectures are described in the paper and evaluated either by using raw waveform or Mel spectrogram as input representations. Among others, the study considered the following models: A Fully Convolutional Network that only consists of convolutional layers and no Fully connected (FC) layers, Musicnn where the architecture relies on music domain knowledge to efficiently extract temporal and timbral features (cf. Section 5.4), a Sample-Level CNN which takes raw waveforms as input and uses squeeze- and excitation blocks for enhanced prediction (cf. Section 5.3), a Convolutional Recurrent Neural Network which is a combination of CNNs and Recurrent Neural Networks (RNNs) or a self-attention based approach using an adaption of the Transformer Encoder of natural language processing which. The length of the input samples, so whether to use song-level- or chunk-level training, is another important aspect for the setup of each model. The results show that models trained with shorter audio excerpts outperformed the other models. The studys authors suspect that this might be due to the increased training set size when splitting each sample into smaller pieces. By applying different audio deformations like pitch shifting or time stretching to the input samples, the robustness of the models was tested in another experiment to determine the generalization abilities of each model. This resulted in a different ranking of the models regarding their performance for each deformation, but no model outperformed the others in terms of robustness.

In Pons et al., 2018, the authors tested the effect of the amount of training data used for different CNN architectures in music auto-tagging. They trained models with different input sizes, from 100,000 to 1.2 million audio samples of the private 1.2M-Songs dataset. The first CNN model uses raw waveforms, whereas the second one uses Mel spectrograms as input. They designed the latter to heavily rely on musical domain knowledge, whereas the former has minimal assumptions over the task. Both architectures relied on previous research and were slightly adapted. The results show that the classification accuracy could be improved significantly until a training-set size of one million data samples.

For smaller dataset sizes, both models performed equally well, but given enough data, the assumption-free model processing raw waveforms outperformed the other one. The two models were tested on MTAT and MSD too. The Mel spectrogram-based model outperformed the raw waveform-based model in both datasets on all tested configurations, which encouraged the conclusion that domain knowledge can be beneficial for designing models, especially for smaller datasets.

2.3 Preprocessing methods

In [Choi et al., 2018], the authors compare the effects of several audio preprocessing methods for music autotagging. For this, a CNN model was used and trained on MSD. STFT and Mel spectrogram served as input representations. The same CNN architecture was used for all experiments to guarantee the comparability of the results. The evaluation of the results, aggregated over all tags, shows only minor differences in the classification performances, depending on the training data usage. However, none of the two input representations outperformed the other one. In another experiment, the effect of using log-compressed magnitudes was tested, motivated by the human perception of loudness, referring to decibel scaling. As a result, the log-compressed versions outperformed the linear ones in all tested configurations significantly.

A more comprehensive comparison of time-frequency-based representations of sound was conducted in [Huzaifah, 2017] in the context of environmental sound classification. The datasets used for this consist of sound recordings, each belonging to exactly one class, which belongs to a major group like animals, humans or natural soundscapes. This is a big difference from music auto-tagging, which is a multi-labeling problem where each music sample gets labeled with a set of different tags instead of only one. Different configurations for STFT, Mel spectrogram, CQT, Continuous Wavelet Transform, and MFCCs were evaluated in the experiments, using different CNN configurations for training. For the configuration of the input representations, the main focus was to compare wideband and narrowband transformations, which refer to the length of the sliding windows of the STFT. The evaluation shows that in nearly all configurations, the shallower CNN models outperformed the deeper ones. The authors of the study explained this with significant over-fitting of the deeper model. The results also show that Mel spectrograms performed consistently well across all variations, while STFT and CQT did well on only a few models. In general, all input representations produced better results than MFCCs. Evaluating the different lengths of the sliding windows led to different results for both configurations in nearly all cases, but none outperformed the other one in general. However, there was a slight tendency that wideband configurations produced better results than narrowband configurations.

The effects of the reduction of frequency and time resolution of Mel spectrograms for CNN architectures in the context of music auto-tagging were investigated in [Ferraro et al., 2021]. The authors used the MTAT dataset for performance comparisons, and then the selected configurations were compared on the larger MSD. Two different state-of-the-art CNN

architectures served as a reference for the comparison of the different frequency and time resolutions of the Mel spectrograms. The results suggest that despite reducing the frequency bands from 128 to 48, the performance loss was negligibly slight. This fact could be important for future decisions on the trade-off between the accuracy of the models, data storage size, and training time.

In Chapter 3, the relevant preprocessing methods for audio signals are described in detail.

2.4 Dilated CNNs

CNNs play an essential role in image classification and segmentation due to their ability to extract spatial features. However, one of the main problems of basic CNNs is their high resource consumption. To overcome this problem, dilated convolutions can be used to extract broader features while using lighter filter kernels.

[Lei et al., 2019] proposes a CNN model based on stacked dilated convolutions with different dilation rates for image classification. Compared to the original CNN, this model could significantly improve the classification accuracy by 2.86% on average and decrease the training time by 12.99%.

In [Muhammad et al., 2021], the authors propose an attention-based Long short-term memory (LSTM) network to recognize human actions in videos based on dilated convolutions. The latter use more inclusive receptive fields than standard convolutions to recognize bigger image segments better. On average, the proposed model could improve the benchmark scores between 1-2%.

In [Zhang et al., 2017], the authors use dilated convolutions for environmental sound classification. This task requires filters with big receptive fields, and existing studies have predominantly used very deep CNNs to tackle this problem. The dilated convolutions can solve this with acceptable resource consumption. The results on different datasets show a performance improvement of up to 10% on different datasets compared to existing deep CNNs but also with a higher computation complexity and bigger storage.

2.5 Summary

This section has shown that DL and especially CNNs have become the central research subjects in music autotagging over the last few years. As shown in [Won et al., 2020], there are many variants of CNNs, each of which provides differently good classification results. Apart from that, the studies have shown that there are other important factors influencing the classification performance like the sample length, number of training samples, or the preprocessing method for the audio signals. Since the focus of this thesis is the comparison of the different audio input representations, the influence of the other factors should be kept as small as possible. Therefore, five different models are used for the experiments, each with different input sample lengths from 3 to 30 seconds. The preprocessing methods are tested on each of those models. To guarantee the general

validity of the results, the experiments are run on two datasets of different sizes. As shown in Section 2.3, every input representation has its advantages and disadvantages, and it is hard to decide which one to choose for a specific model. However, the five preprocessing methods investigated in this thesis are used throughout most research on music autotagging. The insights in [Ferraro et al., 2021] have shown that for Mel spectrograms, the number of frequency bands for the preprocessing is negligible. The other studies also mainly used the same default values for the respective preprocessing methods as the preprocessing library Librosa [McFee et al., 2015] does, so in the experiments of this thesis, those default values are used as well. Different approaches to using dilated convolutions in visual computing and MIR have been presented in Section 2.4. The main ideas behind the application of the dilated convolutions in those models, like the stacked parallel dilated convolutions, will also be considered for the experiments of this thesis.

The following chapter presents the foundations of the different audio signal preprocessing methods.

CHAPTER 3

Audio Signal Preprocessing Methods

The focus in MIR research often lies in optimizing DL models and hyperparameter-tuning. The choice of the audio preprocessing method is often not evidence-based and just decided ad-hoc. Libraries like Librosa [McFee et al., 2015] further facilitate the preprocessing stage, providing one-liner methods for processing Fourier Transformations and other relevant methods and come with helpful default parameter settings. In the research process, typically less time is spent on it and more focus is given to the development of the model itself.

This section introduces the most commonly used audio preprocessing methods in MIR in more detail. In order to get a better understanding of the respective methods, the concrete processing stages are described first. Furthermore, the most important parameters are discussed regarding their impact on the resulting signal and their adjustment for the experiments. The first of the following five preprocessing methods is just the one-dimensional raw waveform of the audio signal. The others are two-dimensional time-frequency spectrograms.

3.1 Raw waveform

In its most basic form, the sound is a one-dimensional signal, representing a vibration that propagates as a wave in a medium like air or liquids. It can be described as a function of the amplitude over time (cf. Figure 3.1), usually with values between -1 and 1 referring to the signals voltage for digitized audio signals [Rocchesso, 2003]. Important parameters that can be affected during preprocessing of those signals are the sampling rate, which is described in the following subsection and the bit rate. Another common option is to use equalizers to manipulate the audio files frequency spectra, but this is

not a subject of this study and is therefore not discussed further. The last subsection presents a few recent studies in MIR that use this input representation for DL models.

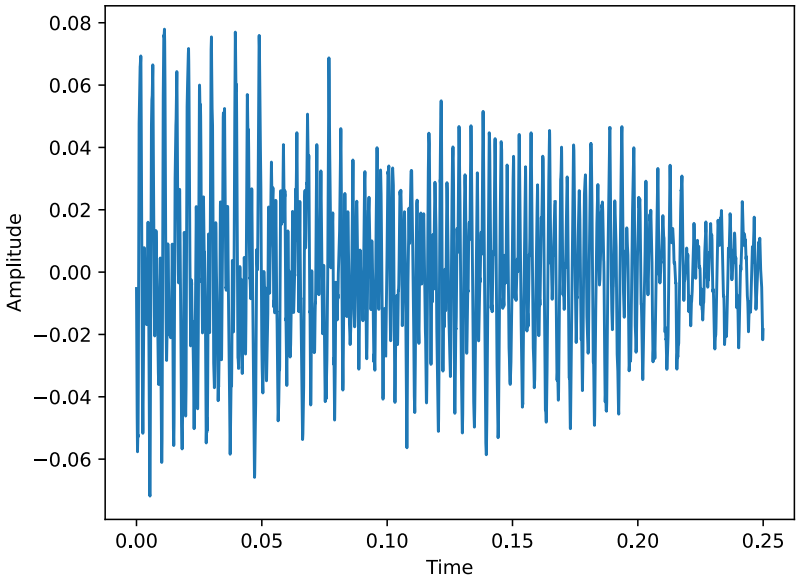


Figure 3.1: Audio waveform of 0.25s sound-clip

3.1.1 Sampling Rate

In order to digitize an analog audio signal, it has to be sampled and quantized. The frequency in which one takes those samples is called sampling rate and can be determined by the Nyquist-Shannon-Theorem. This theorem states that it is necessary to take $2 * f_{max}$ equidistant samples to describe an arbitrary signal with a maximum frequency of f_{max}. It can be generalized by:

$$f_A \geq 2 * f_{max}, \tag{3.1}$$

where f_A is the sampling rate. The human ear perceives audio signals from 20Hz up to 20kHz. The typical sampling rates for audio-CDs are 44.1kHz or 48kHz. It is important to note that a little additional buffer here is added to the 40kHz so anti-aliasing filters can still be applied to the signal to prevent aliasing effects [Görne, 2008]. The sampling rate directly affects the total size of an audio file, so in surroundings where the file size cannot be arbitrarily large, lower sampling rates might be necessary. However, every reduction directly leads to information loss, which can significantly affect the audio quality, as we have seen before.

3.1.2 Raw waveform in MIR

Sample-level DL networks have been used throughout all subtopics of MIR, reaching good results. In music autotagging, there are several approaches for one-dimensional CNNs. For example, in [Yu et al., 2021], they surpass all existing state-of-the-art models. This model consists of strided convolutional layers for extracting local features and reducing temporal dimension and residual blocks to recognize more complex features. A broader investigation was conducted in [Lee et al., 2017a], where two sample level CNNs are evaluated for the three most common sub-domains in sound-classification: music auto tagging (MTAT), speech command recognition (Speech Commands Dataset) and acoustic scene tagging (DCASE 2017 Task 4). The first model is a very basic CNN with convolutional- and pooling blocks, and the second one consists of additional residual- and squeeze- and excitation blocks. The latter one reached results close to the state-of-the-art for all three models at the time the study was published. RNNs are an alternative to CNNs for audio classification with raw waveforms due to their ability to model long-term dependencies. In [Avramidis et al., 2021], a CNN model was combined with an RNN model and outperformed other pure CNNs and RNNs on the task of polyphonic instrument classification. The paper states that RNNs better model longer-term temporal structures, whereas CNNs are better for temporally local correlations, and the enhancement results from the combination. It seems that since the emergence of DL and its overwhelming ability to extract features, raw waveforms have become an attractive alternative to time-frequency spectrograms and are gaining much more attention in research than ten years ago. However, the preprocessing methods based on the STFT are still the most commonly used ones and will be described in more detail in the following sections, starting with the basic STFT.

3.2 Short Time Fourier Transformation

The raw audio waveform is very hard to interpret in its unprocessed form. Upon visual inspection, humans can only detect dynamics and maybe separate the chorus from the verse, but it is nearly impossible to recognize melodies or frequencies without further analysis. The STFT transforms the audio signal into a two-dimensional amplitude versus time-frequency spectrogram, which is easier to interpret visually. Figure 3.2 shows an STFT diagram of a 29 second long audio-file where one can see the current frequencies for each point in time. In the following subsection, the processing steps for the STFT are described in more detail. Afterward, the most important parameters will be described in more detail: the number of samples per Fast Fourier Transformation (FFT) (n_fft), the hop-length and the window-length. For all parameters the default values of Librosa are used [1].

[1] http://librosa.org/doc/main/generated/librosa.stft.html, accessed 15-January-2022

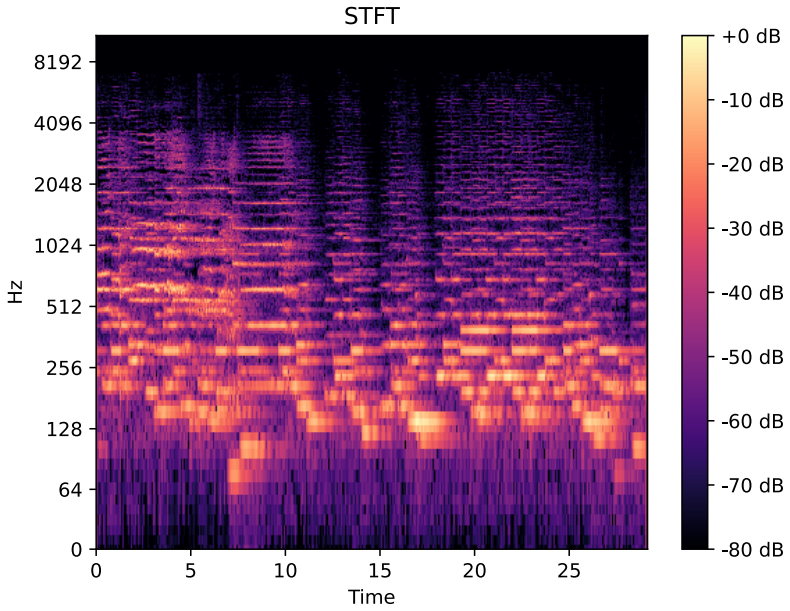


Figure 3.2: STFT of a 29s audio-file

3.2.1 Calculation of the STFT

The basis for the STFT is the Discrete Fourier Transformation (DFT) which forms the inner product of the analyzed function $x(n)$ with the harmonic oscillation $\exp j2\pi$, as given in the following equation:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nk}, \quad with\, W_N = e^{-i2\pi/N}, \tag{3.2}$$

where $X[k]$ refers to the amplitude of the k-th frequency-bin, $x[n]$ is the function at time n, and i is the imaginary unit defined by $i^2 = -1$. The DFT results in an array of amplitudes for each frequency of the whole sequence. The STFT uses a window-function $g[k]$, sliding over the signal-sequence in regular steps and applying the DFT on those sub-sequences. Equation 3.3 describes the computation for $X_{STFT}[m, n]$ with m and n referring to the time- and frequency shifts in discrete steps.

$$X_{STFT}[m, n] = \sum_{k=0}^{L-1} x[k] g[k - m] e^{-i2\pi nk/L} \tag{3.3}$$

L refers to the L-point window function g, $x[k]$ to the signals amplitude at time k and $g[k-m]$ to the value of the window function at the current position in the sliding window. Usually, g uses a Hamming window defined by [Podder et al., 2014]:

$$W[n] = 0.54 - 0.46 * cos(\frac{2\pi n}{N-1}), n = 0, ..., N-1 \tag{3.4}$$

Figure 3.3 illustrates the calculation of the STFT by calculating the Fourier Transformation of the sliding windows $g(t)$ of the signal $x(t)$, producing the output in the lower part of the picture, which visualizes the amplitude of each frequency per time of the signal [Kehtarnavaz, 2008]. Important to note here is that the imaginary part of the output of the DFT, which refers to the phase, is discarded because only the amplitude contains the important perceptual information needed [Logan, 2000].

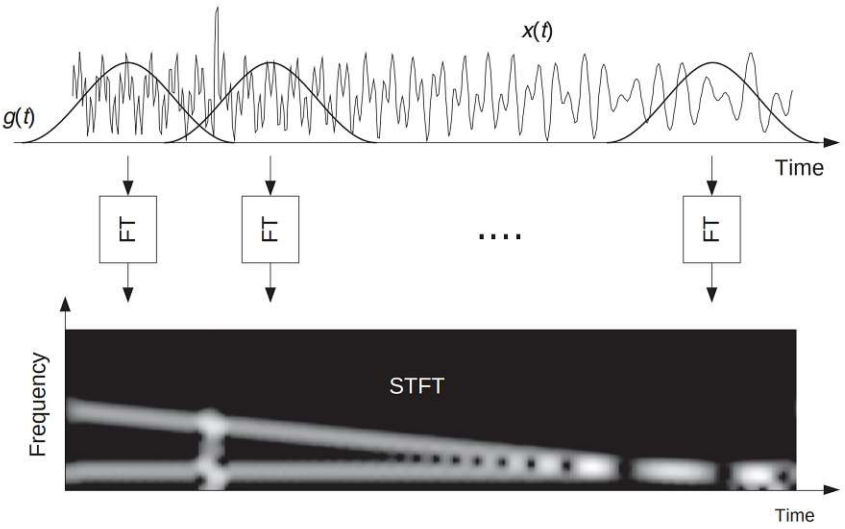


Figure 3.3: Calculation of the STFT [Kehtarnavaz, 2008]

3.2.2 Parameters

n_fft

This parameter defines the number of samples per FFT. Moreover, it also determines the number of frequency rows in the resulting matrix (1+n_fft/2). By default, Librosa uses 2,048 samples for this parameter, corresponding to a duration of 93ms per FFT for a sampling rate of 22,050Hz and 128ms for 16,000Hz. According to the documentation this value is well adapted for music signals.

win_length

Each audio frame of the STFT has a window of the size win_length, and to match the n_fft parameter, the window is by default zero-padded. A smaller window-size can improve the temporal resolution and therefore close tones in time can be better discriminated. However, this decreases also the frequency resolution, which is the ability to discriminate closely spaced tones in frequency. This effect is inevitable and is called the time-frequency localization trade-off, and the challenge for tuning this parameter is to find a good trade-off for all frequencies on average. The default value in Librosa is win_length = n_fft = 2048.

hop_length

The hop length defines the distance between the start of two adjacent FFT windows. It determines the number of columns which increases when choosing a smaller value or decreases for higher values. So smaller values provide a better temporal resolution but also require more space. The default value in Librosa is window-length / 4, so 512.

3.2.3 STFT in MIR

Compared to the other spectrogram input representations discussed in this thesis, pure STFT leads to a minor loss of information. Therefore, it also has the most extensive resource load, leading to a high memory consumption and training time. This might be the reason that STFT is usually not the method of choice. However, there are still a few studies using the STFT as preprocessing method like in [Choi et al., 2018], where it was compared to Mel spectrograms in the context of their performance on music autotagging. They evaluated the performance of both input representations for different usage-proportions of training data of the MSD. The study concludes that the classification performance is dependent on the preprocessing method, but it does not state which one performs better. However, when using the whole dataset the Mel spectrogram outperformed the STFT by a 0.2% higher ROC-AUC score. A similar investigation was conducted in [Huzaifah, 2017], where different time-frequency representations for CNNs are compared to classify environmental sound. Additionally to the MFCCs baseline, four frequency-time representations, each with a wideband and a narrowband configuration are used: STFT, Mel spectrogram, CQT, and Continuous Wavelet Transform (CWT). A bigger sliding-window size of 2,048 for the STFT is used for the wideband configuration, and for narrowband, it is 512. Mel spectrograms lead to the highest accuracies for almost all configurations, followed by CQT and then STFT. For most configurations, MFCCs performed worse than CQT, and both have significantly worse results than all other preprocessing methods. The evaluation further shows that wideband outperformed narrowband on all models and input representations. The following section describes the Mel spectrogram in more detail, which is an adaption of the STFT.

3.3 Mel spectrogram

To overcome the issue of the relatively big size of the basic STFT spectrogram, the Mel Scale can be used to reduce the frequency domain. This scale was designed to fit the human hearing capabilities since humans can detect differences of low frequencies, for example between 200Hz and 300Hz, much better than high frequencies like 8000Hz and 8100Hz. In contrast to the basic STFT, this signal representation was specially designed for audio signals [Moore, 2013]. The Mel spectrogram is an STFT spectrogram where the frequency channels are mapped to the corresponding Mel bins [Dörfler et al., 2017]. Figure 3.4 displays a Mel spectrogram of the same song as the STFT spectrogram in Figure 3.2. The following subsections describe the calculation and the most important parameters, and afterward, several studies using Mel spectrograms in different areas of MIR.

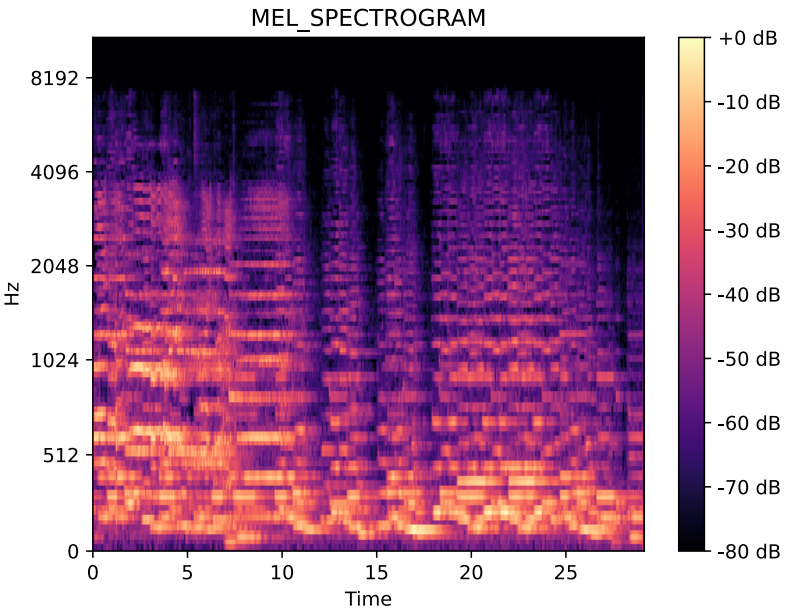


Figure 3.4: Mel spectrogram of 29s audio-file

3.3.1 Calculation of the Mel spectrogram

The first step for calculating the Mel spectrogram is to apply the STFT on the audio signal, which is explained in more detail in Section 3.2.1. Afterward, the frequency spectrum is binned into n_mels frequency-bins of equal size concerning the human perception of the sound. The amplitude for these bins is then derived by taking weighted averages over those frequency channels [Dörfler et al., 2017]. For this, the Mel scale is used, which defines the Mel unit so that values twice as large are perceived twice as high. For example,

a pitch of 500 Mels is subjectively twice as high as 250 Mels [Stevens et al., 1937]. The pitch function displaying the relation of frequency to Mels is displayed in Figure 3.5. It is based on the following equation, where Z refers to the Mel sound unit and f to the frequency [Pfister and Kaufmann, 2008]:

$$Z = 2595 * log_{10}(1 + f/700) \tag{3.5}$$

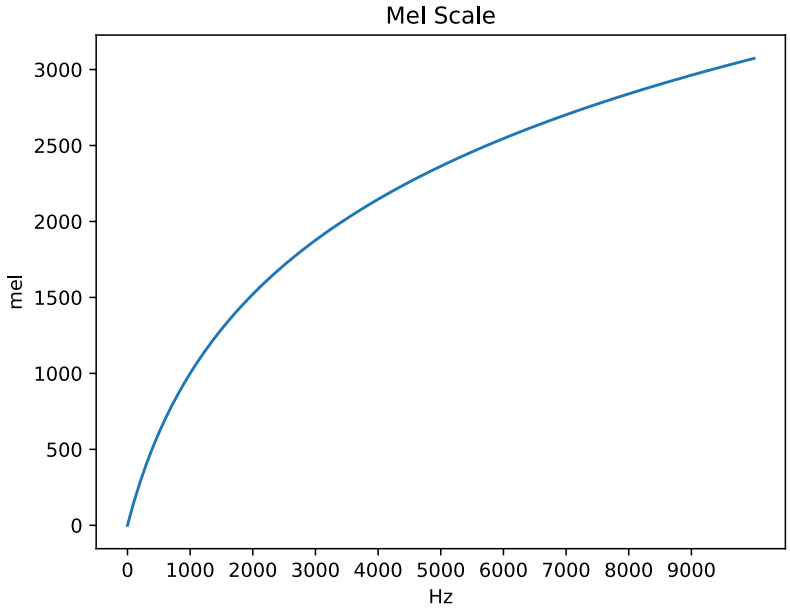


Figure 3.5: Mel scale

3.3.2 Parameters

The parameters of the Mel spectrogram are the same as for the STFT, described in Section 3.2.2. One essential additional parameter is the number of Mels, often set to 96 or 128 in MIR research for Mel spectrograms [Pons and Serra, 2019, Choi et al., 2018]. In [Ferraro et al., 2021], an investigation of the effects of using different Mel numbers and sampling rates on a VGG- (cf. Section 5.2) and a Musicnn-model (cf. Section 5.4) shows that for 12kHz audio using 96 Mels performs slightly better. For 16kHz, 128 Mels perform better for VGG and are equally good as 96 Mels on Musicnn. Therefore, no Mel number configuration significantly outperforms the other. It seems that the majority of other studies in MIR are using 96 Mels, so for this study, this configuration is used too. The following subsection presents some recent research using or focusing on Mel spectrograms.

3.3.3 Mel spectrogram in MIR

Although the Mel spectrogram is a lossy representation for audio data, it seems to be the most commonly used one throughout MIR. This might be due to its comparably good performance and still relatively compact size. In [Choi et al., 2016], different input representations, namely STFT, Mel spectrogram, and MFCCs, are compared on CNN models with different numbers of hidden layers on the MTAT dataset. Mel spectrograms outperformed the other preprocessing methods by far with a ROC-AUC score of 0.894 compared to 0.846 and 0.862 for STFT and MFCCs, respectively. A way to interpret the results of music autotagging is proposed in [Won et al., 2019] using self-attention. They use a frontend-backend architecture, comparing raw waveform and Mel spectrogram based model for the frontend and a CNN- and an Attention-model as backend. The best combination was the one with Mel spectrogram as frontend and a CNN as backend. The attention backend performed slightly worse but has the advantage of opening up the probability of creating attention heat maps. These provide information to highlight the most relevant sub-parts of the input spectrogram to predict a given tag. This information can be especially interesting for further research on specific topics in music autotagging, primarily when focusing on certain tags or tag categories. As seen before, it often outperforms all other preprocessing methods, although it is a lossy and relatively compact audio input representation. An even more compact representation is MFCCs which are described in the next section.

3.4 Mel Frequency Cepstral Coefficients

As seen in the previous section, Mel spectrograms usually consist of 96 to 128 Mels to map the frequencies for each point of time. The MFCCs are an even more compact input representation that reduces the frequency dimension down to 20 values. Around the year 2000, MFCCs have been the most dominant feature in speech recognition and have shown promising results in music analysis [Logan, 2000]. This input representation builds upon the Mel spectrogram and describes the overall shape of the power spectrum per frame in a compact form [Logan, 2000]. However, the resulting output vector for each point of time does not directly refer to sub-segments of the Mel scale. It is rather a feature vector [Pfister and Kaufmann, 2008]. This will be further described in the following subsection, followed by the description of the most important parameters and previous applications of MFCCs in MIR. Figure 3.6 shows the MFCCs of the same song as for the previous spectrograms.

3.4.1 Calculation of the MFCCs

The calculation for the MFCCs starts with the same steps as the Mel spectrogram, cf. Section 3.3.1. The last step is then to build the cepstral vectors is to apply the Discrete Cosine Transform (DCT) to the Mel spectral vectors to decorrelate their components and further reduce the number of parameters without losing too much relevant information. By default Librosa uses the DCT-2 transformation, cf. [Logan, 2000].

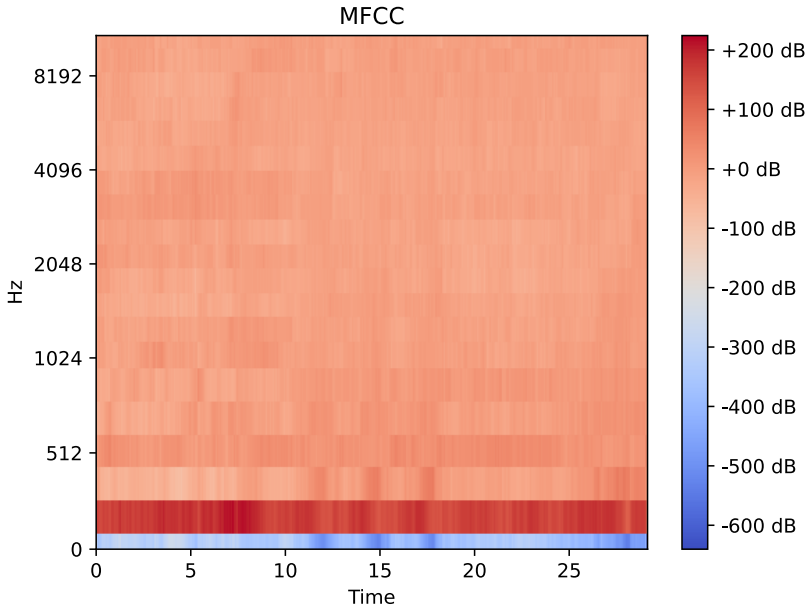


Figure 3.6: MFCCs of 29s audio-file

3.4.2 Parameters

The set of parameters is the same as for the Mel spectrogram (cf. Section 3.3.2) and the most important additional parameter is the total number of MFCCs. The default value in Librosa is 20, which is also used for this thesis [2].

3.4.3 MFCCs in MIR

Genre- instrument- and mood classification are common use cases for MFCCs, and there are many approaches, mainly using CNNs, e.g. [Dutta and Chanda, 2021], [Wu et al., 2017], [Vishnupriya and Meenakshi, 2018]. Another important topic in MIR research is music recommendation systems. In [Han et al., 2018], a recommendation algorithm relying on the eigenvalue similarity of the MFCCs is presented to deal with missing user data. The preprocessing method is used to apply vector quantization and perform K-means clustering to determine similarity scores used for the recommendation. The system was evaluated on a dataset with 1,044 songs and six different emotion categories to recommend songs from the same category as a given song. In total, the experiments reached an accuracy of 87% for this task. A rather exotic topic where MFCCs can be helpful is dance generation for songs. In [Tendulkar et al., 2020], a system was developed which creates a dance choreography for music with a greedy search algorithm based on an alignment score that measures the linear association between the input representation

[2] https://librosa.org/doc/main/generated/librosa.feature.mfcc.html, accessed 15-January-2022

and a predefined dance matrix, consisting of different dance moves. These movements and actions are executable in parallel or sequential. A human evaluation showed that the generated dances are 61% to 99% more creative than different baselines, acting more or less randomly and synced to the beat.

3.5 Constant-Q Transformation

As for the previous input spectrograms, the CQT transforms an input audio signal into a time-frequency representation. This transformation should tackle the problem that the DFT used by the previously described preprocessing methods yields frequency components separated by a constant frequency difference. Therefore, they are not mapping the tone scale of Western music. Figure 3.7 displays a CQT spectrogram of the same song as for the previous spectrograms. The CQT is calculated similar to the FFT but provides a constant ratio of the center frequency to resolution. The following section describes this calculation in more detail followed by the most important parameters where the default values of Librosa are used for the experiments [3]. The last subsection presents recent research using CQT in music autotagging and auto-transcription [Brown, 1991].

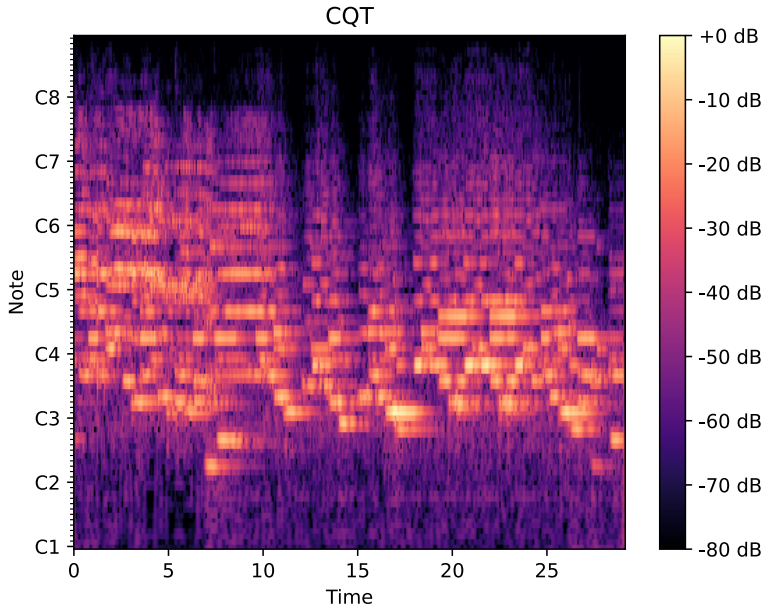


Figure 3.7: CQT of 29s audio-file

[3] http://librosa.org/doc/main/generated/librosa.cqt.html, accessed 15-January-2022

3.5.1 Calculation of CQT

The calculation of the CQT is very similar to the DFT, but to provide the constant ratio of the center frequency to resolution there are a few crucial differences. The frequency of the k-th spectral component is defined by:

$$f_k = (2^{1/24})^k f_{min}, \tag{3.6}$$

where the exponent 1/24 determines the resolution, so now it refers to quarter-tone spacing, and it would be 1/12 for half-tone spacing. f_{min} is the minimum frequency that can be chosen and determines where the frequency information in the spectrogram begins. The resolution, also called bandwidth δf equals the sampling rate divided by the window size for the FFT. These parameters have to be adapted for CQT so that the ratio of frequency to bandwidth, denoted as Q, to be constant. This relation can be described by the following equation (for quarter-tone resolution):

$$Q = f/\delta f = f/(1 - (2^{1/24}))f = 34 \tag{3.7}$$

Using this constant Q the variable window-function N[k] is defined as followed:

$$N[k] = S/\delta f_k = (S/f_k)Q \tag{3.8}$$

The final equation for each frequency component X[k] can be now defined by using these new definitions on the FFT. As for the other spectrograms, the default window function W[n] is the Hamming window (cf. Equation 3.4). Since the number of terms varies with k, the sum is normalized by dividing it by N[k], which leads to the following final equation:

$$X[k] = \frac{1}{N[k]} \sum_{n=0}^{N[k]-1} W[k,n]x[n]exp(-\frac{j2\pi Qn}{N[k]}) \tag{3.9}$$

The following subsection lists the most important parameters for the calculation by their name in Librosa and their value for the experiments conducted in this study [Brown, 1991].

3.5.2 Parameters

hop_length

As for the STFT, the hop-length defines the distance between the start of two adjacent FFT windows. This determines the number of columns that increases when choosing a smaller value or decreases for higher values. So higher values provide a better temporal resolution but also require more space. Librosa uses a value of 512 by default.

fmin

Defines the minimum frequency where the CQT starts. This value can be chosen depending on the individual use case. The default value of 32.70Hz already provides a good starting point for music-classification tasks because this is approximately where the human perception of sound starts [Görne, 2008].

bins_per_octave

The number of bins per octave. It determines the frequency resolution of the CQT, e.g., 12 is used for half-tone resolution, 24 for quarter-tone, etc. For the experiments, 12 is used, to match the size of the Mel spectrogram and increase comparability.

n_bins

The number of total frequency bins. Together with the bins_per_octave and the fmin parameters, determines the total frequency-bandwidth covered by the CQT. In order to get the same size as for the Mel spectrogram, 96 bins are used, so with 12 bins per octave, there are 8 octaves from 32Hz to 8,000Hz in the resulting spectrogram.

3.5.3 CQT in MIR

The CQT transformation is commonly used for automatic music-transcription systems as presented in [Meng and Chen, 2020], which uses both MFCCs and CQT as input. It uses a CNN for classification and trains the network with a monophonic dataset of 2,000 audio samples of different instruments. MFCCs are used for timbre classification to recognize the instrument, and CQT is used for pitch classification. Since only monophonic training- and test data were used, the system reached very good prediction-accuracies for instrument- and pitch classification of 0.9394 and 0.9472, respectively. Comparing different time-frequency representations for environmental sound classification using CNNs in [Huzaifah, 2017] CQTs reached results close to Mel spectrograms and outperformed STFT and MFCCs. (For more information, see Section 4.1.2)

CHAPTER 4

Datasets

For the purpose of this study, two pre-tagged music datasets are used, MagnaTagATune ("MTAT") [Law et al., 2009] and MTG-Jamendo ("MTGJ") [Bogdanov et al., 2019c]. One main selection criteria for the datasets is that they contain tags belonging to all of the three different categories "genre", "instrument" and "mood" which seem to be the most important ones for music auto-tagging. One could also name "artist" as another important category, but since the occurrence of the corresponding tags would, in most cases, be very low, it is not used [Bertin-Mahieux et al., 2010, Marques et al., 2011]. Since both datasets are commonly used in music autotagging research, there are many reference values, i.e., for the classification performance for various deep-learning models, as shown in the following subsections. The experiments are evaluated on MTAT to answer the research questions, validate those results. To ensure a better generality they are additionally run on MTGJ afterward. The two datasets are discussed in more detail in the following sections. Finally, other popular datasets for music autotagging are briefly presented.

4.1 MagnaTagATune

The datasets were created in the course of a game called TagATune, which launched in May 2008 [Law et al., 2007]. It collects music tags from players, packaged as a two-player online game. The game´s objective is to annotate a piece of music with a set of tags and then, after seeing the tags of the other player, to decide if he or she had the same song to annotate or not, which can change randomly for each round. When no other player is available, the player is paired with a computer bot that uses recent user annotations or generates them by an algorithm. Ultimately tags are accepted for a particular track if more than two players independently agree on them [Law et al., 2009].

4.1.1 Description

MTAT consists of 5,223 songs, divided into 25,863 song-chunks with a length of 29 seconds and a sampling rate of 16kHz in the mp3-format. The audio signals are monophonic and have a bit rate of 32kB/s [Güçlü et al., 2016]. Compared to standard CD-quality with a sampling rate of 44.1kHz at 16 bit, which results in a bit-rate of 1411.2kB/s for a stereo signal, this is relatively low quality. It is observable auditorily that the sound quality of the audio files is relatively poor, which is also stated in [Gus Berry, 2021] that below a bit-rate of 90kB/s, one can hear a significant deterioration in the audio signal quality. The worse quality will probably affect the classification performance but significantly reduce the file size, which can benefit memory consumption and classification speed. This may be an advantage for researchers who want to compare their results with others but can be problematic for production applications [Sinclair, 2000].

Each song-chunk provides binary annotations of 188 tags, belonging to genres like ”classical”, ”techno” or ”rock”, to instruments like ”guitar”, ”strings” or piano ”, to moods like ”slow”, ”loud” or ”soft” and some to none of them like ”solo” or ”english”. Many of those tags are just synonyms for each other so in a first preprocessing-step tags such as ”choir” and ”choral” or ”horn” and ”horns” are merged, resulting in a logical or-connection of the merged labels.

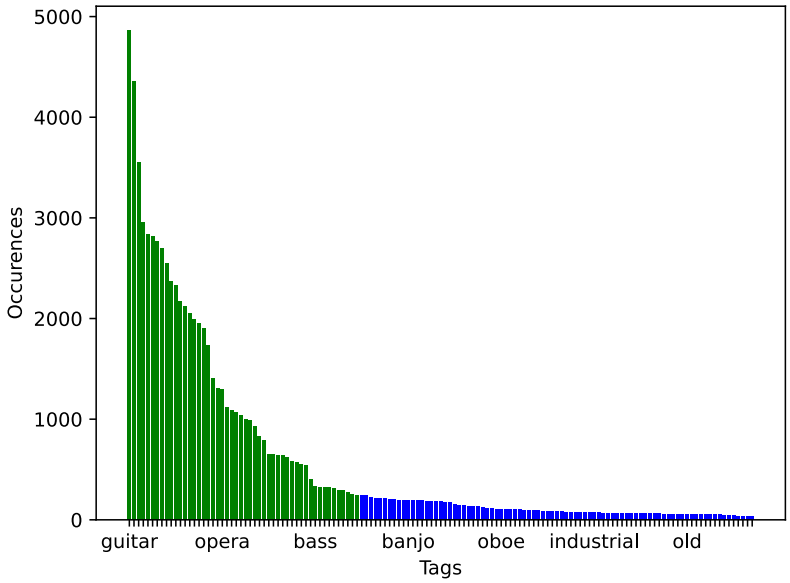


Figure 4.1: Frequency of tags in the MTAT dataset

The number of occurrences of each tag in all song-chunks after tag-merging is depicted in Figure 4.1. We can observe that this number is unequally distributed, and tags like ”guitar” occur nearly 100-times more often than, for example, ”clarinet”. This

may lead to poor prediction performances for less frequent tags because it is hard for a DL model to generalize classifications learned from 50 training samples compared to 5,000. To overcome this problem, only the top 50 tags are used for training and classification which is a common approach in numerous other studies [Kim et al., 2018, Choi et al., 2016, Lee and Nam, 2017]. This results in 17 genre-, 22 instrument-, 9 mood- and two uncategorized tags. For evaluating each category, the number of average song-chunks per tag for that category can also be an important indicator to understand and interpret the results better. As shown in Figure 4.2, genre and mood tags have a similar average number of song-chunks, whereas instruments have about a third more. This could slightly influence the classification performance in favor of instrument tags since more training samples are available. However, the training-set size is only one of many factors. For example, it also heavily depends on the input representation of the songs or the semantics of the categories and the tags itself. This will be further discussed in Chapter 7.

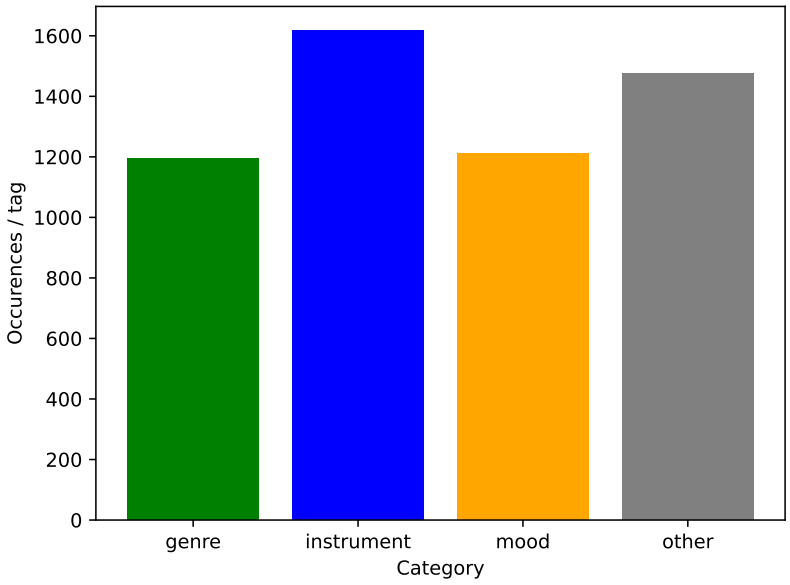


Figure 4.2: Average number of song-chunks per tag for each category - MTAT

The datasets are structured into 16 different folders of approximately the same size. All song chunks for each song are located in the same folders. A common approach for data-splitting is to use the first twelve folders as train-, one as validation- and three as test-set [Won et al., 2020]. To use the same strategy for both datasets, a 60:20:20 data split is used, which leads to a slightly smaller train- and test-set than in some reference studies using MTAT and which seems to be a popular approach in DL. This data split is also used for the predefined splits of MTGJ and is, therefore, a good reference value [Bogdanov et al., 2019c]. Important to note is that the dataset must not be randomly

shuffled before splitting because otherwise, the results get distorted when chunks of the same songs are present in different data splits.

4.1.2 Previous findings

Since the dataset has existed for over a decade now and there are only a few comparable alternatives for music autotagging, it has become one of the primary reference datasets in this research area. Apart from music autotagging, there is some research regarding music similarity for recommendation systems [Wolff and Weyde, 2012, Wolff et al., 2015]. In [Nam et al., 2015], a two-stage learning model is presented, using unsupervised-learning procedures for preprocessing followed by a Dense Neural Network (DNN) on the extracted bag-of-features. This resulted in ROC-AUC scores ranging from 0.845 to 0.888 for the different experiments. An alternative approach to CNNs with the commonly used preprocessing methods in MIR was used in [Song et al., 2018], where scattering transform as preprocessing method served as input for an RNN, resulting in an ROC-AUC score of 0.909. A more extensive evaluation of CNN-based music auto-tagging models was conducted in [Won et al., 2020]. The ROC-AUC and PR-AUC scores range from 0.8703, resp. 0.3625 for a Convolutional Recurrent Neural Network (CRNN) up to 0.9129, resp. 0.4614 for a short-chunk CNN with residual connections. In total, twelve models were evaluated and compared, and the results are summarized in Table 4.1. Compared to other recent studies, the PR-AUC scores are significantly better. For example, in [Pons et al., 2018], a ROC-AUC of 89.05 and a PR-AUC of 34.92 were reported for a waveform model and 90.40 and 38.11 for a spectrogram model, respectively. A reason for the significantly better PR-AUC results in [Won et al., 2020] could be a different preprocessing and subset creation of the data, but the exact steps are not stated there.

Methods	MTAT		MSD		MTG-Jamendo	
	ROC-AUC	PR-AUC	ROC-AUC	PR-AUC	ROC-AUC	PR-AUC
FCN	0.9005	0.4295	0.8744	0.2970	0.8255	0.2801
FCN (with 128 Mel bins)	0.8994	0.4236	0.8742	0.2963	0.8245	0.2792
Musicnn	0.9106	0.4493	0.8803	0.2983	0.8226	0.2713
Musicnn (with 128 Mel bins)	0.9092	0.4546	0.8788	0.3036	0.8275	0.2810
Sample-Level	0.9058	0.4422	0.8789	0.2959	0.8208	0.2742
Sample-Level + SE	0.9103	0.4520	0.8838	0.3109	0.8233	0.2784
CRNN	0.8722	0.3625	0.8499	0.2469	0.7978	0.2358
CRNN (with 128 Mel bins)	0.8703	0.3601	0.8460	0.2330	0.7984	0.2378
Self-attention	0.9077	0.4445	0.8810	0.3103	0.8261	0.2883
Harmonic CNN	0.9127	0.4611	**0.8898**	**0.3298**	0.8322	0.2956
Short-chunk CNN	0.9126	0.4590	0.8883	0.3251	**0.8324**	**0.2976**
Short-chunk CNN + Res	**0.9129**	**0.4614**	**0.8898**	0.3280	0.8316	0.2951

Table 4.1: Performance evaluation results of CNN-models in [Won et al., 2020]

4.2 MTG-Jamendo

MTGJ includes only royalty-free music for commercial use, which is not the case for MTAT or the MSD (cf. Section 4.3). A subset of the annotations is gathered from an open API, but it is not stated exactly how the rest was labeled [Bogdanov et al., 2019c].

4.2.1 Description

The dataset contains 55,709 full-length songs of 3,565 different artists. Every song is at least 30s in length and 224s on average, resulting in 3,777 hours of audio. The sampling rate is 44.1kHz, encoded in stereo-mp3, and the songs have a total bit rate of 320kB/s. Since this is a very high audio quality, the dataset has a size of 509GB which is a considerable high value, especially for dealing with multiple DL models and preprocessing methods [Bogdanov et al., 2019c]. To reduce the size significantly, the audio is downsampled to 16kHz like MTAT, resulting in a bit rate of 127kB/s. Furthermore, for this study, only the first 29s of each audio segment is used, which strongly reduces training- and test-data size. Since the experiments include 25 different combinations of CNN models and preprocessing methods, this is a necessary preprocessing step to evaluate them in a reasonable amount of time. The downside of those reductions in quality and quantity of the dataset is that classification performances worsen and the comparability to previous studies is only possible to a limited extent. However, the purpose of this work is not to reach new state-of-the-art results but rather to compare the different preprocessing methods to each other.

In total, there are 195 different tag annotations for each song, each belonging to one of the three categories ”genre”, ”instrument”, or ”mood”. The set of tags is already cleaned up and does not contain tags with the same meaning since the dataset´s creators already performed an extensive re-mapping of them [Bogdanov et al., 2019c].

Figure 4.3 depicts the number of songs for each tag. The distribution is very similar to MTAT, and it is unequally distributed. Therefore, as suggested in [Bogdanov et al., 2019c], only the top 50 most frequent tags are used for the experiments. There are 31 genre-, 14 instrument- and 5 mood-tags. Taking a more detailed look into the numbers, it is essential to note that genre- and instrument tags appear much more often than mood tags. Figure 4.4 compares the average song number per tag for each category and underlines this observation. As stated before, this could affect the classification performance to the disadvantage of mood tags, but it is only one of several factors. In Chapter 7, this will be further discussed.

The Github-Repository of MTGJ [Bogdanov et al., 2019c] contains all the music data and provides a collection of useful scripts for preprocessing, including five predefined random dataset splits. They ensure that no track appears in more than one subset, tracks of the same artists are only present in one subset, the same tags are present in all subsets, and that there are at least 40 and 20 tracks from 10 and 5 artists for each tag in all subsets, respectively. The split ratios are approximately 60:20:20, and to ensure

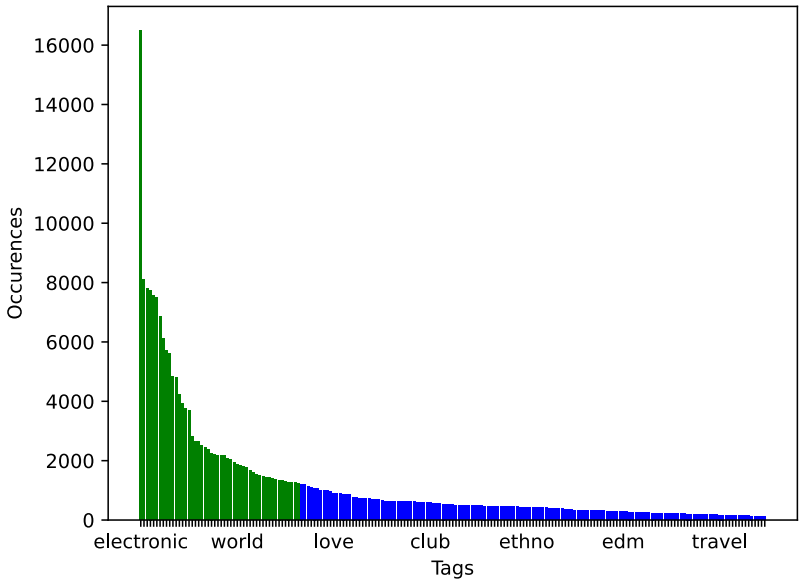


Figure 4.3: Frequency of tags in the MTGJ dataset

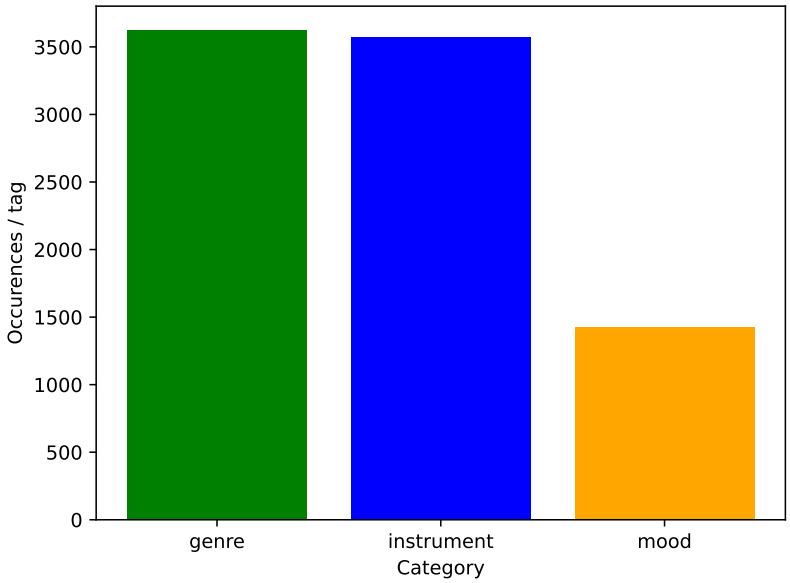


Figure 4.4: Average number of song-chunks per tag for each category - MTGJ

better comparability to recent research, the first of the given sub-splits is used for the experiments [Bogdanov et al., 2019c].

4.2.2 Previous findings

There are several papers [Dipani et al., 2020, Gerczuk et al., 2020] regarding Emotion and Theme Recognition in Music because the authors of MTGJ have been organizing research challenges for this topic since 2019 [Bogdanov et al., 2019b]. The creators of the dataset provide some baseline results for different subsets of the database for which they used a simple 5-layer CNN with 3x3-filters proposed in [Choi et al., 2016]. The evaluation for all 195 tags with all tracks leads to ROC-AUC score of 0.7197 and a PR-AUC score of 0.736. The latter one is rather low because the great majority of tags are very sparse and therefore hard to predict. The evaluation of the top-50 tags shows an ROC-AUC score of 0.7549 and a PR-AUC score of 0.1924. Important to note here that only one centered 29.1s audio segment per song was used here as well, which leads to good comparability to the results of this study. However, for this baseline, full audio quality was used compared to the down-sampling to 16kHz for the experiments of this study [Bogdanov et al., 2019c]. The results in [Won et al., 2020] discussed in Section 4.1.2 and depicted in Table 4.1 show significantly higher ROC-AUC and PR-AUC values for MTGJ. This might be because the full songs are used there, but it is not precisely stated if this is the case. Results for ROC-AUC and PR-AUC range from 0.7978 and 0.2358 up to 0.8324 and 0.2976, resp.

4.3 Other datasets

There are numerous different music datasets annotated with either genre-, mood-, or instrument tags. However, only a few contain annotations for all three categories covered in this section. A small but accurately annotated dataset is Cal500. It contains 500 songs annotated with 135 musically-relevant tags, spanning from instruments, genres, and emotions to song concepts, describing, for example, the acoustic qualities of the recording and usage terms, i.e., in which personal surroundings this song could be appropriate. The tags are annotated manually and checked by three people, and there is a version with 10,000 songs as well, but it contains only the annotations and no song data [Turnbull et al., 2007]. A very popular dataset in MIR is the MSD which was created to help researchers transfer MIR technologies into the commercial world by providing a vast dataset. As the name says, it contains metadata for one million popular music tracks from 44,745 unique artists. There are many variations and subsets available online, like the Last.fm dataset that consists of 943,347 songs from MSD and is labeled with 522,366 unique tags [Bertin-Mahieux et al., 2011]. A common use case in music auto-tagging is using MSD as a second reference dataset for testing the evaluation results on large quantities of songs. A typical combination is to use MTAT, which is much more lightweight and easier to work with together with MSD to validate the results [Lee et al., 2017b, Song et al., 2020, Pons et al., 2018]. Other

popular datasets that do not necessarily provide all of the three tag-categories are FMA, consisting of 106,574 tracks annotated with 161 genres [Defferrard et al., 2017], Music4All, consisting of 109,269 songs, annotated with genre- and mood tags [Santana et al., 2020] and AcousticBrainz, which only contains metadata for genre annotations for about two million songs [Bogdanov et al., 2019a].

CHAPTER 5

CNN models

The previous section discussed the most important datasets for music autotagging. Most of the research done on these datasets is based on CNNs. CNNs are one of the most popular deep neural networks and have become state-of-the-art in many classification tasks dealing with grid-like topologies, especially in the area of image recognition and computer vision [Albawi et al., 2017]. As stated in the previous sections, they provide good results in several MIR-areas, and especially in music autotagging, they are the most dominant artificial neural network used. In this chapter, the CNN and its main components are described in more detail, and the five CNN architectures used for the experiments are presented.

5.1 The CNN structure

The main component of the network is called convolution, which is a linear mathematical operation between matrices. The other basic layers are the pooling layer, the FC layer, also called dense layer, and the activation function. The network learns the parameters of the convolutional filters and the FC layer. At the same time, the pooling reduces the input size for deeper layers, and the activation function provides the necessary non-linearity [Albawi et al., 2017]. A typical CNN architecture is displayed in Figure 5.1. In the following subsections, those components are described in more detail regarding their functionality, role, and parameters.

5.1.1 Convolution

The primary purpose of the convolutional layer is to extract spatial information, usually out of a multidimensional array, and generate feature maps that are modified versions of the original image. For this purpose, one uses convolutional filters applied with the convolutional operation on the input image, which generate one feature map per filter

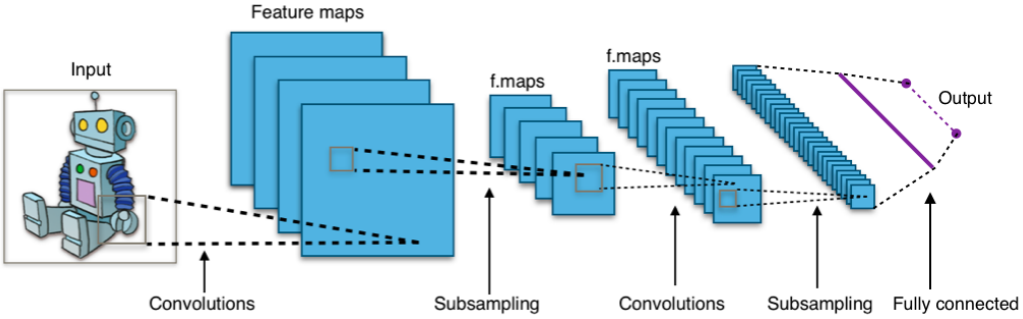


Figure 5.1: Typical CNN architecture [Aphex34, 2015]

[Kim, 2017] (cf. Figure 5.1). For a two-dimensional image I as input, this operation can be expressed through the following equation:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n), \tag{5.1}$$

where $S(i,j)$ is the resulting feature-map with height and width i and j respectively and K is the filter-kernel of height m and width n [Goodfellow et al., 2016, p. 322-324]. Figure 5.2 provides a visualization of a sample image I of size 7x7 with a 3x3 filter-kernel K. K is applied without padding (cf. Section 5.1.1) and a stride of 1 (cf. Section 5.1.1).

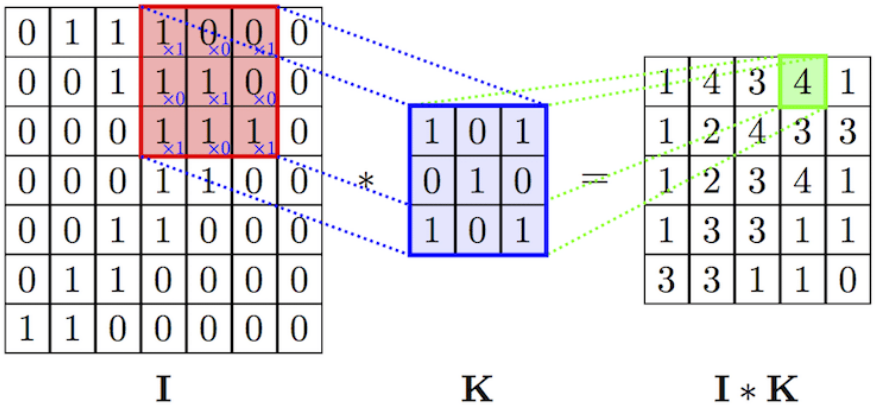


Figure 5.2: Convolution example [Mohamed, 2017]

During training, the filter kernels serve as trainable parameters, so there are $p = f * m * n$ parameters in total for a convolutional layer, where f is the number of filter kernels and m and n are the height and width of the kernels. A common practice is to double the number of filter kernels after each layer and use pooling to reduce the size of feature maps to prevent running out of memory which is described in the following section [Kim, 2017]. Beneath the number of filters and the kernel size, the padding and stride are essential parameters, which are described in the following subsections.

Padding

Working with convolutions, one important aspect is how to deal with the borders of the input. This is determined by the padding parameter. There are two basic padding-strategies, ”valid” and ”same” padding. ”Valid” padding is the simplest form, where no constants are added to the input and the filter kernel is applied until the edge-values of the input, leading to a smaller output depending on the input size used in Figure 5.2. ”Same” or also called ”Zero” padding is used, when the output layer shall have the same size as the input layer. for this, exactly as many zeros are inserted around the input, so that the size remains the same after the convolution operation [Sewak et al., 2018].

Stride

The stride defines the displacement of the filter kernel after each step and has a default value of 1 for each dimension. A bigger value means bigger space and less overlap between the convolutional filter steps. Furthermore, the bigger value leads to smaller output values and usually more information loss. The stride can be defined for each spatial dimension separately [Google, 2022].

5.1.2 Pooling

Pooling layers are used to reduce the size of the input array by combining neighboring pixels to single representative values. As for the convolution, filter kernels of an adjustable size are used. The two most important variants are mean- and max pooling, which use average or maximum values, respectively, visualized in Figure 5.3. By default, the stride for the filter kernels is set the same size as the kernel size itself so that there is no overlap. The padding works identically as for the convolution, described in Section 5.1.1.

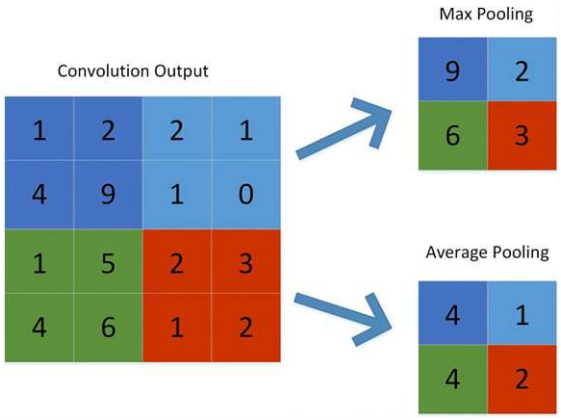


Figure 5.3: Example mean- and max pooling [Jiang et al., 2020]

5.1.3 Activation function

To learn complex mappings from the data, neural networks use activation functions. The type of activation function that is used determines the prediction accuracy of the neural network. When using linear functions or no functions at all, the networks would work just like linear regression models. However, for the most real world problems, the errors possess non-linear characteristics and therefore non-linear activation functions are preferred over linear ones [Sharma et al., 2017]. To apply the back-propagation algorithm, those functions need to be differentiable. This algorithm is used for computing the gradient of the cost function during training of the network. This gradients can then be used for applying optimization algorithms such as the stochastic gradient descent to perform learning, cf. [Goodfellow et al., 2016, p. 197-198]. The most important activation functions include the Hyperbolic Tangent-, the Sigmoid- and different types of the ReLU function. ReLU is the most commonly used function because of its simplicity and efficiency. It is defined by:

$$\begin{cases} f(x) = x & x \geq 0 \\ f(x) = 0 & x < 0 \end{cases} \tag{5.2}$$

and it is used for all non-output function throughout the experiments of this thesis. For CNNs, the activation function is applied directly after each convolutional layer [Sharma et al., 2017].

5.1.4 Fully connected layer

After the last convolution- and pooling block, the feature maps are flattened to one dimension and then fully connected to every neuron of the next layer. For the most basic CNN architecture, this part of the model can be called the classifier network compared to the previous feature extraction network. So it uses the extracted features from the convolutions and generates the final output. The classifier network can be arbitrary deep but usually contain no more than four layers, depending on the CNN architecture [Kim, 2017].

5.2 VGG-16

To assess the performances of the different models and input representations on the respective tag categories, it is important to have a baseline as a reference for each configuration . For this purpose, the VGG-16 model defined in [Simonyan and Zisserman, 2015] is used. Around 2015 this model had achieved state-of-the-art results in image-classification tasks, and compared to the other models, it is a straight-forward deep CNN with 16 layers.

The standard VGG-16 architecture is designed for fixed-size 224x224 input images. Since this model serves only as a baseline, there are no significant structural changes, only minor adaptions. As depicted in Figure 5.4, the model consists of five convolutional

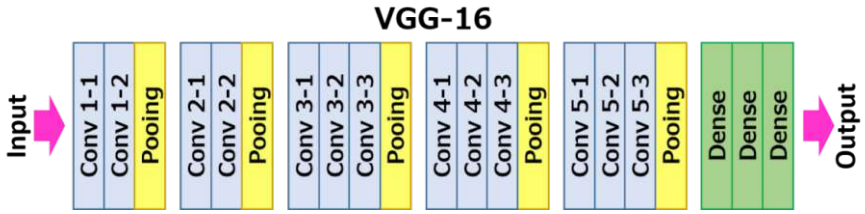
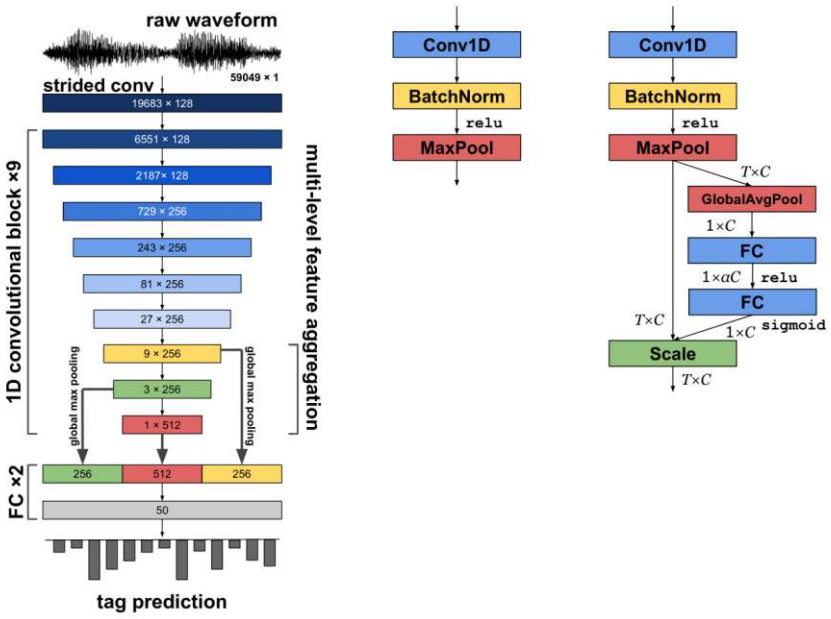


Figure 5.4: VGG16-architecture [ul Hassan, 2018]

blocks, each with several convolutional layers, followed by one FC block. Each convolution uses a 3x3 filter kernel, and in each block, the same number of filters is used for every layer, which is 64, 128, 256, and 512 for the last two, respectively. Important to note for this and all following networks: when speaking of 3x3 or 2x2 filter kernels, this always refers to the two-dimensional inputs and implies a 3x1 and 2x1 filter for one-dimensional inputs, respectively, if not stated further. At the end of each convolution block, a 2x2 max-pooling filter is applied to reduce the input size and enable more filters in the next block. In the last block, the flattened output of the previous convolutional block passes two FC layers with 4,096 channels each and the output layer with 50 for predicting the tags [Simonyan and Zisserman, 2015]. ReLU is used as an activation function for all layers, and dropout- and batch-normalization layers are added after each block to enable the necessary regularization for the different input representations.

5.3 Squeeze and Excitation Network

This model is based on the sample-level CNN architecture proposed in [Kim et al., 2018], which outperformed all existing state-of-the-art models for the MTAT dataset in 2018. It uses one-dimensional raw audio-data as input and consists of 9 basic blocks followed by two fully connected layers as shown in Figure 5.5. As shown in Figure 5.5, a basic block consists of a convolutional layer with filter size 3x3, followed by batch-normalization and a max-pooling layer with filter size 2x2 for 2D and 3 for 1D-input. The outputs of the last three blocks are each max-pooled globally and concatenated to serve as input for the FC layers. The Sigmoid function is used as an output function to provide results between 0 and 1. To increase the representational power of each block, Squeeze and Excitation (SE) blocks are used as an extension of each basic block, as shown in Figure 5.5c. These blocks shall improve the learning of convolutional features by explicitly modeling channel inter-dependencies to increase the sensitivity to informative features. The squeeze operation takes the output of a basic block $U \in \mathbb{R}_{T \times C}$ as input and applies global average pooling on it to compress the global spatial information of each channel into a single channel descriptor. Global average pooling is preferred to global max pooling because experiments in [Hu et al., 2018] showed that the performance of SE blocks is robust to the choice of the specific aggregation operator and that it slightly outperformed global max pooling. Formally, this can be described by the equation



(a) Overview of the architecture (b) Basic block (c) SE block

Figure 5.5: Squeeze and Excitation Network (SENet)-architecture overview Kim et al., 2018

$$z_c = \frac{1}{T}\sum_{i=1}^{N} u_c(i), \tag{5.3}$$

with u_c denoting the different filter channels. The result $z_c \in \mathbb{R}_{1\times C}$ then serves as input for the excitation operation, where channel-wise dependencies should be captured to use the information aggregated before. This step consists of two fully-connected layers with a ReLU activation function and an amplification-factor α to allow variation in the capacity of SE-blocks and then the Sigmoid-function to form the output $S_c \in \mathbb{R}_{1\times C}$. Finally, the former output of the basic-block U get scaled with the output of the SE-block by simple channel-wise multiplication of S_c and U [Kim et al., 2018]. Due to the relatively deep layer architecture, the model does not work for all used input representations without minor adaptions because the pooling operation reduces the dimensionality in each layer. To overcome this problem, more compressed preprocessing methods like MFCCs horizontal filter kernels, with a size of, for instance, 1x2 are used for several layers.

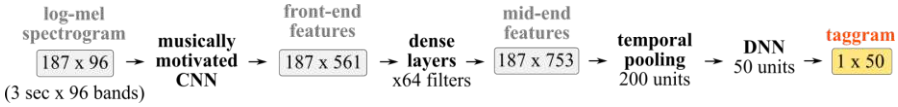


Figure 5.6: The Musicnn-architecture [Pons and Serra, 2019]

5.4 Musicnn

Musicnn is a pretrained CNN for music autotagging and is reproduced and slightly adapted for this thesis. As depicted in Figure 5.6, the architecture is divided into three parts: a musically motivated CNN frontend, a midend consisting of three densely connected convolutional layers, and a backend using temporal pooling. The model originally uses Mel spectrograms with a length of three seconds and 96 Mel-bins as input [Pons and Serra, 2019].

Frontend Experiments in [Pons et al., 2016], [Pons et al., 2018] suggest that the use of musical domain knowledge in the architectural design of the network can increase the classification performance, especially for smaller datasets. Therefore, the frontend uses convolutional filters with different horizontal and vertical kernels to learn the timbral and temporal patterns present in the spectrograms. For Mel spectrogram inputs, for example, kick-drum-patterns are captured by 7x38 filter kernels. 7x67 filters serve for instruments with a broader frequency spectrum like string ensembles. To capture different time-scale representations, the model uses horizontal filters with different sizes, i.e., 128x1, 64x1 and 32x1 [Pons et al., 2018]. The sizes of the timbral filter kernels are adapted for all other two-dimensional input representations and scaled accordingly to capture the same proportions of the total frequency spectrum. For the one-dimensional input, the frontend-architecture as proposed in [Pons et al., 2018] is used, consisting of seven 3x1 convolutions, combined with max-pooling layers.

Midend Three densely connected convolutional layers extract higher-level representations from the low-level features of the frontend. As proposed in [Huang et al., 2017], densely connected means that each layer is connected with every other layer in the network via concatenation of each input with the previous outputs. However, the Musicnn-implementation is slightly adapted and uses residual connections between the layers, so summation is used instead of concatenation. Only the output layer concatenates all previous layers of the midend.

Backend The backend uses temporal pooling for the final output. It combines average- and max-pooling over the temporal dimension of the output of the midend concatenates their output-matrices and then uses two dense-layers with finally a Sigmoid output-function for the final prediction [Pons et al., 2018].

5.5 ResNet

With increasing depth, the classification performance of CNN models has improved significantly over the last few years in image classification and other visual recognition tasks. The downside of this is the increased complexity and difficulty in training. The usage of residual connections allows going deeper and reducing complexity at the same time [He et al., 2016].

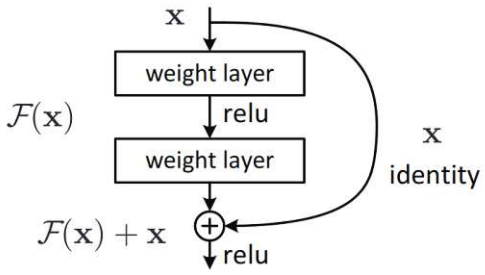


Figure 5.7: Residual connection: building block [He et al., 2016]

A residual building block, as depicted in Figure 5.7, can formally be defined as:

$$y = \mathcal{F}(x, W_i) + x. \tag{5.4}$$

The function $\mathcal{F}(x, W_i)$ represents the residual mapping, and x and y denote the input and output matrices of the building block. When using two layers per residual block, as in Figure 5.7, $\mathcal{F} = W_2\sigma(W_1x)$ where σ denotes the activation function, i.e., ReLU and the bias. After the element-wise summation of $\mathcal{F} + x$, the second activation function is applied. The number of layers used for residual blocks can vary [He et al., 2016].

For this study, a ResNet-101 model is used, consisting of 101 layers. For this, the implementation of [He et al., 2016] was adapted for Tensorflow. The basic building block here is a Bottleneck-layer which consists of a 1x1, 3x3, and another 1x1 convolution and a residual connection from the input to the output of the last layer. The 1x1 layers reduce the input dimensions about four times for the 3x3-layer and increase it again afterward, which is an efficient way of creating deeper networks without running out of memory. After the first convolutional layer with a 7x7 filter kernel to downsize the input, the model consists of 3 bottleneck-blocks with 64 filter channels, 4 with 128, 23 with 256, and 3 with 512, followed by the Sigmoid output layer.

5.6 Dilated CNN

This section presents the developed dilated CNN during the experiments in Section 6.3.4. Therefore, this is one of the main contributions of this thesis. Dilated convolutions are widely used for image segmentation and getting more attention for image classification.

They enable to use wider filter kernels without increasing resource consumption. Using images as input, a dilation factor l=2 can be described as ”skipping” one pixel between each other in the filter-kernel (l=1 means no dilation). Figure 5.8 visualizes the projection of the dilated convolution to the output. With increasing l, the number of pixels between each other rises by the same number, so l=4 will skip 3 pixels and result in a total kernel width of 9, using a 3x3 filter-kernel [Lei et al., 2019].

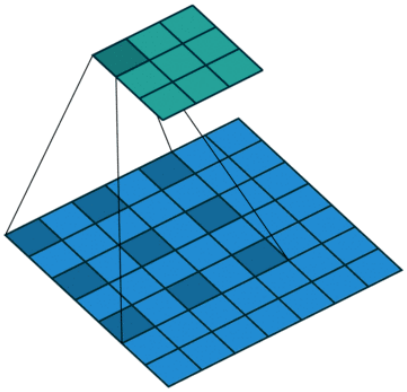


Figure 5.8: Dilated Convolution with l=2 [Tsang, 2018]

Dilated CNNs can be interesting for music autotagging because genre- and especially mood-tags intuitively depend stronger on longer audio segments than on short ones. Therefore, using bigger filter kernels might help extract features that are relevant for the classification. However, the bigger the filters are, the higher the resource consumption of the network. Since the audio input representations usually have quite a big size this is an essential factor for designing a CNN. Dilated convolutions can help capture such features and still keep resource consumption at an acceptable level. For example, a filter kernel of size 21x21 can be replaced by a dilated kernel of size 5x5 and a dilation factor of 5. This decreases the filter size by over 94% and still captures long-term features. However, while the training speed increases and memory consumption decreases, the classification performance might also decrease due to sparser filters. To tackle this issue, stacked dilations can be used to increase the resolution of the dilated filter kernels and still keep the resource consumption as low as possible. In [Lei et al., 2019], an architecture using six stacked convolutional layers is used for image classification. They apply pooling after each layer and use different dilation rates between 1 and 5. With this, the accuracy increases, and the training time decreases simultaneously compared to the traditional CNN. An important factor for the performance of stacked dilations is the choice of the dilation factors to avoid gridding effects which can cause important loss of information for the classification. To facilitate finding the best dilation rates in [Schuster et al., 2019], parallel stacked dilations are used for dense matching of pixel positions in images to find pixel-wise correspondences across different images. For this purpose, one stacked dilated convolution block consists of four parallel convolutions with different dilation rates from

1 to 4, each with the same input. Their output is concatenated and serves as input for the next block. The architecture built from these building blocks outperformed recent heuristic image descriptors in terms of accuracy and robustness [Schuster et al., 2019].

To evaluate the effects of dilated CNNs, the Musicnn model described in Section 5.4 is used as a basis and is adapted accordingly. It is chosen because it is expected to bring the best results of all investigated models and also because of its specific architecture. Only the frontend of Musicnn is adapted for the dilated CNN, while the rest of the model architecture stays the same. The frontend is used for the extraction of temporal and timbral features. It contains big filter kernels (e.g., 128x1 or 7x38), leading to rather big memory consumption and a longer training time. The dilations can be used to capture those long temporal and timbral features while reducing the resource consumption and training time. Figure 5.9 displays the basic architectural layout of the new frontend for the dilated CNN. The first three convolutions Conv1-3 are used to extract temporal features, Conv4-7 for the timbral features. Then, the all outputs are concatenated and serve as input for the midend. All convolutions at the first level have 26 filter kernels and 51 on the second one. The same dilation rates are used for the respective stacked layers. As described for the stacked dilated convolution block in [Schuster et al., 2019], the outputs are concatenated after the first parallel convolutions before the second convolution block is applied.

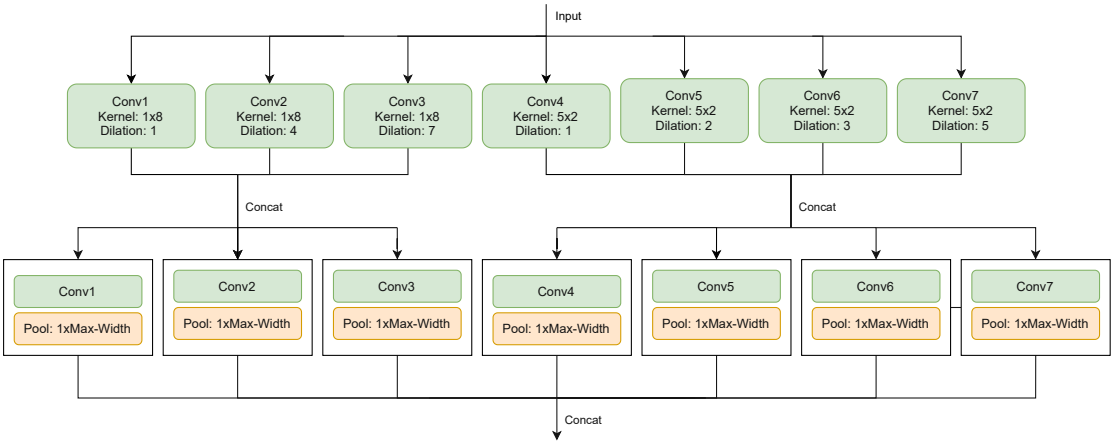


Figure 5.9: Dilated CNN architecture for 2D-input

For the one-dimensional input, the Musicnn frontend was adapted differently. It consists of seven sequential stacked dilated convolutions to reduce the input size sufficiently. Each of those blocks consists of four parallel convolutions stacked on two levels displayed in Figure 5.10.

This dilated CNN and the other described models will be used throughout all experiments described in the following chapter.

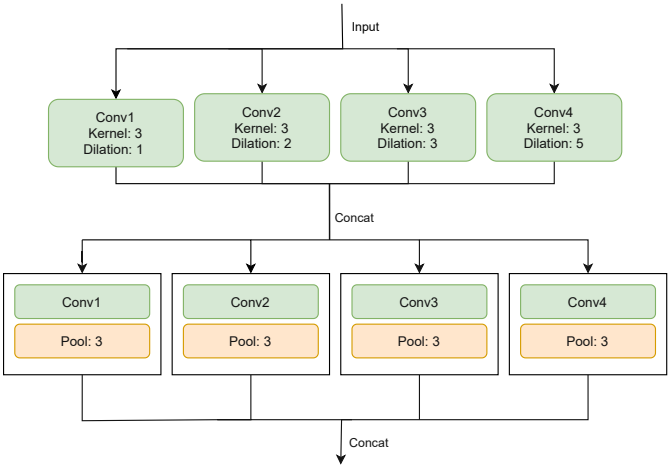


Figure 5.10: Dilated CNN block for 1D-input

CHAPTER 6

Experiments

This chapter describes the concrete methodology and procedure to answer the research questions presented in Section 1.2. Afterward, the experiments are defined in detail using the models presented in the previous Chapter 5.

6.1 Methodology

To answer the first two research questions (cf. Section 1.2) a comparative analysis is conducted. Five different CNNs are trained on the five audio input representations, and their results are evaluated over all tags and for each tag-category separately. For the last question regarding the usefulness of dilated CNNs for music autotagging, an existing model (Musicnn, cf. Section 5.4) is adapted accordingly to exploit the advantages of the dilated convolutions. The experiments are conducted on two datasets to provide a better generality of the results, described in Chapter 4.

The choice of the five input representations described in Chapter 3 is based on recent research on different topics of MIR. The main selection criteria are that the respective preprocessing method plays a remarkable role in recent studies and that the different methods chosen vary in their generation and size. For the CNNs, it is essential that they use different approaches and different input lengths (for further explanations, see Chapter 5).

The MTAT dataset, described in Section 4.1 in more detail is mainly used during the training pipeline development, tuning of the hyperparameters of each input-model combination and for the experiments with the dilated CNN. Audio snippets of either three seconds or the full 29.1 seconds are used for both datasets, depending on the model. After the training pipelines basic setup, the hyperparameters for each input-model combination are tuned on the MTAT dataset. The tuning is critical since the different input representations require a different configuration and size of the model.

Regularization with dropout- and batch-normalization layers is used to adapt the models for the respective preprocessing methods and prevent them from over- and underfitting. The tuned parameter values are then used as a starting point for the second dataset MTGJ and are only adapted in case of poor classification results. For the evaluation, ROC-AUC and PR-AUC are used to compare the classification performances, which seem to be the most commonly used metrics in research on music autotagging [Pons et al., 2018, Won et al., 2020]. They are described in more detail in Section 6.2.3. Afterward, all models are trained with those hyperparameters values on all input representations. Each configuration is trained and tested seven times on MTAT and five times on MTGJ using different initialization seeds. During the evaluation, the results of the respective input representations are compared to each other for all tags and for each tag category separately to answer the first two research questions. Two-way ANOVA analysis compares the impact of the preprocessing methods and the models on the results. Furthermore, the Tukey-HSD range test compares the performances of the different preprocessing methods. For the evaluation of the dilated CNN, t-tests are used to compare the performance to the original Musicnn model to investigate the third research question.

6.2 Experiments setup

The experiments are conducted on two computers, a private one with an Nvidia Geforce RTX 2060 SUPER and the second one from the TU Wien DL cluster with an Nvidia Geforce RTX 2080 Ti.

The DL pipeline for conducting the experiments is written in Python with Tensorflow. The most important preprocessing library used for the input representations is Librosa [McFee et al., 2015] which provides the functionality needed for generating the preprocessing methods. Weights & Biases [Weights-and Biases, 2021] is used to track and manage all experiments, by logging all configurations and results. Moreover, it provides a tool called Sweeps which is used for hyperparameter tuning. In the following subsections, the implementation details for the respective experimentation steps are described: the preprocessing, hyperparameter tuning, training and testing of the model.

6.2.1 Preprocessing

Before the training of the models starts, the input representations for all audio files are generated. The exact calculation steps for the investigated preprocessing methods are described in detail in Chapter 3. For this, the library Librosa is used, which offers a way to generate and manipulate all common audio input representations [McFee et al., 2015]. The preprocessing is done in advance to prevent a possible bottleneck during the data loading. All input representations are calculated and then saved to files containing arrays that can be efficiently loaded during training.

6.2.2 Parameter tuning

Since one central part of the experiments is to compare the performances of different input representations on the same models, the problem of the different sizes of the inputs arises. For example, the STFT representation is approximately 50 times larger than the MFCCs. Without further adaptions, this would lead to either overfitting the one or underfitting the other model trained with those preprocessing methods. One way would be to adapt the models for each input representation and increase or decrease the size of their parameters. However, to guarantee better comparability, additional dropout- and batch-normalization layers are used for regularization. For this, the configurations with smaller input representations are usually regularized stronger than the bigger ones. Before the training of each model-input combination, the regularization rates and the learning rate are determined. For the dropout rate, values between 0 and 0.5 are possible, and for the batch normalization it was only possible to choose if to apply it for all or no layers. Another regularization method used is L2-regularization but it showed minimal effect for the experiments, so it is not discussed further here. For this hyperparameter tuning, the tool "Sweeps" is used, part of the experiment tracker tool "Weights & Biases" used throughout this thesis. Since there are 25 different input-model combinations to tune, a temporal limit of about one day is set for each combination not to exceed the time frame of this study. For this purpose, a random parameter search is used. Figure 6.1 visualizes such an example "Sweep" for the hyperparameter tuning of SENet with Mel spectrograms.

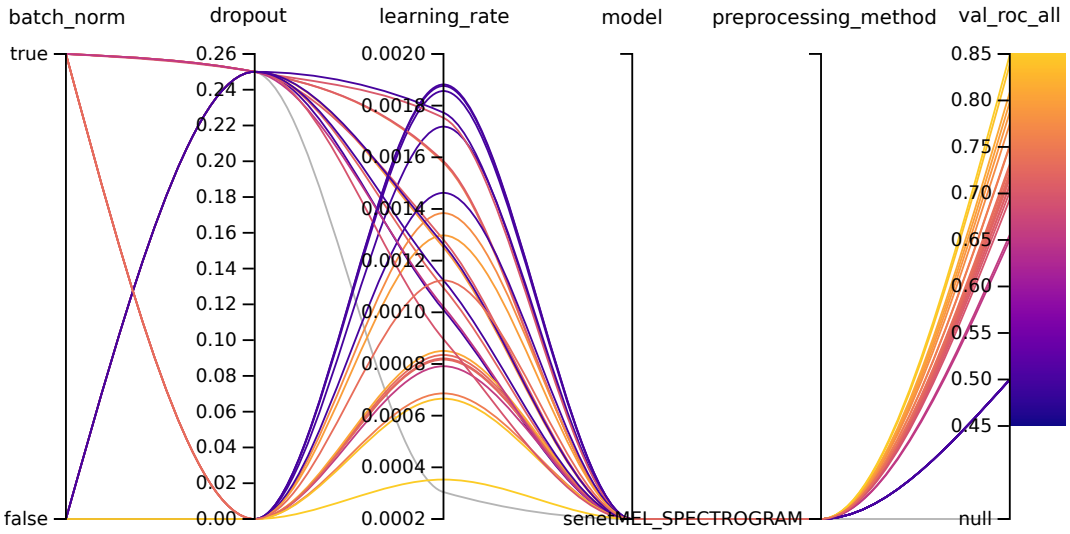


Figure 6.1: Example Sweep - SENet with Mel spectrograms

6.2.3 Training & Testing

In this section, important aspects for the training and testing are described: the optimizer, batch-loading, early-stopping and evaluation metrics. The optimizer and batch-loading are considered separately for each dataset.

Optimizer

One of the main difficulties of this study is the high amount of different combinations of models, input representations, and datasets to train and test.

The Adam optimizer is very efficient and converges fast to a good solution. However, the Stochastic Gradient Descent (SGD) generalizes better after more training epochs. In [Won et al., 2019] they propose a hybrid approach to get good results with Adam quickly and then apply SGD for fine-tuning and a good generalization. This technique is used for the MTAT dataset where the first eight epochs Adam is used, and then it switches to SGD with one fifth of the original learning rate. Since the MTGJ dataset has double the size and a much longer training time, it only uses Adam optimizer until convergence. The worse generalization is taken into account because this dataset only serves to validate the findings of the MTAT dataset, and there is no need to compare the results with recent studies. Since this strategy is used for all configurations of MTGJ, they should be good enough to compare them with each other. For both Adam and SGD, a learning rate scheduler reduces the learning rate step-wise after all 2 to 4 epochs by 50% to 75%. The exact step-sizes and values have been found heuristically by experimenting with a few configurations.

Batch-loading

All configurations use the maximum batch-size possible on the respective GPU to reduce the execution time as much as possible. Since the highest value is 64 for MFCCs on different models, there is little to no effect on the classification performance when reducing the batches tested heuristically. Therefore, the advantage of the execution time weights out possible improvements of classification performance with the additional tuning of this parameter.

The separate chunks are taken randomly from the full audio sequences for the models that use chunk-level inputs with audio snippets of three seconds. For the MTAT dataset, ten chunks per audio file are used for each epoch, and they are chosen randomly but uniformly across the whole file length. There are ten times more training samples in this case, which significantly increases the training time (although bigger batch sizes can be used due to the smaller input sizes). The problem with this is that it is harder to detect plateaus and stop the training early because the model is validated only after every epoch. For MTAT, this is still acceptable, but due to the bigger size of MTGJ, it is necessary to detect this stagnation as soon as possible to limit the training time. Therefore, for this dataset, only one random chunk per audio file is used for each epoch, resulting in approximately three to four times higher amount of total epochs but enabling

recognizing plateaus earlier. For testing chunk-level models, a majority voting of all ten randomly but approximately uniformly taken chunks is used to determine the respective tags on both datasets.

Early-stopping

To limit the training time as much as possible, early stopping is used. It is implemented so that training stops if the sum of ROC-AUC and PR-AUC have not reached a new peak for 3 or 6 epochs (for MTAT and MTGJ, respectively).

Evaluation metrics

In order to provide good comparability to other studies, ROC-AUC and PR-AUC are used as evaluation metrics [Pons et al., 2018, Won et al., 2020]. The optimizers are configured to maximize the sum of these two values. The metrics are calculated over all tags and for each category separately to evaluate the models accordingly.

AUC stands for Area Under the Curve, so the ROC-AUC is the area under the ROC curve, a commonly used technique to visualize the performance of classifiers. This graph is generated by plotting the true positive rate on the y-axis against the false positive rate using different thresholds for the classification results. Both rates range between 0 and 1, so the ROC-AUC is 1 at maximum as well. A value of around 0.5 is achieved by random classification, and values below should usually not occur (otherwise, the results could just be inverted to reach values higher than 0.5). The true positive rate is defined by:

$$TPR = \frac{TP}{TP + FN} \tag{6.1}$$

and the false positive rate by:

$$FPR = \frac{FP}{FP + TN}, \tag{6.2}$$

where TP stands for the number of True Positive predictions, FN for false negative, FP for false positive and TN for true negative [Fawcett, 2004].

The PR-curve plots the Precision on the y-axis against the Recall on the x-axis and is thresholded like the ROC curve. The precision is defined by:

$$P = \frac{TP}{TP + FP} \tag{6.3}$$

and the Recall by:

$$R = \frac{TP}{TP + FN} \tag{6.4}$$

The precision can be interpreted as how many tags that are assigned are appropriate for their songs and the Recall as how many tags are assigned correctly of the total number of tags that should be assigned [Powers, 2011]. Important to note is that unlike for the ROC-AUC, the PR-AUC value might be smaller than 0.5. It makes sense to use both metrics for evaluation because the PR-AUC score can be more informative on highly skewed datasets, i.e., when there are many songs with only very few tags (which is the case for both datasets) or with nearly all tags assigned [Davis and Goadrich, 2006].

6.3 Experiments description

This section introduces the experiments that are conducted to answer the respective research questions. For the first part, it is described how the models training, testing, and evaluation are prepared and conducted while each experiment refers to one dataset. The experiments in the second part are about developing a dilated CNN from the Musicnn model described in Section 5.4.

6.3.1 Part 1: Comparison of input representations

6.3.2 Experiment 1.1: Comparison on MTAT

This experiment aims to train and test all models described in Chapter 5 with all input representations described in Chapter 3 on the MTAT dataset to answer the first two research questions. The first step is to tune the hyperparameters for all configurations as described in Section 6.2.2. Afterward, those configurations are used to train and test those models as described in Section 6.2.3. For each configuration, seven runs with different initialization seeds are conducted. Two-way ANOVA analysis is used to assess the impact of the models and preprocessing methods on the results. The Tukey-HSD range then compares the preprocessing methods.

6.3.3 Experiment 1.2: Comparison on MTGJ

The results of experiment 1.1 are validated on a second dataset with the same models and configurations. For this, MTGJ is used, and apart from that, the experiment is equal to the previous one except for the tuning phase. The hyperparameters found in the previous experiment are used as a starting point to save time, as discussed in Section 6.2.2. Another important difference to experiment 1.1 that affects the evaluation of the results is that they will probably be worse than in recent studies because of the limited hyperparameter tuning, the lower audio quality, and the training-set size reduction as described in Section 4.2. For each configuration, five runs with different initialization seeds are conducted, and as for the previous experiment, Two-way ANOVA analysis and Tukey-HSD range test are used for evaluation.

6.3.4 Part 2: Dilated CNN

Experiment 2.1: Dilated CNN with ResNet

Experiments 2.1 and 2.2 aim to use dilated convolutions to extend an existing model designed for music autotagging so that either the classification performance increases, the training time decreases (while the classification performance stays the same), or both. For this, Musicnn, described in Section 5.4, is used because it provides the best results of all investigated models and reached state-of-the-art performances in 2019 [Pons and Serra, 2019]. It consists of two parts, a frontend to extract temporal and timbral features and a backend for the classification. [1]

In this first experiment, a dilated ResNet developed in [Yu et al., 2017] is used as backend instead of the original Musicnn backend. The idea is that it may better classify the extracted features from the frontend than the original backend, cf. [Yu et al., 2017]. As for the original ResNet, described in Section 5.5, there are different variations, and here the variant drn_d_105, available at [Yu, 2020], is used. In the frontend, all convolutions are extended with dilation to lighten up the filter kernels while providing as much information as possible to reduce training time. The model is developed using the MTAT dataset. For simplification, the dilated ResNet is only trained and tested with MFCCs, Mel spectrogram, and raw waveform. As for all models in the following experiments, the model is extended with dropout and batch-normalization layers. Those parameters are tuned according to the description in Section 6.2.2 for each input representation. Afterward, the model is trained, tested, and evaluated if it can improve training time and classification performance of the original model.

Experiment 2.2: Dilated CNN with parallel stacked dilations

As for the first experiment, the goal is to develop a dilated CNN by adapting Musicnn, described in Section 5.4. Now the frontend is replaced with stacked parallel dilated convolutions which reached good results on image matching in [Schuster et al., 2019]. MFCCs are used during experimentation to accelerate the training time due to their small size. The final architecture is described in Section 5.6. Again, the hyperparameters are tuned, and the model is trained and tested on both datasets. To compare the created dilated CNN with the original model, five and seven runs with different initialization seeds are conducted on MTAT and MTGJ, resp. T-tests are used to assess the performance difference for each preprocessing method for the evaluation.

[1] This is a simplification of the actual network architecture which contains a midend too, which is included in the backend here.

CHAPTER 7

Results and Discussion

After conducting the experiments, the results are presented and discussed in this chapter. The relevant evaluation metrics are described in Section 6.2.3: the ROC-AUC, the PR-AUC, and the training time per epoch. The first part in Section 7.2 presents the comparison of the input representations, and the second part in Section 7.1 deals with creating the dilated CNN.

7.1 Part 1: Comparison of input representations

7.1.1 Experiment 1.1: Comparison on MTAT

In the first experiment, the classification performances of all input representations on all models are compared on MTAT in the first subsection to make statements about their suitability for music autotagging on CNN models in general and for the individual tag categories. Afterward, the results are evaluated for each tag category separately.

Comparison over all tags

Figure 7.1 displays the ROC-AUC scores for all seven runs of each input representation on each model, visualized in different colors. Additionally, it shows the average classification performance per preprocessing method with the brown squares. The distribution of the PR-AUC scores looks very similar, as shown in Figure 7.2. From looking at the diagrams, one can see that the dilated CNN dominates the other models for the audio spectrograms, closely followed by Musicnn. The results for all other models suggest that it depends on the model and input representation combination. It is also hard to tell which input representation performs the best overall.

Two-way ANOVA is used to make further statements about the results on the factors model and input representation and which input representations outperform others. The

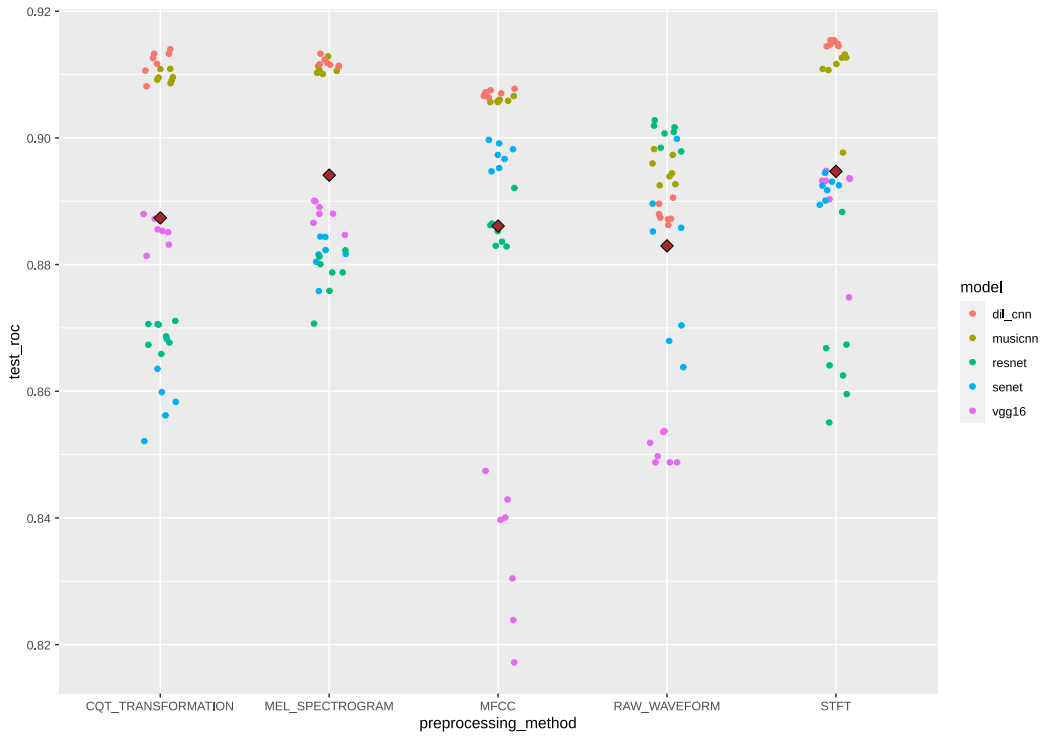


Figure 7.1: ROC-AUC scores for all preprocessing methods and all models on MTAT

statistical software R and the Scipy library in Python are used to calculate the analysis and check its assumptions. For this, only the ROC-AUC scores are used. The *aictab*() function in R is used to find the best ANOVA model. Two models are tested, one considering the interaction between the two independent variables model and input and one without that interaction. The output of the *aictab*() function suggests that the interaction model is preferable since it shows a lower "Akaike information criterion" score which is a quality test for models [Bevans, 2020]. Next, the assumptions for the ANOVA analysis are checked using this model. The independence of the observations assumption is met due to the experimentation design. Each run-configuration for the different model and input representation combinations is independent of the others. The Levene test for equality of variances is applied to the data to check for the homoscedasticity of the values. The p-value is $< 1e-05$, so there is a strong significance that this assumption is not met. Therefore, a data transformation is applied to the values, and the following transformation results in a p-value of 0.1536 for the Levene test: $\frac{exp(x)^{10}}{log10(x)}$ The disadvantage of this transformation is that it is not possible anymore to make clear statements about the exact differences between the results. The results of the various significance tests are the same with and without the transformation. This might be because the sample data is balanced in terms of the number of samples per group. Therefore, the heterogeneity of variances may not be that important for the validity of the results.

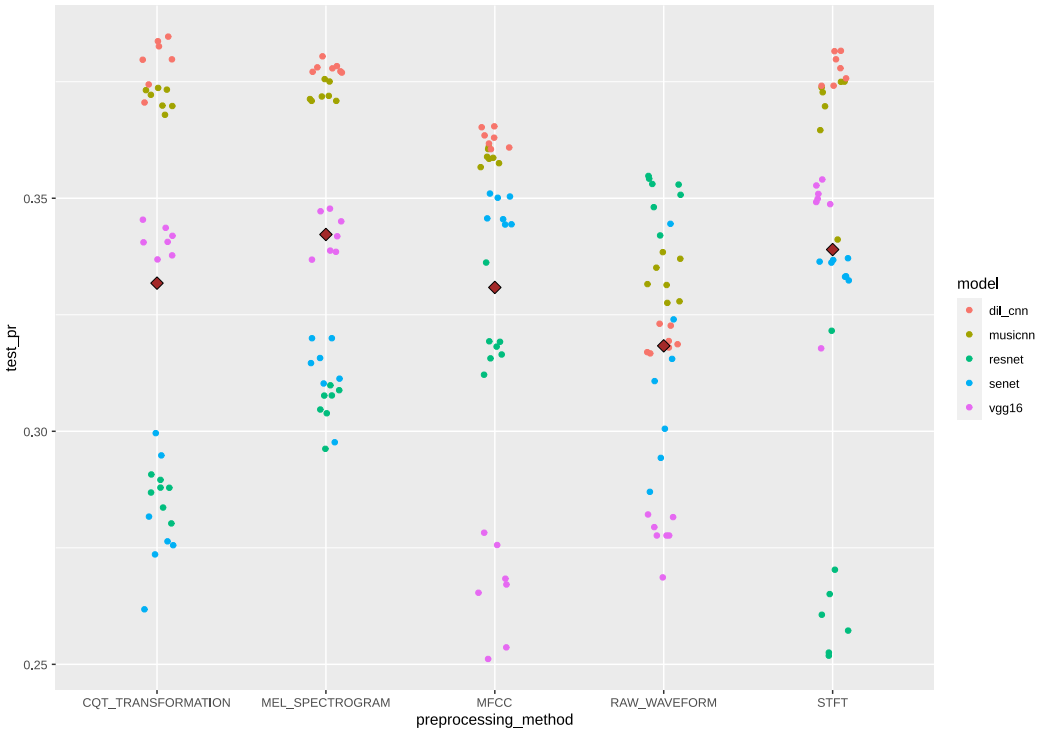


Figure 7.2: PR-AUC scores for all preprocessing methods and all models on MTAT

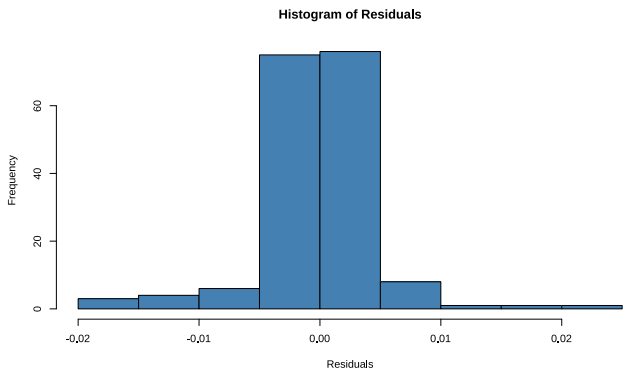


Figure 7.3: Distribution of residuals for Two-way ANOVA on MTAT

Figure 7.3 shows that the residuals are approximately normally distributed, so the last assumption is fulfilled too [Kozak and Piepho, 2018].

Source	Sum Sq	DF	MS	F	p-unc	np2
preprocessing_method	2.83E+10	4	7.08E+09	57.64	2.19E-29	0.61
model	3.16E+11	4	7.89E+10	642.13	3.08E-93	0.94
preprocessing_method * model	1.51E+11	16	9.45E+09	76.86	9.08E-64	0.89
Residual	1.84E+10	150	1.23E+08			

Table 7.1: Two-way ANOVA results on MTAT with transformation

Table 7.1 displays the results for the Two-way ANOVA analysis. The p-values for the preprocessing methods and the models both show a strong statistical significance for those variables and the interaction between them. Therefore, it can be stated that the model or the input representation influences the classification results and, more crucial, the combination of them. Due to the data transformation, the sum of squares (Sum Sq) and the mean squares (MS) are nearly impossible to interpret and very high. The most interesting values, apart from the p-value, are the partial eta-square effect sizes (np2) for each source. The np2-values indicate a strong effect of each source on the classification results. However, the model has the most substantial effect, followed by the interaction of the model and preprocessing method.

	input diff (transf.)	p-value (transf.)	input diff (untransf.)	p-value (untransf.)
MEL-CQT	13926.88	4.92E-06	0.0068	5.11E-07
MFCC-CQT	222.24	1.0000	-0.0013	0.8135
RAW-CQT	-17197.21	1.18E-08	-0.0044	0.0024
STFT-CQT	19305.97	1.69E-10	0.0074	4.22E-08
MFCC-MEL	-13704.64	7.21E-06	-0.0080	2.05E-09
RAW-MEL	-31124.09	0	-0.0112	2.85E-14
STFT-MEL	5379.09	0.2569	0.0006	0.9869
RAW-MFCC	-17419.45	7.63E-09	-0.0031	0.0651
STFT-MFCC	19083.73	2.68E-10	0.0086	1.35E-10
STFT-RAW	36503.18	0	0.0118	3.12E-14

Table 7.2: Tukey-HSD ROC-AUC results for preprocessing methods on MTAT

Tukey-HSD range test is a statistical procedure to compare the means of each group with each other. It has the same assumptions on the data as the Two-way ANOVA analysis. Table 7.2 shows the results without the data transformation to get information about the absolute difference between the groups and also with the data transformation to validate or reject those results. The worst overall preprocessing method is the raw waveform which performs worse than the other methods. It is 0.44% worse than the CQT transformation with a p-value of 0.0024 and significantly worse for the transformed input with a p-value of 1.18e-08. Compared to the MFCCs, the raw waveform performs 0.0031% worse, but with a p-value of 0.0651, so this difference is insignificant for the untransformed input. However, with the transformed input, there is a strong significance with a p-value of 7.63e-09 that the MFCCs outperform the raw waveform to an equal extent as the CQT transformation does. Comparing the MFCCs with the CQT transformation, neither of

both outperforms the other for the transformed input and the untransformed with very high p-values of 1.00 and 0.8135. As displayed before in Figure 7.1, the former performs better on all models except for the VGG-16 model than the latter, where it shows a very poor performance compared to all other input representations. Mel spectrograms in average perform 0.8% better than MFCCs with a p-value of 2.05e-09 and 0.68% better than the CQT transformation with a p-value of 5.11e-07. These strong significance levels are also validated with the transformed inputs. There is no significant difference between the Mel spectrogram and the STFT with high p-values of 0.9869 and 0.2569 for untransformed and transformed inputs, respectively. Table 7.3 shows similar results for the PR-AUC scores, so that it will be not further discussed here.

	input diff (transf.)	P-value (transf.)	input diff (untransf.)	P-value (untransf.)
MEL-CQT	4.8898	0.0023	0.0104	1.22E-05
MFCC-CQT	-3.4019	0.0743	-0.0009	0.9917
RAW-CQT	-14.6395	3.55E-15	-0.0135	1.04E-08
STFT-CQT	4.6687	0.0042	0.0072	0.0057
MFCC-MEL	-8.2917	2.39E-08	-0.0114	1.58E-06
RAW-MEL	-19.5293	0	-0.0239	0
STFT-MEL	-0.2211	0.9998	-0.0032	0.5217
RAW-MFCC	-11.2376	1.22E-13	-0.0125	1.01E-07
STFT-MFCC	8.0706	5.66E-08	0.0081	0.0012
STFT-RAW	19.3082	0	0.0207	3.03E-14

Table 7.3: Tukey-HSD PR-AUC results for preprocessing methods on MTAT

	input diff (transf.)	P-value (transf.)	input diff (untransf.)	P-value (untransf.)
musicnn-dil_cnn	-4834.96	0.3634	-0.0005	0.9938
resnet-dil_cnn	-84812.56	0	-0.0269	0
senet-dil_cnn	-78691.83	0	-0.0243	0
vgg16-dil_cnn	-100118.83	0	-0.0370	0
resnet-musicnn	-79977.60	0	-0.0264	0
senet-musicnn	-73856.87	0	-0.0238	0
vgg16-musicnn	-95283.87	0	-0.0365	0
senet-resnet	6120.73	0.1475	0.0026	1.93E-01
vgg16-resnet	-15306.27	4.24E-07	-0.0101	1.32E-13
vgg16-senet	-21427.00	1.95E-12	-0.0127	1.54E-14

Table 7.4: Tukey-HSD ROC-AUC results for models on MTAT

Table 7.4 shows the results for the Tukey-HSD range test on the different models. VGG-16 performs significantly worse than all other models. On average, it performs 1.01% worse than ResNet and 1.27% worse than SENet with p-values of $< 1e-06$ for both transformed and untransformed scores. There is no significant difference between the latter two because the p-value is 14.75% for the transformed input. Musicnn and the dilated CNN significantly outperform all other models with $> 2.4\%$ difference. As already seen in the previous Section 7.2.1, there is no significant difference between those two models with p-values of 99.38% and 36.34% for untransformed and transformed scores, respectively.

Comparison for individual tag categories

To assess the performance of the input representations for the different tag categories, the Two-way ANOVA analysis is run separately on the respective ROC-AUC scores as well as the Tukey-HSD range test. For each category, the assumptions for the variance analysis have to be checked first, and then the results are presented. The model - preprocessing method interaction model is used for the ANOVA test, as described in the previous section.

Genre Figure 7.4 shows the distribution of the residuals for the ANOVA model, which is normally distributed, so this assumption is met. Since there Levene Test for homoscedasticity shows a significant heterogeneity of variances, the data is transformed using the following transformation: $\frac{1}{log10(x)}$. Analyzing the model diagnostic plots in Figure 7.5 shows that the model fits the assumption of homoscedasticity. The red lines representing the means of the residuals have to be approximately horizontally, and the normal Q-Q plot represents the regression to a perfect homoscedastic model. Therefore, a close slope of 1 is ideal, which is the case here [Bevans, 2020].

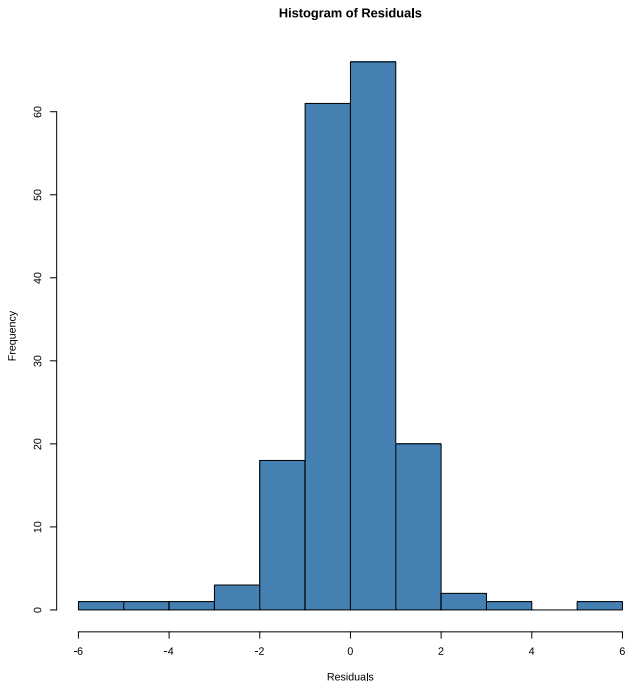


Figure 7.4: Distribution of residuals for Two-way ANOVA on MTAT using genre tags

Table 7.5 shows the results for the transformed ANOVA analysis. It shows very similar p-values and np2-values for all sources as in the comparison overall tags in the previous subsection. Therefore, all sources significantly impact the classification performance of genre tags. Figure 7.6 displays the respective ROC-AUC scores. The Tukey-HSD results

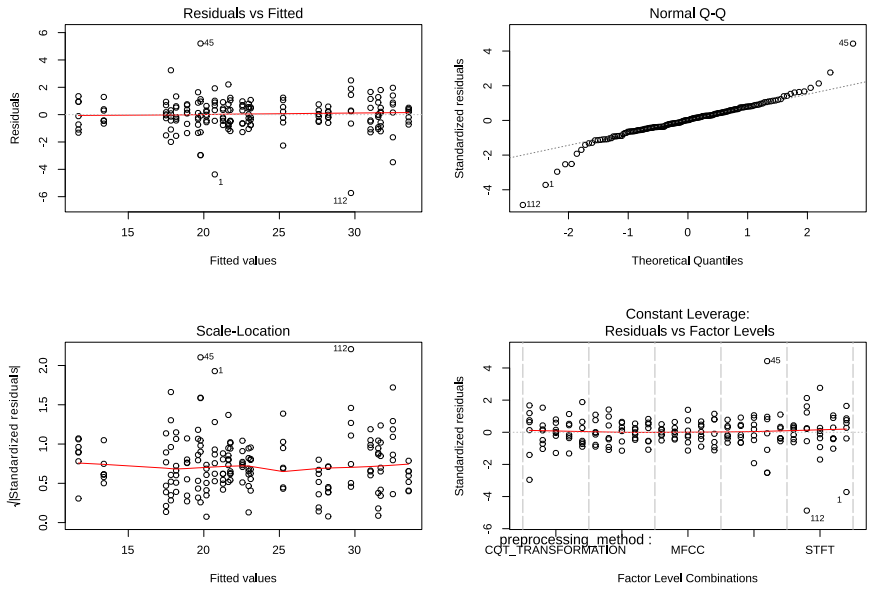


Figure 7.5: Residual-plots for ROC-AUC on MTAT using genre tags

Source	Sum Sq	DF	MS	F	p-unc	np2
preprocessing_method	513.3274	4	128.3319	78.9064	6.59E-36	0.6779
model	4010.0751	4	1002.5188	616.4105	5.55E-92	0.9427
preprocessing_method * model	1626.0242	16	101.6265	62.4863	6.63E-58	0.8695
Residual	243.9573	150	1.6264			

Table 7.5: Two-way ANOVA results on MTAT with transformation for genre tags

analysis for the preprocessing methods is presented in Table 7.6. The STFT performs equally well for genre tags as the Mel spectrogram and the CQT transformation. What strikes out is that especially the latter one performs better than overall tags, where it shows significantly worse results than the other two input representations. Again, the raw waveform performs worse than all other preprocessing methods, followed by the MFCCs.

Instrument Figure 7.7 shows the distribution of the residuals for the ANOVA model, which is approximately normally distributed, so this assumption is met. Since there Levene Test for homoscedasticity shows a significant heterogeneity of variances, the data is transformed using the following transformation: $\frac{exp(x)^5}{log10(x)}$. This transformation results in a p-value of 0.0936 for the Levene Tests. Therefore the assumption of homoscedasticity is fulfilled as well for the transformed model.

Table 7.7 shows the results for the transformed ANOVA analysis on the instrument tags. It shows similar p-values and np2-values for all sources as in the comparison overall tags in the previous subsection and as for the genre tags. Therefore, all sources

	input diff (transf.)	P-value
MEL-CQT	0.1630	0.3910
MFCC-CQT	-0.6889	4.99E-11
RAW-CQT	-1.1318	0.0000
STFT-CQT	0.1263	0.6440
MFCC-MEL	-0.8520	3.18E-14
RAW-MEL	-1.2948	0.0000
STFT-MEL	-0.0368	0.9945
RAW-MFCC	-0.4428	3.33E-05
STFT-MFCC	0.8152	4.64E-14
STFT-RAW	1.2581	0.0000

Table 7.6: Tukey-HSD ROC-AUC results for preprocessing methods on MTAT using genre tags

Source	Sum Sq	DF	MS	F	p-unc	np2
preprocessing_method	1805704.8659	4	451426.2165	56.2512	6.56E-29	0.6000
model	18345906.2275	4	4586476.5569	571.5100	1.15E-89	0.9384
preprocessing_method * model	11312538.5879	16	707033.6617	88.1018	1.02E-67	0.9038
Residual	1203778.4975	150	8025.1900			

Table 7.7: Two-way ANOVA results on MTAT with transformation for instrument tags

significantly impact the classification performance of instrument tags. Figure 7.8 displays the respective ROC-AUC scores. Table 7.8 presents the results for the Tukey HSD test on the preprocessing methods. As for the genre- and overall tags, the raw waveform performs worse than all other inputs. In contrast to the genre tags, the CQT transformation performs equally well as the MFCCs. Both input representations perform significantly worse than the STFT and the Mel spectrogram, which perform equally well. Therefore, the classification of the instrument tags for each preprocessing method is very similar to the classification overall tags.

	input diff (transf.)	P-value
MEL-CQT	49.4190	2.06E-11
MFCC-CQT	6.6873	0.8376
RAW-CQT	-31.5998	2.39E-05
STFT-CQT	47.0505	1.57E-10
MFCC-MEL	-42.7318	5.72E-09
RAW-MEL	-81.0189	0.0000
STFT-MEL	-2.3685	0.9961
RAW-MFCC	-38.2871	1.91E-07
STFT-MFCC	40.3633	3.80E-08
STFT-RAW	78.6503	0.0000

Table 7.8: Tukey-HSD ROC-AUC results for preprocessing methods on MTAT using instrument tags

Mood Figure 7.9 shows the distribution of the residuals for the ANOVA model, which is approximately normally distributed, so this assumption is met. Since there Levene

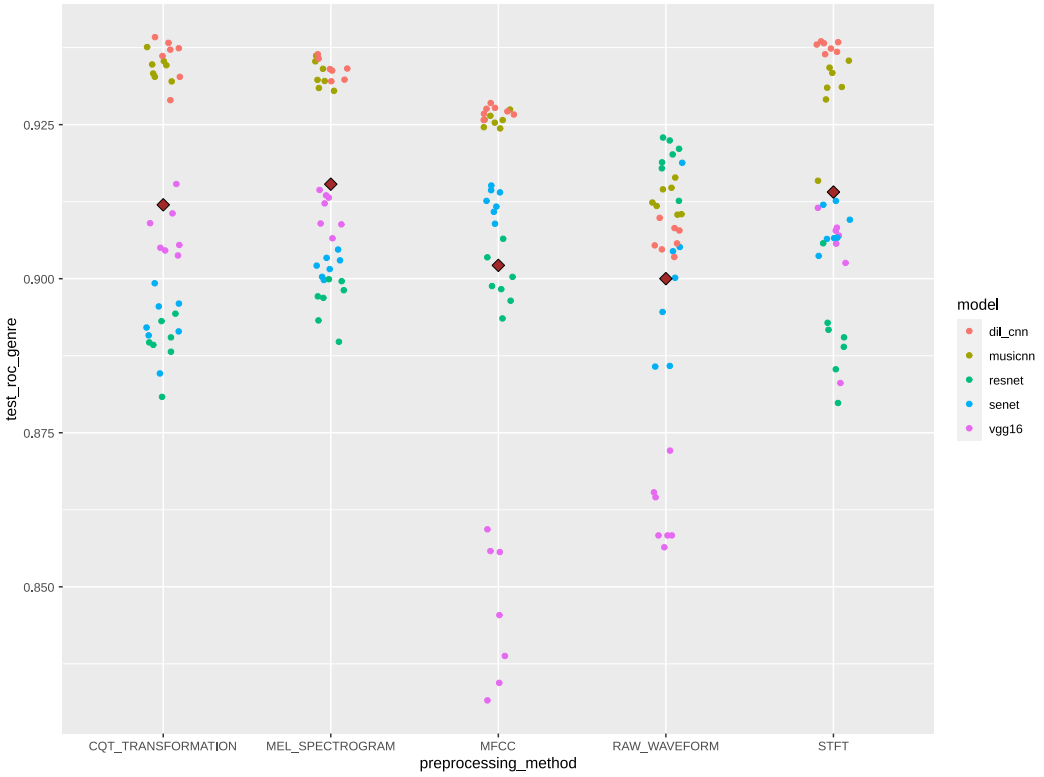


Figure 7.6: ROC-AUC scores for all preprocessing methods and all models on MTAT using genre tags

Test for homoscedasticity shows a significant heterogeneity of variances, the data is transformed using the following transformation: $\frac{1}{log10(x)}$. Analyzing the model diagnostic plots in Figure 7.10 shows that the model fits the assumption of homoscedasticity. The red lines representing the means of the residuals are all approximately horizontally, and the normal Q-Q plot roughly follows the slope of 1 [Bevans, 2020].

Source	Sum Sq	DF	MS	F	p-unc	np2
preprocessing_method	119.1147	4	29.7787	51.5857	2.91E-27	0.5791
model	418.1733	4	104.5433	181.1007	2.39E-56	0.8285
preprocessing_method * model	229.5219	16	14.3451	24.8501	2.78E-34	0.7261
Residual	86.5899	150	0.5773			

Table 7.9: Two-way ANOVA results on MTAT with transformation for mood tags

Table 7.9 shows the results for the transformed ANOVA-analysis on the mood tags. It shows similar p-values and np2-values for all sources as for all previous tag-categories. Therefore, all sources significantly impact the classification performance of mood tags. Figure 7.11 displays the respective ROC-AUC scores. Table 7.10 presents the results for the Tukey-HSD test on the preprocessing methods. The results for the mood tags

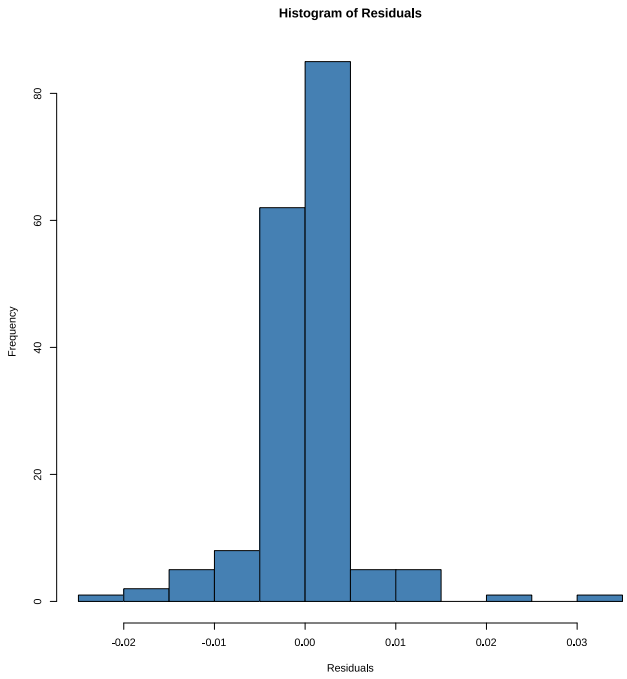


Figure 7.7: Distribution of residuals for Two-way ANOVA on MTAT using instrument tags

deviate strongly from the other tag categories. The CQT transformation performs significantly worse than all other input representations, followed by the raw waveform and the Mel spectrogram, which have equally good results. Surprisingly, MFCCs significantly outperform the latter two, and STFT significantly shows the best overall results. Since MFCCs are much smaller than the other input representations it is worth considering this preprocessing method for mood-classification tasks. However, if the goal is only to reach the maximum classification performance, then the best choice for all other tag categories would be to use STFT. The results presented in this section are validated or rejected on the MTGJ dataset in the following section.

7.1.2 Experiment 1.2: Comparison on MTGJ

The primary purpose of this experiment is to validate or reject the findings of experiment 1.1. Therefore, the observations and conclusions of the previous experiment are discussed further based on the results of the MTGJ dataset. This experiment is structured equally to the previous one, so the first section compares all tags and the second section with the individual tag categories.

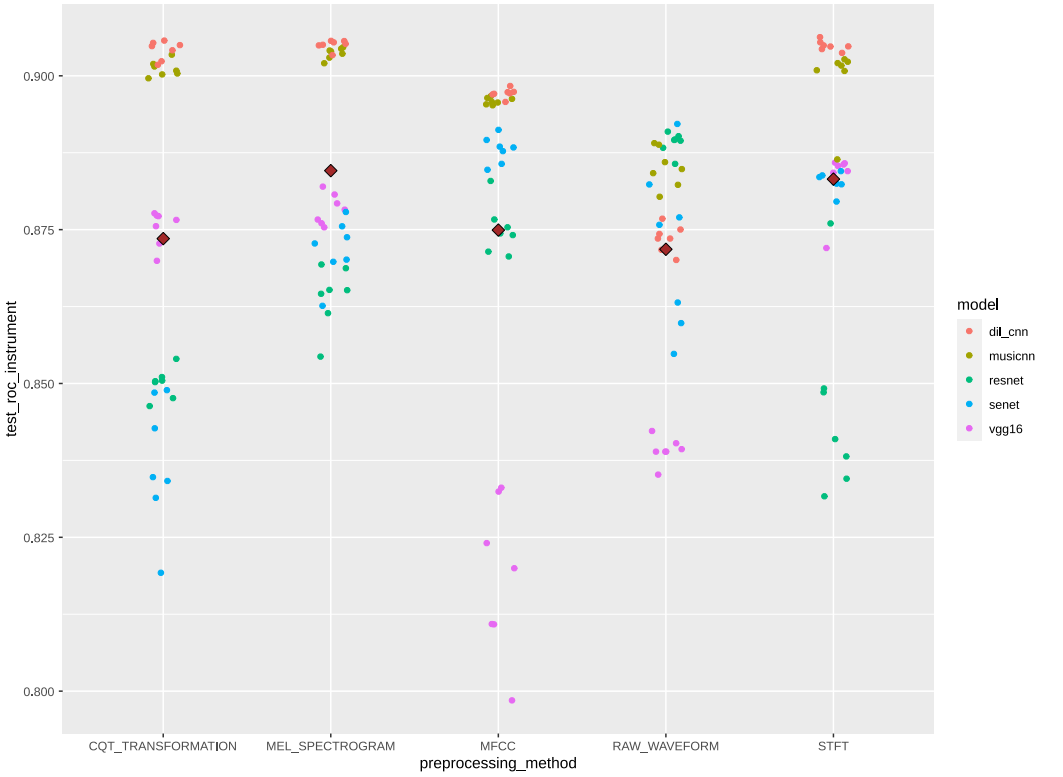


Figure 7.8: ROC-AUC scores for all preprocessing methods and all models on MTAT using instrument tags

Comparison over all tags

Figure 7.12 shows the ROC-AUC scores of all five runs on the MTGJ dataset, and Figure 7.13 shows all PR-AUC scores. As for MTAT, the raw waveform performs the worst and is again followed by MFCCs. All other three models approximately perform equally well with the best average results for STFT. Especially noticeable is the excellent performance of the CQT transformation compared to the other dataset. It even provides the best results for Musicnn and dilated CNN.

The Two-way ANOVA analysis is conducted similarly to the previous experiment. Again, the interaction model is preferable to the other without the interaction. Next, the required assumptions are checked to guarantee the validity of the results. Again, the independence of observations assumption is met due to the experimental design. The Levene Test to check the homoscedasticity shows a p-value of 0.1921 which means that this assumption holds without further data transformations. Figure 7.14 shows that the distribution of the residuals for the models is approximately normally distributed with only a slight skewness, so this last assumption is fulfilled as well.

Table 7.11 displays the results for the Two-way ANOVA analysis. As for the MTAT

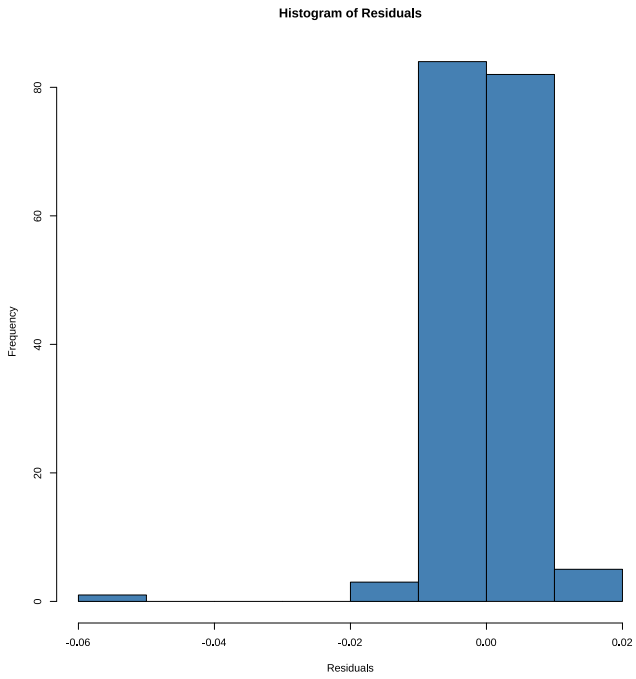


Figure 7.9: Distribution of residuals for Two-way ANOVA on MTAT using mood tags

	input diff (transf.)	P-value
MEL-CQT	0.5968	0.0109
MFCC-CQT	1.7685	2.89E-14
RAW-CQT	0.8743	3.50E-05
STFT-CQT	2.3011	0.0000
MFCC-MEL	1.1717	1.44E-08
RAW-MEL	0.2775	0.5461
STFT-MEL	1.7042	2.96E-14
RAW-MFCC	-0.8942	2.17E-05
STFT-MFCC	0.5325	0.0314
STFT-RAW	1.4268	7.11E-12

Table 7.10: Tukey-HSD ROC-AUC results for preprocessing methods on MTAT using mood tags

dataset, the p-values all show a strong statistical significance for all sources. Therefore, both models and preprocessing methods significantly influence the results as well as the combination of them. The trends of the effect sizes are also very similar to the previous experiment. The model has the most significant impact, followed by the interaction of both values.

Table 7.12 shows the results for the Tukey-HSD range test on the preprocessing methods. The results are very similar to MTAT. Raw waveform again performs worse than all other inputs, but now the significance for MFCCs outperforming the raw waveform is much

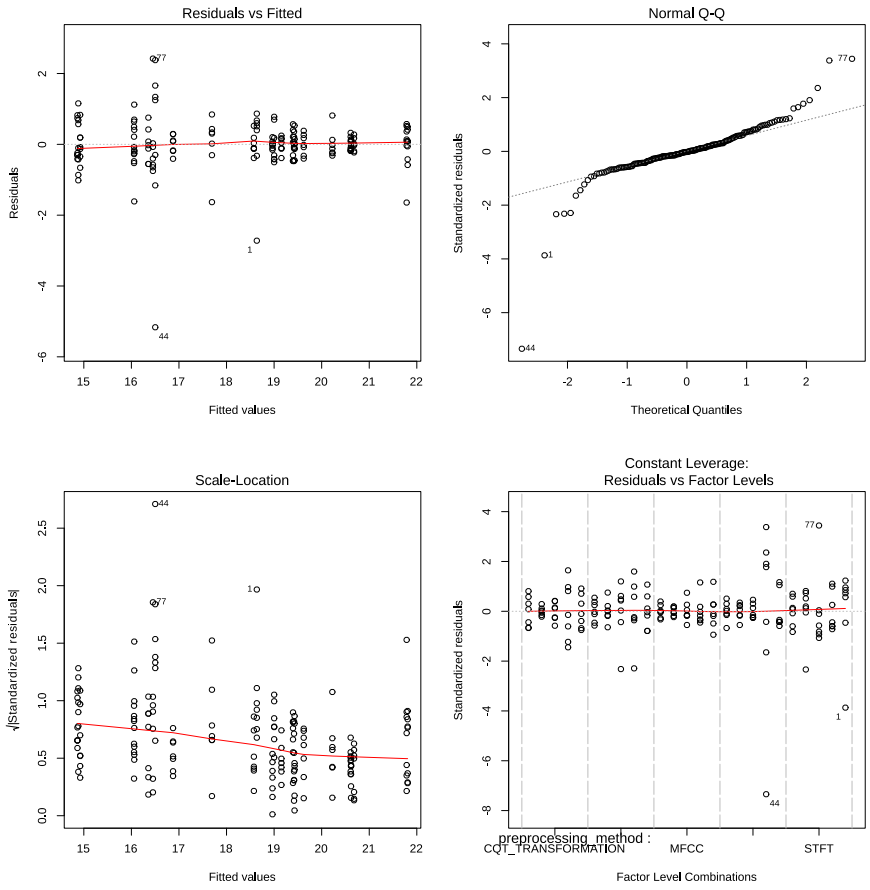


Figure 7.10: Residual-plots for ROC-AUC on MTAT using mood tags

Source	Sum Sq	DF	MS	F	p-unc	np2
preprocessing_method	0.0080	4	0.0020	63.6155	1.23E-26	0.7179
model	0.0260	4	0.0065	205.5154	2.63E-47	0.8915
preprocessing_method * model	0.0101	16	0.0006	20.0451	2.61E-24	0.7623
Residual	0.0032	100	3.16E-05			

Table 7.11: Two-way ANOVA results on MTGJ

stronger, with a p-value of 8.22e-05 for having a 0.75% higher ROC-score. On average, the Mel spectrogram significantly performs 0.85% better than the MFCCs, and the CQT transformation performs 1.1% better than the latter. In contrast to the previous dataset, the Mel spectrogram and the CQT transformation perform equally well. The STFT significantly performs 0.62% better than the Mel spectrogram. However, for the CQT transformation, there is no statistical significance the STFT outperforms it with a p-value of 0.1472. The PR-AUC score are very similar to the ROC-AUC scores, as shown in

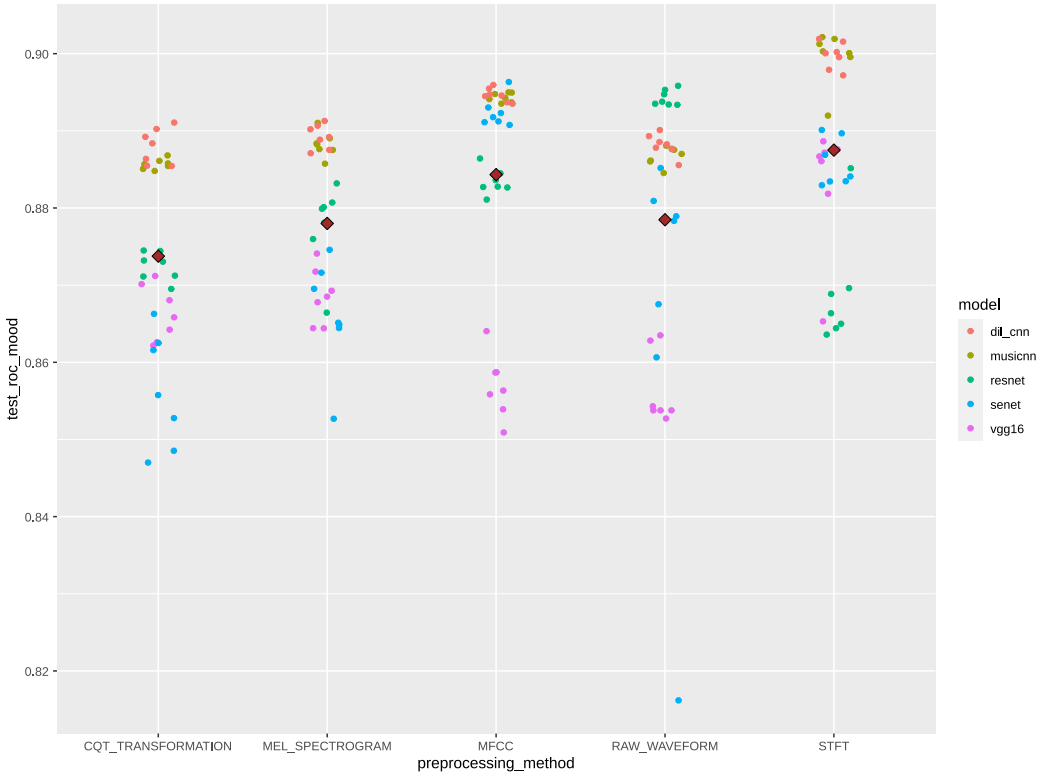


Figure 7.11: ROC-AUC scores for all preprocessing methods and all models on MTAT using mood tags

Table 7.13, so they will not be further discussed here.

The Tukey-HSD range test on the different models displayed in Table 7.14 shows the same ranking as for the MTAT dataset. Again, there is no significant difference between the Musicnn and the dilated CNN, which perform the best overall. Vgg-16 performs worse than all other models, followed by ResNet and SeNet.

Overall, it can be stated that the results of this experiment validate the results of the previous one for the most part. The STFT again performs the best of all input representations. However, the CQT transformation show similarly good results, which is the biggest difference to the MTAT dataset, where the Mel spectrogram performs significantly better than the former one.

Comparison for individual tag categories

To validate the results of the MTAT dataset in Section 7.1.1, the Two-way ANOVA analysis is run separately for each tag category on the respective ROC-AUC scores as well as the Tukey-HSD range test. For each category, the assumptions for the variance

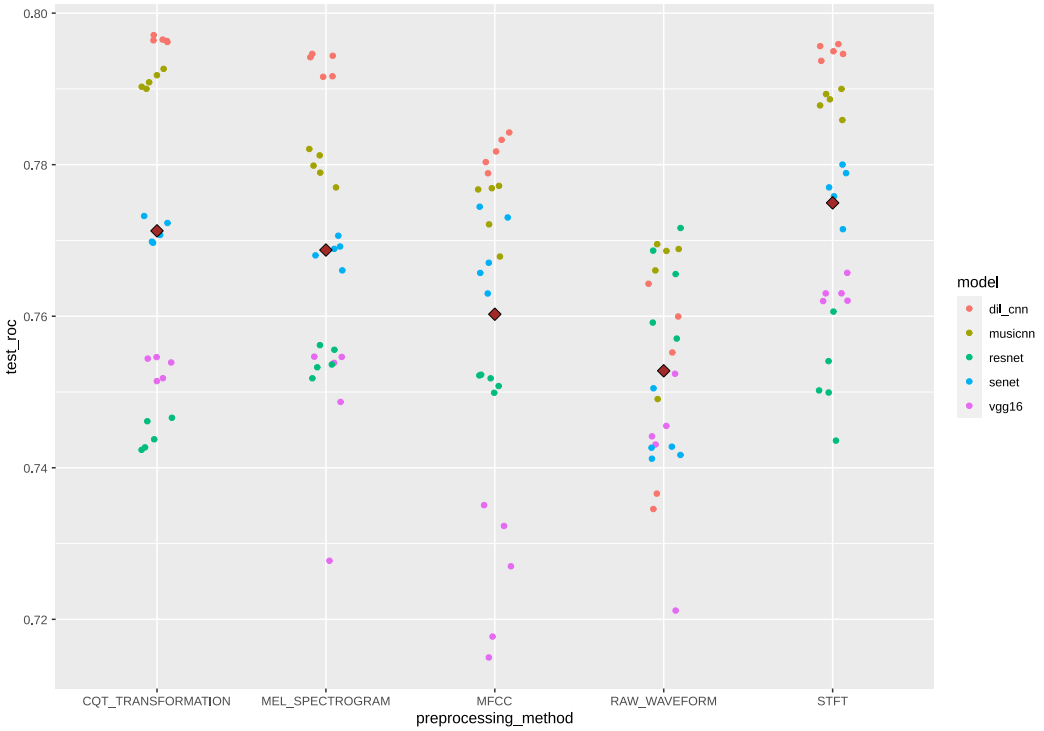


Figure 7.12: ROC-AUC scores for all preprocessing methods and all models on MTGJ

	input diff	P-value
MEL-CQT	-0.0025	0.5060
MFCC-CQT	-0.0110	4.53E-09
RAW-CQT	-0.0185	1.21E-10
STFT-CQT	0.0037	0.1472
MFCC-MEL	-0.0085	6.10E-06
RAW-MEL	-0.0159	1.21E-10
STFT-MEL	0.0062	0.0015
RAW-MFCC	-0.0075	8.22E-05
STFT-MFCC	0.0147	1.22E-10
STFT-RAW	0.0222	1.21E-10

Table 7.12: Tukey-HSD ROC-AUC results for preprocessing methods on MTGJ

analysis have to be checked first, and afterward, the results are presented. The model - preprocessing method interaction model is used for the ANOVA test.

Genre Figure 7.15 shows the distribution of the residuals for the ANOVA model, which is approximately normally distributed, so this assumption is met. The Levene test shows a p-value of 0.2614, so the assumption of homoscedasticity is fulfilled too.

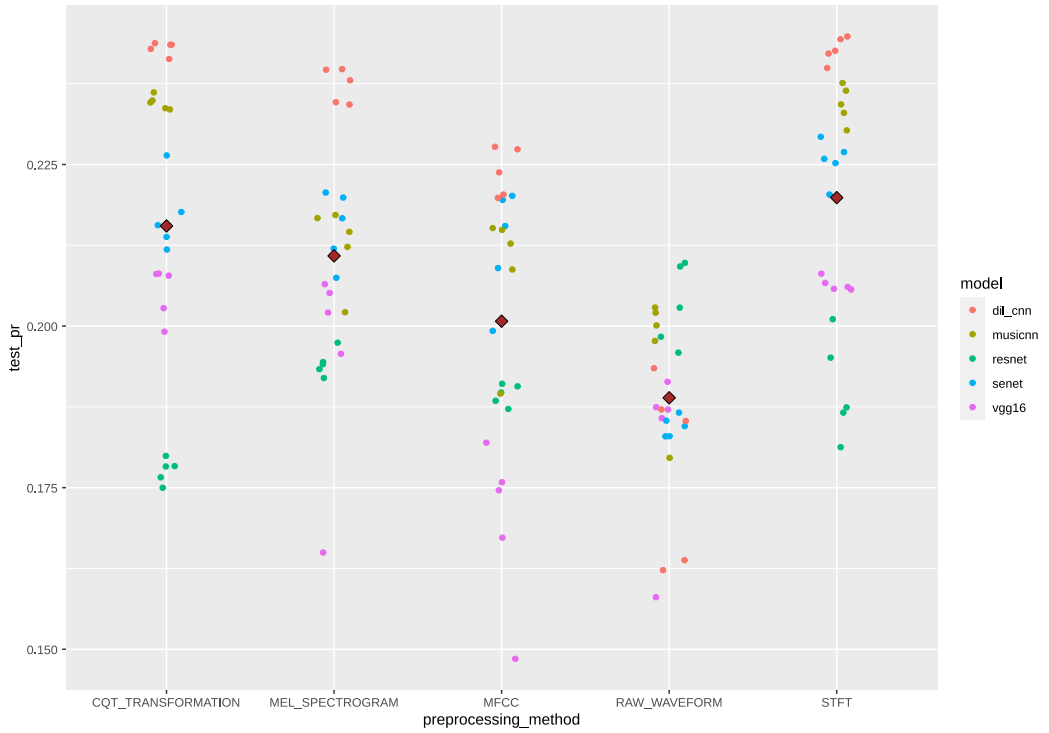


Figure 7.13: PR-AUC scores for all preprocessing methods and all models on MTGJ

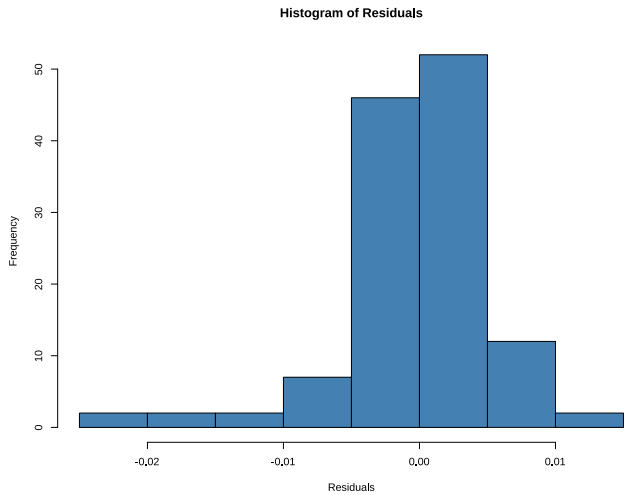


Figure 7.14: Distribution of residuals for Two-way ANOVA on MTGJ

	input diff	P-value
MEL-CQT	-0.0046	0.1975
MFCC-CQT	-0.0147	4.31E-09
RAW-CQT	-0.0266	1.21E-10
STFT-CQT	0.0044	0.2451
MFCC-MEL	-0.0101	6.53E-05
RAW-MEL	-0.0220	1.21E-10
STFT-MEL	0.0090	0.0005
RAW-MFCC	-0.0119	2.11E-06
STFT-MFCC	0.0191	1.22E-10
STFT-RAW	0.0310	1.21E-10

Table 7.13: Tukey-HSD PR-AUC results for preprocessing methods on MTGJ

	model diff	P-value
musicnn-dil_cnn	-0.0037	0.1371
resnet-dil_cnn	-0.0301	1.21E-10
senet-dil_cnn	-0.0176	1.21E-10
vgg16-dil_cnn	-0.0371	1.21E-10
resnet-musicnn	-0.0264	1.21E-10
senet-musicnn	-0.0138	1.22E-10
vgg16-musicnn	-0.0334	1.21E-10
senet-resnet	0.0126	1.57E-10
vgg16-resnet	-0.0070	0.0003
vgg16-senet	-0.0196	1.21E-10

Table 7.14: Tukey-HSD ROC-AUC results for models on MTGJ

Source	Sum Sq	DF	MS	F	p-unc	np2
preprocessing_method	0.0099	4	0.0025	67.7536	1.27E-27	0.7305
model	0.0229	4	0.0057	156.3477	4.13E-42	0.8621
preprocessing_method * model	0.0113	16	0.0007	19.2833	1.06E-23	0.7552
Residual	0.0037	100	3.66E-05			

Table 7.15: Two-way ANOVA results on MTGJ for genre tags

Table 7.5 shows the results of the ANOVA analysis. The results are very similar to the test over all tags, and all sources have a significant impact, which is also the case for the MTAT dataset. The np2-value shows almost identical trends as the previous experiment, with the strongest effect size for the model. Figure 7.16 displays the ROC-AUC scores for genre tags. The Tukey-HSD range test in Table 7.16 shows the same ranking for the input representations as in the other dataset. The only difference is that the STFT now significantly outperforms the CQT transformation and the Mel spectrogram, which perform equally well. Raw waveforms significantly perform the worst, followed by MFCCs.

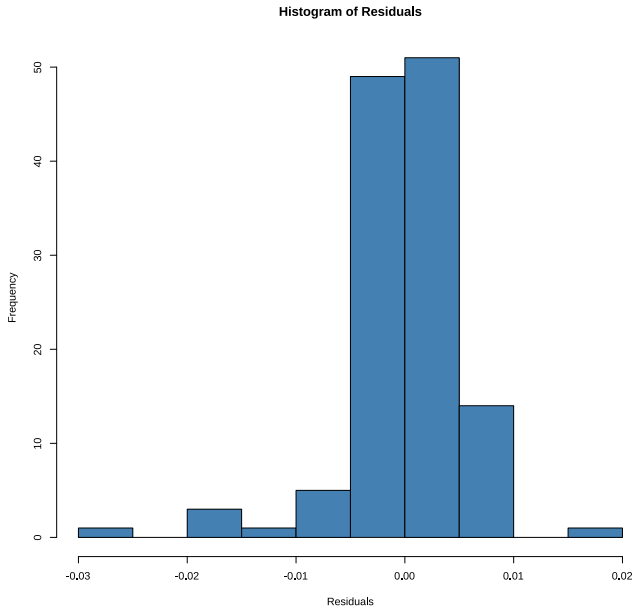


Figure 7.15: Distribution of residuals for Two-way ANOVA on MTGJ using genre tags

	input diff	P-value
MEL-CQT	-0.0023	0.6693
MFCC-CQT	-0.0114	1.49E-08
RAW-CQT	-0.0200	1.21E-10
STFT-CQT	0.0051	0.0294
MFCC-MEL	-0.0091	6.09E-06
RAW-MEL	-0.0177	1.21E-10
STFT-MEL	0.0074	0.0004
RAW-MFCC	-0.0086	2.31E-05
STFT-MFCC	0.0165	1.21E-10
STFT-RAW	0.0251	1.21E-10

Table 7.16: Tukey-HSD ROC-AUC results for preprocessing methods on MTGJ using genre tags

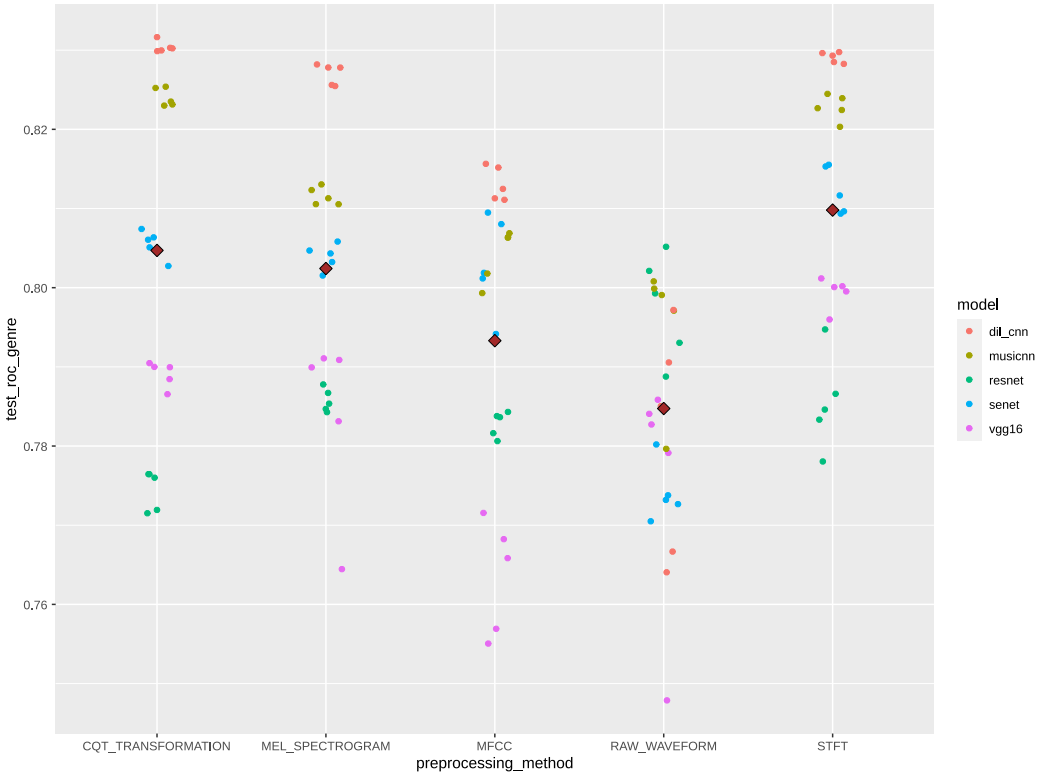


Figure 7.16: ROC-AUC scores for all preprocessing methods and all models on MTGJ using genre tags

Instrument Figure 7.17 shows the distribution of the residuals for the ANOVA model, which is approximately normally distributed, so this assumption is met. The Levene test results in a p-value of 0.1911, so the assumption of homoscedasticity is fulfilled too.

Source	Sum Sq	DF	MS	F	p-unc	np2
preprocessing_method	0.0058	4	0.0015	37.0539	5.60E-19	0.5971
model	0.0369	4	0.0092	235.7249	5.66E-50	0.9041
preprocessing_method * model	0.0102	16	0.0006	16.2895	4.03E-21	0.7227
Residual	0.0039	100	3.92E-05			

Table 7.17: Two-way ANOVA results on MTGJ for instrument tags

Table 7.7 shows the results of the ANOVA analysis. Again, all sources significantly impact the results. The np2 values show similar trends as all previous analyses, with models having the strongest impact, followed by the interaction of model and preprocessing method. The results of the Tukey-HSD range test are presented in Table 7.18, which shows a slightly different picture than in the MTAT dataset. Figure 7.18 displays the ROC-AUC scores for instrument tags. Again, the best preprocessing method is the STFT, with a significantly better average ROC-score over 0.61% than Mel spectrograms.

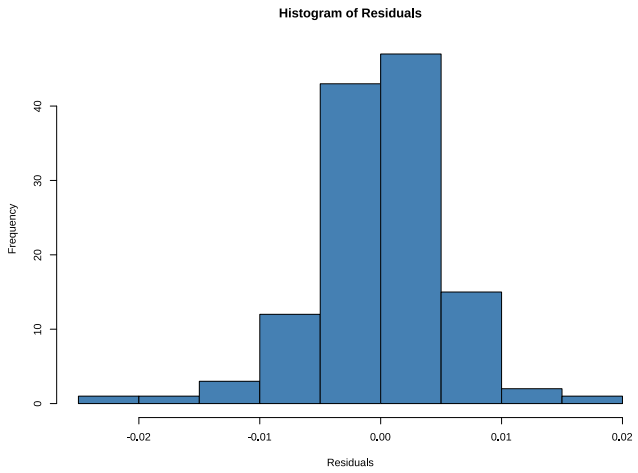


Figure 7.17: Distribution of residuals for Two-way ANOVA on MTGJ using instr-tags

However, the CQT transformation is not significantly worse than the former with a p-value of 0.6132 but does not outperform the Mel spectrogram. It has a higher average ROC-score of 0.36%, but the difference is insignificant, with a p-value of 0.2516. Raw waveform significantly performs 0.57% worse than MFCCs with a p-value of 0.0144, which shows much worse results than the other three spectrograms. All in all, it can be stated that STFT has the best results across both datasets and raw waveform the worst. However, there is too much difference in between the other preprocessing methods to make further statements.

	input diff	P-value
MEL-CQT	-0.0036	0.2516
MFCC-CQT	-0.0103	6.72E-07
RAW-CQT	-0.0160	1.22E-10
STFT-CQT	0.0025	0.6132
MFCC-MEL	-0.0067	0.0024
RAW-MEL	-0.0124	2.99E-09
STFT-MEL	0.0061	0.0067
RAW-MFCC	-0.0057	0.0144
STFT-MFCC	0.0128	9.90E-10
STFT-RAW	0.0186	1.21E-10

Table 7.18: Tukey-HSD ROC-AUC results for preprocessing methods on MTGJ using instrument tags

Mood Figure 7.19 shows the distribution of the residuals for the ANOVA model, which is approximately normally distributed, so this assumption is met. The Levene test results

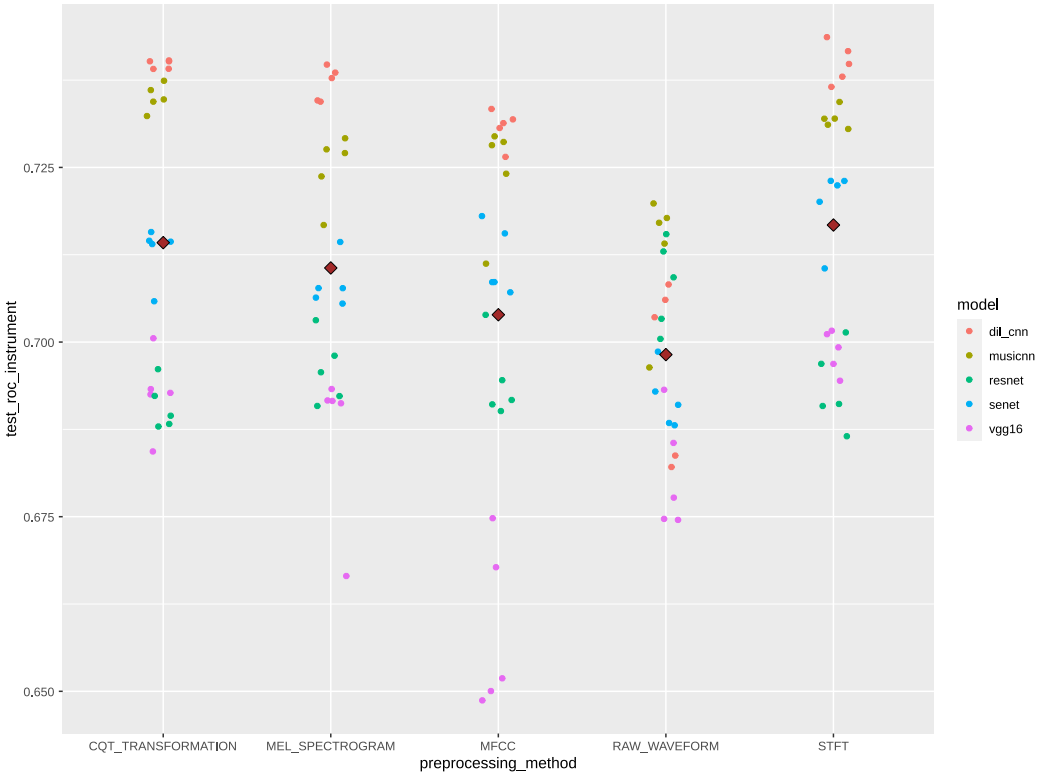


Figure 7.18: ROC-AUC scores for all preprocessing methods and all models on MTGJ using instrument tags

in a p-value of 0.0122, showing a slight significance for a heteroscedastic distribution. Since this is an edge case considering the p-value and all other assumptions are fulfilled, no additional transformation is applied to the data. Therefore, for the interpretation of the results, only very strong significance levels, e.g., p-values < 0.001, will be accepted to draw conclusions. As discussed in the previous experiment, this should not strongly influence the results due to the balanced sample data.

Source	Sum Sq	DF	MS	F	p-unc	np2
preprocessing_method	0.0050	4	0.0012	18.8436	1.42E-11	0.4298
model	0.0219	4	0.0055	83.2362	5.95E-31	0.7690
preprocessing_method * model	0.0078	16	0.0005	7.3901	4.62E-11	0.5418
Residual	0.0066	100	6.57E-05			

Table 7.19: Two-way ANOVA results on MTGJ for mood tags

Table 7.19 presents the results for the ANOVA-analysis for mood tags, resulting in a strong significance for the impact of all sources. Again, the model has the strongest effect on the ROC scores with an np2-value of 0.7690, followed by the interaction of the preprocessing method and the model with a value of 0.5418 and the preprocessing method

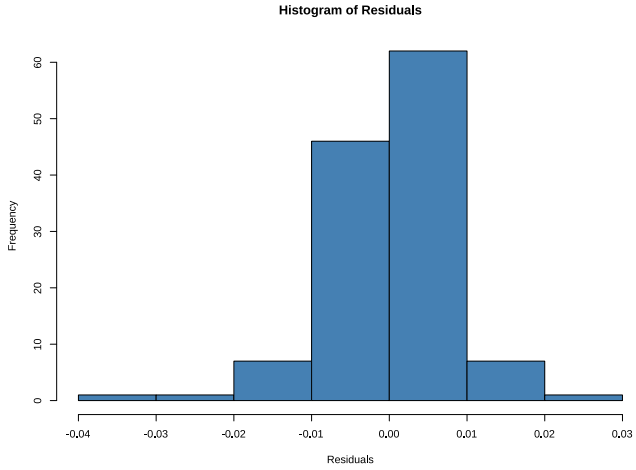


Figure 7.19: Distribution of residuals for Two-way ANOVA on MTGJ using mood tags

with 0.4298. The Tukey-HSD test results in Table 7.20 show very weak significance levels across almost all combinations. The STFT, CQT transformation, and Mel spectrogram perform equally well and significantly better than the raw waveform. Compared to MFCCs, the CQT transformation has the lowest p-value of 0.001, with an average 1.04% higher ROC-AUC score. The Mel spectrogram shows a strong significance compared to MFCCs with a p-value of 0.0007, but for the STFT, it is only 0.0024, which is already an edge case due to the slight heteroscedasticity. The MFCCs do not significantly outperform the raw waveform due to a p-value of 0.1190. The results differ strongly from the MTAT dataset, and it is hard to make general statements. The STFT shows good classification scores across both datasets. Apart from that, as displayed in Figure 7.20, the average ROC-AUC scores lie closely together, and the MFCCs might still be a good choice for tasks requiring a lower resource consumption since the difference to the other spectrograms is $< 1\%$.

7.2 Part 2: Dilated CNN

7.2.1 Experiment 2.1: Dilated CNN with ResNet

To evaluate the results of the models developed during this and the following experiment, the results of the Musicnn model, displayed in Table 7.21, serve as a reference.

The dilated CNN model is trained and tested with MFCCs, Mel spectrogram, and raw waveform as input. The results are shown in Table 7.22. Both the PR- and ROC-values are about 0.5%-2.0% worse for the two-dimensional inputs of the dilated CNN than for Musicnn. This could be due to the slightly lighter frontend with fewer trainable parameters caused by the dilations. Raw waveforms perform slightly better for the dilated CNN than for Musicnn, which might be due to the much deeper backend of the ResNet

	input diff	P-value
MQL-CQT	-0.0010	0.9925
MFCC-CQT	-0.0104	0.0001
RAW-CQT	-0.0160	3.61E-09
STFT-CQT	-0.0017	0.9415
MFCC-MEL	-0.0094	0.0007
RAW-MEL	-0.0150	2.73E-08
STFT-MEL	-0.0007	0.9976
RAW-MFCC	-0.0055	0.1190
STFT-MFCC	0.0087	0.0024
STFT-RAW	0.0142	1.23E-07

Table 7.20: Tukey-HSD ROC-AUC results for preprocessing methods on MTGJ using mood tags

input	PR-all	PR-gen	PR-instr	PR-mood	ROC-all	ROC-gen	ROC-instr	ROC-mood
CQT	0.3714	0.3563	0.4411	0.2453	0.9097	0.9343	0.9011	0.8857
MEL	0.3725	0.3544	0.4424	0.2506	0.9109	0.9330	0.9037	0.8882
MFCC	0.3588	0.3390	0.4245	0.2499	0.9059	0.9257	0.8959	0.8943
RAW	0.3327	0.3105	0.3905	0.2448	0.8950	0.9130	0.8851	0.8867
STFT	0.3674	0.3489	0.4295	0.2642	0.9099	0.9300	0.8995	0.8996

Table 7.21: Avg. results for Musicnn on MTAT

than of the original Musicnn. All in all, the dilated ResNet backend does not achieve the intended improvements. For the next experiment, the focus lies in enhancing the frontend with stacked dilations.

input	PR-all	PR-gen	PR-instr	PR-mood	ROC-all	ROC-gen	ROC-instr	ROC-mood
MEL	0.3626	0.3448	0.4285	0.2463	0.9061	0.9301	0.8970	0.8846
RAW	0.3429	0.3324	0.4006	0.2322	0.9018	0.9255	0.8898	0.8885
MFCC	0.3423	0.3199	0.4019	0.2466	0.9002	0.9176	0.8901	0.8922

Table 7.22: Results Dilated CNN with ResNet on MTAT

7.2.2 Experiment 2.2: Dilated CNN with parallel stacked dilations

The model developed in experiment 2.1 has already improved the training time but only for an average worse classification accuracy. In order to overcome this issue, stacked parallel dilated convolutions are used to keep the feature extraction in the frontend as good as possible while reducing the training time. The experiments are conducted on the MTAT dataset in de first subsection and then validated in the second one.

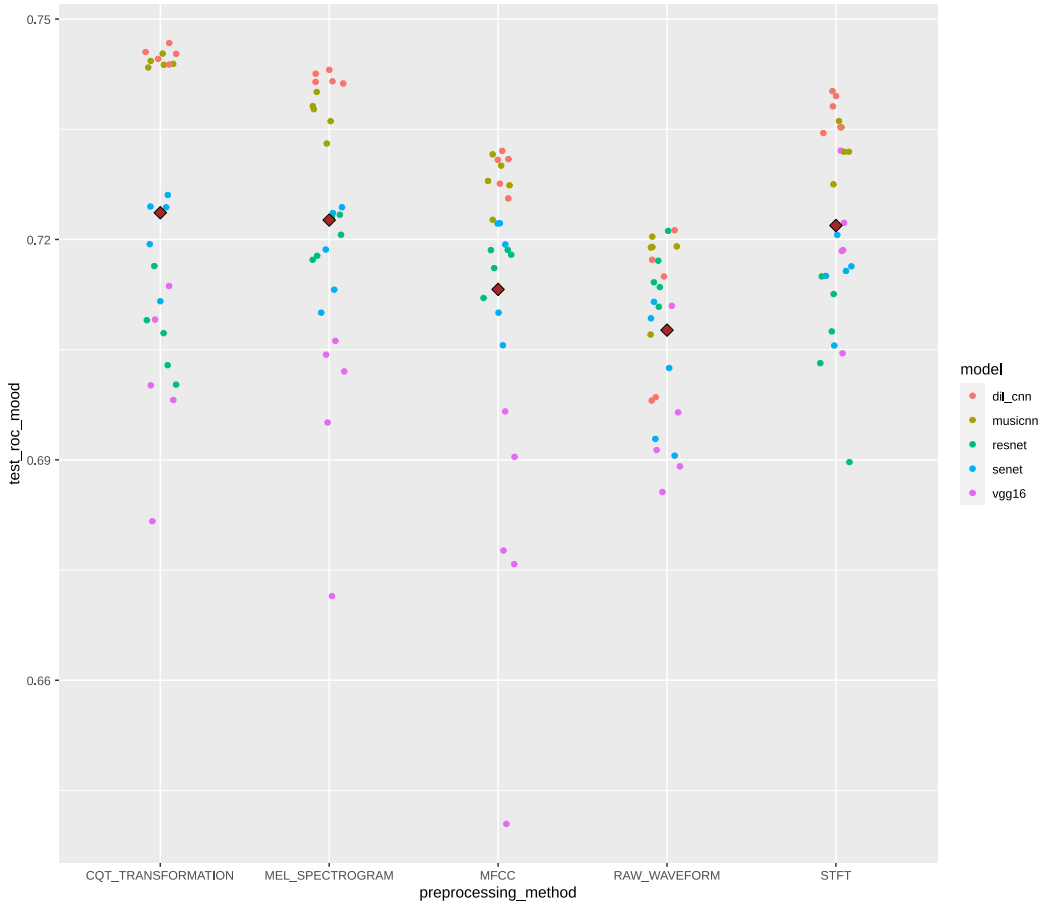


Figure 7.20: ROC-AUC scores for all preprocessing methods and all models on MTGJ using mood tags

input	PR-all	PR-gen	PR-instr	PR-mood	ROC-all	ROC-gen	ROC-instr	ROC-mood
CQT	0.3793	0.3676	0.4479	0.2501	0.9119	0.9357	0.9042	0.8880
MEL	0.3780	0.3618	0.4477	0.2555	0.9119	0.9341	0.9050	0.8893
MFCC	0.3629	0.3439	0.4284	0.2536	0.9071	0.9272	0.8971	0.8946
RAW	0.3194	0.3018	0.3638	0.2510	0.8880	0.9065	0.8736	0.8882
STFT	0.3778	0.3592	0.4445	0.2667	0.9149	0.9377	0.9049	0.8998

Table 7.23: Avg. results for Dilated CNN on MTAT

Experiment on MTAT

Table 7.23 displays the average results for the dilated CNN grouped by the preprocessing methods. Compared to Table 7.21, all metrics for all tag categories slightly improved on the two-dimensional spectrograms. However, almost all values are worse for the raw waveform than for the original Musicnn model. The only exception is the mood tags

results, which slightly increased too. The better classification of mood tags might be due to the better extraction of broader features of the dilated convolutions since mood tags intentionally depend on longer audio segments.

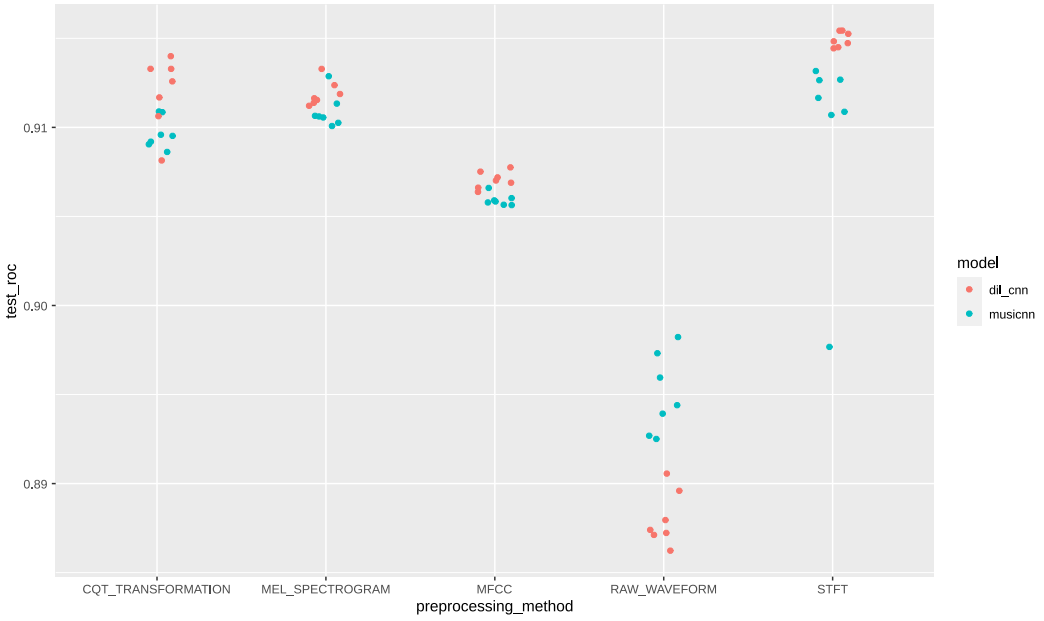


Figure 7.21: ROC-AUC scores of dilated CNN vs. Musicnn on MTAT

A graphical visualization of the ROC-AUC scores for both models is displayed in Figure 7.21. The graphic shows the same trends: the increase in classification performance on the spectrogram inputs and a decrease for the raw waveform.

The results are validated using T-tests for the means of two independent variables. Since the results differ greatly between the two- and one-dimensional preprocessing methods, the significance-tests are performed for each input representation separately. The following four assumptions have to be fulfilled to guarantee the meaningfulness of the results: interval- or ratio scaled data, normality/symmetry of the population-distributions, equality of variances, and skewness and kurtosis [Rietveld and van Hout, 2015]. The assumption of a ratio scaled data is trivially fulfilled for ROC-AUC and PR-AUC since both are decimals between 0 and 1.0. The tests for normal distribution hold for all ROC-AUC and PR-AUC scores on all preprocessing methods except for STFT and Mel spectrogram. The non-normal distribution of the former one is probably due to an outlier, as shown in Figure 7.21. For the latter one, the p-values around 0.03 are very close to the threshold of 0.05. These results still suggest applying the T-tests as planned and considering those violations of the assumption in the discussion of the T-test scores. As shown in the result tables, the assumption of the equality of variances holds since there is no difference between the models higher than a factor of two for any input representation. Since we have a normal distribution for almost all configurations and two borderline

cases, the fourth assumption is negligible to check. The tables show that the t-statistic value is larger than two for all two-dimensional inputs, and the p-values are all <= 0.05. Therefore the dilated CNN performs significant better than Musicnn for those input representations. For the STFT (cf. Table 7.25), the p-values for both metrics are close to 0.05. Considering the violation of the normal distribution, more samples would be necessary to draw clear conclusions, which is not possible in the scope of this thesis due to resource limitations. However, the mean value is about 0.5% higher than for Musicnn for ROC-AUC, and around 1% higher for PR-AUC. For the Mel spectrogram (cf. Table 7.24), the p-value for the ROC-AUC score is also close to the threshold of 0.05. However, the p-value of the PR-AUC score shows a very strong statistical significance with 3.77e-05. Therefore the dilated CNN clearly outperforms the Musicnn for Mel spectrograms. For the raw waveform Table 7.27 shows a strong statistical significance for the outperformance of Musicnn over the dilated CNN.

	ROC-AUC				PR-AUC			
model	var	mean	t_stat	p-val	var	mean	t_stat	p-val
dil_cnn	4.40E-07	0.9119	2.2011	0.0480	1.23E-06	0.3780	6.3302	3.77E-05
musicnn	7.76E-07	0.9109	-2.2011	0.0480	3.32E-06	0.3725	-6.3302	3.77E-05

Table 7.24: T-Tests for dilated CNN vs. Musicnn on MTAT using Mel spectrogram

	ROC-AUC				PR-AUC			
model	var	mean	t_stat	p-val	var	mean	t_stat	p-val
dil_cnn	1.55E-07	0.9149	2.4229	0.0321	9.13E-06	0.3778	2.1892	0.0491
musicnn	2.57E-05	0.9099	-2.4229	0.0321	1.27E-04	0.3674	-2.1892	0.0491

Table 7.25: T-Tests for dilated CNN vs. Musicnn on MTAT using STFT

	ROC-AUC				PR-AUC			
model	var	mean	t_stat	p-val	var	mean	t_stat	p-val
dil_cnn	2.00E-07	0.9071	5.1207	0.0003	3.31E-06	0.3629	4.3089	0.0010
musicnn	9.28E-08	0.9059	-5.1207	0.0003	2.04E-06	0.3588	-4.3089	0.0010

Table 7.26: T-Tests for dilated CNN vs. Musicnn on MTAT using MFCCs

	ROC-AUC				PR-AUC			
model	var	mean	t_stat	p-val	var	mean	t_stat	p-val
dil_cnn	1.99E-06	0.8880	-6.8489	1.78E-05	5.69E-06	0.3194	-7.0720	1.30E-05
musicnn	4.25E-06	0.8950	6.8489	1.78E-05	1.57E-05	0.3327	7.0720	1.30E-05

Table 7.27: T-Tests for dilated CNN vs. Musicnn on MTAT using raw waveform

Table 7.29 shows the average epoch training times for each input representation for the dilated CNN and Musicnn. [1] The training times for the dilated CNN are approximately

[1] To ensure the comparability of the values, only the values of the Nvidia GTX 2080 are used for this comparison

	ROC-AUC				PR-AUC			
model	var	mean	t_stat	p-val	var	mean	t_stat	p-val
dil_cnn	3.50E-06	0.9119	2.7251	0.0184	2.29E-05	0.3793	3.7296	0.0029
musicnn	6.62E-07	0.9097	-2.7251	0.0184	4.25E-06	0.3714	-3.7296	0.0029

Table 7.28: T-Tests for dilated CNN vs. Musicnn on MTAT using CQT

half of the size as for Musicnn for CQT, Mel spectrogram, and MFCCs. For STFT, it is almost only a quarter the size. The only exception is the raw waveform, where the training time is 10% longer than for Musicnn. Therefore, not only the classification performance could be increased for the two-dimensional input representations, but also the training time could be decreased significantly.

input	time_dil_cnn (minutes)	time_musicnn (minutes)
CQT	52.21	105.75
MEL	51.38	106.39
MFCC	16.94	25.21
RAW	122.60	109.49
STFT	65.24	215.82

Table 7.29: Avg. epoch times for dilated CNN vs. Musicnn on MTAT

To sum up, the results suggest that dilated convolutions and more concretely stacked parallel dilated convolutions can be beneficial for the classification performance and the training time of CNNs for music autotagging. Especially when using two-dimensional input representations, it is worth considering them for the architectural design of the network. However, the raw waveforms did not benefit from the extension with the dilated convolutions.

Experiment on MTGJ

Figure 7.22 shows the results for the dilated CNN and Musicnn for all input representations. The trends are very similar to the previous dataset, and again all spectrograms perform better than the raw waveform input. The latter is also the only one there the Musicnn performs better than the dilated CNN. It is also striking that the CQT transformation performs even slightly better than the STFT on both models.

T-tests are used to validate or reject the MTAT dataset results. Again the following four assumptions must be checked first: interval- or ratio scaled data, normality/symmetry of the population-distributions, equality of variances, and skewness and kurtosis [Rietveld and van Hout, 2015]. The first assumption is again trivially fulfilled for ROC-AUC and PR-AUC since both are decimals between 0 and 1.0. The tests for normal distribution hold for all ROC-AUC and PR-AUC scores on all preprocessing methods except for the raw waveform on Musicnn, probably due to an outlier. This violation will be considered during the discussion of the results. As shown in the result tables, the

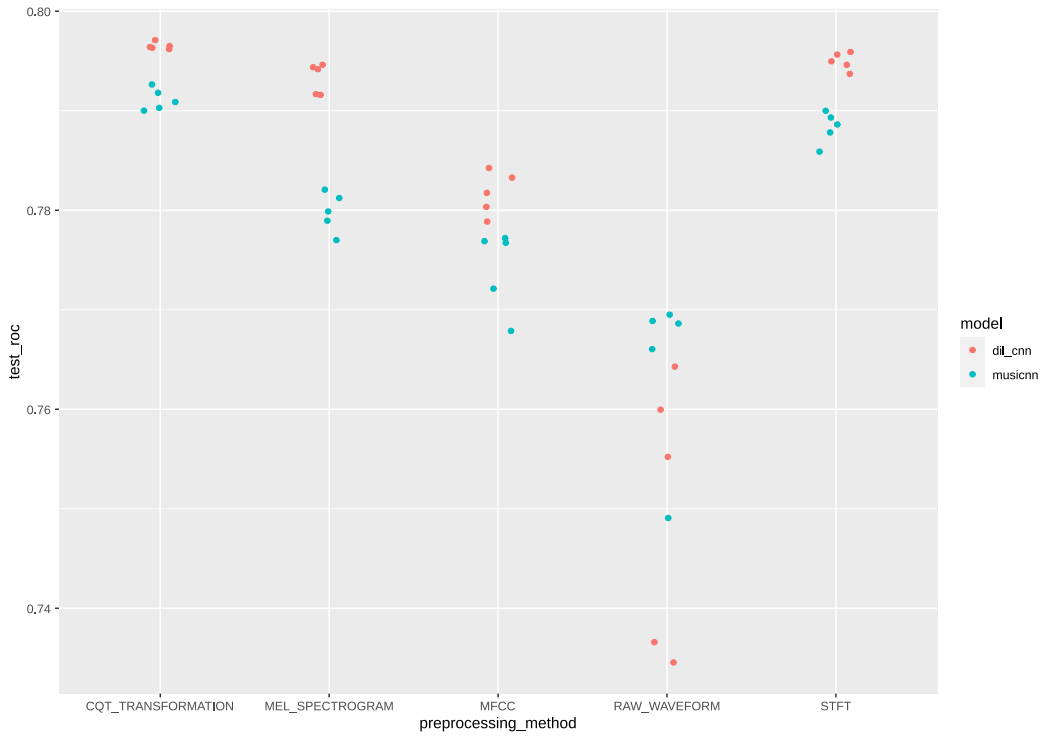


Figure 7.22: ROC-AUC scores of dilated CNN vs. Musicnn on MTGJ

assumption of the equality of variances holds since there is no difference between the models larger than a factor of two for any input representation. Since we have a normal distribution for almost all configurations, the fourth assumption is negligible to check.

	ROC-AUC				PR-AUC			
model	var	mean	t_stat	p-val	var	mean	t_stat	p-val
dil_cnn	1.85E-06	0.7933	12.0371	2.09E-06	5.68E-06	0.2373	8.2365	3.54E-05
musicnn	3.15E-06	0.7798	-12.0371	2.09E-06	3.03E-05	0.2126	-8.2365	3.54E-05

Table 7.30: T-Tests for dilated CNN vs. Musicnn on MTGJ using Mel spectrogram

	ROC-AUC				PR-AUC			
model	var	mean	t_stat	p-val	var	mean	t_stat	p-val
dil_cnn	6.14E-07	0.7950	8.1952	3.67E-05	3.04E-06	0.2428	5.4314	0.0006
musicnn	2.01E-06	0.7883	-8.1952	3.67E-05	6.66E-06	0.2343	-5.4314	0.0006

Table 7.31: T-Tests for dilated CNN vs. Musicnn on MTGJ using STFT

The results of the t-tests are consistent with the previous results of the MTAT dataset. For all input spectrograms, the dilated CNN significantly outperforms the Musicnn model, and it performs worse for the raw waveform. Therefore, this underlines the

	ROC-AUC				PR-AUC			
model	var	mean	t_stat	p-val	var	mean	t_stat	p-val
dil_cnn	3.78E-06	0.7817	3.6332	0.0067	1.12E-05	0.2238	3.0632	0.0155
musicnn	1.34E-05	0.7742	-3.6332	0.0067	9.24E-05	0.2082	-3.0632	0.0155

Table 7.32: T-Tests for dilated CNN vs. Musicnn on MTGJ using MFCCs

	ROC-AUC				PR-AUC			
model	var	mean	t_stat	p-val	var	mean	t_stat	p-val
dil_cnn	0.0001	0.7501	-1.9729	0.0840	0.0002	0.1784	-2.3388	0.0475
musicnn	6.03E-05	0.7644	1.9729	0.0840	7.42E-05	0.1965	2.3388	0.0475

Table 7.33: T-Tests for dilated CNN vs. Musicnn on MTGJ using raw waveform

	ROC-AUC				PR-AUC			
model	var	mean	t_stat	p-val	var	mean	t_stat	p-val
dil_cnn	9.90E-08	0.7965	10.4859	5.95E-06	7.95E-07	0.2430	12.9611	1.19E-06
musicnn	9.55E-07	0.7911	-10.4859	5.95E-06	8.98E-07	0.2346	-12.9611	1.19E-06

Table 7.34: T-Tests for dilated CNN vs. Musicnn on MTGJ using CQT

previous conclusion that the stacked parallel dilated convolutions can be beneficial for the classification performance of CNNs for music autotagging. However, this also validates the observation that models using the raw waveform as input might not benefit from dilated convolutions.

7.3 Summary

To sum up, the evaluation in this chapter shows that both model and preprocessing methods significantly impact the classification results during all experiments. Both experimental parts are conducted on two different datasets to validate the results. Therefore, this section is structured into consistent and inconsistent findings.

7.3.1 Consistent Findings

The STFT provides the best overall results on almost all tested configurations. This comes to the price of higher resource consumption due to the larger size of this input representation. Raw waveforms show the worst results on all configurations which might be because all models are originally designed for two-dimensional inputs and are adapted for this study. Despite their small size, MFCCs perform well across almost all models, so they might be a good choice for tasks requiring low resource consumption. Mel spectrograms show solid performances across all models but are still significantly worse than the STFT. However, due to their around ten times smaller size, they are much less resource-consuming which might be the reason why they are the more commonly used

input representation. The dilated CNN significantly outperforms the original Musicnn on the two-dimensional preprocessing methods and performs worse on the raw waveform.

7.3.2 Inconsistent Findings

On MTAT the CQT transformation has the second-worst results but on MTGJ it performs equally well as the STFT. Therefore, no general conclusion can be drawn about the performance of this input representation whose size is comparable to the Mel spectrogram. Besides that, the evaluation of the individual tag categories differed too much between the two datasets to make further statements. For genre tags, similar results can be observed between MTAT and MTGJ but there is no particular trend identifiable compared to the evaluation across all tags. The results suggest that mood classification differs the most from the other tag categories. For example, MFCCs perform the second-worst for genre- and instrument classification but the second-best on mood tags. However, this was not the case for MTGJ, where MFCCs only show slightly better results than the raw waveform.

The following chapter summarizes this thesis and presents the main insights regarding the research questions.

CHAPTER 8

Conclusion

8.1 Summary

In this thesis, I investigated the suitability of different preprocessing methods on CNN models for music autotagging. Moreover, I discussed why dilated CNNs could be beneficial for this task and developed a new stacked dilated CNN from an existing model.

Chapter 1 introduced the topic, and I formulated the problem statements there. In Chapter 2, the most relevant work on music autotagging and other related topics useful for the investigation were presented. The most relevant basic knowledge of audio signals was described in Chapter 3, as well as the description, calculation, and recent research of the relevant preprocessing methods. Chapter 4 extensively discussed the relevant datasets MTAT and MTGJ for this study, the respective preprocessing steps, and some other datasets for music autotagging and related tasks. Some basic knowledge about CNNs was presented in Chapter 5 and relevant models for this thesis, including the developed dilated CNN. In Chapter 6, I proposed the concrete methodology for the experiments and evaluation to answer the research questions. Moreover, this chapter presented the concrete setup and preconditions for the experiments and a detailed description for each one. The goal of the first experiment was to develop a dilated CNN that could enhance the performance of existing models without dilated convolutions. In the second part of the experiments, five different CNN models are evaluated with all five input representations on both datasets. The results were presented and discussed in Chapter 7. The experimentation with dilated convolutions has shown that they can be useful for music autotagging models, especially for the two-dimensional input representations. Comparing the different inputs and models in the second experimental part has shown that the STFT input performed very well in all configurations. The raw waveform performed worse than the other inputs except for a few configurations. MFCCs have shown a solid performance across all models except for the Vgg-16. Since it is a comparatively small input representation, it is worth considering for use cases requiring low resource

consumption. The performance metrics have shown that the performance of the inputs depends on the dataset used, so for example, CQT transformation was rather bad for MTAT but outperformed Mel spectrogram and MFCCs on MTGJs. Therefore, I could not conclude if CQT transformation or Mel spectrogram perform better overall. The results show a different picture between the two datasets for the specific tag categories. As for all tags, the STFT provides good classification scores across all categories, and the raw waveform always has the worst average score. The results suggest that genre- and instrument tags show similar trends as the evaluation overall tags do, whereas the mood tags differentiate stronger between the input representations. For example, MFCCs on MTAT provided the second-worst results for genre,- instrument,- and overall tags but second best on mood tags. However, this was not the case for the MTGJ dataset, where the CQT transformation had the best average ROC-AUC score and MFCCs performed only slightly better than the worst raw waveform.

8.2 Insights regarding research questions

To conclude this thesis, the three main research questions are answered using the insights gained in this work.

> Which audio input representations generally provide better results for CNNs than others, and to what extent?

Looking at the results in Chapter 7, it can be seen that the STFT shows the best performance among all input representations on almost all models. Raw waveforms did perform comparatively badly on both datasets. The results on MTAT suggest that this input can lead to good results on some models, but it probably requires an appropriate adaption. The other three two-dimensional input representations performed worse than STFT on most configurations. Therefore, it can be stated that for maximum classification performance, STFT might be the best choice for CNNs. CQT outperformed Mel spectrogram and MFCCs on the MTGJ dataset, whereas, on average, it performed worse than all other two-dimensional preprocessing methods on MTAT. Mel spectrograms showed a consistent and solid performance throughout both datasets but rarely top results. MFCCs performed well on MTAT, but on the bigger MTGJ, it was the worst two-dimensional input. However, the average training time was much lower than for the other input representations, and also, for tasks with limited resources, it might be a good option.

> How much difference in classification performance is between different audio input representations for certain tag categories?

As discussed in the previous subsection, there was too much difference between the two datasets to make general statements for the performance on the separate tag categories.

On MTAT, for example, the MFCCs outperformed all other preprocessing methods, except for STFT, on mood tags, whereas it performed worse on the other categories. On MTGJ, this was not the case, and moreover, the CQT transformation outperformed all other input representations for mood tags. On MTAT, this input performed worse than all other preprocessing methods for mood tags, which contradicts the other dataset. All in all, it can be stated that there is not enough evidence for any input representation to perform better on one tag category than on the others. There are recognizable differences, but further research on this question is needed on different datasets and models to give a more concrete answer to it.

> To what extent are dilated convolutions beneficial for music autotagging in terms of classification performance and training time?

As seen in the first experimental part in Section 7.1, dilated convolutions can enhance both classification performance and training time. The enhanced Musicnn with stacked parallel dilated convolutions outperformed the original model in both aspects on all two-dimensional input representations. However, for raw waveforms, it even performed worse, so for this one-dimensional input representation, more research is required to see if dilated convolutions can be beneficial for it too. Overall, the dilated CNN could significantly improve the classification and training time, and therefore it is worth considering those dilated convolutions for future research on music autotagging.

8.3 Limitations and outlook

8.3.1 Limitations

The investigated scope of this thesis was rather extensive and included five different input representations on five different models and two datasets. The goal was to cover the most important preprocessing methods and test them on different CNN architectures to keep the generalization of the results as high as possible. However, this inevitably led to huge resource consumption, and therefore it was only possible to run five to seven runs per configuration which is a relatively small number for statistical evaluations.

The experiments have shown that it is hard to compare the one-dimensional raw waveforms with the two-dimensional inputs. Such a comparison might require more extensive adaptions on the respective models than it was possible in the scope of this thesis. It has been shown that this input could reach good results on specific models like ResNet on MTAT, so it is hard to say if this was only the case for this individual model or if the other models need more adjustment for reaching good results.

The investigation on the effect of dilated convolutions for music autotagging models was only one part of this thesis and therefore had to be restricted. Instead of developing a completely new model, the existing Musicnn that reached state-of-the-art performance during its publication in 2019 was adapted accordingly to reduce training time and

increase classification performance [Pons and Serra, 2019]. It is not enough to assess the effects of dilated convolutions on one or two models, so the results reveal only their potential advantages.

8.3.2 Outlook

Future research on this topic could focus on a smaller number of input representations. An interesting comparison would be STFT versus raw waveforms since the STFT provided the best results among all two-dimensional preprocessing methods. This would allow to better adapt the models for the one-dimensional input and test more different models. Important would also be to test different adaptions for all investigated models to unleash both models´ full potential and ensure that no model is under- or overfitting. Not part of this thesis was the effect of different parameter configurations and scaling techniques for the input representations. There is already some research on this topic, like in [Ferraro et al., 2021], where Mel spectrograms are tested with different numbers of Mel bands against different sampling rates. It would also be interesting to compare different window lengths for the STFT and different resolutions and octave numbers for CQT.

The effects of dilated convolutions for music autotagging models also require further investigation. Moreover, it would be interesting to develop an entirely new CNN architecture based on the insights gained in this thesis that exposes the strengths of dilated convolutions.

List of Figures

List of Tables

Acronyms

CNN Convolutional Neural Network. xi, xiii, 1–3, 5–8, 13, 16, 19, 20, 23, 28, 29, 33, 34, 36, 37, 39–43, 45, 46, 50, 51, 53, 57, 63, 66, 74–90

CQT Constant-Q Transformation. xi, xiii, 2, 7, 16, 21–23, 56, 57, 59, 60, 62, 63, 65, 66, 69, 72, 74, 79, 81, 82, 84–87, 90

CRNN Convolutional Recurrent Neural Network. 28

CWT Continuous Wavelet Transform. 16

DCT Discrete Cosine Transform. 19

DFT Discrete Fourier Transformation. 14, 15, 21, 22

DL Deep Learning. 5, 6, 8, 11–13, 27, 29, 46

DNN Dense Neural Network. 28

FC Fully connected. 6, 33, 37

FFT Fast Fourier Transformation. 13, 15, 16, 21, 22

LSTM Long short-term memory. 8

MFCC Mel Frequency Cepstral Coefficient. xi, xiii, 2, 5, 7, 16, 19, 20, 23, 38, 47, 48, 51, 56, 57, 59, 60, 62–65, 69, 72, 74, 78, 79, 81–85, 87, 89, 90

MIR Music Information Retrieval. xi, xiii, 1, 9, 11–13, 16–20, 23, 28, 31, 33, 45

ML Machine Learning. 5

MSD Million Song Dataset. 6, 7, 16, 29, 31

MTAT MagnaTagATune. xi, xiii, 3, 6, 7, 13, 19, 25–27, 29, 31, 37, 45, 46, 48–51, 53–66, 69, 71, 74–80, 82–85, 87–90

MTGJ MTG-Jamendo. xi, xiii, 3, 6, 25, 27, 29–31, 46, 48–51, 62, 63, 65, 67–76, 79–85, 87–90

RBM Restricted Boltzmann machine. 5

RNN Recurrent Neural Network. 6, 13, 28

SE Squeeze and Excitation. 37, 38

SENet Squeeze and Excitation Network. 38, 47, 57, 87

SGD Stochastic Gradient Descent. 48

STFT Short Time Fourier Transformation. xi, xiii, 2, 7, 13–19, 22, 23, 47, 57, 59, 60, 62, 63, 65, 66, 69, 71, 72, 74, 78–87, 89, 90

Bibliography

[Albawi et al., 2017] Albawi, S., Mohammed, T. A., and Al-Zawi, S. (2017). Understanding of a convolutional neural network. In *Proceedings of the 2017 International Conference on Engineering and Technology (ICET)*, pages 1–6.

[Aphex34, 2015] Aphex34 (2015). Typical cnn architecture. https://commons.wikimedia.org/wiki/File:Typical_cnn.png. [Online; accessed 16-October-2021].

[Avramidis et al., 2021] Avramidis, K., Kratimenos, A., Garoufis, C., Zlatintsi, A., and Maragos, P. (2021). Deep convolutional and recurrent networks for polyphonic instrument classification from monophonic raw audio waveforms. In *Proceedings of the ICASSP 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3010–3014.

[Barrington et al., 2008] Barrington, L., Turnbull, D., and Lanckriet, G. (2008). Auto-tagging music content with semantic multinomials. In *Proceedings of the 9th International Conference on Music Information Retrieval, ISMIR 2008*, Philadelphia, PA, USA.

[Bertin-Mahieux et al., 2010] Bertin-Mahieux, T., Eck, D., and Mandel, M. (2010). Automatic tagging of audio: The state-of-the-art. *Machine Audition: Principles, Algorithms and Systems*, pages 334–352.

[Bertin-Mahieux et al., 2011] Bertin-Mahieux, T., Ellis, D., Whitman, B., and Lamere, P. (2011). The million song dataset. In *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011*, pages 591–596.

[Bevans, 2020] Bevans, R. (2020). Anova in r. https://www.scribbr.com/statistics/anova-in-r/. [Online; accessed 14-February-2022].

[Bogdanov et al., 2019a] Bogdanov, D., Porter, A., Schreiber, H., Urbano, J., and Oramas, S. (2019a). The acousticbrainz genre dataset: Multi-source, multi-level, multi-label, and large-scale. In *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR 2019)*, Delft, The Netherlands.

[Bogdanov et al., 2019b] Bogdanov, D., Porter, A., Tovstogan, P., and Won, M. (2019b). Mediaeval 2019: Emotion and theme recognition in music using jamendo. In *MediaEval 2019, Multimedia Benchmark Workshop*, Sophia Antipolis, France.

[Bogdanov et al., 2019c] Bogdanov, D., Won, M., Tovstogan, P., Porter, A., and Serra, X. (2019c). The mtg-jamendo dataset for automatic music tagging. In *Machine Learning for Music Discovery Workshop, International Conference on Machine Learning (ICML 2019)*, Long Beach, CA, United States.

[Brown, 1991] Brown, J. C. (1991). Calculation of a constant q spectral transform. In *Journal of the Acoustical Society of America*, volume 89, pages 425–434.

[Camacho-Collados and Pilehvar, 2018] Camacho-Collados, J. and Pilehvar, M. T. (2018). On the role of text preprocessing in neural network architectures: An evaluation study on text categorization and sentiment analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 40–46, Brussels, Belgium.

[Choi et al., 2016] Choi, K., Fazekas, G., and Sandler, M. (2016). Automatic tagging using deep convolutional neural networks. In *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR 2016*, pages 805–811, New York City, USA.

[Choi et al., 2018] Choi, K., Fazekas, G., Sandler, M., and Cho, K. (2018). A comparison of audio signal preprocessing methods for deep neural networks on music tagging. In *Proceedings of the 26th European Signal Processing Conference, EUSIPCO 2018*, pages 1870–1874, Italy. IEEE.

[Davis and Goadrich, 2006] Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240.

[Defferrard et al., 2017] Defferrard, M., Benzi, K., Vandergheynst, P., and Bresson, X. (2017). Fma: A dataset for music analysis. In *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017*, pages 316–323, Suzhou, China.

[Dipani et al., 2020] Dipani, A., Iyer, G., and Baths, V. (2020). Recognizing music mood and theme using convolutional neural networks and attention. In *MediaEval Benchmarking Initiative for Multimedia Evaluation*, online.

[Dörfler et al., 2017] Dörfler, M., Bammer, R., and Grill, T. (2017). Inside the spectrogram: Convolutional neural networks in audio processing. In *2017 international conference on sampling theory and applications (SampTA)*, pages 152–155. IEEE.

[Dutta and Chanda, 2021] Dutta, J. and Chanda, D. (2021). Music emotion recognition in assamese songs using mfcc features and mlp classifier. In *Proceedings of the 2021 International Conference on Intelligent Technologies (CONIT)*, pages 1–5. IEEE.

[Dörfler et al., 2017] Dörfler, M., Bammer, R., and Grill, T. (2017). Inside the spectrogram: Convolutional neural networks in audio processing. In *Proceedings of the 2017 International Conference on Sampling Theory and Applications (SampTA)*, pages 152–155.

[Ellis et al., 2013] Ellis, K., Coviello, E., Chan, A. B., and Lanckriet, G. (2013). A bag of systems representation for music auto-tagging. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(12):2554–2569.

[Fawcett, 2004] Fawcett, T. (2004). Roc graphs: Notes and practical considerations for researchers. *Machine learning*, 31(1):1–38.

[Ferraro et al., 2021] Ferraro, A., Bogdanov, D., Serra, X., Jeon, J. H., and Yoon, J. (2021). How low can you go? reducing frequency and time resolution in current cnn architectures for music auto-tagging. In *Proceedings of the 28th European Signal Processing Conference, EUSIPCO 2020*, pages 131–135, Amsterdam, The Netherlands.

[Gerczuk et al., 2020] Gerczuk, M., Amiriparian, S., Ottl, S., Rajamani, S. T., and Schuller, B. (2020). Emotion and themes recognition in music with convolutional and recurrent attention-blocks. In *MediaEval Benchmarking Initiative for Multimedia Evaluation*, online.

[Goodfellow et al., 2016] Goodfellow, I. J., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press, Cambridge, MA, USA. http://www.deeplearningbook.org.

[Google, 2022] Google (2022). 2d convolution layer. https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv2D#used-in-the-notebooks_1. [Online; accessed 01-February-2022].

[Güçlü et al., 2016] Güçlü, U., Thielen, J., Hanke, M., and van Gerven, M. A. J. (2016). Brains on beats. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS 2016, page 2109–2117, Red Hook, NY, USA.

[Gus Berry, 2021] Gus Berry, P. R. (2021). Unsere anleitung zur bitrate von audiodateien. https://www.adobe.com/at/creativecloud/video/discover/audio-bitrate.html. [Online; accessed 22-September-2021].

[Görne, 2008] Görne, T. (2008). *Tontechnik: Schwingungen und Wellen, Hören, Schallwandler, Impulsantwort, Faltung, Sigma-Delta-Wandler, Stereo, Surround, WFS, Regiegeräte, tontechnische Praxis*. Hanser Fachbuch, München.

[Han et al., 2018] Han, H., Luo, X., Yang, T., and Shi, Y. (2018). Music recommendation based on feature similarity. In *Proceedings of the 2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, pages 650–654. IEEE.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, Nevada, USA.

[Hu et al., 2018] Hu, J., Shen, L., and Sun, G. (2018). Squeeze-and-excitation networks. In *Proceedings of the 2018 IEEE conference on computer vision and pattern recognition (CVPR)*, pages 7132–7141, Salt Lake City, Utah, USA.

[Huang et al., 2017] Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, Hawaii, USA.

[Huzaifah, 2017] Huzaifah, M. (2017). Comparison of time-frequency representations for environmental sound classification using convolutional neural networks. *arXiv preprint arXiv:1706.07156*.

[Jiang et al., 2020] Jiang, X., Lu, M., and Wang, S. (2020). An eight-layer convolutional neural network with stochastic pooling, batch normalization and dropout for fingerspelling recognition of chinese sign language. In *Multimedia Tools and Applications*, volume 79, pages 15697–15715.

[Kehtarnavaz, 2008] Kehtarnavaz, N. (2008). *Digital Signal Processing System Design, Second Edition: LabVIEW-Based Hybrid Programming*. Academic Press, Inc., USA.

[Kim, 2017] Kim, P. (2017). *MATLAB Deep Learning: With Machine Learning, Neural Networks and Artificial Intelligence*, chapter Convolutional Neural Network, pages 121–147. Apress, Berkeley, CA.

[Kim et al., 2018] Kim, T., Lee, J., and Nam, J. (2018). Sample-level cnn architectures for music auto-tagging using raw waveforms. *Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 366–370.

[Kozak and Piepho, 2018] Kozak, M. and Piepho, H.-P. (2018). What's normal anyway? residual plots are more telling than significance tests when checking anova assumptions. *Journal of agronomy and crop science*, 204(1):86–98.

[Law et al., 2009] Law, E., West, K., Mandel, M., Bay, M., and Downie, J. (2009). Evaluation of algorithms using games: The case of music tagging. In *Proceedings of the 10th International Conference on Music Information Retrieval, ISMIR 2009*, pages 387–392, Kobe, Japan.

[Law et al., 2007] Law, E. L., Von Ahn, L., Dannenberg, R. B., and Crawford, M. (2007). Tagatune: A game for music and sound annotation. In *Proceedings of the 8th International Conference on Music Information Retrieval, ISMIR 2007*, volume 3, pages 361–364, Vienna, Austria.

[Lee et al., 2017a] Lee, J., Kim, T., Park, J., and Nam, J. (2017a). Raw waveform-based audio classification using sample-level cnn architectures. *arXiv preprint arXiv:1712.00866*.

[Lee and Nam, 2017] Lee, J. and Nam, J. (2017). Multi-level and multi-scale feature aggregation using pretrained convolutional neural networks for music auto-tagging. *IEEE signal processing letters*, 24(8):1208–1212.

[Lee et al., 2017b] Lee, J., Park, J., Kim, K. L., and Nam, J. (2017b). Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms. *In Proceedings of the Sound Music Computing Conference (SMC)*, pages 220–226.

[Lei et al., 2019] Lei, X., Pan, H., and Huang, X. (2019). A dilated cnn model for image classification. In *IEEE Access*, volume 7, pages 124087–124095.

[Logan, 2000] Logan, B. (2000). Mel frequency cepstral coefficients for music modeling. In *In International Symposium on Music Information Retrieval*, Plymouth, United States.

[Mandel et al., 2011] Mandel, M., Pascanu, R., Larochelle, H., and Bengio, Y. (2011). Autotagging music with conditional restricted boltzmann machines. *arXiv preprint arXiv:1103.2832*.

[Marques et al., 2011] Marques, G., Domingues, M., Langlois, T., and Gouyon, F. (2011). Three current issues in music autotagging. In *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011*, pages 795–800, Miami,Florida, USA.

[McFee et al., 2015] McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., and Nieto, O. (2015). librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, volume 8, pages 18–25, Austin, Texas, USA.

[Meng and Chen, 2020] Meng, Z. and Chen, W. (2020). Automatic music transcription based on convolutional neural network, constant q transform and mfcc. In *Journal of Physics: Conference Series*, volume 1651, cf. 012192. IOP Publishing.

[Mohamed, 2017] Mohamed, I. S. (2017). *Detection and Tracking of Pallets using a Laser Rangefinder and Machine Learning Techniques*. PhD thesis, University of Genova, Italy.

[Moore, 2013] Moore, B. (2013). *An Introduction to the Psychology of Hearing: Sixth Edition*. Brill, Leiden, Niederlande.

[Muhammad et al., 2021] Muhammad, K., Mustaqeem, Ullah, A., Imran, A. S., Sajjad, M., Kiran, M. S., Sannino, G., and de Albuquerque, V. H. C. (2021). Human action recognition using attention based lstm network with dilated cnn features. In *Future Generation Computer Systems*, volume 125, pages 820–830.

[Nam et al., 2015] Nam, J., Herrera, J., and Lee, K. (2015). A deep bag-of-features model for music auto-tagging. *arXiv preprint arXiv:1508.04999*.

[Pfister and Kaufmann, 2008] Pfister, B. and Kaufmann, T. (2008). *Sprachverarbeitung: Grundlagen und Methoden der Sprachsynthese und Spracherkennung*. Springer, Berlin.

[Podder et al., 2014] Podder, P., Khan, T. Z., Khan, M. H., and Rahman, M. M. (2014). Comparative performance analysis of hamming, hanning and blackman window. *International Journal of Computer Applications*, 96(18).

[Pons et al., 2016] Pons, J., Lidy, T., and Serra, X. (2016). Experimenting with musically motivated convolutional neural networks. In *Proceedings of the 14th International Workshop on Content-Based Multimedia Indexing, CBMI 2016*, pages 1–6, Bucharest, Romania.

[Pons et al., 2018] Pons, J., Nieto, O., Prockup, M., Schmidt, E., Ehmann, A., and Serra, X. (2018). End-to-end learning for music audio tagging at scale. In *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018*, pages 637–644, Paris, France.

[Pons and Serra, 2019] Pons, J. and Serra, X. (2019). musicnn: Pre-trained convolutional neural networks for music audio tagging. In *Late Breaking of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019*, Delft, The Netherlands.

[Powers, 2011] Powers, D. M. W. (2011). Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation. In *Journal of Machine Learning Technologies*, volume 2, pages 37–63.

[Purwins et al., 2019] Purwins, H., Li, B., Virtanen, T., Schlüter, J., Chang, S.-Y., and Sainath, T. (2019). Deep learning for audio signal processing. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):206–219.

[Rietveld and van Hout, 2015] Rietveld, T. and van Hout, R. (2015). The t test and beyond: Recommendations for testing the central tendencies of two independent samples in research on speech, language and hearing pathology. In *Journal of Communication Disorders*, volume 58, pages 158–168.

[Rocchesso, 2003] Rocchesso, D. (2003). *Introduction to Sound Processing*. Mondo estremo, Italy.

[Santana et al., 2020] Santana, I. A. P., Pinhelli, F., Donini, J., Catharin, L. G., Mangolin, R. B., Costa, Y. M. G., Feltrim, V. D., and Domingues, M. A. (2020). Music4all: A new music database and its applications. In *Proceedings of the 2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, pages 399–404, Rio de Janeiro, Brazil.

[Schuster et al., 2019] Schuster, R., Wasenmuller, O., Unger, C., and Stricker, D. (2019). Sdc-stacked dilated convolution: A unified descriptor network for dense matching tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2556–2565, Long Beach, CA, USA.

[Sewak et al., 2018] Sewak, M., Karim, M. R., and Pujari, P. (2018). *Practical convolutional neural networks: implement advanced deep learning models using Python*. Packt Publishing Ltd.

[Sharma et al., 2017] Sharma, S., Sharma, S., and Athaiya, A. (2017). Activation functions in neural networks. *Towards Data Science*, 6(12):310–316.

[Simonyan and Zisserman, 2015] Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In Bengio, Y. and LeCun, Y., editors, *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015*, San Diego, CA, USA.

[Sinclair, 2000] Sinclair, I. (2000). *Audio and hi-fi handbook*. Course Technology, Oxford, England.

[Song et al., 2020] Song, G., Wang, Z., Han, F., Ding, S., and Gu, X. (2020). Music auto-tagging using scattering transform and convolutional neural network with self-attention. In *Applied Soft Computing*, volume 96, page 106702.

[Song et al., 2018] Song, G., Wang, Z., Han, F., Ding, S., and Iqbal, M. A. (2018). Music auto-tagging using deep recurrent neural networks. *Neurocomputing*, 292:104–110.

[Stevens et al., 1937] Stevens, S. S., Volkmann, J. E., and Newman, E. B. (1937). A scale for the measurement of the psychological magnitude pitch. *Journal of the Acoustical Society of America*, 8:185–190.

[Tendulkar et al., 2020] Tendulkar, P., Das, A., Kembhavi, A., and Parikh, D. (2020). Feel the music: Automatically generating a dance for an input song. In *Proceedings of the Eleventh International Conference on Computational Creativity, ICCC 2020*, Coimbra, Portugal.

[Tsang, 2018] Tsang, S.-H. (2018). Review: Dilatednet — dilated convolution (semantic segmentation). https://towardsdatascience.com/review-dilated-convolution-semantic-segmentation-9d5a5bd768f5. [Online; accessed 29-September-2021].

[Turnbull et al., 2007] Turnbull, D., Barrington, L., Torres, D., and Lanckriet, G. (2007). Towards musical query-by-semantic-description using the cal500 data set. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 439–446, Amsterdam, The Netherlands.

[ul Hassan, 2018] ul Hassan, M. (2018). Vgg16 – convolutional network for classification and detection. https://neurohive.io/en/popular-networks/vgg16/. [Online; accessed 18-October-2021].

[Vishnupriya and Meenakshi, 2018] Vishnupriya, S. and Meenakshi, K. (2018). Automatic music genre classification using convolution neural network. In *Proceedings of the 2018 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–4, Bristol, United Kingdom.

[Weights-and Biases, 2021] Weights-and Biases, I. (2021). Weights and biases. https://wandb.ai/site. [Online; accessed 26-October-2021].

[Wolff et al., 2015] Wolff, D., MacFarlane, A., and Weyde, T. (2015). Comparative music similarity modelling using transfer learning across user groups. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015*, pages 24–30, Malaga, Spain.

[Wolff and Weyde, 2012] Wolff, D. and Weyde, T. (2012). Adapting similarity on the magnatagatune database: effects of model and feature choices. In *Proceedings of the 21st international conference on world wide web*, pages 931–936, Lyon, France.

[Won et al., 2019] Won, M., Chun, S., and Serra, X. (2019). Toward interpretable music tagging with self-attention. *arXiv preprint arXiv:1906.04972*.

[Won et al., 2020] Won, M., Ferraro, A., Bogdanov, D., and Serra, X. (2020). Evaluation of cnn-based automatic music tagging models. In *Proceedings of the 17th Sound and Music Computing Conference (SMC2020)*, Toronto, Ontario, Canada.

[Wu et al., 2017] Wu, Y., Wang, Q., and Liu, R. (2017). Music instrument classification using nontonal mfcc. In *Proceedings of the International Conference on Frontiers of Manufacturing Science and Measuring Technology*, pages 24–25, Taiyuan, China.

[Yu, 2020] Yu, F. (2020). Dilated residual network. https://github.com/fyu/drn/. [Online; accessed 27-October-2021].

[Yu et al., 2017] Yu, F., Koltun, V., and Funkhouser, T. (2017). Dilated residual networks. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 636–644, Los Alamitos, CA, USA.

[Yu et al., 2021] Yu, Y.-b., Qi, M.-h., Tang, Y.-f., Deng, Q.-x., Mai, F., and Zhaxi, N. (2021). A sample-level dcnn for music auto-tagging. In *Multimedia Tools and Applications*, volume 80, pages 11459–11469.

[Zhang et al., 2017] Zhang, X., Zou, Y., and Shi, W. (2017). Dilated convolution neural network with leakyrelu for environmental sound classification. In *2017 22nd international conference on digital signal processing (DSP)*, pages 1–5.