UNIVERSITY OF BERGEN
DEPARTMENT OF INFORMATICS

# Automatic Drum Transcription using Deep Learning

*Author:* Runar Fosse
*Supervisor:* Pekka Parviainen

Add correct NT faculty, instead of MAT-NAT

UNIVERSITETET I BERGEN
*Det matematisk-naturvitenskapelige fakultet*

April, 2025

## Abstract

Lorem ipsum dolor sit amet, his veri singulis necessitatibus ad. Nec insolens periculis ex. Te pro purto eros error, nec alia graeci placerat cu. Hinc volutpat similique no qui, ad labitur mentitum democritum sea. Sale inimicus te eum.

No eros nemore impedit his, per at salutandi eloquentiam, ea semper euismod meliore sea. Mutat scaevola cotidieque cu mel. Eum an convenire tractatos, ei duo nulla molestie, quis hendrerit et vix. In aliquam intellegam philosophia sea. At quo bonorum adipisci. Eros labitur deleniti ius in, sonet congue ius at, pro suas meis habeo no.

Write proper abstract

## Acknowledgements

Est suavitate gubergren referrentur an, ex mea dolor eloquentiam, novum ludus suscipit in nec. Ea mea essent prompta constituam, has ut novum prodesset vulputate. Ad noster electram pri, nec sint accusamus dissentias at. Est ad laoreet fierent invidunt, ut per assueverit conclusionemque. An electram efficiendi mea.

Runar Fosse

Tuesday 1st April, 2025

Write proper acknowledgements

# Contents

**1 Introduction** **4**

1.1 Thesis statement . . . . . 4

**2 Background** **6**

2.1 Automatic Drum Transcription . . . . . 6

2.2 Audio . . . . . 7

2.2.1 Fourier Transform . . . . . 8

2.2.2 Discrete Fourier Transform . . . . . 9

2.2.3 Nyquist frequency . . . . . 9

2.2.4 Fast Fourier Transform . . . . . 10

2.2.5 Short-time Fourier Transform . . . . . 10

2.2.6 Spectrogram . . . . . 11

2.2.7 Filters . . . . . 12

2.3 Transcription . . . . . 12

2.3.1 Sheet Music . . . . . 12

2.3.2 MIDI Annotations . . . . . 12

2.3.3 Activation Functions . . . . . 12

2.4 Performance Measure . . . . . 13

**3 Architectures** **14**

3.1 Recurrent Neural Network . . . . . . . . . . . . . . . . . . . . . . . . . 14

3.2 Convolutional Neural Network . . . . . . . . . . . . . . . . . . . . . . . 14

3.3 Convolutional Recurrent Neural Network . . . . . . . . . . . . . . . . . 14

3.4 Convolutional Transformer . . . . . . . . . . . . . . . . . . . . . . . . . 14

3.5 Vision Transformer . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

**4 Datasets** **15**

4.1 ENST-Drums + MDB Drums . . . . . . . . . . . . . . . . . . . . . . . 15

4.2 EMG-D . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

4.3 Slakh . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

4.4 ADTOF-YT . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

**5 Methods** **16**

5.1 Task . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

5.2 Pipeline . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

5.2.1 Preprocessing . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

5.2.2 Training . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

5.2.3 Postprocessing . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

5.2.4 Performance Measures . . . . . . . . . . . . . . . . . . . . . . . 17

5.3 Experiments . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

5.3.1 Architecture experiment . . . . . . . . . . . . . . . . . . . . . . 18

5.3.2 Dataset generalization experiment . . . . . . . . . . . . . . . . . 18

5.3.3 Ablation experiments? . . . . . . . . . . . . . . . . . . . . . . . 18

**6 Results** **19**

6.1 Architecture experiment . . . . . . . . . . . . . . . . . . . . . . . . . . . 19

6.2 Dataset generalization experiment . . . . . . . . . . . . . . . . . . . . . . 20

6.3 Ablation experiments . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20

**List of Acronyms and Abbreviations** **21**

**Bibliography** **22**

# Chapter 1

# Introduction

Within the field of Music Information Retrieval (MIR), the task of Automatic Music Transcription (AMT) is considered to be, both an important, and challenging research problem. It describes the process of generating a symbolic notation from audio. The majority of instruments are melodic, where key information for transcription would be to discern pitch, onset time, and duration. This stands in contrast to percussive instruments, where instead of pitch and duration one would focus on instrument classification and onset detection. This sets the stage for Automatic Drum Transcription (ADT), which is a subfield of AMT, specifically focusing on drums and percussive instruments. [13]

Previously, a popular approach to ADT was using signal processing, which later developed into using classical machine learning methods [13]. However in later years, deep learning has shown to be quite effective. Therfore, the recent focus of most authors has been to find the best performing deep learning approaches by either; constructing and analysing the best performing model architectures, or by finding datasets which allow models to generalize the best. [14]

Provide a good introduction into the master thesis, mentioning AMT, ADT and why deep learning is suited for such a task.

## 1.1 Thesis statement

This leads us to two primary questions. Which deep learning architecture is the best suited for solving a task like this? And, what makes a dataset optimal by making models generalize? These are two of the questions we will try to answer in this thesis.

For the former, we will train different model architectures on different, well-known ADT datasets. Specifically, recurrent neural networks, convolutional neural networks, convolutional-recurrent neural networks, convolutional transformers and vision transformers. By comparing their performances we could be able to gauge the one best suited for an ADT task.

For the latter, we will select the best performing model architecture from the first question, and train it over several different combination of the ADT datasets. By performing zero-shot evaluations, we could analyse and figure out what makes a good ADT dataset and how it would supplement a suitable model architecture.

In addition to these, we will also analyse two standard approaches when it comes to ADT and see how effective they really are, through ablation studies. These are, usage of log-filtered spectrograms, and frequency-based, dynamic timestep loss-weighting during training.

Present the aim of the thesis here. And the questions! How do we train a model capable of solving such a task at a high performing level. More specifically:

What architectures are suited for learning such a task? What datasets / combination of datasets makes the model generalize best? Of the many techniques made to help models learn this task, which ones actually help? (Ablation)

**Remember the concrete What do we want to figure out.**

# Chapter 2

# Background

## 2.1 Automatic Drum Transcription

As mentioned, ADT describes the task of transcribing symbolic notation for drums from audio. To be even more descriptive, ADT can be split into further tasks. From least to most complex we have: Drum Sound Classification (DSC), where we classify drum instruments from isolated recordings. Drum Transcription of Drum-only Recordings (DTD), where we transcribe audio containing exclusively drum instruments. Drum Transcription in the Presence of Additional Percussion (DTP), where we transcribe audio containing drum instruments, and additional percussive instruments which the transcription should exclude. Finally, we have Drum Transcription in the Presence of Melodic Instruments (DTM), which describes the task of drum transcription with audio containing both drum, and melodic instruments. [13]

In this thesis, we will focus on the most complex of these, namely DTM. Intuitively, we want to develop a deep learning model which, given input audio, has the ability to detect and classify different drum instrument onsets (events), while selectively ignoring unrelated, melodic instruments.

This task comes with difficulties not seen in the less complex tasks. Zehren et al. [14] describes one example, in where *"melodic and percussive instruments can overlap and mask eachother..., or have similar sounds, thus creating confusion between instruments"*.

Deep learning has shown to be a promising method to solve such a task, and several different approaches have been tried, many with great success. Vogl et al. [12, 11]

displayed good results with both a convolutional, and a convolutional-recurrent neural network. Zehren et al. [14, 15] focused on datasets, showing that the amount of data and quality of data are equally important to get good performance. Most recently, Chang et al. [5] explored an autoregressive, language model approach. This approach explored multi-instrument transcriptions, but their results on ADT were notable.

This reinforces the fact that there still exist many approaches to attempt, which could lead to a general improvement on ADT models.

## 2.2 Audio

Sound has be described as *"the sensation caused in the nervous system by vibration of the delicate membranes of the ear."* [1]. In short, sound is the human perception of acoustic waves in a transition medium, like air. These waves, consisting of vibrating molecules, get sensed by our auditory organs and perceived by the brain.

Thus sound can be described as the propogation and perception of waves. Mathematically, waves can be studied as signals [4]. To represent these sounds digitally, as *audio*, one can express these waves as a signal, giving rise to the *waveform*. The waveform is a representation of a signal as a graph, and charts the amplitude, or strength of the signal, over time.

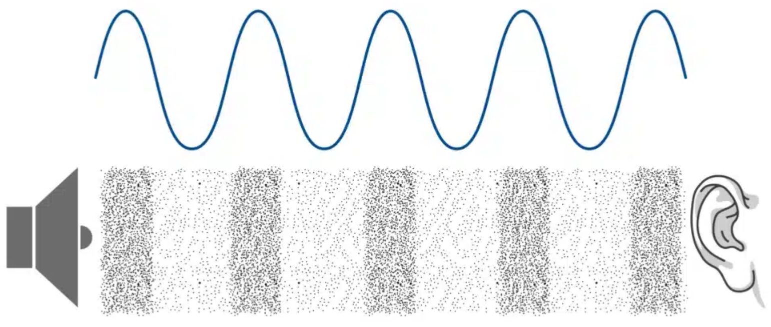
Figure 2.1: Soundwave to waveform relationship

For monophonic sound, this waveform is a one-dimensional representation. Even though this is an excellent way of storing audio digitally, it is very compact. There have been deep learning models working directly with these waveforms, e.g. Oord et al.'s WaveNet [10], however the task of parsing and perceiving such a signal is a complex one.

### 2.2.1 Fourier Transform

The Fourier Transform is a mathematical transformation which, given a frequency, computes its significance, or intensity, in a given signal. As we've established, audio is represented as a signal, and we can therefore use this transform to turn this audio signal into frequency space.

The fourier transform is a complex transformation, but we can attempt an intuitive explanation. Trigonometric functions have the property that they are *orthogonal* to eachother if they have different integer frequencies. In laymen's terms, the *area under the curve* of the product between to sines (or cosines) are zero if they have a different frequency.

$$\int_{\infty}^{\infty} \sin(ax)\sin(bx)dx = 0,\ \ a \neq b \wedge (a,b) \in \mathbb{Z}$$

Continue explanation but maybe not as mathy? Define and explain the core Fourier transform equation.

By doing such a transform, we turn our temporal data into spectral data. This initively *untangles* our signal into its respective base frequencies. Such an transformation could lessen the complexity of the task, making *understanding* of audio easier.

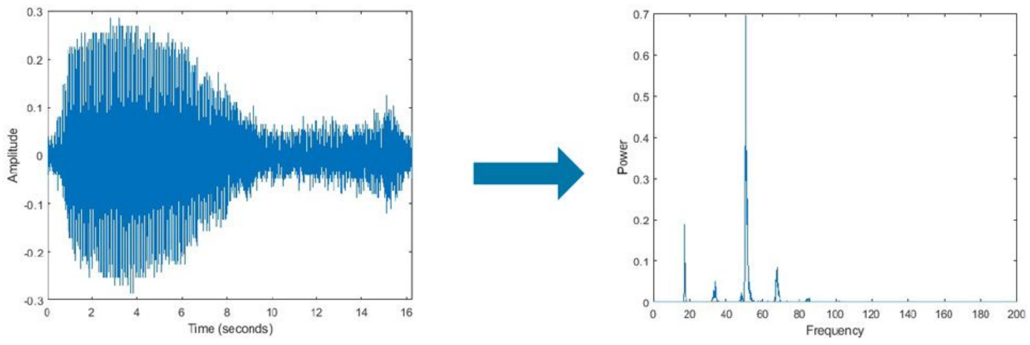

Figure 2.2: Application of a Fourier Transform

The information obtained from the Fourier Transform is *complex*. This means every value consists of a *real* part and an *imaginary* part. The real part consists of the amplitude of a certain frequency, where as the imaginary part consists of the phase.

### 2.2.2 Discrete Fourier Transform

The Fourier Transform is defined as an integral over continuous time. On computers, instead of storing signals continuously we store signals using a discrete number of samples. Each signal's *sampling rate* describes how many samples a signal contains per second of audio, and is denoted in *Hz*.

To extract frequency values from these signals, we instead have to use the Discrete Fourier Tranform (DFT). Intuitively this works as the normal Fourier Transform, but ported to work on discrete-valued signals. It is given by the formula

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi \frac{k}{N} n}$$

, where $k$ denotes the frequency and $N$ the number of discrete samples.

### 2.2.3 Nyquist frequency

When we discretize a signal, e.g. when going from continuous audio waves in the air to discrete audio signals on a computer, we could lose some information. T discrete representation of the signal is an *approximation*, which quality is directly dependent on the sampling rate. The higher the sampling rate, the *closer* we are the original, continuous signal. However a higher sampling rate comes at the cost of needing to store these signals at a higher precision. A lower sampling rate would need less information stored, but this also mean a less precise signal approxmation.

*Aliasing* is the phenomena where new frequencies seem to emerge in undersampled signals. For a given discrete signal, the *Nyquist frequency*, equal to half the sampling rate, is the maximum frequency a signal accurately can represent. Thus to prevent aliasing, one would need to store a signal with a sampling rate of at least double the maximum frequency.

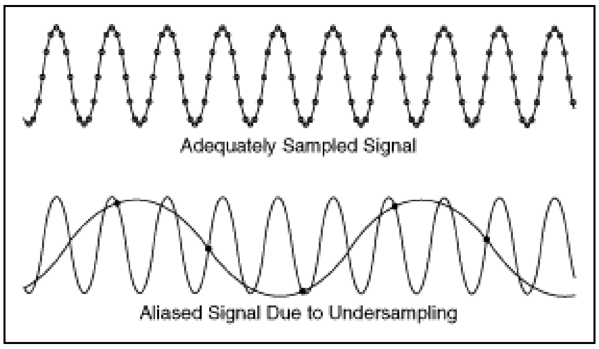
Adequately Sampled Signal
Aliased Signal Due to Undersampling

Figure 2.3: Example of aliasing in an undersampled signal.

Regarding the DFT, it here directly follows that the maximum frequency we accurately could extract information about is proportional to the sampling rate of the signal.

### 2.2.4 Fast Fourier Transform

Keen-eyed computer scientists may have spotted that the DFT runs in $\mathcal{O}(n^2)$ time as we, for every frequency in the range $[0, N]$ have to sum over $N$ different values. In other words, the DFT algorithm scales quite poorly. Take into account that the standard sampling rate for audio is 44.1kHz, i.e. 44100Hz, then we can see that the DFT could be inefficient. [2]

The Fast Fourier Transform (FFT) is an algorithm which solves this problem, and instead computes the DFT of a signal within $\mathcal{O}(n \log n)$ time. Described by Gilbert Strang as *"the most important numerical algorithm of our lifetime"* [9], this practically solves our scaling problem, and allows us to efficiently extract spectral information from a signal regardless of sampling rate.

There exist many different implementations of the FFT. However the Cooley-Tukey algorithm is by far the most used FFT and optimizes calculations through a *divide and conquer* approach, utilizing previous calculations to compute others. [6]

### 2.2.5 Short-time Fourier Transform

The Fourier Transform comes with some drawbacks, notably how by moving from time space into frequency space, we lose temporal information. For certain tasks this might be sufficient, but the temporal dimension is vital when working with transcriptions and ADT tasks. We've seen how the Fourier Transform computes the frequencies of a signal, but what happens if we had applied the same transform to smaller, *partitions* of a signal.

This leads us to the Short-time Fourier Transform (STFT). By instead of transforming the whole signal, we transform smaller *windows*, we could gain insight into the frequency space while keeping temporal information relatively intact. This turns our data from being one-dimensional into two-dimensional, giving us insight into the intensities of different frequencies, along different timesteps.

Talk more about the partitioning. The window functions applied, and why. Spectral leakage..

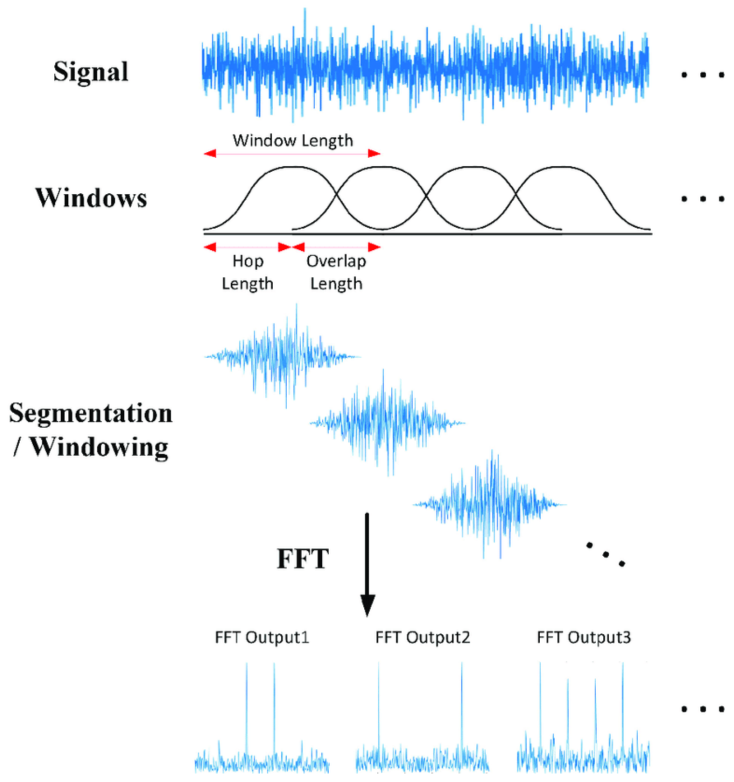

Figure 2.4: Example of the STFT

## 2.2.6 Spectrogram

The STFT, as the standard Fourier Transform, returns the data as complex values. To turn these into strictly real values without discarding data, we could compute the spectrogram. This is done by squaring the absolute value of each complex number.

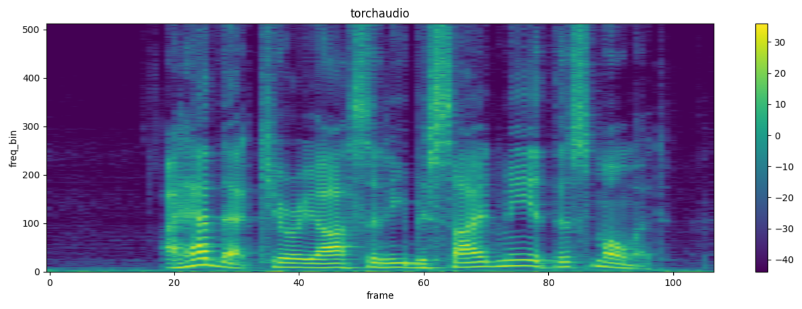

Figure 2.5: The spectrogram of an audio signal

Mention something about how we are losing some, or all phase information.

### 2.2.7 Filters

Explain about spectrogram filtering and filterbanks. Mention mel-filters, logarithmic-filters, and their usages in ADT.

#### Mel Spectrograms

#### Logarithmic Filters

## 2.3 Transcription

Explain what a transcription is, and what formats they usually are on. Explain what our model is predicting.

### 2.3.1 Sheet Music

### 2.3.2 MIDI Annotations

### 2.3.3 Activation Functions

Not the usual "Activation Functions" talked about in Deep Learning, but rather the output of the models. The *confidence* of a class' activation. "Probability of happening".

## 2.4 Performance Measure

Mention the different performance measures we have, what they are good for, and which ones are suitable here.

Also mention what is deemed a ”correct prediction”, and why we can allow a certain timeframe of ”correctness”.

# Chapter 3

# Architectures

Mention the different architectures, figures of their components, hyperparameters we tune on them, and motivation.

## 3.1 Recurrent Neural Network

## 3.2 Convolutional Neural Network

## 3.3 Convolutional Recurrent Neural Network

## 3.4 Convolutional Transformer

## 3.5 Vision Transformer

# Chapter 4

# Datasets

Mention the different already existing datasets used and information about them.

## 4.1 ENST-Drums + MDB Drums

The ENST-Drums dataset (ENST-Drums) by Gillet et al. [7] has become a staple dataset within ADT.

The same can be said for The MedleyDB Drums dataset (MDB Drums), from Southall et al. [8]. This dataset is built on top of Bittner et al.'s MedleyDB dataset [3], but re-annotated and specialized for ADT related tasks.

Due to the small size of these two datasets, they are in this thesis combined into one.

## 4.2 EMG-D

## 4.3 Slakh

## 4.4 ADTOF-YT

# Chapter 5

# Methods

## 5.1 Task

Precisely explain the task we are solving. Explain what the input data is, what the labels are. Give intuition into what exactly we want our model to predict.

Here we also explain the input and output, i.e. the data and the labels. What do they look like in their un-preprocessed form and predictions?

Here we can also give a figure into the pipeline itself, for better intuition.

## 5.2 Pipeline

Talk about the general ADT pipeline.

### 5.2.1 Preprocessing

Now explain what we do to the data before prediction. Explain the preprocessing steps we do afterwards (normalization, etc).

And explain how we preprocess the labels (target widening, etc.).

### 5.2.2 Training

Mention the loss function used, and why we use this (BCEWithLogitsLoss).

Mention the computation of infrequency weights, i.e. how they are computed, why they are computed, the intuition into how they will help us...

### 5.2.3 Postprocessing

Mention how model outputs a "confidence in event happening" distribution, which we want to discretize into events. I.e. explain Vogl's peak picking algorithm [12].

### 5.2.4 Performance Measures

Mention that we use F-measure (F1-score) is the most used and why. Compare this to accuracy, balanced accuracy. Mention precision, recall and their meaning.

Mention the difference in class-wise, micro- and macro-F1, and why we choose to focus on class-wise and micro in this thesis. Also mention how these are all computed.

## 5.3 Experiments

Here we mention the setups for each of the experiments.

Mention that we use RayTune to train, with PyTorch models. Mention that we only used RayTune's FIFOScheduler, and how for random search / grid search we used their built in parameter space functionality.

Mention that every single experiment was trained for at most 100 epochs, with a early stopping if validation loss didn't decrease within 10 epochs. Mention the learning rate scheduler, where we reduce the learning rate by a factor of 5 if the model hasn't improved in the last 3 epochs.

Mention that we perform early stopping on the validation loss (and why, like the smooth nature, overfitting prevention, etc.), where as we store the best performing model based on the validation F1-score (due to this representing overall prediction performance).

Mention that every experiment is model selected using hold-out validation, and best model is chosen based on micro F1-score.

### 5.3.1 Architecture experiment

Every sub-section below should be a separate chapter.

Shortly mention what we do, what the goal is, what we want to figure out.

#### Architectures

Mention the different architectures trained, and at what hyperparameters they were trained over.

#### Datasets

Mention the different datasets used, and tested over.

### 5.3.2 Dataset generalization experiment

Shortly mention why, what, like in the previous experiment.

#### Architectures

Mention which architectures we now use, and why we chose them. (And hyperparameters)

#### Datasets

Now mention which datasets / combination of datasets we use. Mention how we now use zero-shot testing (and maybe why).

### 5.3.3 Ablation experiments?

#### Technique 1

#### Technique 2

#### Technique 3

# Chapter 6

# Results

## 6.1 Architecture experiment

Display a table of results, class-wise and micro-F1: Best archicecture per dataset is bolded.

| | Dataset 1 | Dataset 2 | Dataset 3 | Dataset 4 |
|---|---|---|---|---|
| CNN | 0.5 | | | |
| RNN | 0.4 | | | |
| Conv-RNN | 0.8 | | | |
| Conv-Attention | 0.9 | | | |
| Transformer | **0.95** | | | |

Display the results in a barplot, to easily capture well-performing models.

Also plot enough information to be able to conclude about overall performance of models, performance on rarer instruments, etc.

## 6.2 Dataset generalization experiment

Display a table of results, possibly both class-wise and micro-F1 (or maybe just micro-F1): Zero-shot tests have a grayed background, best zero-shot test are bolded.

| | Dataset 1 | Dataset 2 | Dataset 3 | Dataset 4 |
|---|---|---|---|---|
| Dataset 1 | 0.5 | 0.3 | | |
| Dataset 2 | 0.4 | 0.8 | | |
| Dataset 1+2 | 0.8 | 0.7 | | |
| Dataset 3 | **0.9** | 0.6 | | |
| Dataset 1+2+3 | 0.95 | 0.8 | | |
| Dataset 1+4 | 0.95 | **0.75** | | |
| Dataset 1+2+3+4 | 0.95 | 0.82 | | |

Display the results in a barplot, to easily capture well-performing models.

Also plot enough information to be able to conclude about overall performance of models, performance on rarer instruments, etc.

## 6.3 Ablation experiments

Display results and data to be able to conclude if techniques help training / give better end results.

I.e., do we converge faster? Do we converge to a better minimum?

Could plot some loss over epochs? Need to be thorough (or average) to ensure that gains/losses are due to technique (and not hyperparameter choice, etc.).

# List of Acronyms and Abbreviations

**ADT** Automatic Drum Transcription.

**AMT** Automatic Music Transcription.

**DFT** Discrete Fourier Tranform.

**DSC** Drum Sound Classification.

**DTD** Drum Transcription of Drum-only Recordings.

**DTM** Drum Transcription in the Presence of Melodic Instruments.

**DTP** Drum Transcription in the Presence of Additional Percussion.

**ENST-Drums** The ENST-Drums dataset.

**FFT** Fast Fourier Transform.

**MDB Drums** The MedleyDB Drums dataset.

**MIR** Music Information Retrieval.

**STFT** Short-time Fourier Transform.

# Bibliography

[1] *Fundamentals of Telephony.* United States, Department of the Army, 1953.
**URL:** `https://books.google.no/books?id=8nvJ6qvtdPUC`.

[2] Pras Amandine and Guastavino Catherine. Sampling rate discrimination: 44.1 khz vs. 88.2 khz. *Journal of the Audio Engineering Society*, (8101), may 2010.

[3] Rachel Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Pablo Bello. Medleydb sample, October 2014.
**URL:** `https://doi.org/10.5281/zenodo.1438309`.

[4] Pragnan Chakravorty. What is a signal? [lecture notes]. *IEEE Signal Processing Magazine*, 35(5):175–177, 2018. doi: 10.1109/MSP.2018.2832195.

[5] Sungkyun Chang, Emmanouil Benetos, Holger Kirchhoff, and Simon Dixon. Yourmt3+: Multi-instrument music transcription with enhanced transformer architectures and cross-dataset stem augmentation, 2024.
**URL:** `https://arxiv.org/abs/2407.04822`.

[6] James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19(90):297–301, 1965. ISSN 00255718, 10886842.
**URL:** `http://www.jstor.org/stable/2003354`.

[7] Olivier Gillet and Gaël Richard. Enst-drums: an extensive audio-visual database for drum signals processing, October 2006.
**URL:** `https://doi.org/10.5281/zenodo.7432188`.

[8] Carl Southall, Chih-Wei Wu, Alexander Lerch, and Jason Hockman. Mdb drums: An annotated subset of medleydb for automatic drum transcription. 2017.

[9] Gilbert Strang. Wavelet transforms versus fourier transforms, 1993.
**URL:** `https://arxiv.org/abs/math/9304214`.

[10] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio, 2016.
**URL:** `https://arxiv.org/abs/1609.03499`.

[11] Richard Vogl, Matthias Dorfer, Gerhard Widmer, and Peter Knees. Drum transcription via joint beat and drum modeling using convolutional recurrent neural networks. In *International Society for Music Information Retrieval Conference*, 2017.
**URL:** `https://api.semanticscholar.org/CorpusID:21314796`.

[12] Richard Vogl, Gerhard Widmer, and Peter Knees. Towards multi-instrument drum transcription, 2018.
**URL:** `https://arxiv.org/abs/1806.06676`.

[13] Chih-Wei Wu, Christian Dittmar, Carl Southall, Richard Vogl, Gerhard Widmer, Jason Hockman, Meinard Müller, and Alexander Lerch. A review of automatic drum transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(9):1457–1483, 2018. doi: 10.1109/TASLP.2018.2830113.

[14] Mickaël Zehren, Marco Alunno, and Paolo Bientinesi. High-quality and reproducible automatic drum transcription from crowdsourced data. *Signals*, 4(4):768–787, 2023. ISSN 2624-6120. doi: 10.3390/signals4040042.
**URL:** `https://www.mdpi.com/2624-6120/4/4/42`.

[15] Mickaël Zehren, Marco Alunno, and Paolo Bientinesi. Analyzing and reducing the synthetic-to-real transfer gap in music information retrieval: the task of automatic drum transcription, 2024.
**URL:** `https://arxiv.org/abs/2407.19823`.