Technische Universität Berlin
Institute for Speech and Communication
Audio Communication Group

Thomas Holz

Automatic Drum Transcription with Deep Neural Networks

Master's thesis in the program
"Audio Communication and Technology" (M. Sc.)

Submitted on December 23, 2021

Reviewer:

Prof. Dr. Stefan Weinzierl	–	Technische Universität Berlin
Dr. Andreas Arzt	–	Apple

Abstract

Automatic Drum Transcription with Deep Neural Networks

by Thomas HOLZ

The field of Music Information Retrieval (MIR) has gained a lot of importance over the last years. One important sub-field of MIR is the subject of Automatic Music Transcription (AMT) which focuses on extracting a musical score or a symbolic representation (e.g. MIDI) from an audio signal. This thesis explores the ability of deep neural networks to automatically transcribe drums – also known as Automatic Drum Transcription (ADT). In order to do so, this thesis proposes a deep neural network (DNN) architecture that is inspired by related work in the field of ADT. The backbone of this model is a convolutional recurrent neural network (CRNN) that is supposed to learn both spectral as well as structural patters. This model is compared to different deep neural network architectures, all of which try to extract latent feature representations from an audio signal and learn patterns to identify and distinguish between different drum instruments. The benchmark models are utilized to gauge the performance of the proposed model on evaluation datasets. The foundation of the training of the model is the publicly available dataset E-GMD introduced by Callender, Hawthorne, and Engel (2020) as it features 444.5 hours of labeled audio files.

This thesis demonstrates the importance of data in a supervised deep learning setting by showing peculiarities of E-GMD that can impact the performance of the proposed model negatively. Furthermore, a new dataset is created consisting of random drum sequences (RGDD) to show that adding new data to an already existing dataset can be a viable regularization approach. Additionally, it is shown that random data can serve as standalone datasets and even outperform models that have been trained on sequences that were played by humans. For this, the proposed model is evaluated in three different settings. The first setting sheds light on the performance of the proposed model in a drums-only scenario, that is audio files that contain multiple drum instruments. After this, the same model is evaluated on files that only contain single instrument hits which are here referred to as stems. Lastly, it is investigated if models that were trained on drums-only sequences can generalize well in full-mix settings where accompanying instruments like bass, synthesizer, guitar, or vocals are present. In each of the aforementioned scenarios it is shown that training a model to perform more fine grained instrument predictions and later group them to a coarser instrument grouping can improve the overall performance of the model. That means that a model that was trained to transcribe for example 7 instruments is later used to only transcribe 3 distinct classes by grouping the corresponding predictions.

Zusammenfassung

Das Gebiet des Music Information Retrieval (MIR) hat in den letzten Jahren stark an Bedeutung gewonnen. Ein wichtiges Teilgebiet von MIR ist die automatische Musiktranskription (AMT), die sich auf die Extraktion einer Partitur oder einer symbolischen Darstellung (z.B. MIDI) aus einem Audiosignal konzentriert. Diese Arbeit untersucht die Fähigkeit von neuronalen Netzen, Schlagzeug automatisch zu transkribieren - auch bekannt als Automatic Drum Transcription (ADT). Zu diesem Zweck wird in dieser Arbeit eine Architektur für neuronale Netze (DNN) vorgeschlagen, die von verwandten Arbeiten aus dem Gebiet der ADT inspiriert ist. Das Grundgerüst dieses Modells ist ein Convolutional Recurrent Neural Network (CRNN), das sowohl spektrale als auch strukturelle Muster lernen soll. Dieses Modell wird mit verschiedenen neuronalen Netzwerkarchitekturen verglichen, die alle versuchen, latente Merkmalsrepräsentationen aus einem Audiosignal zu extrahieren und Muster zu lernen, um verschiedene Schlagzeuginstrumente zu identifizieren und zu unterscheiden. Die Benchmark-Modelle werden verwendet, um die Ergebnisse des vorgeschlagenen Modells auf Evaluationsdatensätzen zu beurteilen. Die Grundlage für das Training des Modells ist der öffentlich zugängliche Datensatz E-GMD, der von Callender, Hawthorne, and Engel (2020) vorgestellt wurde und 444,5 Stunden annotierte Audiodateien enthält.

Diese Arbeit demonstriert die Bedeutung von Daten in einer überwachten Deep-Learning-Umgebung, indem sie Besonderheiten von E-GMD aufzeigt, die sich negativ auf die Ergebnisse des vorgeschlagenen Modells auswirken können. Außerdem wird ein neuer Datensatz erstellt, der aus zufälligen Schlagzeugsequenzen (RGDD) besteht, um zu zeigen, dass das Hinzufügen neuer Daten zu einem bereits vorhandenen Datensatz ein praktikabler Regularisierungsansatz sein kann. Darüber hinaus wird gezeigt, dass Zufallsdaten als eigenständige Datensätze dienen können und sogar Modelle übertreffen, die auf Sequenzen trainiert wurden, die von Menschen gespielt wurden. Zu diesem Zweck wird das vorgeschlagene Modell in drei verschiedenen Situationen evaluiert. Das erste Experiment beleuchtet die Ergebnisse des vorgeschlagenen Modells in einem reinen Schlagzeugszenario, d. h. Audiodateien, die mehrere Schlagzeuginstrumente enthalten. Danach wird dasselbe Modell für Dateien untersucht, die nur einzelne Instrumentenschläge eines einzigen Instruments enthalten, die hier als Stems bezeichnet werden. Schließlich wird untersucht, ob Modelle, die auf reinen Schlagzeugsequenzen trainiert wurden, auch in vollständigen Mischungen mit Begleitinstrumenten wie Bass, Synthesizer, Gitarre oder Gesang gute Ergebnisse erzielen können. In jedem der zuvor genannten Szenarien zeigt sich, dass das Training eines Modells für feinere Instrumentenprediktionen und deren spätere Gruppierung zu einer gröberen Instrumentengruppierung das Gesamtergebnis des Modells verbessern kann. Das bedeutet, dass ein Modell, das für die Transkription von beispielsweise 7 Instrumenten trainiert wurde, später durch Gruppierung der entsprechenden Prediktionen nur für die Transkription von 3 verschiedenen Klassen verwendet wird.

Acknowledgements

First and foremost, I would Iike to thank my supervisor Dr. Andreas Arzt for his outstanding support over the period of my internship. He enabled me to do things which I had not thought about doing before. Regardless of how trivial my questions or obstacles were, he was always there to help. Thank you for trusting me and giving me the freedom to explore different facets of ADT.

At the same time, I would like to thank Pierre Fournier who went out of his way to help me grow during my time at Apple and for giving me the opportunity to combine my passion for music with my Master's thesis while creating an incredible environment to work in. I genuinely enjoyed every day of working on your team.

I would also like to thank everyone on the ML / DSP Music Apps team for creating such a supportive and pleasant atmosphere. Their work has been truly inspiring. I especially want to thank Johannes Böhler for helping me with onset detection.

I would also like to thank Prof. Dr. Stefan Weinzierl his for constant support and consultation during my thesis.

Last but not least, I want to thank my family for believing in me and supporting me no matter what.

Contents

List of Figures

List of Tables

List of Abbreviations

ADT	**A**utomatic **D**rum **T**ranscription
AMT	**A**utomatic **M**usic**T**ranscription
BE	**Be**ll
BiGRU	**Bi**-directional **G**ated **R**ecurrent **U**nit
BiLSTM	**Bi**-directional **L**ong **S**hort-**t**erm **M**emory
BiRNN	**Bi**-directional **R**ecurrent **N**eural **N**etwork
BPM	**B**eats **P**er **M**inute
CNN	**C**onvolutional **N**eural **N**etwork
CRNN	**C**onvolutional **R**ecurrent **N**eural **N**etwork
CY	**Cy**mbal
DNN	**D**eep **N**eural **N**etwork
EDM	**E**lectronic **D**ance **M**usic
FFT	**F**ast **F**ourier **T**ransform
FN	**F**alse **N**egative
FT	**F**ourier **T**ransform
FP	**F**alse **P**ositive
GRU	**G**ated **R**ecurrent **U**nit
HH	**H**i-**H**at
HHM	**H**idden **M**arkov **M**odel
KD	**K**ick **D**rum
LDA	**L**inear **D**iscriminant **A**nalysis
LSTM	**L**ong **S**hort-**t**erm **M**emory
MFCC	**M**el **F**requency **C**epstral **C**oefficient
MIDI	**M**usical **I**nstrument **D**igital **I**nterface
MIR	**M**usic **I**nformation **R**etrieval
ML	**M**achine **L**earning
MLLR	**M**aximum **L**ikelihood **L**inear **R**egression
NN	**N**eural **N**etwork
NMF	**N**on-negative **M**atrix **F**actorization
PSA	**P**rior **S**ubspace **A**nalysis
RD	**R**i**d**e
RNN	**R**ecurrent **N**eural **N**etwork
SD	**S**nare **D**rum
STFT	**S**hort-**t**ime **F**ourier **T**ransform
SVL	**S**cramdisk **V**olume **F**iles
SVM	**S**upport **V**ector **M**achine
TN	**T**rue **N**egative
TP	**T**rue **P**ositive
TT	**T**om
WAV	**Wav**eform Audio File Format
XML	E**x**tensible **M**arkup **L**anguage
ZCR	**Z**ero **C**rossing **R**ate

List of Symbols

b	bias	
dB	decibel	dB
e	Euler's number	
E	error	
f	frequency	Hz
f_s	sampling frequency	Hz
j	imaginary unit	
η	learning rate	
ω	angular frequency	$\mathrm{rad\,s^{-1}}$
ϕ	activation function	

Chapter 1

Introduction

Today, the applications of machine learning algorithms are more versatile than ever before. One major field is the topic of image recognition as demonstrated by Krizhevsky, Sutskever, and Hinton (2012). There, deep neural networks (DNNs) are used to find patterns in images that set them apart from others, and hence try to classify them correctly according to their labels which is referred to as *supervised learning*. This fact paves the way for deep learning in the audio domain as the same classification techniques can be applied to audio. Here, the short-time Fourier transform (STFT) creates a visual representation or "image" of the audio signal which is often times the foundation for deep learning tasks related to audio. Nevertheless, visual representations of audio are just one option among others to work with in deep learning. For example, raw audio can also be used for this. The structure of DNNs is hierarchical in nature which is why they are well suited to explore patterns in data. Music information retrieval (MIR) focuses on exactly that in the realm of audio. It is an example of an interdisciplinary research field that combines statistics, signal processing, psycho-acoustics, and music theory to gain an in-depth view of the spectral composition of an audio signal over time so that it can be further analyzed or processed. *Automatic music transcription* (AMT) is a fundamental part of MIR (Benetos et al. (2019)). It is used to extract an automatically generated symbolic transcription from an audio signal by using deep neural networks while offering various different use-cases like *score following* (Li and Duan (2016)), audio to *score alignment* (Niedermayer and Widmer (2010)), *score-informed source separation* (Duan and Pardo (2011)), or *remixing* volume balances between individual drum instruments (Dittmar and Müller (2016)) to name a few. One sub-field of AMT is *automatic drum transcription* (ADT) which focuses on transcribing drums or percussion instruments from audio files. The word *transcription* is used in thesis to refer to the detection of drum onsets and their classification. Figure 1.1 shows a simplified high-level overview of ADT and the fundamental steps taken in this thesis to obtain an automatically generated drum transcription.

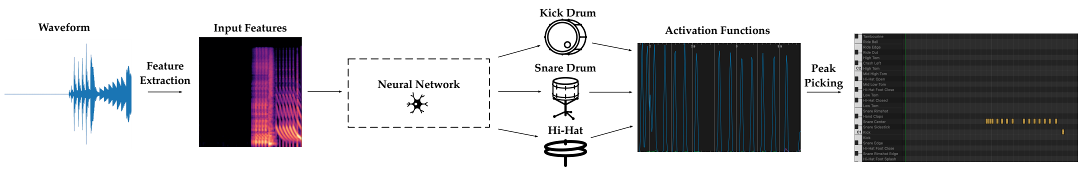


FIGURE 1.1: Exemplary building blocks of automatic drum transcription (ADT) based on the file *drummer1/session1/6_jazz-funk_116_fill_4-4_57.wav* from *E-GMD*. This 3-hit high-level overview depicts the main concepts that are used in this thesis to automatically generate a transcription from an audio signal.

The starting point of ADT in thesis is an audio file which is transformed into a spectrogram that serves as input features to a deep neural network. This audio file can either contain *drums-only* or other instruments / vocals in addition to drums (here referred to as *full-mixes*). This "image" of the audio contains both spectral and temporal information and lays the foundation for convolutional and recurrent blocks inside the neural network. In the next step, the neural network extracts low level features or latent representations with the aid of multiple hidden layers that allow the neural network to eventually make predictions about where a certain instrument is in the spectrogram. This is done frame-wise across the entire spectrogram. The outcome of this is an activation vector that represents the probabilities of where the neural networks predicts an instrument to be. Finally, in the post-processing steps a peak picking algorithm is applied to those probabilities so that a transcription of the remaining peaks can be created.

Motivation and Outline

The goal of this thesis is to thoroughly analyze state-of-the-art supervised deep learning techniques while proposing an own model architecture. Two benchmark architectures are compared with the proposed model which is evaluated on datasets that are commonly used in the ADT literature. This includes datasets that either contain *drums-only* mixes, which can be hits of single instruments or beats and fills of entire drum kits, or entire songs that feature a band consisting of multiple instruments and a singer. The latter of which are referred to as *full-mixes*. At the same time, the importance of data in deep learning is emphasized by a comprehensive analysis of the main training dataset called *E-GMD* introduced by Callender, Hawthorne, and Engel (2020). On top of this, a new dataset that contains randomly generated drum sequences from MIDI notes is introduced to investigate if randomly generated data are a legitimate foundation for ADT and to see if adding random sequences to already existing training datasets can have an impact on a model's performance. Finally, this thesis explores the impact of training a DNN to initially discriminate between more instruments than are finally used in the automatic drum transcription. Note however, that the focus of this thesis is the automatic transcription of drums while neglecting the velocity of events.

The use cases for automatically generated drum transcriptions are manifold. Vogl, Dorfer, and Knees (2016) name possible applications like the extraction of sheet music for music students, MIDI generation/re-synthesis, or utilizing the derived meta-information for other MIR related tasks like genre classification. At the same time, ADT allows for re-arranging drum sequences after a recording has already been finished.

This thesis is subdivided into 6 chapters. Following this introductory part is Chapter 2 which gives a high-level overview of the aspects of deep learning that are necessary to understand the main concepts of this thesis. It talks about the two main concepts used herein namely *convolutional neural networks* (CNNs) and *recurrent neural networks* (RNNs) while touching on the training and testing process of the DNN using an audio example. Chapter 2 is purposefully kept concise in order to put the main focus on the datasets, methods, and results presented in this thesis as well as to highlight the application of deep neural networks with regard to audio and music. Nevertheless, references for self-study and more in-depth explanations are given where needed.

After that, Chapter 3 gives an overview of the employed datasets used for training and evaluation purposes. In order to understand the results presented in this thesis, this chapter sheds light on characteristics like for example event-, genre-, and tempo-distribution as well as overall size and the creation of the datasets where applicable.

Chapter 4 talks about the entire processing pipeline starting with the introduction of the features used for training purposes. Following this is the introduction of the employed pre-processing steps including cutting and padding, annotation filtering, augmentation, and target computation. Afterwards, the proposed network architecture is presented and compared to two baseline networks to finally conclude the chapter by talking about post-processing steps as well as evaluation metrics used to gauge the overall performance of the proposed model.

Chapter 5 presents and discusses the results of various experiment settings and compares the outcome of the evaluation on the datasets presented in Chapter 3 of the proposed model to baseline models where possible. In the first section of this chapter, the performance of the proposed model is evaluated in a 3-, 7-, and 25-event setting focusing on transcribing drums from *drums-only* mixes. After that, the results of an explorative approach which investigates whether a model that was trained on drums-only mixes also performs well on drum *stems* are presented and discussed. Lastly, the proposed model which is trained on drums-only mixes is applied to full-mixes in order to investigate its ability to generalize in a setting it was not trained in.

Finally, Chapter 6 concludes this thesis by reflecting on the outcome of the investigate experiments. Key findings are summarized and evaluated while giving an outlook on future work as well as talking about open topics.

Chapter 2

Background & Related Work

This chapter first talks about the fundamentals of deep learning and lays the foundation of background knowledge that is later necessary to better understand the architecture choices in Chapter 4. It serves as a concise high-level overview of the fundamental concepts of deep learning without going into too much detail. A more complete and theoretical view on individual parts can be found in Goodfellow, Bengio, and Courville (2016). The concepts described here are underlined by a clear focus on audio. However, this does not mean that they are not applicable or transferable to similar tasks or domains. As mentioned in Chapter 1, this section will not cover an in-depth view of the mathematical background on machine learning either. However, a great resource for this is given by Deisenroth, Faisal, and Ong (2020). Finally, this chapter concludes by presenting related work as well as the current state of research.

2.1 Deep Learning

Finding patterns in data is at the core of *machine learning* (ML) algorithms. Conventional ML approaches are applied to structured data for which the underlying patterns have to be understood so that features can be extracted in order for the data to be further processed or classified under a predefined set of rules. However, this approach presupposes knowledge about the data and its structure which is not always given. With regard to large unstructured datasets, for which a manual feature extraction in a classification scenario increases in difficulty, this problem becomes even more pronounced. *Deep learning* tries to solve this by combining both the feature extraction as well as the classification by enabling an algorithm to "learn" the rules to classify data by optimizing an error function. The field of deep learning is a sub-field of machine learning and employs algorithms that were originally inspired by the function and anatomy of the brain (McCulloch and Pitts (1943); Hebb (1949)) which is why deep learning algorithms are used to create so-called *artificial neural networks*. Deep learning is a method of representation learning that enables models that consist of an *input layer*, multiple processing layers (also called *hidden-layers*), as well as an *output layer* to extract abstract feature representations that are learned from data during the training process. That way, multiple layers of abstractions are created which suppress irrelevant features as well as propagate important ones for a given problem. Besides applications in the realm of audio, *deep neural networks* (DNNs) are employed in a broad spectrum of different use cases like computer vision (Krizhevsky, Sutskever, and Hinton (2012)), medicine (Esteva et al. (2021)), or natural language processing (Bahdanau, Cho, and Bengio (2015)). The term *deep* refers to the fact that the neural network consists of multiple hidden layers between input and output layers. An example of a deep neural network with multiple layers is shown in Figure 2.1.

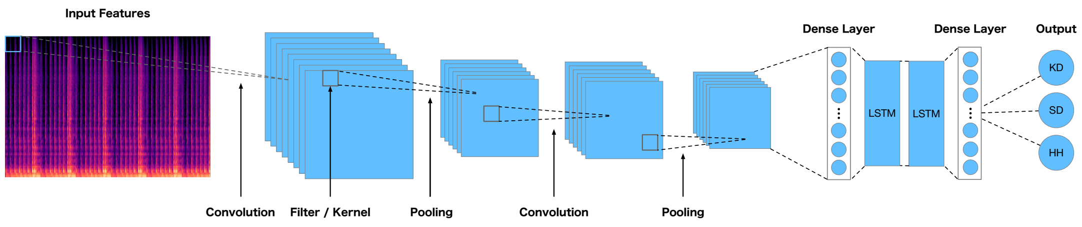


FIGURE 2.1: Exemplary deep neural network that has 3 output prediction nodes for kick drum (KD), snare drum (SD), and hi-hat (HH). This DNN is a reduced and abstracted version of the architecture of the proposed model.

This approach is motivated by the hierarchical structure of many audio signals as shown by Burred and Lerch (2004). On the other hand, typical machine learning algorithms used to require hand-crafted feature extractors that only then could cope with raw input data (LeCun, Bengio, and Hinton (2015)). Deep learning is able to circumvent this problem by enabling neural networks to learn the best feature extractions (latent features) based on the given data during training with the aid of hidden-layers. The types of hidden-layers that are used in this thesis are *dense-*, *convolutional-*, *pooling-*, *recurrent-*, and *dropout-layers* which are explained in the following part.

Dense Layer

The basis of a *dense layer* as shown in Figure 2.1 is an *artificial neuron* where multiple instances of them arranged in a layer constitute a dense layer with a width of N, where N is the number of neurons. The artificial neuron can be mathematically described as follows:

$$y = \phi(\sum_{i=1}^{n} w_i x_i + b) = \phi(\mathbf{w}^T \mathbf{x} + b) \tag{2.1}$$

where:

ϕ = activation function
w = weights
x = input vector
b = bias

Equation 2.1 shows that the output y of a single artificial neuron is computed by passing the dot product of a weight vector $\mathbf{w}$ and an input vector $\mathbf{x}$ plus an additional bias term b through a non-linear activation function $\phi(\cdot)$ to enable the deep neural network to learn complex functional mappings from data. The parameters b and $\mathbf{w}$ are the ones that are adjusted or "learned" during the training process. In this thesis, the *SELU* activation function, introduced by Klambauer et al. (2017), is used in both *convolutional-* and *dense layers* due to its self-normalizing characteristics. Therefore, no additional batch-normalization layers (Ioffe and Szegedy (2015)) are employed in the proposed model. On top of that, *SELU* solves the vanishing gradient problem as described by Hochreiter (1998) for deep neural networks which may cause the training process to stop. Another commonly used activation function employed to solve the vanishing gradient problem is the *ReLU* function which introduced by Jarrett et al. (2009). Besides that, the *sigmoid* activation function is used in the final dense layers of the DNN as it outputs a probability that is useful for binary or *multi-class multi-label* classification tasks. The latter of which describes the scenario for ADT where the output is not mutually exclusive meaning that there can be for example two different instruments occurring at the same time. Since the sum of the predicted outputs does not have to add up to 1, it is an indicator of how certain the neural network is about classifying one or multiple instruments in a given frame.

Convolutional Layer

A particularly important component in deep learning tasks is the *convolutional layer*. Research of the visual cortex of a cat by Hubel and Wiesel (1959) as well as Hubel and Wiesel (1962) has shown that depending on the location and orientation of an input the cells in the cortex fire differently and therefore motivating the birth of *convolutional neural networks* (CNNs). Convolutional layers are the building blocks of CNNs as they enable deep neural networks to recognize patterns in images. LeCun et al. (1998) paved the way for CNNs by introducing *LeNet-5*, an artificial neural network that is able to recognize digits. CNNs have a local receptive field that allows the network to extract local relations like for example edges. The name *convolutional neural network* stems from the mathematical operation called *convolution* which describes an overlap of one function over another.

$$x * w = \sum_{-\infty}^{\infty} x[m] \cdot w[n-m] \tag{2.2}$$

Equation 2.2 demonstrates the discrete version of the convolution for 1-dimensional problems. However, in deep learning the input features as well as the kernels are usually multi-dimensional arrays. Therefore, when adopting the concept of convolution to artificial neural networks Equations 2.1 and 2.2 are combined. The input of a convolutional layer is a grid of pixels which is then convolved with a *filter* or *kernel* that is "learned" during the training process based on the optimization of a loss function. An exemplary convolution is shown in Figure 2.2.

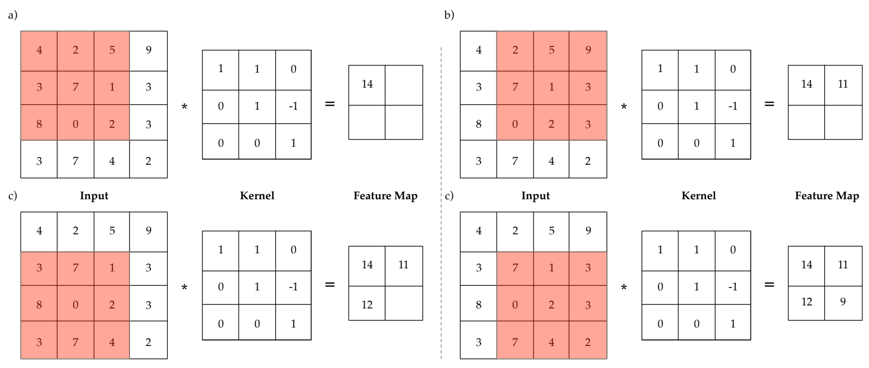


FIGURE 2.2: Convolution of a 4x4 input vector with a 3x3 kernel and a stride of 1. See Dumoulin and Visin (2018) for a detailed description on convolution arithmetic.

Convolutional layers are typically followed by *pooling layers* which help reduce the dimensions of the previously extracted feature map while reducing the memory footprint of the DNN at the same time. Two commonly employed pooling techniques are *average-* and *max-pooling*, the first of which takes the average while the latter of which takes maximum value of a filter that is applied to a feature map. Gholamalinezhad and Khosravi (2020) give a more detailed overview of the variety of pooling methods.

Recurrent Layer

Sequential data like for example text or audio are usually processed using *recurrent neural networks* (RNNs) for CNNs struggle to model the relationship between past and present states as they assume an independence of inputs and outputs. RNNs are capable of doing exactly that by forwarding a previous cell state to the next cell and therefore providing the neural network some sort of memory. In case of music for example, a chord that was played before can influence the chord that occurs next. In this unidirectional scenario, only past events influence the determination of the output of a given sequence. Figure 2.3 demonstrates how a cell state at point t is influenced by the cell state.

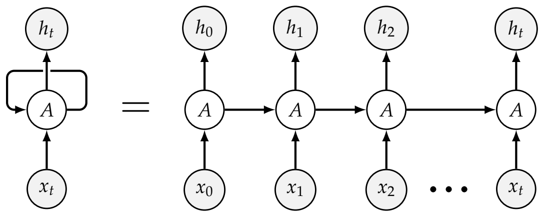


FIGURE 2.3: Schematic representation of an unrolled recurrent neural network (RNN).

Similar to CNNs, the calculated error is backpropagated into the network to update the weights and biases of the DNN. However, in case of simple RNNs the error is summed at each time step for they share parameters across each layer. This circumstance can cause *vanishing* or *exploding gradients* during the optimization process and therefore eventually stop the training. This problem motivated Hochreiter and Schmidhuber (1997) to introduce a way to store information over a longer period of time called *long short-term memory* (LSTM) by using so-called *gates* in a recurrent cell as shown in Figure 2.4.

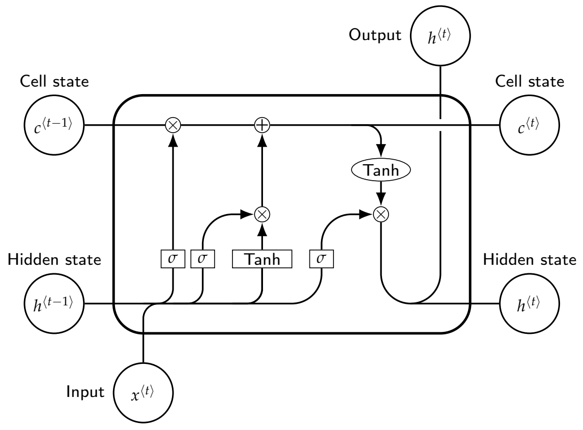


FIGURE 2.4: Schematic representation of an long-short term memory (LSTM) cell.

The first gate in an LSTM cell is the *forget gate* which controls what information in the cell state to forget based on new information at time step t. For this, the hidden state of the previous cell is processed with the concatenated input vector x by the first sigmoid function and then multiplied with the cell state. In a following processing step, the *input gate* determines what additional information to encode into the cell state. Hence, the hidden state and the input are transformed by the adjacent sigmoid and tanh functions to be then added to the cell state. Finally, the *output gate* multiplies the final cell state at time step t (previously passed through a tanh function) with the concatenated and sigmoid processed hidden state and input vector to result in the new hidden state or output of the LSTM cell.

The concept of introducing some kind of memory of past time steps can also be extended to time steps in the future which is especially useful for input data like music as it enables the DNN to learn from context that lies in the future and not only in the past. This type of architecture is called *bidirectional recurrent neural network* (BiRNN) and was introduced by Schuster and Paliwal (1997). In this thesis, the proposed model introduced in Section 4.3 features 3 layers of bidirectional LSTM layers.

Dropout Layer

Overfitting has been a widely discussed topic in the literature (Roelofs et al. (2019); Salman and Liu (2019)) and describes the problem of DNNs performing well on the training data but poorly on unseen data and therefore lacking the ability to generalize. One commonly used technique established by Srivastava et al. (2014) to prevent a model from overfitting is *dropout*. Dropout disables neurons inside the neural network by randomly multiplying them with zero which forces the network to no longer rely on certain nodes and look for different paths inside the trained model. The concept of this approach is demonstrated in Figure 2.5.

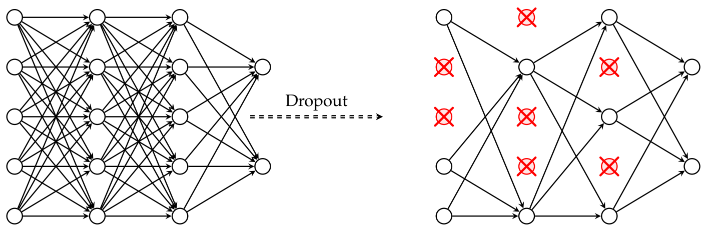


FIGURE 2.5: A comparison between a NN with (right) and without (left) dropout. This is a typical approach to reduce overfitting.

In this thesis, regular dropout is used after each max-pooling layer of a convolutional stack as well as after the last bi-directional LSTM layer. In contrast to regular dropout, which sets individual bins of the spectrogram to zero with a predefined probability when used in a convolutional stack, Tompson et al. (2015) introduced *spatial dropout* which drops an entire 2D feature map instead of individual bins of the spectrogram. Contrary to intuition, spatial dropout did not show an advantage over regular dropout in preliminary investigations for this thesis.

However, implementing dropout is only one technique to regularize the DNN among others (Benning and Burger (2018); Jacques and Roebel (2019); Thakkar and Lohiya (2021); Jain et al. (2021)). Additional regularization techniques used in this thesis are augmenting training data as described in Section 4.2 or increasing the number of training instances by adding the dataset presented in Section 3.2.

Supervised Learning

The form of deep learning that is used in this thesis is called *supervised learning* which means that for every drum event in a dataset there is a corresponding label (also referred to as *ground truth* or *annotations*). The basic concept behind supervised learning is visualized in Figure 2.6.

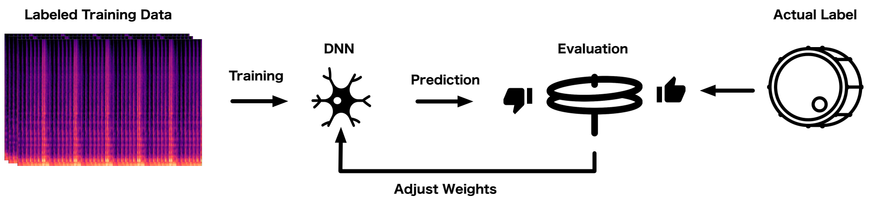


FIGURE 2.6: Schematics of a supervised deep learning algorithm based on audio data.

The first step in a supervised learning algorithm is to feed labeled training data to the deep neural network. In case of this thesis, it is a 12 s log-mel magnitude spectrogram which is computed prior to the training and then propagated through multiple layers of the network. The model then outputs a vector of predictions (*activations vector*) for each individual instrument it is trained to transcribe. This thesis investigates multiple classification scenarios, namely classifying 3, 7, and 25 different instruments. A *loss-* or *error-function*, here *binary cross-entropy loss*, computes the error between the predicted activations vector and the ground truth vector. It is important to note, that the different drum instrument classes are predicted independently of each other because multiple instruments can occur simultaneously. The parameters that are trained and adjusted after each iteration of this iterative training process are the *weights* or *filters* which are ultimately "learned" by the deep neural network to minimize the aforementioned error. The model adjusts its weights by computing a gradient vector by employing *backpropagation* as it enables the neural network to propagate the error back into the model and therefore improve its performance based on the calculated error. The foundations of this optimization technique were laid by Kelley (1960) and Bryson (1961) while the first implementation of backpropagation in neural networks goes back to Rumelhart, Hinton, and Williams (1986). It has since then been the foundation of optimizing the parameters inside a deep neural network. Equation 2.3 introduces the main concept behind backpropagation.

$$\mathbf{W}|_t = \mathbf{W}|_{t-1} - \eta \left.\frac{\partial E}{\partial \mathbf{W}}\right|_{t-1} \tag{2.3}$$

where:

η	= learning rate
$\mathbf{W}$	= weights & biases
E	= error / loss

The *learning rate*, which determines the size of the gradient descent steps in order to minimize the error, is among the most important parameters to tune with respect to training a DNN. Smith (2017) introduced the so-called *cyclical learning rates* to circumvent the problem of finding the right setting for this hyper-parameter. The optimization algorithm employed together with the aforementioned cyclical learning rates in this thesis is *Adam* which was introduced by Kingma and Ba (2015).

More information on gradient descent based optimization algorithms is given by Ruder (2016).

Testing
A test set of unseen data is fed to the deep neural network after the training process is completed to evaluate the performance of the trained model. This final step gives indication of how well the model can generalize on data that the model has not seen before. It is important to note that in supervised learning a deep neural network cannot generalize well on data that was not present in the training dataset in any way, meaning that predicting for example a tom in an audio file when there are only kick drum (KD), snare drum (SD), and hi-hat (HH) events in the training data will not yield promising results. This fact indicates that data and its variance is a key factor when it comes to deep learning and generalizability.

2.2 Related Work

Before the advent of deep neural networks, different approaches to tackle ADT tasks were applied. One of the very first steps in that direction was undertaken by Schloss (1985) with a focus on modeling attack characteristics by implementing an automatic slope detector based on the envelope of an audio signal. Once the onsets of the drums are identified, an exponential decay constant is determined to discriminate between either dampened or undamped strokes so that the pitch of the drum can be figured out.

However, not until the work of Gouyon, Pachet, and Delerue (2000) did ADT experience a renaissance. This time, the approach was based on using the *zero-crossing rate* (ZRC) to classify percussive sounds as either snare drum-like or bass drum-like. They have found that the zero-crossing rate of the decay region is a proper feature to discriminate the two aforementioned classes in a clean, non-noisy setting.

Later work focused more on *prior subspace analysis* (PSA) as for example demonstrated by Fitzgerald, Lawlor, and Coyle (2003). Their technique showed that when prior knowledge about the sources in the audio in the form prior frequency subspace is available, the results can be improved when compared to *independent subspace analysis* (ISA) as presented by Fitzgerald, Coyle, and Lawlor (2002).

One year after that, Van Steelant et al. (2004) used *support vector machines* (SVM) to classify kick and snare drums in audio signals. This method serves as a supervised learning algorithm that is applied in binary classification tasks. Multi-dimensional feature vectors of those two classes are computed to then calculate a *hyperplane* or *decision boundary* so that in the best case scenario all instances belonging to the same class (either -1 or 1) are on the same side of the hyperplane. The vectors with the smallest distance to this hyperplane are called *support vectors*. Note, that this classification is optimized by maximizing the distance or *margin* between both support vectors and the hyperplane.

In contrast to that, Paulus and Virtanen (2005) apply an unsupervised approach to transcribe drums that combines both PSA as well as *non-negative matrix factorization* (NMF), the latter of which extracts two non-negative matrices that encode the information of both the average spectral energy distribution (source spectra) of an instrument as well as the activation or probability of each instrument per time frame of the magnitude spectrogram of the audio. The multiplication of these two matrices is an approximation of the input spectrogram. This separation process therefore estimates the proportion or gain of an instrument contributing to the overall signal at each time frame. The divergence between the prediction and the observed spectrum is minimized in an iterative manner. Finally, after normalizing, compressing, differentiation, low-pass filtering, and peak-picking the resulting onsets above a predefined threshold are extracted. Through this method they were able to surpass the performance of SVMs and PSAs.

Another approach was used by Paulus and Klapuri (2009). Here, hidden Markov models (HMMs) are used to transcribe drums from polyphonic music. For this, the input audio is first processed by *sinusoids-plus-residual modelling* to suppress tonal spectral components that are likely to be not part of drums. After that, *mel-frequency cepstral coefficients* (MFCCs) are extracted from the audio as input features to the HMMs to be then reduced in dimensionality by applying a *linear discriminant analysis* (LDA). Following this is a *maximum likelihood linear regression* (MLLR) to adapt the HMMs. Finally, applying the Viterbi algorithm yields the predicted drum transcription based on the state transitions of the HMMs.

Kaliakatsos-Papakostas et al. (2012) proposed a method to transcribe drums in real-time using amplifiers as well as band-pass filters that are estimated in an iterative process to determine a characteristic frequency response of a drum onset. Contrary to defined frequency ranges as suggested by Tzanetakis, Kapur, and McWalter (2005), these optimized filter ranges allow for a more nuanced and appropriate range to work in. In order to determine if a certain drum is played, the amplitude responses of these filters are observed and if a predefined threshold is exceeded, the corresponding drum is detected.

While this thesis tries to give an overview of techniques to transcribe drums prior to the advent of deep neural networks, only a selection can be given. However, Wu et al. (2018) give a more detailed overview of the work that was done prior to the first approaches towards using deep neural networks like the one by Gajhede, Beck, and Purwins (2016).

Current state-of-the-art ADT models have a few things in common. Firstly, they calculate an activation vector for each instrument that is under investigation. These activation vectors serve as pseudo-probabilities for each investigated frame of the audio signal indicating whether or not a note onset is present. They are necessary to compare them to the target activation vectors to eventually compute the error between both. With the aid of a peak-picking algorithm during post-processing, the final model predictions can be extracted. Secondly, it is common to have as little processing steps as possible, and finally, they use a gradient-descent-based approach for the optimization of the training (Vogl (2018)).

Unlike Gajhede, Beck, and Purwins (2016), Vogl, Dorfer, and Knees (2016) did not use CNNs to solve ADT tasks but employed four different recurrent neural networks (RNNs) since they are well suited to process sequential data as shown by Böck, Krebs, and Widmer (2016). Vogl, Dorfer, and Knees (2016) focus on predicting three classes of drums, namely KD, SD, and HH, while showing the best evaluation scores for the RNN containing a time-shift parameter. Vogl, Dorfer, and Knees (2017) further underline the successful implementation of RNNs with an additional label time-shift (tsRNN). In this case, Vogl, Dorfer, and Knees (2017) show that their model is superior to other models like the ones from Wu and Lerch (2015) or Southall, Stables, and Hockman (2016) that use either partially fixed NMF or bi-directional RNNs when transcribing drums from polyphonic music. Following the work on CNNs and RNNs is the work by Vogl, Widmer, and Knees (2018) who put emphasis on transcribing multiple instruments of a drum set while at same time showing that convolutional recurrent neural networks (CRNNs) deliver the best results for almost every publicly available dataset regardless of the number of drum instruments to be transcribed. This approach combines the properties of CNNs that are capable of extracting local dependencies in the spectrogram whereas RNNs can capture temporal long-term dependencies to some extent. Ishizuka, Nishikimi, and Yoshii (2021) try to further improve this by introducing a self-attention mechanism as transfer learning in their transcription model which is based on an encoder-decoder model.

All of the abovementioned approaches are supervised in nature which inherently require large labeled datasets to work with. One of the first steps towards working with unlabeled data with regard to ADT was done by Wu and Lerch (2017) where they show that labels that were generated by a teacher model can indeed be used by a student model and therefore have the student outperform the teacher. Approaches like the one proposed by Wang et al. (2020) introduce a *semi-supervised* method called *few-shot learning* with minimal human input that is able to recognize percussive instrument classes that are not present during training. On the other hand, Choi and Cho (2019) utilized a complete unsupervised approach with which they are able to score competitive results on a publicly available dataset. These methods are a promising outlook because a bottleneck of supervised methods is the fact that large amounts of labeled data are either difficult to acquire or time-consuming to create.

Table 2.1 gives an overview of the selection of related work presented in this section.

TABLE 2.1: A selection of related work on ADT presented in this thesis. A more in-depth overview is given by Wu et al. (2018).

Author(s)	**Approach**
Schloss (1985)	Modeling attack characteristics
Gouyon, Pachet, and Delerue (2000)	ZCR
Fitzgerald, Coyle, and Lawlor (2002)	ISA
Fitzgerald, Lawlor, and Coyle (2003)	PSA
Van Steelant et al. (2004)	SVM
Paulus and Virtanen (2005)	NMF
Paulus and Klapuri (2009)	HMM
Kaliakatsos-Papakostas et al. (2012)	Band-pass filtering
Gajhede, Beck, and Purwins (2016)	CNN
Vogl, Dorfer, and Knees (2016)	RNN
Southall, Stables, and Hockman (2016)	BiRNN
Wu and Lerch (2017)	Student-teacher learning
Vogl, Widmer, and Knees (2018)	CRNN
Choi and Cho (2019)	CRNN
Wang et al. (2020)	Few-shot learning
Ishizuka, Nishikimi, and Yoshii (2021)	Self-attention

Chapter 3

Datasets

As mentioned in Chapter 2, data is key to deep learning. Therefore, a wide selection of different datasets are used in this thesis to either train a DNN or evaluate its performance and its capability to generalize on unseen data respectively. The characteristics of each dataset are highlighted which includes for example the number of files and the musical genre they feature, the tempo where information was provided, as well as instrument complexity. The latter refers to the number of different instrument classes or event types that are present in the dataset. On top of that, a new dataset that comprises randomly generated single event sequences and random full-mixes is introduced that was created to tackle generalization issues.

3.1 E-GMD

In this thesis, the *Expanded Groove MIDI Dataset* (E-GMD)[1] is used for training purposes of the model that is presented in Section 4.3. It was introduced by Callender, Hawthorne, and Engel (2020) and is an expansion of the Groove MIDI Dataset (GMD)[2] by Gillick et al. (2019). It is the first drum transcription dataset that features human drum performances with velocity annotations, however, they are not used in this thesis as the focus solely lies on classifying drum events. Nevertheless, this topic should be investigated in future work. The dataset consists of audio recordings and their corresponding annotations in MIDI format. The dataset was created using an electronic drum set[3] featuring 43 different drum kit sounds ranging from electronic to acoustic drum kits, all of which are present in the train, test, and validation split.

TABLE 3.1: Average duration of *Beats* (47.1 %) and *Fills* (52.9 %) of E-GMD.

Type	**# Files**	**∅ in seconds**
Beats	18,576	82
Fills	26,961	2.85
Total	45,537	35

Generally, this dataset is a *drums-only* dataset which means that there are no instruments other than drums present in the recordings. The recordings were done at 44.1 kHz, 24 bit and were aligned within 2 ms of the original MIDI files. The dataset distinguishes between *Beats* and *Fills* with *Beats* referring to longer musical phrases

[1]https://magenta.tensorflow.org/datasets/e-gmd, last accessed July 16, 2021
[2]https://magenta.tensorflow.org/datasets/groove, last accessed July 16, 2021
[3]https://www.roland.com/us/products/td-17/, last accessed July 16, 2021

and *Fills* being short sequences that are usually used in music to transition from one part of the song to another which is underlined by their average file duration shown in Table 3.1.

An important aspect about data in deep learning is variance. Table 3.2 gives on overview of the distribution of sequences across train, test, and validation splits of E-GMD and their duration in hours.

TABLE 3.2: An overview of the E-GMD splits and their duration according to Table 2 by Callender, Hawthorne, and Engel (2020). It indicates that although the number of sequences/files is large, the actual number of unique sequences is low (2.3 %).

Split	**# Unique Sequences**	**# Total Sequences**	**Duration (hours)**
Train	819	35,217	341.4
Test	123	5,289	50.9
Validation	117	5,031	52.2
Total	1,059	45,537	444.5

It becomes obvious that although the dataset features a large amount of total sequences the number of unique ones only amounts to roughly 2.3 % across every split.

In addition to the variety of 43 different drum kit sounds, there is also variety with regard to genres which can be seen in Figure 3.1. There, a simplified overview of the genre distribution is shown for the sake of clarity. *Rock* is the most represented genre with a share of 28.7 % among all 17 that are featured in the dataset. The least represented genre is *Middle Eastern* with a share of 0.1 %.

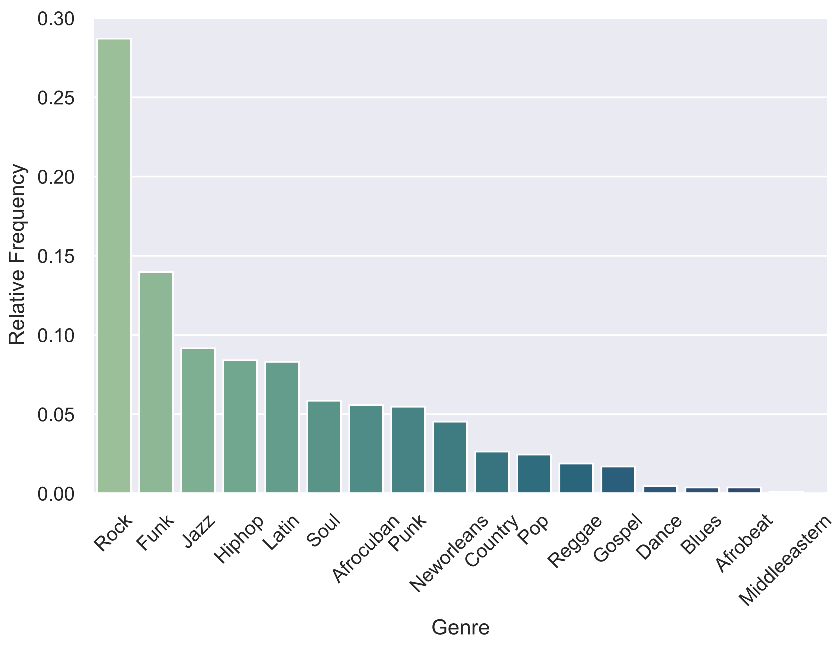


FIGURE 3.1: Coarse genre distribution of E-GMD with *Rock* being the main genre (28.7 %). For this, the first genre of an audio file name was taken into consideration.

Figure 3.1 shows a clear imbalance among genres which might already be indicator for poor performance of underrepresented genres as good classification can only be achieved with a balanced representation of all classes (Kumar et al. (2021)).

Similar to the genres, there is also an imbalance with regard to tempo. The span of the tempo of the sequences in the dataset range from 50 to 290 BPM as can be seen in Figure 3.2. Tempos over 200 BPM were grouped into the category >200 for visual purposes.

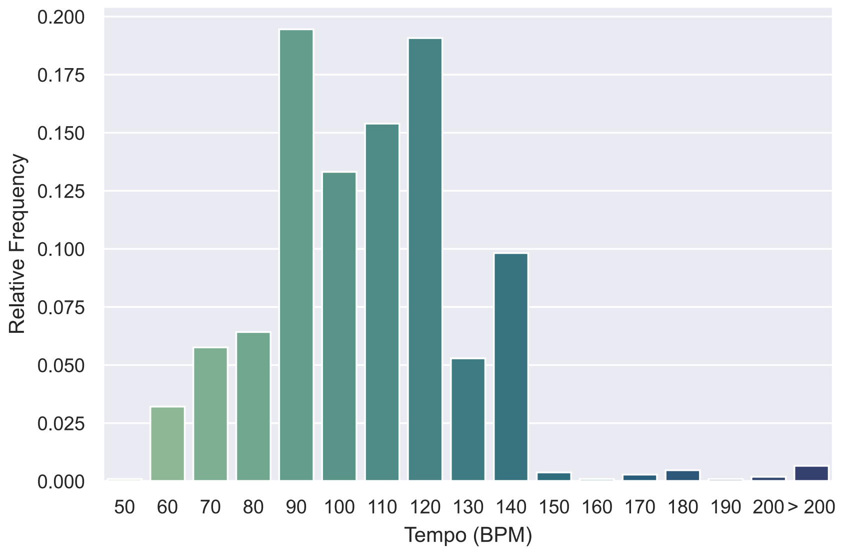


FIGURE 3.2: Absolute tempo distribution of E-GMD grouped into steps of 10 BPM.

The majority of sequences (19.5 %) display an overall tempo of around 90 BPM whereas the minority of sequences (0.1 %) feature a tempo of 43, 160, and 190 BPM. Overall, this Figure 3.2 underlines an emphasis on lower to mid-tempo ranges. All of the abovementioned meta-information on *Beats*, *Fills*, *splits*, *genres*, and *tempo* are provided for each individual audio file in a `csv` file that comes with the dataset.

As mentioned in the introductory statement of the chapter, datasets can have different levels of instrument complexity. In case of E-GMD, Callender, Hawthorne, and Engel (2020) distinguish between 3 different levels of complexity. The most complex and therefore finest level of complexity comprises 25 different event types or classes. Here, every single instrument of the drum set represents its own individual class consisting for example of different kinds of snare drums (SD) and toms (TT). The next coarser level of complexity groups individual instruments into their main class. This means that for example all types of hi-hats (HH) are grouped under the umbrella term (HH) and therefore introduces the first level of information loss with regard to distinguishing between each individual instrument. At this point, a model trained on this grouping can for example no longer distinguish between an open and a closed HH. The last grouping the authors make is to further summarize the remaining classes (KD, SD, TT, HH, CY, RD, BE) into just 3 classes (KD, SD, HH). Note, however, that only the 3 and 7 class ADT scenarios are investigated by Callender, Hawthorne, and Engel (2020) while all scenarios including the 25 classes are investigated in this thesis. Table 3.3 gives a more granular perspective on the groupings of the different event types that are present in E-GMD.

TABLE 3.3: Instrument complexity and grouping of E-GMD according to Callender, Hawthorne, and Engel (2020).

Instrument	25 Hits	7 Hits	3 Hits
Kick Drum	KD	KD	KD
Snare Drum Head	SDH	SD	SD
Snare Drum Rimshot	SDR	SD	SD
Snare Cross-Stick	SDX	SD	SD
Clap	CL	SD	SD
Tom 1	T1	TT	SD
Tom 1 Rim	T1R	TT	SD
Tom 2	T2	TT	SD
Tom 2 Rim	T2R	TT	SD
Tom 3	T3	TT	SD
Tom 3 Rim	T3R	TT	SD
Hi-Hat Open Bow	HOB	HH	HH
Hi-Hat Open Edge	HOE	HH	HH
Hi-Hat Closed Bow	HCB	HH	HH
Hi-Hat Closed Edge	HCE	HH	HH
Hi-Hat Pedal	HP	HH	HH
Tambourine	TB	HH	HH
Crash 1 Bow	C1B	CY	HH
Crash 1 Edge	C1E	CY	HH
Crash 2 Bow	C2B	CY	HH
Crash 2 Edge	C2E	CY	HH
Ride Bow	RBO	RD	HH
Ride Edge	RED	RD	HH
Ride Bell	REB	BE	HH
Cowbell	COW	BE	HH

Note, however, that the mapping of MIDI pitches in E-GMD does not follow the general MIDI mapping[4] but the Roland Mapping[5] which makes a potential re-synthesis of the dataset cumbersome.

To better understand the aforementioned instrument groupings and levels of complexity, it is crucial to take a look at the distribution of instruments across all groupings. Figure 3.3 shows the relative frequencies of each grouping across all splits. In the top row, the simplest scenario features only 3 events, namely kick-drum (KD), hi-hat (HH), and snare drum (SD). Here, the share of each instrument in the individual split is roughly the same. However, it also shows that the 3 different event classes are imbalanced when compared to each other with KD being underrepresented. This imbalance in the dataset can also be seen in the 7 event (middle row) and 25 event (bottom row) groupings. This circumstance is not ideal as it might result in a biased DNN by favoring majority classes as described by Johnson and

[4]`https://www.midi.org/specifications-old/item/gm-level-1-sound-set`, last accessed July 19, 2021

[5]`https://rolandus.zendesk.com/hc/en-us/articles/360005173411-TD-17-Default-Factory-MIDI-Note-Map`, last accessed July 19, 2021

Khoshgoftaar (2019), but it can be counteracted to some degree by applying different loss weights in the model (Fernando and Tsokos (2021)). However, this approach cannot fix major classes imbalances as shown by Cartwright and Bello (2018). A more effective approach would be to over-sample unrepresented instrument classes as for example demonstrated by Mohammed, Rawashdeh, and Abdullah (2020) as well as Shaikh, Changan, and Malik (2021).

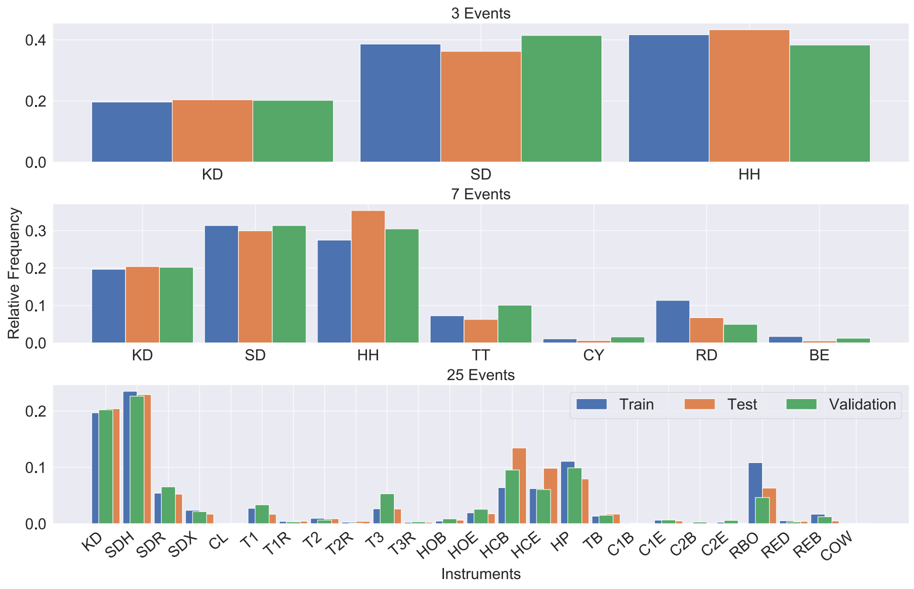


FIGURE 3.3: Relative event distributions of E-GMD among train, test, and validation split. The total number of events is 14,340,500.

In order to be able to compare to the results of this thesis to the results of Callender, Hawthorne, and Engel (2020), the problem of class imbalance is left unchanged.

3.1.1 E-GMD-Clean

As E-GMD serves as the main training dataset for the models in thesis, further analyses were done. They show, however, that the dataset is not free of flaws which is why a cleaned version of E-GMD is presented in this thesis to investigate what the impact of the irregularities might be on the overall performance. For example, there are 43 audio files that are empty and therefore contain no signal even though their MIDI counterpart contains events. These files, all of of which are part of the training splits, would degrade the training process, hence, they are removed from the dataset. An overview of these files is given in Appendix A.1. On top of that, there are a total of 496 MIDI files that contain events occurring after the end of their audio file counterparts. These events are removed from the MIDI files during the pre-processing procedure so that both the audio and MIDI file have the same length to ensure a proper training. A full list of these files can be found in Appendix A.2. Finally, running an onset detection algorithm over all audio files in E-GMD shows that when comparing the detected onset times to the ground truth onset times a wide range of deviation in the alignment between audio and MIDI files ranging from 0 ms up to 112 ms can be detected. This observation was visually confirmed by manually inspecting a selection of files. The average deviation between audio and MIDI files was determined by calculating the median of the 10 onsets in the audio that displayed the highest energy and the onset time of the nearest annotation. An example of the erroneous alignment between audio and MIDI files is presented in Figure 3.4. The distribution of the described discrepancy is shown Figure 3.5.

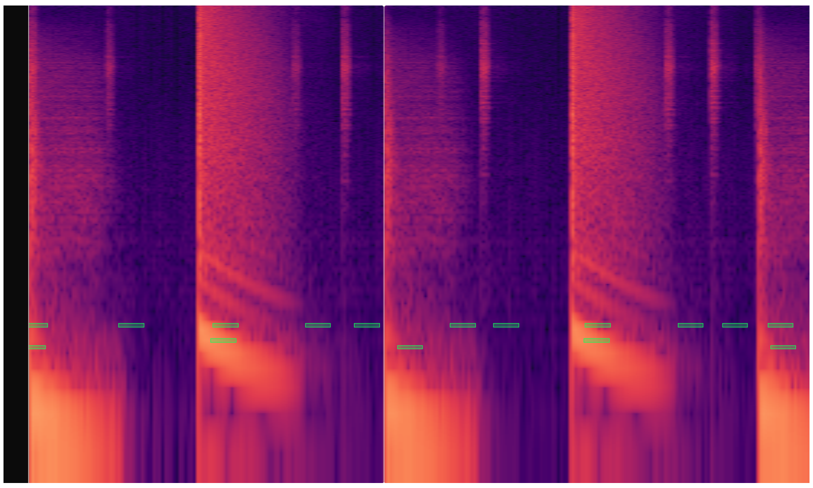


FIGURE 3.4: Example of misalignment between audio and MIDI files (green bars) in E-GMD demonstrated by the example file *drummer8_eval_session_4_soul-groove4-80_beat_4-4_55.wav*. The visualization was done by using the software *Sonic Visualiser* introduced by Cannam, Landone, and Sandler (2010).

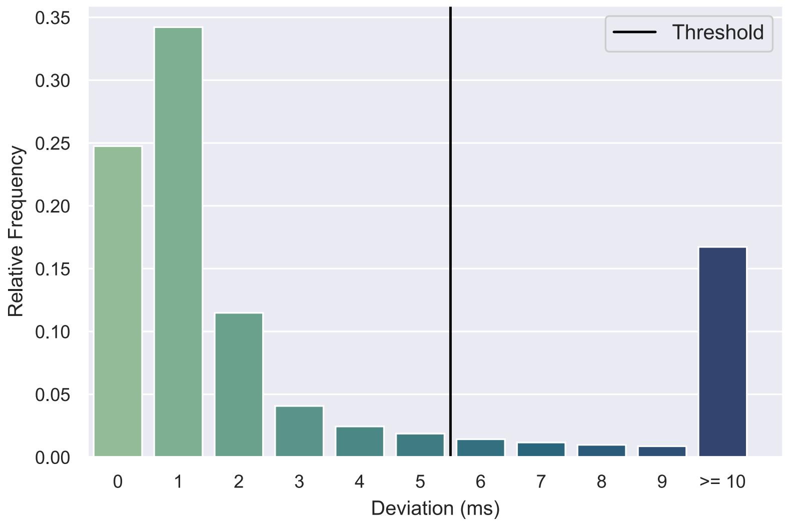


FIGURE 3.5: Distribution of the deviation between audio files and MIDI annotations of E-GMD. Files that display a deviation of $\geq$ 6 ms are removed from the dataset and are therefore not part of E-GMD-Clean.

In order to be able to guarantee a proper training, testing, and validation process, all files where the MIDI annotations deviate less than 6 ms from the audio file are kept in the dataset whereas the remaining files were excluded. The threshold of less than 6 ms is chosen based on the findings from Litovsky et al. (1999) where trial participants start to hear two distinct clicks reliably when the temporal gap between two events is around 8 ms to 10 ms. The remaining data is called *E-GMD-Clean* from this point on.

This filtering process changes the distributions presented for E-GMD. Therefore, updated versions of the overview of unique sequences (Table 3.4), genres (Figures 3.6), tempos (Figure 3.7), and instruments (Figure 3.8) are given.

TABLE 3.4: An overview of the E-GMD-Clean splits and their duration. Compared to E-GMD, the number of total sequences is reduced by 20.5 % while the number of unique sequences is reduced by 13.5 %.

Split	**# Unique Sequences**	**# Total Sequences**	**Duration (hours)**
Train	731	28,844	250.6
Test	83	3,556	29.4
Validation	102	3,799	32.6
Total	916	36,199	312.6

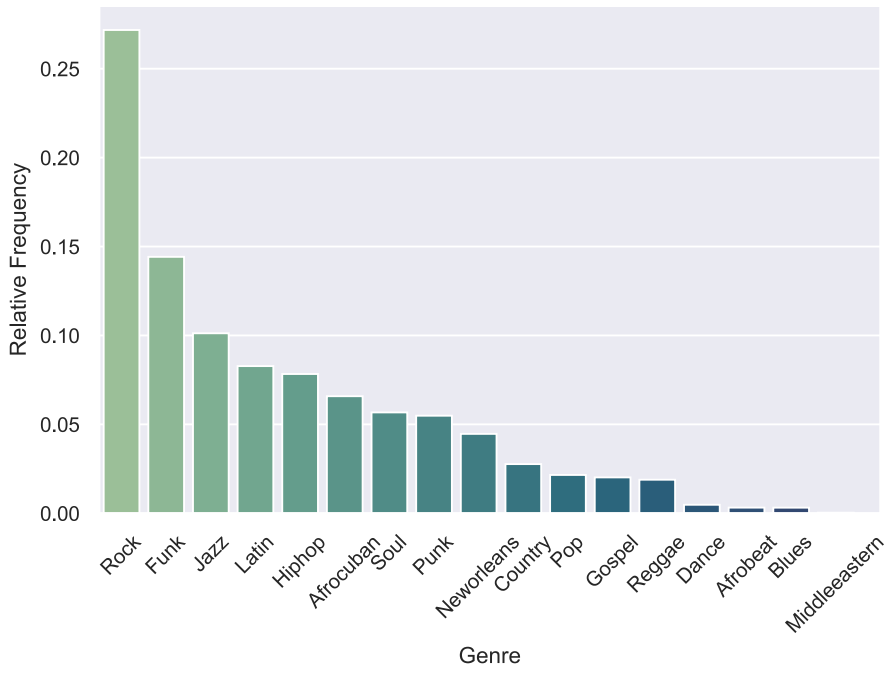


FIGURE 3.6: Coarse genre distribution of E-GMD-Clean with *Rock* being the main genre (28.7 %). For this, the first genre of a file name was taken into consideration.

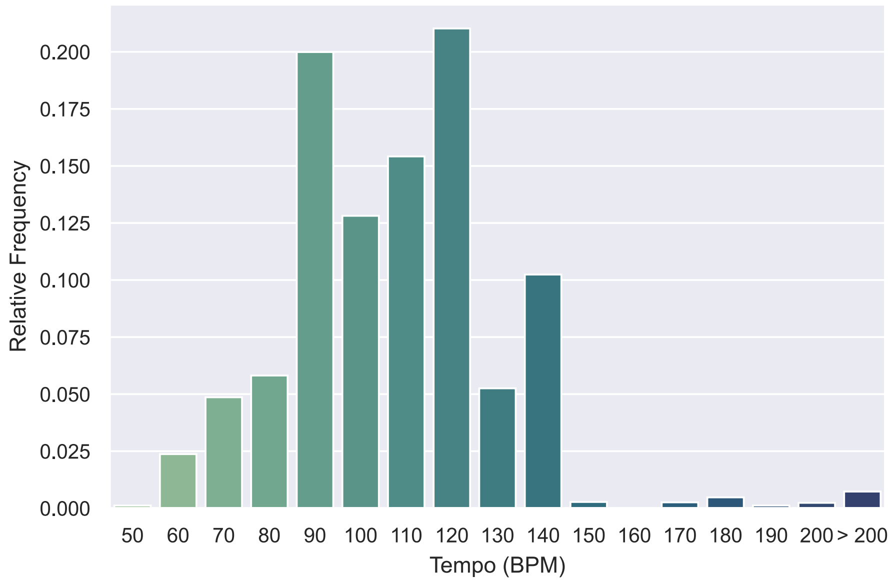


FIGURE 3.7: Relative tempo distribution of E-GMD-Clean grouped into steps of 10 BPM.

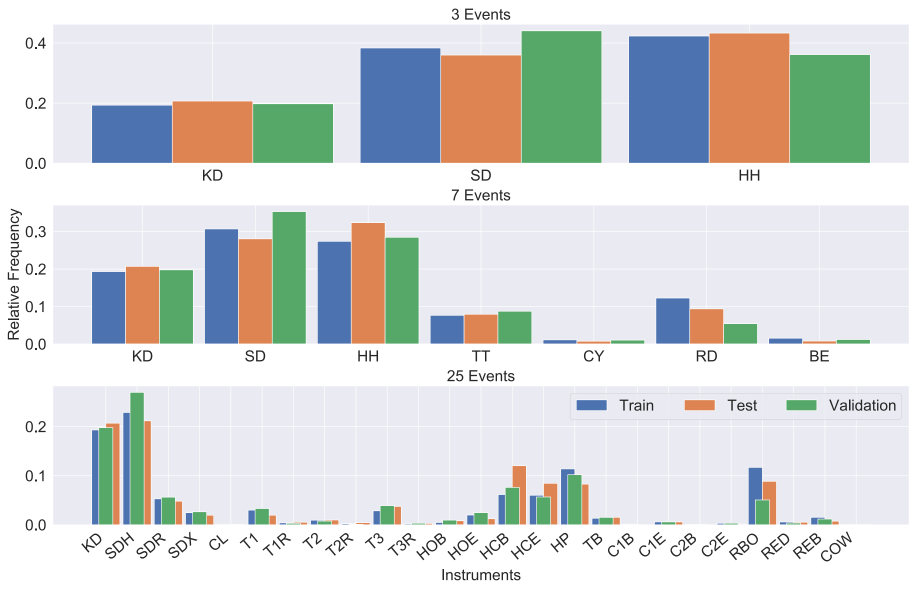


FIGURE 3.8: Relative event distributions of E-GMD-Clean among train, test, and validation split. The total number of events is 10,136,955.

3.2 RGDD

In order to investigate whether real drum sequences perform better than random drum sequences or adding random sequences to other training data can improve the performance of DNNs, a dataset consisting of both acoustic and electronic drum kits is introduced. This new dataset is referred to as *Randomly Generated Drums Dataset*

(RGDD) and was created using the Python package *MIDIUtil*[6]. RGDD comprises 52 different drum sets from Logic Pro[7] that were used to synthesize MIDI files which feature both random drums-only mixes and stems with a duration of 12 s each. Table B.1 shows the exact overview of the employed drum sets. Random mixes are files that feature random drum sequences where events occur both simultaneously as well as sequentially the majority of which are events occurring sequentially though. The maximum number of events that can occur simultaneously is set to 5. All parameters that were set either fixed or randomly for the creation of the dataset are shown in Table 3.5.

TABLE 3.5: An overview of the parameters and their values / ranges used for the creation of RGDD. Unlike ranges of values, single values indicate that this parameter is fixed and not randomly changed from one note to the next. The parameters *duration* and *spacing* are measured in quarter notes. The first range of note spacing is used for the creation of the stems of RGDD whereas the latter is used for the mixes.

Parameter	**Value / Range**
Track Tempo	60 BPM
Note Duration	0.01
Note Velocity	40 – 127
Note Spacing	0.3 – 1.4 / 0.1 – 0.3

The *tempo* of each drum kit track is arbitrarily set to 60 BPM as this parameter does not play an important role when creating random drum sequences. The *duration* of each MIDI event is also arbitrarily set to a value of 0.01 and refers to the duration in quarter notes. Just like *tempo*, *duration* can also be neglected with regard to the creation of the randomly generated dataset as the sustain of drums behaves differently from melodic instruments like for example piano. The first parameter that changes from one note to another is *velocity* with a typical range between 0 and 127. In case of RGDD, the velocity range is set to 40 – 127 to both allow for a wide dynamic range as well as reasonable loud values so that some events do not get masked by others too much. This phenomenon can be examined for E-GMD and is to be avoided. The last randomly set parameter is *spacing* which refers to the time in quarter notes between the onset of one note and its following note. This range was chosen to account for two aspects: to give each individual instrument hit similar space compared to IDMT-Stems for the stems of RGDD, and to have 12 hits of each instrument for stems and 80 hits of a drum set for mixes fit in a 12 s file.

In order to investigate if a DNN that is trained solely or additionally on *stems* can improve the performance of the DNNs that were previously trained on *mixes* only, a portion of RGDD is used as a standalone dataset to form the subset that is referred to as *RGDD-Stems*. For this part of the dataset only the note spacing of 0.3 – 1.4 is used. RGDD-Stems features 175,488 files in total which is comparable to E-GMD-Clean after it is split into 12 s sequences. After splitting RGDD-Stems by a ratio of 0.8 for the train test and 0.1 for the test and validation splits respectively, this yields the distribution among the different splits shown in Table 3.6.

[6]`https://github.com/MarkCWirt/MIDIUtil`, last accessed September 14, 2021
[7]`https://www.apple.com/logic-pro/`, last accessed November 5, 2021

TABLE 3.6: An overview of the RGDD (right column) and RGDD-Stems (left column) splits and their duration. Each of the 12 s long audio files is created randomly and therefore unique.

Split	**# Sequences**	**Duration (hours)**
Train	128,630 / 163,910	429 / 546
Test	23,429 / 35,189	78 / 117
Validation	23,429 / 35,189	78 / 117
Total	175,488 / 234,288	585 / 780

Due to the fact that the General MIDI Level 1 Percussion Key Map does not account for all the instruments listed in the *25 Hits* column in Table 3.3, the finest level of instrument grouping for both RGDD and RGDD-Stems is set to 7 to ensure the ability to compare the results of RGDD to E-GMD and E-GMD-Clean. The resulting grouping is shown in Table 3.7.

TABLE 3.7: Instrument complexity and grouping of RGDD.

Instrument	**7 Hits**	**3 Hits**
Kick	KD	KD
Sub-kick†	KD	KD
Snare Sidestick	SD	SD
Snare Center	SD	SD
Snare Rimshot	SD	SD
Hand Claps	SD	SD
Low Tom	TT	SD
Mid Low Tom	TT	SD
Mid Tom	TT	SD
Mid Hi Tom	TT	SD
Hi Tom	TT	SD
Hi-Hat Open	HH	HH
Hi-Hat Closed	HH	HH
Hi-Hat Foot Close	HH	HH
Hi-Hat Foot Splash	HH	HH
Tambourine	HH	HH
Crash	CY	HH
Ride In	RD	HH
Ride Edge	RD	HH
Ride Out	RD	HH
Ride Bell	BE	HH
Cowbell	BE	HH

†RGDD-Stems exclusive

Figure 3.9 shows the instrument for RGDD whereas Figure 3.10 shows the instrument distribution for RGDD-Stems, both of which distinguish between 3 and 7 classes.

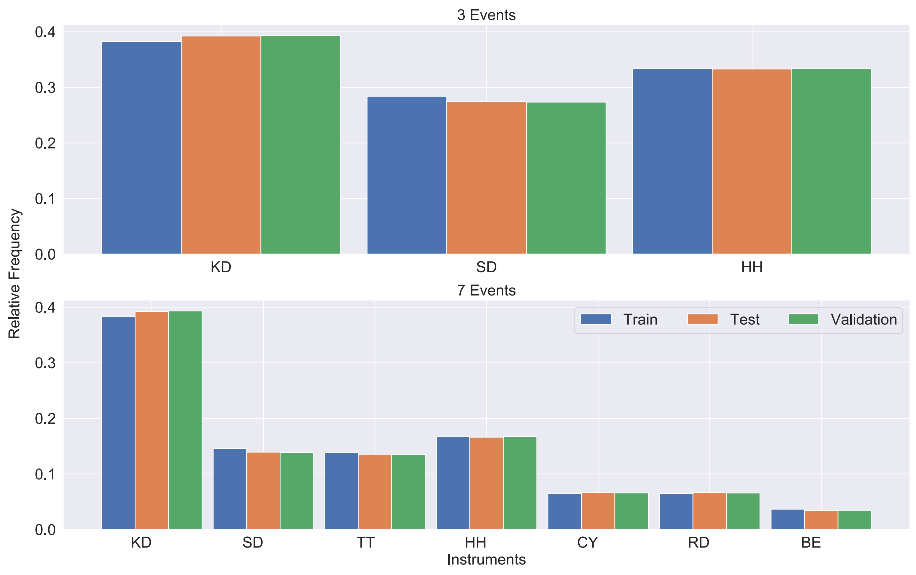


FIGURE 3.9: Relative event distributions of RGDD among train, test, and validation split distinguishing both between 3 (top row) and 7 (bottom row) event grouping. The total number of events is 6,809,856.

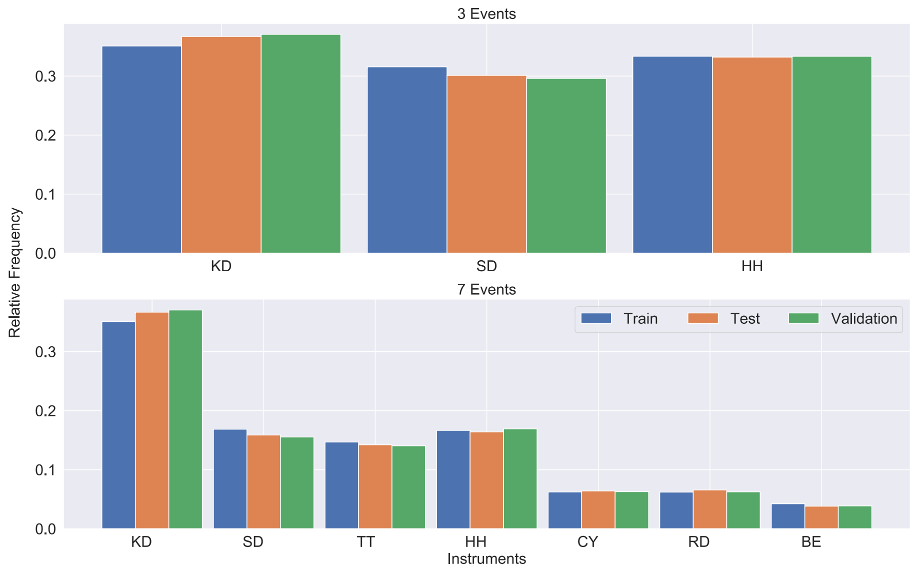


FIGURE 3.10: Relative event distributions of RGDD-Stems among train, test, and validation split distinguishing both between 3 (top row) and 7 (bottom row) event grouping. The total number of events is 2,105,856.

3.3 IDMT

The first dataset used for evaluation purposes is IDMT-SMT-Drums[8] (in this thesis referred to as *IDMT*) introduced by Dittmar and Gärtner (2014) which consists of 608 `wav` files that feature kick drum (KD), snare drum (SD), and hi-hat (HH) events only, 104 of which are polyphonic drum loops similar to the *Beats* and *Fills* of E-GMD. For each of the drum loops there are 3 stem files containing their individual drum hits originally intended for training purposes. This is one major difference compared to E-GMD which only contains files that feature multiple instruments. The corresponding ground truth annotations were manually created and are provided as `xml` and `svl` files. The dataset comprises three different kinds of tracks, that is recorded (*RealDrum*), synthesized (*TechnoDrum*), as well as sampled drums (*WaveDrum*). The recordings were made at a sampling frequency f_s of 44.1 kHz and 16 bit. The drum loops named *TechnoDrum02* and *WaveDrum02* comprise 64 drum loops combined but also feature 3 stem files each with their individual drum hits originally intended for source separation purposes. The stems of *WaveDrum02* are used for evaluation purposes in Section 5.2 to investigate the performance of DNNs when transcribing drums from stems instead of mixes. This part of the dataset is referred to as *IDMT-Stems*. The annotations to these files where extracted from the accompanying `xml` files. Due to unknown reasons, the available dataset only contains a total of 560 `wav` files albeit Dittmar and Gärtner (2014) claim it to be 608. However, this finding is similar to the findings of Vogl, Dorfer, and Knees (2017). Table 3.8 shows the exact distribution of the tracks of the dataset available after the download.

TABLE 3.8: An overview of IDMT and its track distribution.

Drum Type	**# Tracks**	**# Mixes**	**# Stems**
RealDrum	56	14	42
TechnoDrum	44	11	33
WaveDrum	460	70	390
Total	560	95	465

This means 9 out 104 *Mixes* plus their corresponding *Stems* are missing. Here, *Mixes* refers to tracks where all three instrument types occur in a file as a beat (previously referred to as *drum loops*). *Stems*, on the other hand, refers to single instrument tracks only containing for example KD hits. As a result of this, similar to Callender, Hawthorne, and Engel (2020), only the 95 available *Mixes* (here referred to as *IDMT-Mixes*) are used for evaluation purposes with an average duration of 15.14 s. The total duration of this test set is 23.97 min.

Figure 3.11 shows the relative event distribution among KD, SD, and HH events of *IDMT-Mixes*.

[8] `https://www.idmt.fraunhofer.de/en/business_units/m2d/smt/drums.html`, last accessed July 16, 2021

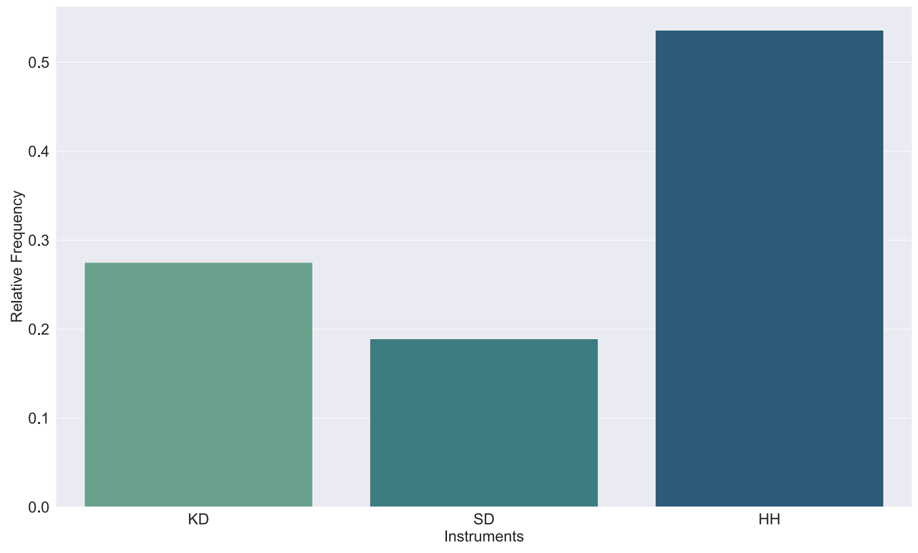


FIGURE 3.11: Relative event distribution of the *IDMT Mixes* used for evaluation purposes. The total number of events is 7,972.

Similar to E-GMD, there is also a class imbalance in IDMT with twice as many HH onsets (54 %) as KD onsets (27 %) and SD (19 %) being the least present instrument in this test set.

3.4 ENST

Another commonly used dataset for ADT is ENST-Drums (here referred to as *ENST*) introduced by Gillet and Richard (2006). It features 3 drummers playing various genres like Rock, Salsa, Funk, and Big-Band. Besides the variation with regard to genres, there is also variation in playing styles that range from sticks, to brushes, to rods. Overall, three different drum sets are used for the recordings. The dataset comes with different kinds of tracks that comprise single hits, short phrases, solos, drums-only tracks (by Gillet and Richard (2006) referred to as *minus-one* tracks), and a track that is the isolated accompaniment to the drums-only tracks. ENST is used to evaluate in two different settings in this thesis to investigate the proposed model's ability to generalize on both drums-only and full-mixes. The first setting is the evaluation on the isolated drums-only mixes in Section 5.1. In contrast to that, in Section 5.3 the drums-only tracks are mixed with their corresponding accompaniment track in an amplitude ratio of 2/3 for drums-only tracks and 1/3 for the accompaniment which is in line with Vogl, Widmer, and Knees (2018) and Paulus and Klapuri (2009). Note, that the *wet_mixes* of the *minus-one* tracks are used where equalization, compression, and reverberation was applied to the audio tracks. This results in a test set of 64 files with an average duration of 57 s and a total duration of roughly 1 h.

ENST features an instrument granularity of overall 20 different instruments. Similar to Callender, Hawthorne, and Engel (2020), all of these instruments are grouped into 7 classes which is different to Vogl, Widmer, and Knees (2018) who group them into 8 classes. However, since their 8th class is *clave/sticks* the evaluation results for ENST can still be compared to each other because there are no annotations for *STICKS* in the dataset which means this class can be neglected. Table 3.9 shows the grouping of all 20 instruments into 7 and 3 classes, respectively.

TABLE 3.9: Instrument complexity and grouping of ENST. Similar to Callender, Hawthorne, and Engel (2020), all instruments are grouped into 7 classes which is different to Vogl, Widmer, and Knees (2018) who apply a 8-class grouping. Unlike Callender, Hawthorne, and Engel (2020), both this thesis as well as Vogl, Widmer, and Knees (2018) investigate a 3-class scenario as well.

Instrument	**20 Hits**	**7 Hits**	**3 Hits**
Kick Drum	BD	KD	KD
Snare Drum	SD	SD	SD
Snare Drum Brush Sweep	SWEEP	SD	SD
Snare Drum Rimshot	RS	SD	SD
Sticks Together	STICKS	SD	SD
Side-Stick	CS	SD	SD
Mid Tom	MT	TT	SD
Mid Tom Rim	MTR	TT	SD
Low Mid Tom	LMT	TT	SD
Low Tom	LT	TT	SD
Low Tom Rim	LTR	TT	SD
Lowest Tom	LFT	TT	SD
Closed Hi-Hat	CHH	HH	HH
Open Hi-Hat	OHH	HH	HH
Crash Cymbal	CR	CY	HH
Splash Cymbal	SPL	CY	HH
Chinese Ride Cymbal	CH	CY	HH
Ride Cymbal	RC	RD	HH
Other Cymbal	C	RD	HH
Cow Bell	CB	BE	HH

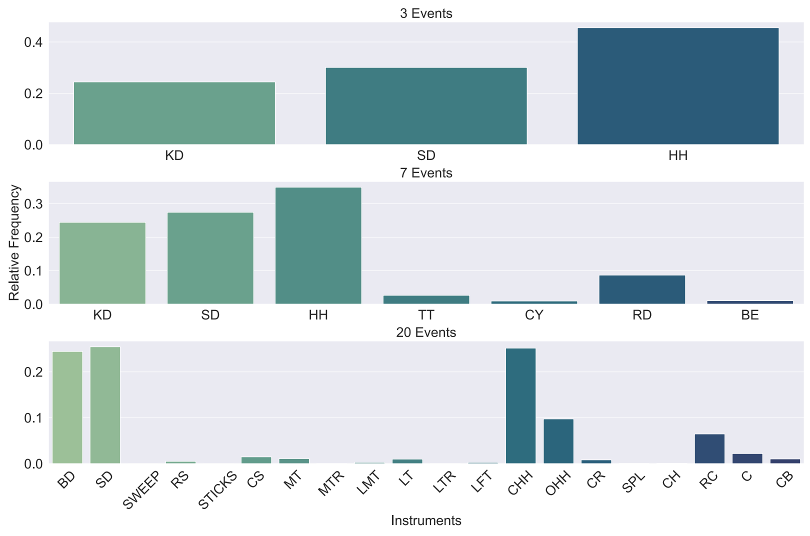


FIGURE 3.12: Relative event distributions of ENST for 3, 7, and 20 event classes. The total number of events is 26,411.

Overall, the most frequently instrument played in ENST is the hi-hat, followed by the snare drum and kick-drum. Figure 3.12 shows that all other instrument classes are sparsely populated. Especially at the most granular level, it becomes obvious that besides the aforementioned majority classes mostly ride and other cymbals are played.

3.5 MDB

MedleyDB Drums (*MDB*) is another commonly used dataset in ADT tasks (Vogl, Widmer, and Knees (2018); Wang et al. (2020); Cartwright and Bello (2018); Wei, Wu, and Su (2021)) and was published by Southall et al. (2017). It consists of 23 tracks featuring various genres (e.g. Country, Disco, Rock, Jazz, and Reggae) and is an annotated subset of the MedleyDB dataset[9] which was published by Bittner et al. (2014). For each of those tracks there are multiple versions that is *drums-only*, *full-mix*, and *multi-track* files. In this thesis, both the *drums-only* and *full-mixes* files are used for evaluation purposes. Drums-only files are used in Section 5.1 whereas full-mixes are used in Section 5.3. The average duration of an audio file is 56.9 s while the overall duration is 21.8 min. Two different drum event groupings are provided with this dataset. The first option groups all events into 6 classes, whereas the second one delivers a finer grouping with a total of 21 classes. However, for the sake of this thesis those 21 classes were grouped into 7 classes similar to the one of E-GMD. Figure 3.13 shows the relative frequency of those two classes plus the additional grouping of 3 classes as this has been the simplest grouping in previous research too.

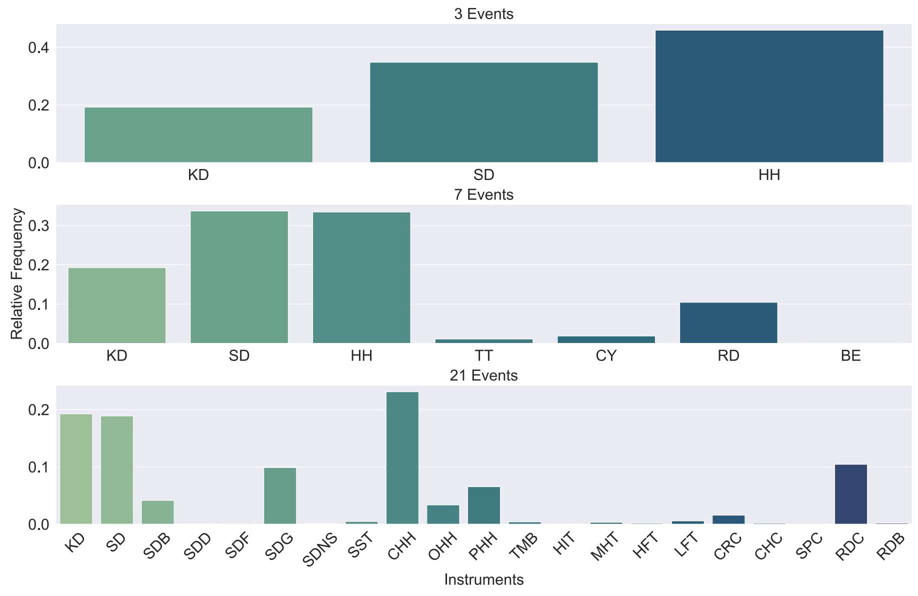


FIGURE 3.13: Relative event distributions of MDB for 3, 7, and 21 event classes. The total number of events is 7,994.

[9]https://medleydb.weebly.com/, last accessed July 16, 2021

TABLE 3.10: Instrument complexity and grouping of MDB.

Instrument	21 Hits	7 Hits	3 Hits
Kick Drum	KD	KD	KD
Snare Drum	SD	SD	SD
Snare Drum Brush	SDB	SD	SD
Snare Drum Drag	SDD	SD	SD
Snare Drum Flam	SDF	SD	SD
Snare Drum Ghost Note	SDG	SD	SD
Snare Drum No Snare	SDNS	SD	SD
Side-Stick	SST	SD	SD
High Tom	HIT	TT	SD
Mid-High Tom	MHT	TT	SD
High Floor Tom	HFT	TT	SD
Low Floor Tom	LFT	TT	SD
Closed Hi-Hat	CHH	HH	HH
Open Hi-Hat	OHH	HH	HH
Pedal Hi-Hat	PHH	HH	HH
Tambourine	TMB	HH	HH
Crash Cymbal	CRC	CY	HH
China Cymbal	CHC	CY	HH
Splash Cymbal	SPC	CY	HH
Ride Cymbal	RDC	RD	HH
Ride Cymbal Bell	RDB	BE	HH

3.6 RBMA13

The Red Bull Music Academy (RBMA13)[10] dataset was introduced by Vogl et al. (2017) with a focus on transcribing various drum types from full-mixes. It consists of 30 multi-tracks used for evaluation purposes and features 22 different drum instruments. These tracks are fully-produced pieces of music and are therefore the most difficult dataset to automatically transcribe drums from. However, three of the multi-tracks (*Leo Aldrey - NY Walk*; *QuietDust - Meridian Lines*; *Louis Baker - The Way*) do not feature any drums and were therefore excluded from this dataset. The featured genres range from electronic dance music to techno to fusion-jazz styled music with an average duration of 3 m 50 s while the overall duration amounts to 1 h 43 m. The annotations to the dataset were created manually and are publicly available[11,12]. Similar to E-GMD, ENST, and MDB, the individual drum instruments were summarized into groups of 3 and 7, respectively. The exact grouping is demonstrated in Table 3.11. Finally, Figure 3.14 gives an overview of the instrument distribution among the two different kinds of groupings.

[10]https://rbma.bandcamp.com/album/various-assets-not-for-sale-red-bull-music-academy-new-york-2013, last accessed August 11, 2021

[11]https://github.com/GiantSteps/RBMA_various_annotations/tree/master/RBMA_VA_2013, last accessed August 11, 2021

[12]http://ifs.tuwien.ac.at/~vogl/datasets/, last accessed August 11, 2021

TABLE 3.11: Instrument complexity and grouping of RBMA13.

Instrument	**22 Hits**	**7 Hits**	**3 Hits**
Kick Drum	KD	KD	KD
Snare Drum	SD	SD	SD
Side-Stick	SST	SD	SD
Hand Clap	CLP	SD	SD
Clave, Sticks, Clicks	CLV	SD	SD
High Tom	HIT	TT	SD
Mid-High Tom	MHT	TT	SD
Low Tom	LOT	TT	SD
Low Mid Tom	LMT	TT	SD
High Floor Tom	HFT	TT	SD
Low Floor Tom	LFT	TT	SD
Closed Hi-Hat	CHH	HH	HH
Open Hi-Hat	OHH	HH	HH
Pedal Hi-Hat	PHH	HH	HH
Tambourine	TMB	HH	HH
Shaker, Maracas	SHK	HH	HH
Crash Cymbal	CRC	CY	HH
China Cymbal	CHC	CY	HH
Splash Cymbal	SPC	CY	HH
Ride Cymbal	RDC	RD	HH
Ride Cymbal Bell	RDB	BE	HH
Cowbell	CWB	BE	HH

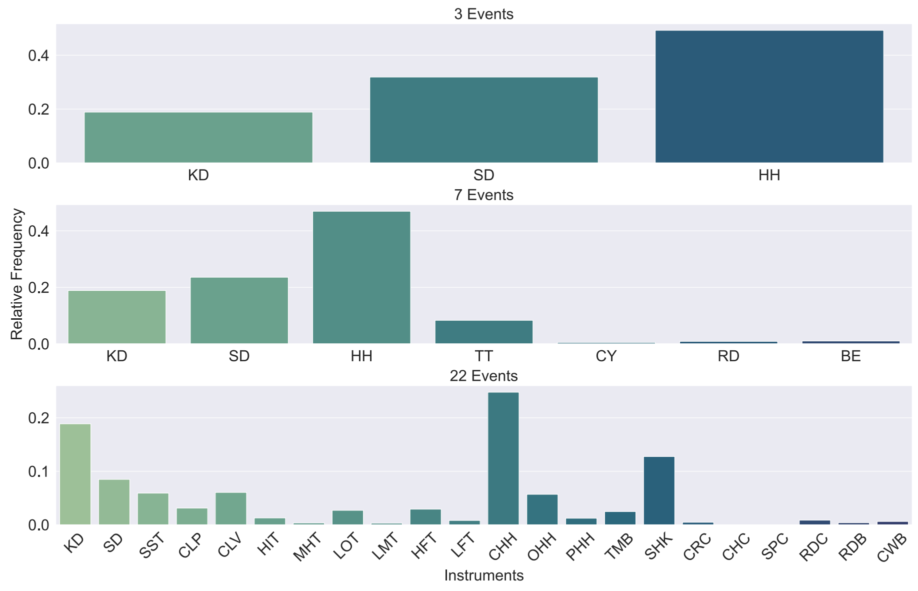


FIGURE 3.14: Relative event distributions of RBMA13 for 3, 7, and 22 event classes. The total number of events is 40,218.

Chapter 4

Methods

The following chapter talks about the pre- and post-processing steps used in this thesis to train and evaluate the performance of the deep neural network used to automatically transcribe drums from audio signals. First, the input features of the DNN are introduced by explaining their computation while at the same time motivating the use of them. After that, pre-processing steps like ensuring that all training instances have the same length as well as filtering odd annotations are presented. In addition to that, an augmentation technique is introduced followed by the description of the target computation. The proposed model architecture used to tackle ADT is compared to benchmark models while highlighting differences between them. Finally, this chapter concludes by presenting a post-processing step to extract detected onsets from an activation vector and introducing the necessary metrics to evaluate the performance of the proposed model on publicly available datasets and therefore be able to compare this to the benchmark models.

4.1 Features

The input features of the deep neural network used in this thesis are log-mel magnitude spectrograms which show the intensity of frequencies over time. The starting point for this is the *Fourier transform* which decomposes a continuous time signal into its constituent frequency components as shown in the following equation:

$$X(\omega) = \int_{-\infty}^{\infty} x(t)\mathrm{e}^{-j\omega t}\, dt \tag{4.1}$$

where:

$\omega = 2\pi f$, angular frequency
x = continuous input signal
t = time
e = Euler's number
j = imaginary unit

The result of the Fourier transform is the complex spectrum consisting of phase and magnitude of the entire input signal x.

However, due to the non-stationary nature of music signals, one single Fourier transform of the entire signal would not yield information on the frequency composition at individual time steps or *frames* which is necessary to make predictions about when in the input signal a certain drum instrument occurs. Therefore, a window of N samples is used to calculate the *short-time Fourier transform* (STFT) centered around sample point n with a step size or *hop length* of H samples, which determines the number of samples the window is shifted across the audio. The STFT is applied to the input signal, which has previously been discretized by sampling the input at a sampling frequency f_s, which results in the following equation:

$$X(m,k) = \sum_{n=0}^{N-1} x(n)w(n-mH)\mathrm{e}^{-j2\pi kn/N} \tag{4.2}$$

where:

x	= discrete input signal
n	= sample point
w	= window function of length N
m	= time frame
H	= hop length
k	= Fourier coefficient
N	= FFT window length in samples

The parameter settings of the feature computation used for this thesis are listed in Table 4.1. Given these settings, the time resolution amounts to $H/f_s = 10\,\mathrm{ms}$ whereas the frequency resolution is $f_s/N \approx 21.5\,\mathrm{Hz}$.

TABLE 4.1: Overview of the employed parameter settings of the log-mel magnitude spectrogram computation.

Parameter	**Value**
Sample rate f_s	44.1 kHz
Hop length (H)	441
FFT window length (N)	2,048
Number of mel bands (M)	128
f_{min}	27.5 Hz
f_{max}	16 kHz

To account for the human perception of sound, two additional post-processing steps are applied to the STFT of the input signal. First, the spectrogram is mapped from a frequency scale to M bands of a mel scale to account for the non-linear perception of pitch. Equation 4.3 demonstrates the conversion from a linear frequency scale to a mel scale where equal distances on this scale have the same perceptual distance.

$$m(f) = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \tag{4.3}$$

where:

m	= frequency in mel
f	= frequency in Hz

Equation 4.3 was introduced by O'Shaughnessy (1987) and maps frequencies up to 1 kHz linearly for humans can discern small changes at low frequencies better than at higher frequencies. As a result of this, higher frequencies are treated approximately logarithmically. In order to convert an STFT spectrogram into mel spectrogram, both the lowest and the highest possible occurring frequencies are converted to their equivalent mel frequency. Between this range, a previously defined number

of mel bands (*M*) is equidistantly spaced. After that, the mel frequencies are converted back to Hz so that an overlapping triangular filter bank of size *(M, N + 1)* is applied to the STFT spectrogram. Second, the resulting amplitude is transformed from a linear to a logarithmic scale as the latter better describes the way humans perceive loudness as demonstrated by Fechner (1860). Note, that for the features employed in this thesis only the magnitude is used which is in line with the work by Vogl, Widmer, and Knees (2018) as well as Callender, Hawthorne, and Engel (2020).

Unlike Callender, Hawthorne, and Engel (2020), the number of mel bands is reduced from 250 to 128 in order to reduce the computational cost during the training process. As the results in Chapter 5 suggest, this reduction in feature complexity does not result in a degradation of the model performance. Similar to Callender, Hawthorne, and Engel (2020), the features are cropped to a length of 12 s which is done offline before the training process by cutting all audio files into 12 s chunks. In case the remainder is under 1 s long, it is discarded otherwise the chunked audio and MIDI files are repeated so many times so that they have a length of 12 s, respectively. The threshold is set to 1 s because the average duration of *Fills* of E-GMD is 2.85 s. That way it is guaranteed that no file from the original dataset is discarded.

These features are commonly used in current ADT tasks (Vogl, Widmer, and Knees (2018); Callender, Hawthorne, and Engel (2020)). In order to compute them, a Python package called *librosa* which was published by McFee et al. (2015) is used. Figure 4.1 shows an example of the employed features based on a file taken from E-GMD.

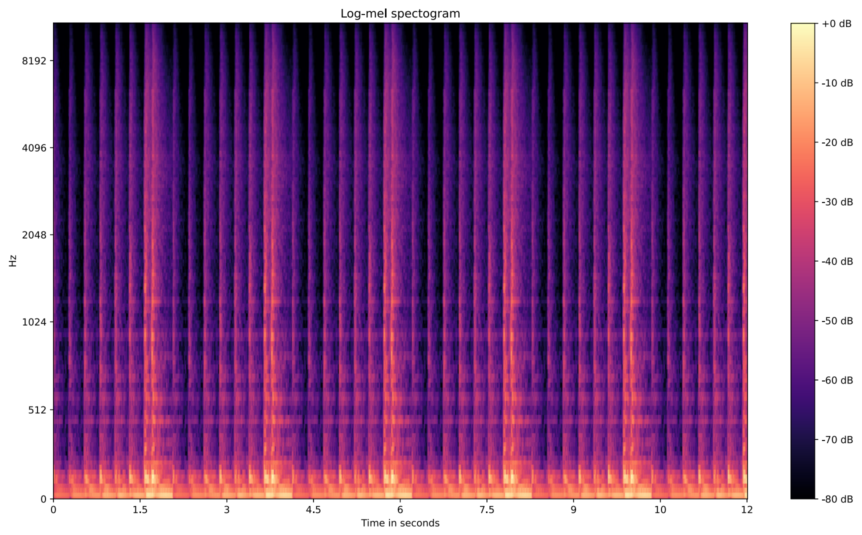


FIGURE 4.1: Log-mel magnitude spectrogram of the repeated 12 s version of *drummer1_session1_6_jazz-funk_116_fill_4-4_57.wav* from E-GMD. The amplitude spectrum is converted to a normalized dB scale for visualization purposes.

4.2 Pre-processing

Cutting & Padding

Since the model always expects the same dimensions during the training process, a cutting and padding process is implemented in the pre-processing pipeline. In case that an audio file is longer than the predefined 1200 frames, which is equivalent to

12 s in this thesis, it is cut into segments of the same length. On the other hand, an audio file is padded with zeros if it is shorter than 12 s to ensure the same length of all training instances.

Annotation Filtering

An analysis of E-GMD has shown that there are files in the dataset that feature MIDI files which have events that occur after the end of the corresponding audio file. As a result of this, part of the pre-processing pipeline is a filter that ensures that MIDI files do not feature any annotations after the corresponding audio that would. If not corrected, this would degrade the quality of the training process.

Augmentation

In order to tackle the issue of overfitting, augmentation is applied to the training datasets of the proposed model. To achieve this, the corresponding audio files were altered using the Python library *audiomentations*[1] prior to the training process. The first change was to add Gaussian noise to the file with a probability of 30 %. Next, a pitch shift of $\pm$ 2 semitones is applied with a probability of 15 % followed by a gain reduction or increase of $\pm$12 dB with a probability of 75 %. Finally, a bandpass filter is applied with a center frequency between 300 Hz and 1000 Hz and a q-factor between 1 and 2 with a probability of 10 %. The augmentation is applied prior to the training to avoid an unnecessary slow-down of the entire training process. Note, that the augmented audio files are used in addition to the original dataset to increase the number of files and therefore the variance within the train set.

Target computation

As described in Section 2.1, the core idea in supervised learning is to compare the predictions of a DNN to the actual label or ground truth of the data. Based on that, a loss is calculated which is then backpropagated to indicate how the weights or kernels in the DNN have to be adjusted in order to minimize this error. In this thesis, a *soft-target vector* is computed which the predictions of the DNN are compared to. The approach for this is to compute a vector that contains target probabilities for each time frame. Unlike a *one-hot target vector*, it specifies probabilities of an onset in neighboring frames in addition to the probability for the frame of the actual onset. This concept is visualized in Figure 4.2.

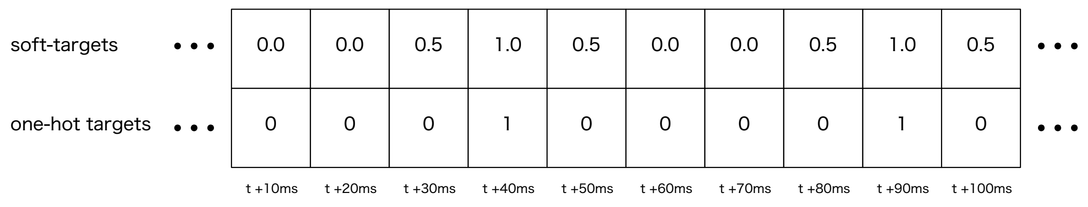


FIGURE 4.2: Conceptual comparison between *soft-targets* (top row) and *one-hot targets* (bottom row).

The decision to use a soft-target vector instead of a one-hot target vector is based on the sparsity of the latter one. Populating the activation vector with additional target probabilities allows for more information during the training process especially in cases where the actual onset of an instrument is at the end of a frame of the computed spectrogram.

[1]https://github.com/iver56/audiomentations, last accessed Nov 24, 2021

4.3 Network Architectures

This thesis proposes a deep neural network architecture that is inspired both by Vogl, Widmer, and Knees (2018) as well as Callender, Hawthorne, and Engel (2020). To better understand the main differences between the different architectures, Figure 4.3 juxtaposes the main building blocks of each of those.

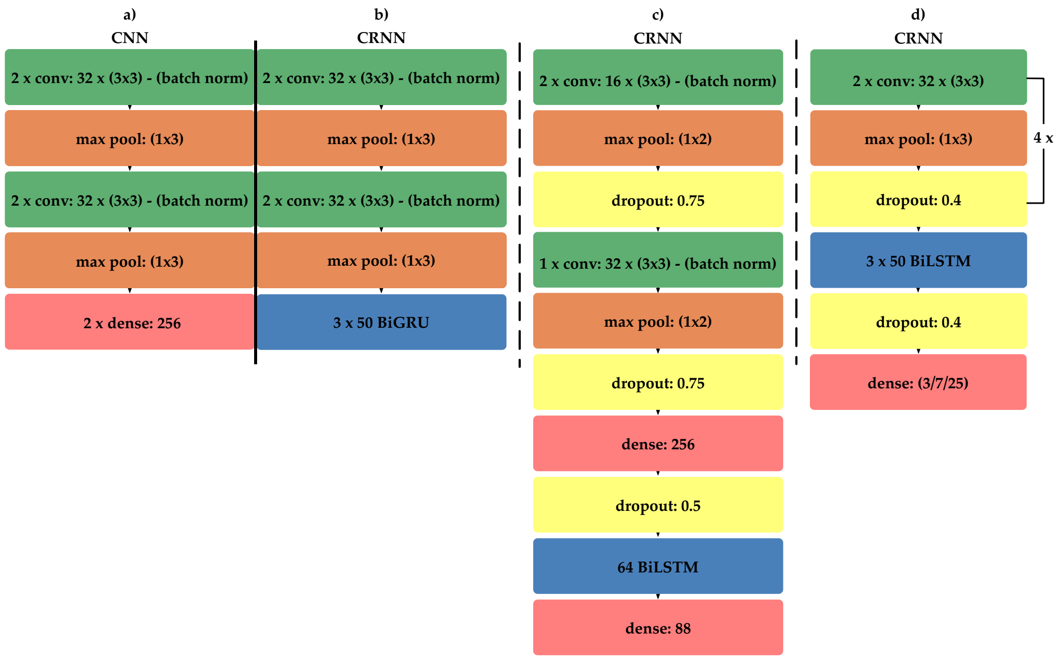


FIGURE 4.3: Comparison of baseline model architectures and a new proposed model. The architectures a) and b) by Vogl, Widmer, and Knees (2018) are used as a baseline for the comparison of the performance on full-mixes of ENST, MDB, and RBMA13, whereas c) is the proposed model architecture by Callender, Hawthorne, and Engel (2020) which is used as a baseline for E-GMD, IDMT, ENST, and MDB in a drums-only mixes setting. In contrast to that, d) shows the architecture of the proposed model in this thesis.

Vogl, Widmer, and Knees (2018) propose two different approaches to tackle ADT, namely a typical CNN and a CRNN both of which are identical with regard to the first three building blocks, only the last one changes from dense layers in the CNN (Figure 4.3a) scenario to bi-directional GRUs in the CRNN setup (Figure 4.3b). These architectures were introduced in a setting to transcribe various numbers of drum instruments from full-mixes from ENST, MDB, and RBMA13 ranging from 3 to 8 to finally 18 instrument classes. These two models serve as benchmarks in the evaluation of the full-mix scenario in Section 5.3 and are here referred to as *Vogl-Drums*.

Figure 4.3c shows the model that serves as a benchmark in the evaluation of the drums-only scenario introduced by Callender, Hawthorne, and Engel (2020) which is called *OaF-Drums*. This model introduces a dense layer similar to the one from Vogl-Drums CNN before a bi-directional LSTM layer that features a dropout of 50 % at its output. Additionally, OaF-Drums uses dropout after max-pooling layers which serve as additional regularization. Besides that, OaF-Drums consists of an additional velocity prediction head which is not displayed here.

Unlike the baselines, the proposed model shown in Figure 4.3d does not employ batch-normalization layers but uses the *SELU* activation function introduced by Klambauer et al. (2017) for normalization purposes instead. Overall, the proposed model combines 4 blocks of convolutional, max pooling, and dropout layers with a block of bi-directional LSTM layers and dropout layers followed by a dense layer. The LSTM layers' input dropout is set to 10 % so that the model does not rely too much on the context it sees during training. Generally, the hyperparameter settings of the proposed model were found through a coarse grid search. Besides that, the motivation for a CRNN in the setting of ADT is based on preliminary experiments that showed a better performance when using a CRNN instead of a CNN.

4.4 Post-processing

The final step in the transcription task is to determine where an onset of a certain instrument is. This is done by applying a *peak-picking* algorithm to the output of the DNN which is an activation array per instrument that indicates the probability that an instrument is played at a certain time frame. In order to do so, a moving average filter is applied to the activation array with a filter size of 15 frames. Next, it is checked at what time frame the activation array is equal or larger than the moving average plus a threshold which is set to 0.1. This value is motivated by findings of Vogl, Dorfer, and Knees (2017). At the indices where this condition is met, the value of the activation function is preserved otherwise it filled with zeros. This process is followed by applying a maximum filter of 5 frames to the previously determined array of matches. Finally, this result is compared to the matches, so that eventually all indices where the values of both the maximum filter array and the matches are identical are returned. Therefore, the final output of this peak-picking algorithm is the points in time at which a certain instrument is classified by the DNN. The described approach is in accordance with Vogl, Widmer, and Knees (2018).

4.5 Evaluation Metrics

In order to gauge the performance of the ADT model, a typical metric called *F-measure* is used. It takes the count of *True Positives* (TP), *False Positives* (FP), and *False Negatives* (FN) into account, which can be summarized as *Precision* and *Recall*. It is a commonly used metric in music transcription tasks as well as imbalanced classification problems (Ferri, Hernández-Orallo, and Modroiu (2009)). In the following, the constituents of the *F-measure* are explained and visually demonstrated.

Precision indicates how many of the positive predictions were in fact positive which is shown in Equation 4.4

$$Precision = \frac{TP}{TP + FP} \tag{4.4}$$

Recall (see. Equation 4.5), on the other hand, describes how many percent of the predictions that should be positive are indeed positive.

$$Recall = \frac{TP}{TP + FN} \tag{4.5}$$

Each one of this metric has its own focus. *Precision* for example focuses on minimizing the amount of *False Positive* predictions whereas *Recall* focuses on minimizing the number of *False Negative* predictions. Depending on the task, one of those metrics can be favorable. Figure 4.4 visualizes the concept between those two metrics.

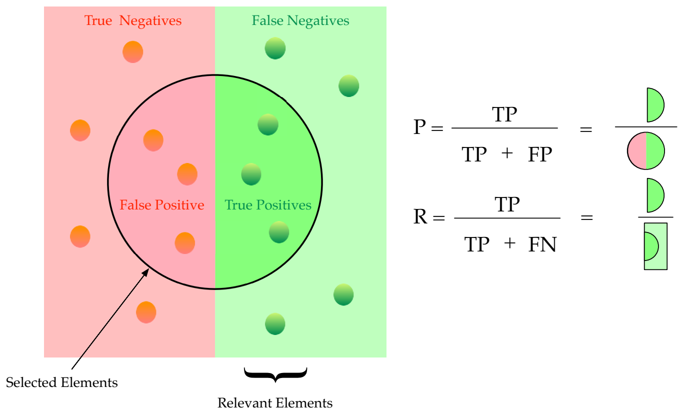


FIGURE 4.4: Overview of how *Precision* and *Recall* are constituted. The harmonic mean of both form the *F-measure*.

If you combine all correct predictions and compare those against the total number of predictions, you get the *Accuracy* of your predictions, shown in Equation 4.6.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.6}$$

However, classification tasks want to reduce both the number of *FP* and *FN*. Due to this reasons, the *F-measure* tries to balance *Precision* and *Recall* and still capture the properties of both of them. Equation 4.7 describes the computation of the *F-measure* which is the harmonic mean of *Precision* and *Recall*.

$$F_{measure} = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{4.7}$$

The Python[2] package *mir_eval*[3] is used to compute the above-mentioned metrics. To determine if there is a matching annotation in the ground truth to a predicted instrument, a tolerance window of 50 ms is used which is consistent with Callender, Hawthorne, and Engel (2020), although some authors use smaller tolerance windows such as 20 ms (Vogl et al. (2017)) which results in lower F-measures. The global *F-measure* is computed by gathering all TPs, FPs, and FNs of all instruments and tracks. This is more favorable than computing the mean of all tracks and instruments because of the sparsity of some instrument classes (Vogl et al. (2017)).

[2]`https://www.python.org`, last accessed August 1, 2021
[3]`https://craffel.github.io/mir_eval/`, last accessed August 1, 2021

Chapter 5

Results & Discussion

In the following three sections, different experiment settings of ADT are investigated. The focus of this work is detecting onsets and classifying the individual instruments while the velocity of the events is neglected. Two datasets are mainly used for training purposes, the first of which is *E-GMD* which was introduced by Callender, Hawthorne, and Engel (2020). The second one is a newly created dataset called *RGDD* which features random drum sequences consisting of both *stems* and *drums-only* mixes.

Section 5.1 presents the results of automatically transcribing drums from *drums-only* mixes, that is recordings which only contain drum instruments and no accompaniment. For this purpose, the mixes from *IDMT*, *ENST*, and *MDB* without accompaniment are used as test datasets in order to evaluate the performance of the deep neural network proposed in this thesis. The results for MDB in the drums-only chapter are presented for the sake of completeness and cannot be compared to a benchmark because to the best of the author's knowledge there is no paper in the literature that evaluates on the drums-only mixes of MDB. Similar to that, the results for ENST are not compared to a benchmark either. Even though Vogl et al. (2017) evaluate their models on the drums-only mixes from ENST, they ignore the annotations of all instruments but KD, SD, and HH. As results of this, a direct comparison with the approach presented in this thesis is not possible.

In contrast to that, Section 5.2 presents the results of a qualitative analysis on automatically transcribing drums from recordings that only contain one single drum instrument, namely kick-drum, snare-drum, or hi-hat, at a time. These recordings are here referred to as *stems*. For this purposes, only the *WaveDrum02* stems from IDMT are used because they are the only stems from IDMT that come with annotations.

Finally, Section 5.3 extends this to recordings including accompaniments like guitar, bass, keyboard, and vocals to gauge the performance of the proposed model in a setting it was not trained in. These recordings are referred to as *full-mixes*. Both Section 5.1 and 5.3 first present the results of a 3-class ADT scenario which is then extended to 7 classes. Only Section 5.1 also investigates the performance of 25 classes since E-GMD is the only dataset used in this thesis that contains 25 drum event types.

As mentioned in Chapter 1, it is investigated if a grouping of predicted activations to a smaller of number of instruments (e.g. reducing 7 predicted classes down to 3 and therefore reducing the instrument complexity) has an influence on the overall performance. That way, it can be analyzed if a model that is trained to transcribe a finer level of instrument complexity can have a benefit on simpler tasks. For this purpose, all predicted activations of a target instrument grouping are collected and compared to eventually take the overall maximum value of the corresponding instrument class at each time step to form the final activations vector. Figure 5.1 visualizes the concept behind this grouping method.

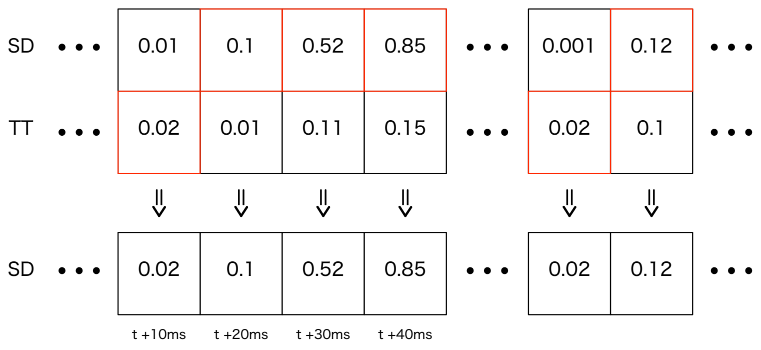


FIGURE 5.1: Exemplary grouping of SD and TT activations to a single SD activation vector by taking the overall maximum value at each time step. This approach is used for models that were initially trained to transcribe more instruments than finally used.

Furthermore, each section presents the results for *unmodified* and *augmented* versions of the training dataset for the sake of investigating the impact augmented training data has on the overall performance in ADT. Note, that the augmentation for OaF-Drums is different from the augmentation for the proposed model. The main differences are discussed in the next subsection. In case of the *augmented* version for the proposed model, the training set consists of both the *unmodified* as well as the *augmented* version to increase the number of training instances and therefore increase the variance. Throughout this thesis, only augmented training datasets are indicated as such.

The selected models for the evaluation were chosen from a series of training runs. For each model / training dataset combination numerous training runs were executed while the one that performed best on its test dataset was chosen to proceed with for further evaluation on IDMT, ENST, MDB, and RBMA13. This means that when the proposed model was for example trained on E-GMD-Clean-3 and RGDD-3 in numerous individual training runs the model that yielded the best overall F-measure on the corresponding test dataset of E-GMD-Clean-3 and RGDD-3 was chosen as the final model. This model was then used for evaluation purposes on the aforementioned datasets. Hence, the results presented in this thesis are based on a single model only.

Following the results section of each of the aforementioned experiments, the results are discussed and evaluated while trying to give answers as to why examining data prior to training is essential and why certain training data setups perform better than others.

5.1 Drums-only Mixes

This section presents the results of automatic transcription of drum events from signals that only contain drum instruments. The starting point of this investigation is a 3-event scenario that is commonly used in ADT tasks where all occurring instruments are grouped into KD, SD, and HH events as demonstrated in Section 3. Models that were trained on datasets that contain "-3" as a suffix can only distinguish between the three aforementioned instruments. In contrast to that, models that were trained on datasets that contain "-7" or "-25" as a suffix were trained to distinguish between the corresponding number of instruments. However, in this part of the thesis these models are employed to transcribe only 3 instruments. Therefore, their resulting predictions are grouped into KD, SD, and HH based on the concept shown in Figure 5.1. This is true for the evaluation results of IDMT, ENST, and MDB while the results of E-GMD always indicate the model's performance on the test set with the corresponding number of instruments.

To compare the performance of the proposed model in a 7-class scenario for E-GMD and ENST and a 3-class scenario for IDMT, *OaF-Drums* is used as a benchmark. Nevertheless, the performance of the proposed model on 3 classes for E-GMD, ENST, and MDB are listed for the sake of completeness. OaF-Drums was introduced by Callender, Hawthorne, and Engel (2020) and is based on a model by Hawthorne et al. (2018) that was designed for onset and frame predictions in a piano transcription task. A detailed overview of the model architecture is shown in Figure 4.3. OaF-Drums was trained on two different versions of E-GMD. The first *unmodified* version of the dataset comprises the unaltered training split chunked into 12 s segments. On the other hand, Callender, Hawthorne, and Engel (2020) employ an augmentation technique called *Shuffled mixup* that is used for regularization purposes. Here, they group two randomly selected audio files, repeat the shorter one so that both files have the same length, and then mix their audio samples as well as annotations. Finally, they split the newly created audio files into 1 s segments, shuffle them randomly, and finally combine them to 12 s segments. This configuration is called E-GMD *ShuMix*.

The results of the proposed model also distinguish between *unmodified* and *augmented*, however, a more traditional approach was chosen for augmentation purposes where noise, pitch shift, gain-boost/reduction, and a bandpass filter were applied to the audio files prior to the training. A detailed description of this augmentation process is given in Section 4.2. Besides that, two F-measures are given for the evaluation on E-GMD, the first of which presents the results when tested on the uncleaned version of E-GMD while the second one indicates the model's performance on the test set of E-GMD-Clean. Again, both results demonstrate the model's performance on the test set with the corresponding number of instruments indicated by the additional suffix.

3-classes

TABLE 5.1: F-measures of the 3-event scenario tested on drums-only input signals. All instruments are grouped into either KD, SD, or HH. Models that were trained on a dataset with the "-7/25" suffix group their predictions for IDMT, ENST, and MDB as shown in Figure 5.1. The first column of the E-GMD results show the performance of the proposed model on the E-GMD test set whereas the second column shows the results on the E-GMD-Clean test set where files with a deviation between the annotations and the audio of ≥ 6 ms were removed from the dataset. The augmentation approach of OaF-Drums and the proposed model are not identical. OaF-Drums employs a technique called *Shuffled mixup* whereas in case of the proposed model noise, pitch shift, gain-boost/reduction, and a bandpass filter were applied to the training data.

		F-measures			
Model	**Training Dataset**	**E-GMD**	**IDMT**	**ENST**	**MDB**
OaF-Drums	E-GMD-3	-	.527	-	-
	E-GMD-3 *ShuMix*	-	.857	-	-
Proposed CRNN	E-GMD-3	.866 / .913	.666	.833	.785
	E-GMD-Clean-3	**.879** / **.927**	.675	.826	.789
	E-GMD-Clean-3 *augm.*	**.879** / .926	.790	.828	.837
	E-GMD-Clean-3 + RGDD-3	.873 / .920	.853	.845	.831
	E-GMD-Clean-3 + RGDD-3 *augm.*	.869 / .919	**.885**	**.855**	**.867**
	RGDD-3	.642 / .678	.803	.792	.801
	RGDD-3 *augm.*	.716 / .747	.838	.791	.841
	E-GMD-7	.834 / .906	.746	.834	.789
	E-GMD-Clean-7	.874 / .921	.766	.830	.794
	E-GMD-Clean-7 *augm.*	.876 / .925	.837	.846	.820
	E-GMD-Clean-7 + RGDD-7	.868 / .913	.844	.841	.840
	E-GMD-Clean-7 + RGDD-7 *augm.*	.849 / .892	.829	.846	.856
	RGDD-7	.580 / .599	.815	.784	.804
	RGDD-7 *augm.*	.622 / .645	.833	.785	.826
	E-GMD-25	.758 / .803	.761	.823	.783
	E-GMD-Clean-25	.798 / .840	.785	.851	.809
	E-GMD-Clean-25 *augm.*	.805 / .850	.798	.832	.807

Table 5.1 suggests that when training the proposed model on E-GMD-Clean instead of E-GMD an increase in performance of 1.3 percentage points can be achieved when tested on the regular, uncleaned E-GMD test set, resulting in an F-measure of .879. However, the proposed augmentation does not impact the performance with regard to E-GMD in this setting. At the same time, these two models are the best performing ones in the 3-event transcription for E-GMD. Comparable results can be achieved with the proposed model trained on E-GMD-Clean-7 (augmented) with resulting F-measures of .874 and .876, respectively. With respect to 25 classes, training the proposed model on E-GMD-Clean leads to an increase in performance from .758 to .798, albeit these results show worse performance when compared to models that were trained to classify a coarser instrument grouping. When adding random drum sequences to E-GMD in the form of RGDD as additional training data to the training split of E-GMD-Clean, the overall performance on the E-GMD test set slightly decreases while at the same time performing 0.4 percentage points worse in case of

augmented training data in the 3-class scenario when compared to training on unmodified data. In case of E-GMD-Clean-7 + RGDD-7 augmented, the performance drops by 1.9 percentage points. This downward trend can also be seen when the model is only trained on RGDD, however, here the augmentation increases the performance by 7.4 percentage points for RGDD-3 and 4.2 percentage points for RGDD-7. The worst performance on the test set of E-GMD is achieved when training on RGDD-7. Similar to training on cleaned data, testing on cleaned data also increases the overall performance. This fact can be seen in all training setups, ranging from +1.9 percentage points for RGDD-7 to +7.2 percentage points for E-GMD-7. The best performing model when evaluating on E-GMD-Clean can be achieved with the proposed model trained on E-GMD-Clean-3 scoring an F-measure of .927 closely followed by its augmented version (.926) as well as its seven class counterpart E-GMD-Clean-7 augmented (.925). Figure 5.2 compares a selection of F-measures to show the discrepancy between models that were trained only on variations of E-GMD, adding RGDD, or RGDD only when tested on E-GMD. It makes clear that cleaning or augmenting the training data in case of E-GMD does not result in a major improvement in this case. At the same time, it shows that when trained only on RGDD the model struggles to generalize.

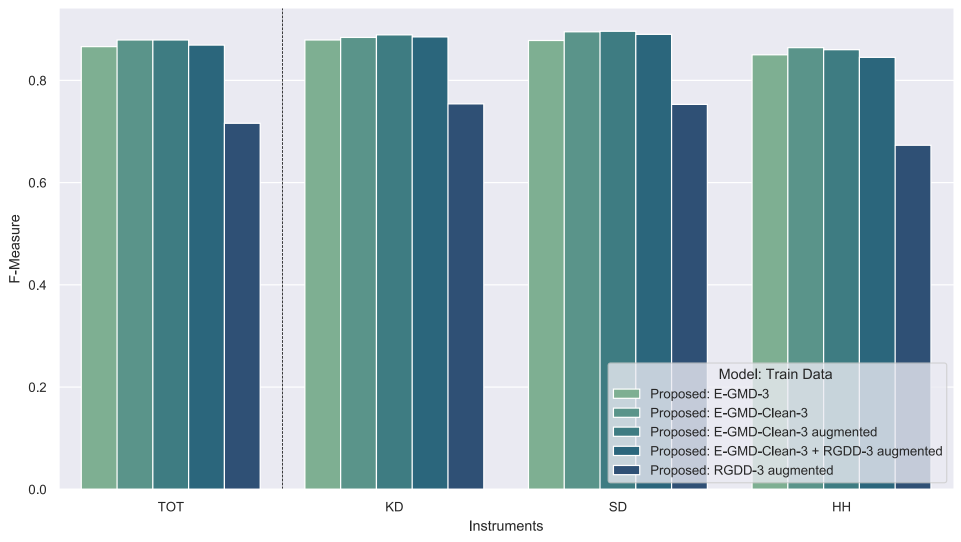


FIGURE 5.2: Overview of selected F-measures for the proposed model evaluated on E-GMD in a 3-class drums-only scenario.

With regard to IDMT, Table 5.1 shows that Callender, Hawthorne, and Engel (2020) are able to achieve an increase in performance of 33 percentage points by augmenting the training data resulting in an F-measure of .857. These values serve as the benchmark for the 3-class scenario from this point forward. It is assumed that Callender, Hawthorne, and Engel (2020) trained their model for the IDMT evaluation to distinguish between only 3 instruments instead of grouping predictions from 7 or 25 to 3 instruments as it is not stated otherwise. In comparison to OaF-Drums, the results show an advantage of the proposed model when trained on the unmodified version of E-GMD. For E-GMD-3 a plus of 13.9 percentage points can be achieved, yet models that were trained on a more granular number of instruments score even higher F-measures. In case of E-GMD-7, an increase of 21.9 percentage points with regard to the unmodified benchmark is scored whereas E-GMD-25 extends this to

23.4 percentage points. The cleaned and augmented version of E-GMD further improves the model's performance in all of the aforementioned setups while reaching its peak at .837 for E-GMD-Clean-7 augmented. In contrast to the evaluation on E-GMD, adding RGDD as additional training data increases the overall performance in all cases but E-GMD-Clean-7 + RGDD-7 augmented. This combination of training data results in the overall best score of .885 for E-GMD-Clean-3 + RGDD-3 augmented and therefore surpasses the benchmark by 2.8 percentage points. While the proposed model trained on RGDD alone does not score the highest F-measure on IDMT, its ability to generalize on IDMT is better than it was on E-GMD. In addition to that, the augmentation of the training data yields an increase in performance of up to 3.5 percentage points. Figure 5.3 summarizes the evaluation results of the proposed model on IDMT. It shows that while increasing the instrument complexity in a 3-class scenario can increase the overall performance, augmenting the training data or adding additional data further improves the ability to generalize on unseen data.

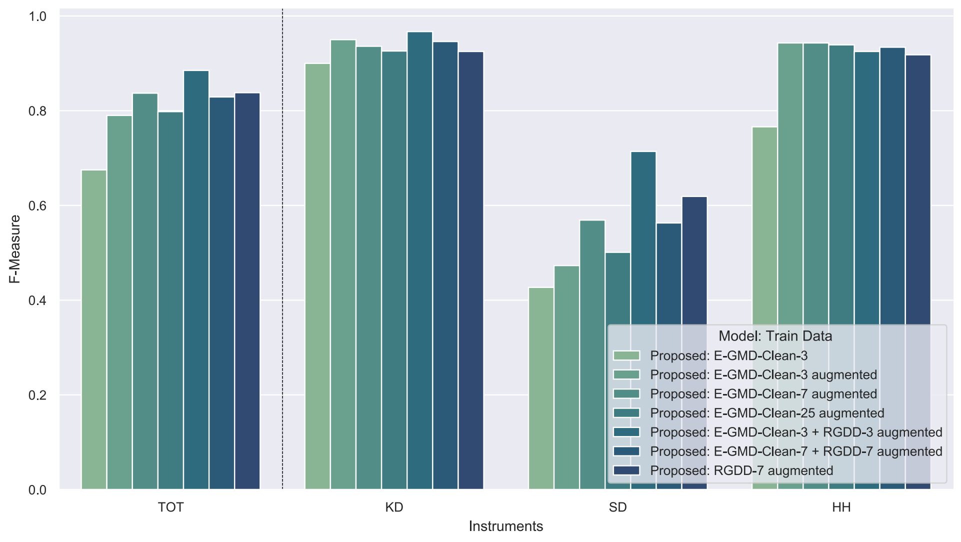


FIGURE 5.3: Overview of selected F-measures for the proposed model evaluated on IDMT in a 3-class scenario.

The evaluation of the proposed model on ENST also shows that training a model to initially transcribe 7 instruments and then use it to group its activation vectors so that it only yields 3 instrument classes can improve the overall performance in case of E-GMD-Clean augmented. The best results of the proposed model on ENST can be achieved when trained on E-GMD-Clean-3 + RGDD-3 with an F-measure of .855 which is closely followed by the model trained on E-GMD-Clean-25 which scores .851. With regard to the impact of the proposed augmentation technique, Table 5.1 shows that in most cases this approach can indeed improve the performance of the model when compared to the unmodified version of the training data, albeit this is not always the case as the example of E-GMD-Clean-25 shows where the augmentation introduces a performance decrease of 1.9 percentage points. With regard to RGDD, the augmentation introduces a performance change of $\pm$ 1 percentage points. A general tendency that can be extracted from the results is that the models trained on E-GMD(-Clean) exclusively mostly perform better on ENST than they did on IDMT. Otherwise, all other models show a slight decrease in performance with the exception of the model trained on RGDD-7 augmented as additional training data. When comparing Figure 5.3 and 5.4, performance with regard to SD can be substantially improved.

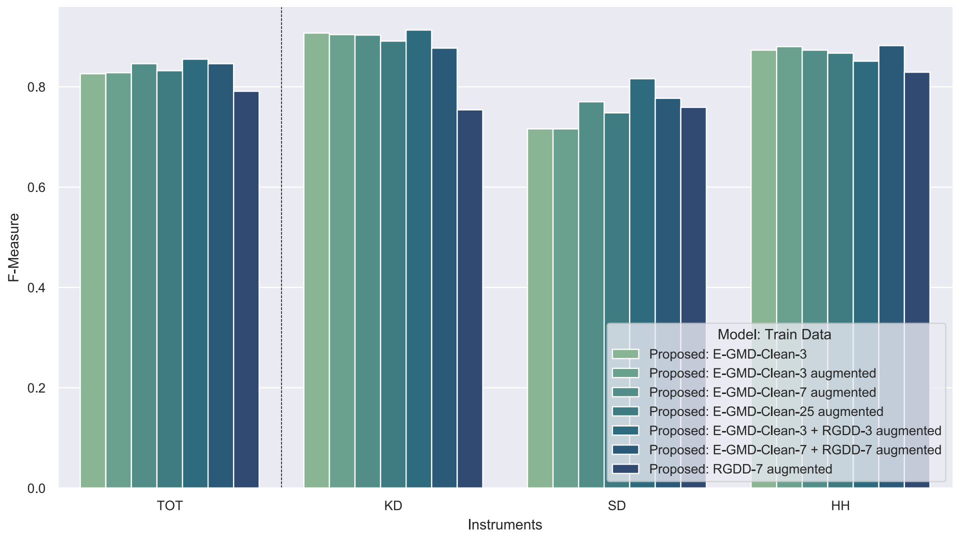


FIGURE 5.4: Overview of selected F-measures for the proposed model evaluated on ENST in a 3-class drums-only scenario.

Finally, the evaluation results of the proposed model on MDB show again that training on the cleaned version of E-GMD can improve the overall performance from .4 percentage points when trained to distinguish between 3 classes to 2.6 percentage points when trained to distinguish between 25 classes. Similar to the evaluation on IDMT and ENST, the model that was trained on E-GMD-Clean-3 + RGDD-3 augmented performs the best scoring an F-measure of .867. The closest results to this are the models trained on E-GMD-Clean-7 + RGDD-7 augmented (.856) or RGDD-3 augmented (.841). Augmenting the training data generally improves the performance ranging from 1.6 for E-GMD-Clean-7 + RGDD-7 to 4.8 percentage points for E-GMD-Clean-3, again, with the exception of E-GMD-Clean-25 where the performance drops by .2 percentage points. The results of the models trained on RGDD indicate an increase in performance on MDB over the performance on ENST for all settings. Especially pronounced is the performance increase with regard to KD as shown in Figure 5.5.

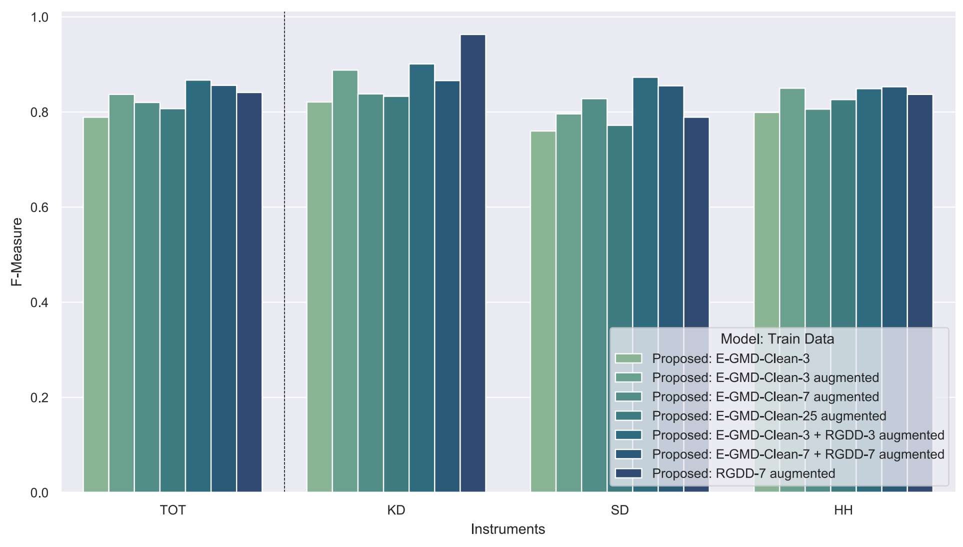


FIGURE 5.5: Overview of selected F-measures for the proposed model evaluated on MDB in a 3-class drums-only scenario.

7-classes

TABLE 5.2: F-measures of the 7-class scenario tested on drums-only input signals. All instruments are grouped into either KD, SD, TT, HH, CY, RD, or BE. Models that were trained on a dataset with the "-25" suffix group their predictions for ENST and MDB as shown in Figure 5.1. The first column of the E-GMD results show the performance of the proposed model on the E-GMD test set whereas the second column shows the results on the E-GMD-Clean test set where files with a deviation between the annotations and the audio of ≥ 6 ms were removed from the dataset. The augmentation approach of OaF-Drums and the proposed model are not identical. OaF-Drums employs a technique called *Shuffled mixup* whereas in case of the proposed model noise, pitch shift, gain-boost/reduction, and a bandpass filter were applied to the training data.

		F-measures		
Model	**Training Dataset**	**E-GMD**	**ENST**	**MDB**
OaF-Drums	E-GMD-7	.631 / -	.674	-
	E-GMD-7 *ShuMix*	.834 / -	.769	-
Proposed CRNN	E-GMD-7	.834 / .906	.753	.735
	E-GMD-Clean-7	.874 / .921	.739	.700
	E-GMD-Clean-7 *augm.*	**.876** / **.925**	.759	.772
	E-GMD-Clean-7 + RGDD-7	.868 / .913	.767	.718
	E-GMD-Clean-7 + RGDD-7 *augm.*	.849 / .892	**.770**	**.774**
	RGDD-7	.580 / .605	.655	.675
	RGDD-7 *augm.*	.663 / .686	.674	.683
	E-GMD-25	.758 / .803	.760	.719
	E-GMD-Clean-25	.798 / .840	.768	.756
	E-GMD-Clean-25 *augm.*	.805 / .850	.753	.738

Similar to Callender, Hawthorne, and Engel (2020), the proposed model was also trained to transcribe more instruments than the standard grouping of KD, SD, and HH. In the following, the results of the 7-class scenario are presented. The suffix "-7" denotes that the model was trained to distinguish between 7 different drum classes. In contrast to that, models with a suffix of "-25" indicate models that were trained to classify 25 instruments but whose predictions are grouped into the desired 7 classes when evaluated on ENST and MDB. Note, that unlike in the 3-class scenario ENST and MDB are used for evaluation purposes as IDMT only features 3 different drum types. The benchmark for both E-GMD and ENST is again Callender, Hawthorne, and Engel (2020) who scored an F-measure of .834 on E-GMD and .769 on ENST when training on augmented data. Table 5.2 shows that they are able to achieve a 20.3 percentage point increase on E-GMD and a 9.5 percentage point increase on ENST when compared to training on unmodified data.

The proposed model achieves an identical F-measure value of .834 when trained and tested on the uncleaned version of E-GMD-7. However, when training on E-GMD-Clean-7 the performance can be further improved by an additional 4 percentage points which reaches its best performance of .876 when training on augmented data. The same model scores an F-measure of .925 when evaluated on the clean version of E-GMD. Slightly worse results can be achieved when training additionally on RGDD-7 yielding an F-measure of .868 for the unmodified training data and .849 in the augmented setup. Here, the augmentation decreases the overall performance by about 2 percentage points. A comparison between the results of the models trained to classify 7 or 25 instruments shows that when tested on E-GMD(-Clean) the performance of the finer instrument complexity that is 25 instruments is worse. This finding is coherent with the results in Table 5.1 of the 3-class scenario. Training the proposed model on random drum sequences yields the overall worst performances which was already confirmed in the 3-class scenario.

When evaluating on ENST, the overall trend is also similar to the 3-class scenario. Training a model on more instruments than are finally classified and later grouping predictions according to the concept demonstrated in Figure 5.1 can improve the performance. Even though the results of the proposed model when trained on E-GMD-7 are better than on E-GMD-Clean-7 by 1.4 percentage points this changes for the better when looking at E-GMD(-Clean)-25 where training on E-GMD-Clean-25 results in a .8 percentage point increase. Furthermore, adding RGDD-7 as additional training data improves the overall performance by 2.8 percentage points over E-GMD-Clean-7. The best performance, however, can be achieved with the proposed model that is trained on E-GMD-Clean-7 + RGDD-7 augmented yielding an F-measure of .770. Finally, the results of the models trained only on RGDD-7 perform better on ENST than they do on E-GMD in this 7-class scenario which is in accordance with the findings from Table 5.1. Nevertheless, they still perform the worst in this setting but are still able to achieve the same results as the unmodified version of OaF-Drums when augmentation is applied.

With regard to MDB, the results indicate no clear trend whether a model that is initially trained to classify 25 instruments performs better than models are that trained to classify 7 instruments. In case of the unmodified version of E-GMD, the 25 class setup performs worse while performing better when it comes to E-GMD-Clean-7. Finally the augmented version of E-GMD-Clean favors a model that is trained on 7 classes and also transcribes 7 classes as compared to the model that is trained on 25 classes and groups its predictions to 7 classes. E-GMD-Clean-25 is also the only instance where augmentation decreases the overall performance. Otherwise, augmentation improves the performance resulting in the best score for E-GMD-Clean-7

+ RGDD-7 augmented with an F-measure of .774.

Figure 5.6 compares the results of the proposed model trained on E-GMD-7 and both versions of E-GMD-Clean-7 to the results of Callender, Hawthorne, and Engel (2020). However, the benchmark results had to be determined visually from Figure 2 from Callender, Hawthorne, and Engel (2020) since the authors do not list concrete numbers. Notwithstanding some inaccuracies of the benchmark results, the overall trend becomes clear.

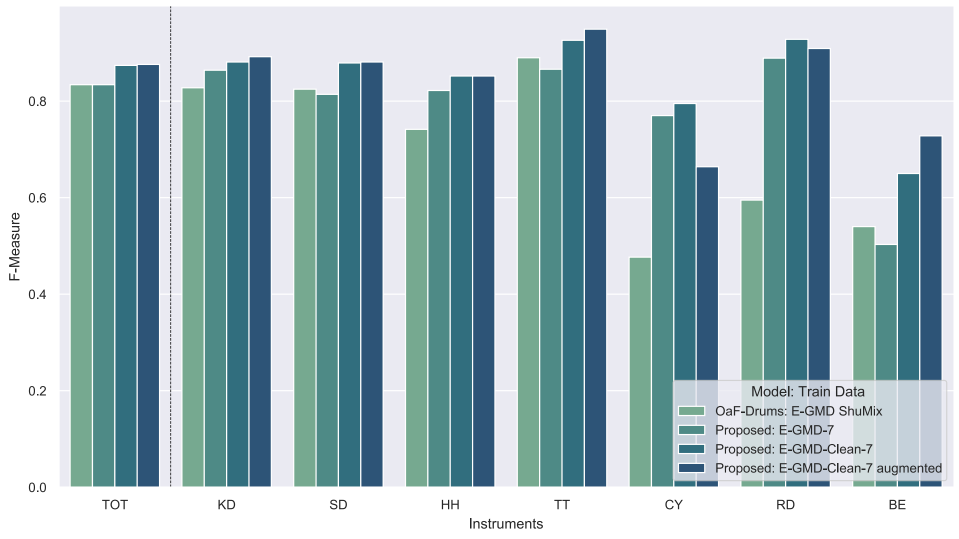


FIGURE 5.6: Selection of F-measures in a 7-class scenario comparing OaF-Drums to the proposed model tested on the uncleaned test set of E-GMD.

Almost every instrument F-measure can be improved by switching to the proposed model architecture. Exceptions to this are SD, TT, and BE in case the proposed is trained on the uncleaned train set of E-GMD. In those instances, the largest deviation is roughly 3.7 percentage points (BE). Otherwise, the proposed performs better with respect to CY and RD. Cleaning the train data on the other hand leads to major improvements especially with regard to high-pitched instruments like CY, RD, and BE.

25-classes

In addition to the aforementioned classification scenarios, a more advanced approach is presented in the following part of the thesis. Here, the proposed model was trained to distinguish between all 25 events of E-GMD as presented in Table 3.3. As a result of this, the outcome cannot be compared to a benchmark because Callender, Hawthorne, and Engel (2020) did not investigate this. On top of this, it is the most fine grained task for the model to solve, so it is expected to perform worse than in the 3- or 7-class scenario. Figure 5.7 shows the results for every single instrument present in E-GMD.

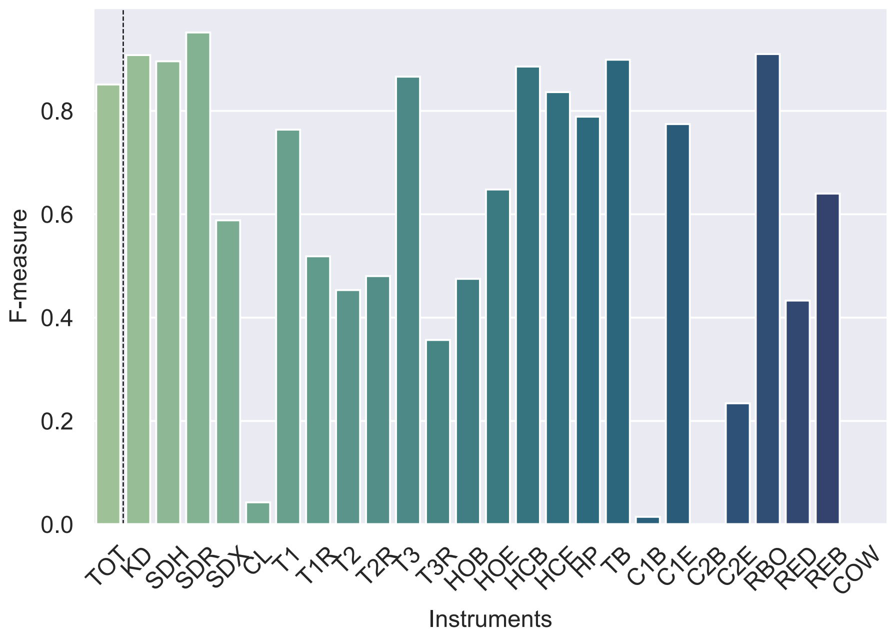


FIGURE 5.7: Overview of all F-measures for all 25 classes present in E-GMD. The results are based on the proposed model which was trained and evaluated on E-GMD-Clean-25

As shown in Tables 5.1 and 5.2, the best results when classifying all instruments of E-GMD(-Clean) can be achieved when training on E-GMD-Clean-25 augmented yielding an overall F-measure of .805 when evaluating on E-GMD-25 whereas an evaluation on E-GMD-Clean-25 results in a score of .850. Figure 5.7 shows that *Crash 1 Bow* (C1B), *Crash 2 Bow* (C2B), *Cowbell* (COW), and *Clap* (CL) display the worst performance among all instrument classes. In contrast to that, among the best performing instruments are *Kick Drum* (KD), *Ride Bow* (RBO), and *Snare Drum Rimshot* (SDR).

5.1.1 Discussion

In the first experiment, the proposed model was trained on various different training data configurations to then automatically transcribe drums from drums-only mixes of E-GMD, IDMT, ENST, and MDB. The results clearly suggest that training on clean data can mostly improve the performance of a model when compared to the performance of the same model trained on uncleaned data. For this, all files that displayed a deviation between the ground truth and the audio files $\geq 6\,\text{ms}$ were excluded from E-GMD. The reason for this decrease in performance with regard to training and testing on uncleaned data is probably the noise that is introduced to the data by the deviating annotations as it basically forces the DNN to guess where the actual onset is. An example of this is the performance of the proposed model trained on E-GMD-Clean-25 which scores an F-measure of .798 on the test split of E-GMD and therefore gaining a 4 percentage point advantage over the same model that is trained on the uncleaned data. This finding is also almost always consistent when evaluating on other datasets. The only exception to this is the evaluation on ENST where the F-measure decreases up to .7 percentage points when trained on E-GMD-Clean-3. However, this does not necessarily mean that this is always the case.

Since training a DNN is a statistical process, it might be that when averaging over a number of different training runs this decrease in performance would no longer be present. Besides that, it was shown that evaluating on clean test data can further improve the overall performance. When evaluating on E-GMD-Clean, results can in fact be improved by 7.2 percentage points compared to an evaluation on E-GMD as shown for the model trained on E-GMD-7. Paying attention to the preciseness of train and test data ultimately leads to an overall best performance of .927 for the proposed model when evaluated on E-GMD-Clean-3.

The results for the 7-class scenario shown in Table 5.2 indicate that the proposed model can in fact score better F-measures when trained on E-GMD by a margin of 20.3 percentage points which is the score *OaF-Drums* achieves only after applying augmentation to the training data. At first glance, this might indicate overfitting to the training data, however, when testing on ENST the results are also better compared to the benchmark when the training data is left unmodified. The same holds true for the results on IDMT.

Generally, the results of the evaluation on IDMT are worst for models that are trained on E-GMD(-Clean) only. The reason for this is probably the variance in quality of recordings of the *RealDrum* mixes and the sound selection of the synthesized drums of *TechnoDrum* and *WaveDrum*. Despite the fact that E-GMD contains electronic and synthetic drum kits as well, the model might overfit to the sounds present in the training data. This becomes especially notable with regard to the performance on SD. Listening to a selection of these files reveals for example that some KD hits sound similar to SD hits which makes it difficult for the DNN to distinguish between these two classes. Since the classification is not exclusive, which means that events can occur simultaneously, it might happen that in fact KD hits are both classified as such and SD hits as well. This intuition is confirmed by the resulting predictions shown Figure 5.8.

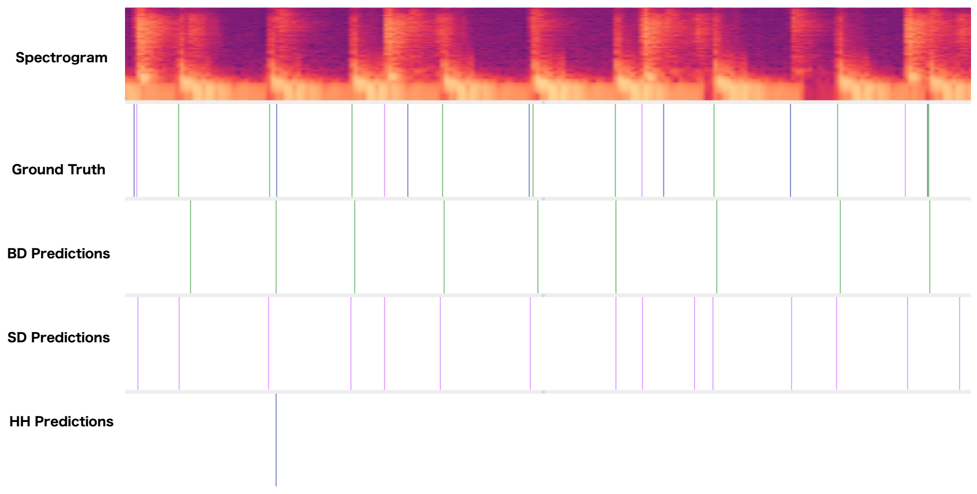


FIGURE 5.8: Juxtaposition of ground truth (upper row) to predictions (bottom 3 rows) of *TechnoDrum01_03#MIX* from IDMT using the proposed model trained on E-GMD-3.

Applying augmentation to the training data can substantially improve a model's performance on IDMT which is shown by an increase of 11.5 percentage points when trained for example on E-GMD-Clean-3 augmented. Nonetheless, this trend decreases the more different classes of instruments the model was initially trained to classify. Nonetheless, a comparison with OaF-Drums shows that the presented form of augmentation in this thesis is not as capable as the *Shuffled Mixup* approach with regard to IDMT and ENST because training on E-GMD-Clean only and applying augmentation to the training data cannot achieve better results, yet results can get close when training the model to solve a more complex task, that is to classify 7 instruments. In spite of this, adding additional data to the E-GMD-Clean training dataset can also serve as augmentation. That way it is possible to achieve results that surpass the benchmark by 2.8 percentage points on IDMT in the 3-class scenario and .1 percentage points for ENST in the 7-class scenario. This can be explained by the introduction of further variance in sounds as RGDD contains 20 different acoustic and 32 electronic drum kits yielding a total of 52 different drum kits which serves a another technique to prevent the proposed model from overfitting to the training data. Further, since the number of unique sequences in E-GMD is only 1,059, adding random sequences increases the overall amount of unique sequences and hence the variance in the data substantially. The performance of the models trained only on RGDD confirms this assumption because all of these models perform better than the ones trained on E-GMD(-Clean) only. On the other hand, they perform worse than the models that combine E-GMD-Clean and RGDD which can probably be explained by the fact that the number of events that occur simultaneously in the mixes of RGDD are fewer than in the mixes of E-GMD(-Clean) and therefore performing worse when multiple events happen at the same time.

The results of the evaluation on ENST show a notably increased performance over the results on IDMT in the 3-class scenario for models trained on E-GMD(-Clean) only. One reason for this might be that ENST does not contain files featuring synthesized but only acoustic drums which might make it easier for the model to transcribe drums from these files as the overlap in the frequency domain might not be as pronounced. Additionally, the quality of recordings does not vary as much as the ones from IDMT. As for models that were trained on RGDD, the performance drops compared the performance on IDMT. A reason for that might be the overall more complex structure of ENST compared to IDMT. Even though ENST features only 3 different drum kits, the number of playing styles varies from brushes, to sticks, and finally rods.

With regard to MDB, a similar behavior to the performance on IDMT can be observed. Models trained solely on E-GMD(-Clean) seem to be less capable of handling the varying quality of recordings from MDB which features recordings with for example a lot of reverberation or varying degrees of volume. This can be counteracted to some degree by applying augmentation to the training data either in the form of adding noise, pitch shifting, gain-boost/reduction, or filtering the signal, or by adding RGDD as additional training data while combining both approaches yields the best results, the latter of which holds true for all test datasets but E-GMD.

The experiments in the drums-only setting show that besides the combination of adding RGDD as additional training data and augmenting training data as described in Section 4.2 training the proposed model to classify exactly the number of target instruments in this setup yields the best results which means that in case of a 3-class scenario it is best to train the model to distinguish between the same 3 classes to begin with. The same is applicable to the 7-class scenario. On the other, should no additional training data be available, training the model on more classes than desired in the final transcription task results is the favorable approach since this can serve as another form of regularization.

5.2 Drum Stems

This section presents the results of a qualitative analysis of the performance of the proposed model on instrument stems which means that the test set only contains one single event type at a time per file, that is either KD, SD, or HH. These kinds of files are here referred to as *stems*. Although most use-cases probably feature drums-only mixes or full-mixes, this exploratory approach investigates a DNN's ability to generalize on data it did not see during training. No benchmark performance metrics can be given for comparison purposes because to the best knowledge of the author no related work has been done in this direction on the datasets presented in Chapter 3. To investigate the model's performance on audio files that only contain stems, four different training data setups are used. The first two approaches feature models that are trained on E-GMD-Clean only and E-GMD-Clean + RGDD, respectively. That way it can be determined if adding random drum sequences consisting of both mixes and stems can improve the model's ability to generalize on unseen data. Thirdly, the proposed model is trained on RGDD only which means that the training data only consists of random drum sequences. Finally, RGDD-Stems — a subset of RGDD — is used for training purposes to see how a model performs when trained on stems only.

In order to evaluate the performance of the various model / training data configurations, the stems of IDMT, first mentioned in Table 3.8, in particular the 180 stems of the *WaveDrum02* files (*IDMT-Stems*), are used. These files are isolated instrument tracks from *WaveDrum02* mixes that were intended to be used for source separation tasks by Dittmar and Gärtner (2014). Similar to the results in Section 5.1 and 5.3, Table 5.3 distinguish between three instrument grouping scenarios. The upper part of the table shows the results of the proposed model to classify three instruments (KD, SD, and HH). In contrast to that, the remaining results show the performance of models that were initially trained to classify 7 or 25 instruments respectively, but where the predictions were summarized down to KD, SD, and HH as demonstrated in Figure 5.1. Again, based on the findings of the drums-only mixes scenario presented in Section 5.1, only models that were either trained on E-GMD-Clean or RGDD are used for evaluation purposes.

The results shown in Table 5.3 present the F-measures of the evaluation with a 50 ms tolerance window so that they can be compared to the results of the evaluation on IDMT-Mixes. These values are identical to the results presented in Table 5.1 and are shown here for comparison purposes. Models that were trained only on E-GMD-Clean show overall the worst performance when evaluated on IDMT-Stems, yet they show a clear trend towards performing better the more instruments they were initially trained to classify. A comparison between these models shows an increase in performance of 12.4 percentage points for the unmodified version and 6.5 percentage points when changing from E-GMD-Clean-3 augmented to E-GMD-Clean-25 augmented yielding the best performance for a model solely trained on E-GMD-Clean with an F-measure of .717 for the model that was trained to classify 25 instruments.

TABLE 5.3: F-measures of an evaluation based on the stems from IDMT *WaveDrum02*. This subset of IDMT contains files that feature one single instrument that is KD, SD, or HH at a time. For the sake of comparison, the results of the same models on IDMT-Mixes from Table 5.1 are listed. All displayed F-measures are computed using a 50 ms tolerance window.

		F-measures	
Model	**Training Dataset**	**IDMT-Mixes**	**IDMT-Stems**
Proposed CRNN	E-GMD-Clean-3	.675	.550
	E-GMD-Clean-3 *augm.*	.790	.652
	E-GMD-Clean-3 + RGDD-3	.853	.859
	E-GMD-Clean-3 + RGDD-3 *augm.*	**.885**	.892
	RGDD-3	.803	.819
	RGDD-3 *augm.*	.838	.921
	RGDD-Stems-3	.455	.890
	RGDD-Stems-3 *augm.*	.582	.928
	E-GMD-Clean-7	.766	.662
	E-GMD-Clean-7 *augm.*	.837	.684
	E-GMD-Clean-7 + RGDD-7	.844	.875
	E-GMD-Clean-7 + RGDD-7 *augm.*	.829	.785
	RGDD-7	.815	.888
	RGDD-7 *augm.*	.833	**.941**
	RGDD-Stems-7	.450	.910
	RGDD-Stems-7 *augm.*	.353	.925
	E-GMD-Clean-25	.785	.674
	E-GMD-Clean-25 *augm.*	.798	.717

Adding RGDD to E-GMD-Clean as additional training data improves the performance of all models compared to models solely trained on E-GMD-Clean ranging from an increase of 10.1 percentage points for E-GMD-Clean-7 augmented to 30.9 percentage points for E-GMD-Clean-3. At the same time, the results show that when RGDD is part of the training dataset the performance of those models can be improved in comparison to IDMT-Mixes. The only exception to this is the model trained on E-GMD-Clean-7 + RGDD-7 augmented where the performance decreases by 4.4 percentage points.

If RGDD is the only dataset the proposed model is trained on, the results are either comparable or better than the models trained on mixed datasets. The model trained on RGDD-3 augmented for example is among the best performing models in this scenario scoring an F-measure of .921 on IDMT-Stems which is an increase of 8.3 percentage points over its performance on IDMT-Mixes. The overall best performance however is achieved when training on RGDD-7 augmented. This training setup yields an F-measure of .941 which is even better than models that were trained on stems only.

Training the proposed model on RGDD-Stems almost always yields F-measures above .900. The only exception to this is the proposed model trained on RGDD-Stems-3 which results in an F-measure of .890. Notwithstanding these good results on IDMT-Stems, it becomes also clear that models trained only on stems perform poorly when evaluated on mixes which can result in a performance decrease ranging from 34.6 percentage points for RGDD-Stems-3 augmented to 57.2 percentage points for RGDD-Stems-7 augmented. At the same time, this holds true for models that were trained on mixes that are then used to transcribe drums from stems. Figure 5.9 underlines the described set of problems based on examples taken from IDMT.

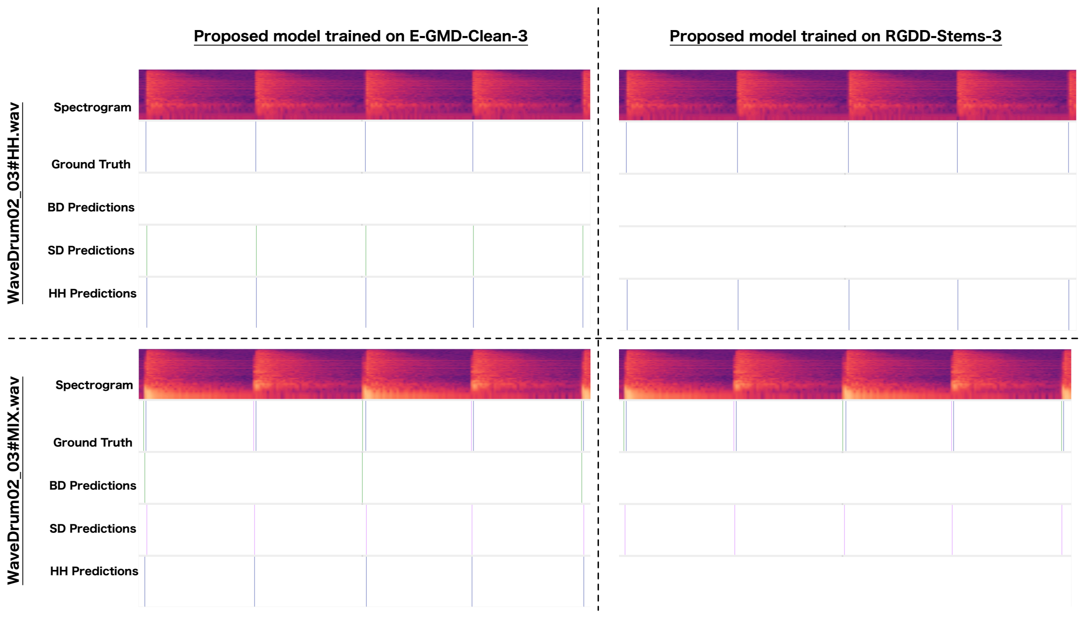
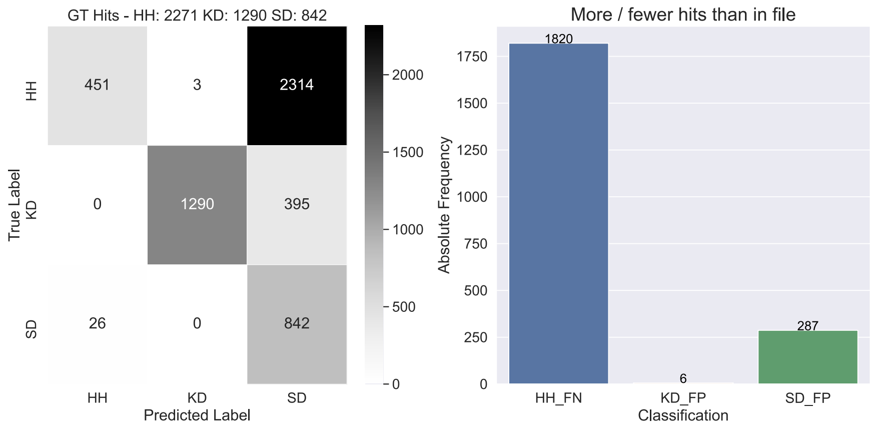
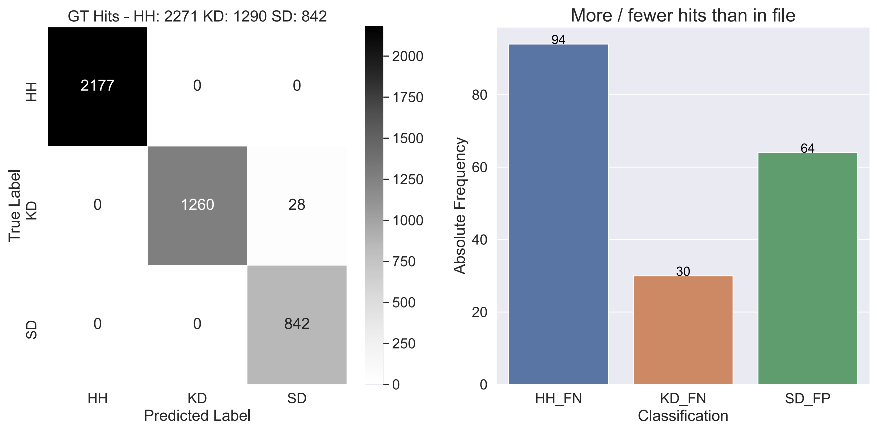


FIGURE 5.9: Comparison of exemplary results of the proposed model trained on either E-GMD-Clean-3 (left column) or RGDD-Stems-3 (right column). The first row presents the automatic drum transcription of these models based on *WaveDrum02_03#HH.wav* which only features hi-hat hits whereas the bottom row shows the results based on *WaveDrum02_03#MIX.wav* which features kick-drum, snare drum, and hi-hat hits. Both of these examples are taken from IDMT. This comparison underlines the problems that a model has when either stems or drums-only mixes are not present in the training data.

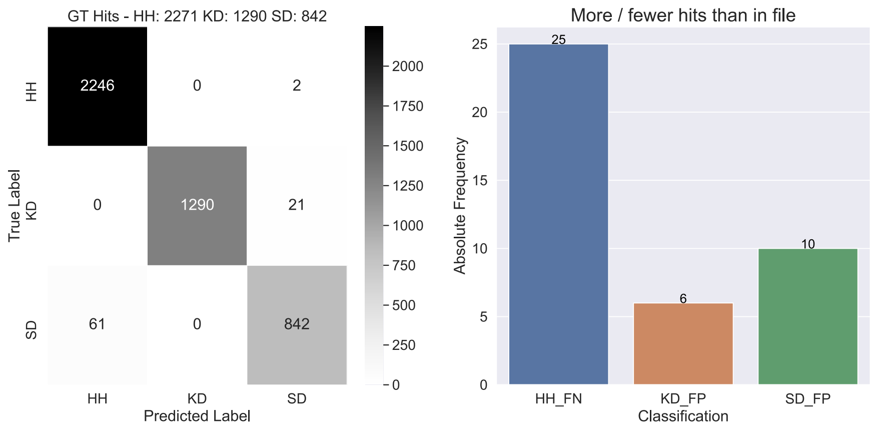
It becomes clear that the model trained on E-GMD-Clean-3 (drums-only mixes) predicts an additional SD for every HH hit in the selected section when evaluated on a stem file as shown in the upper row of Figure 5.9 whereas the model trained on RGDD-Stems-3 (stems only) correctly classifies the HH hits in the selected section. On the other hand, the evaluation on a drums-only mix presented in the bottom row shows that when trained on drums-only mixes the model is able to correctly classify numerous instrument hits with the exception of SD. Compared to this, the model which was trained only on stems is neither capable of transcribing events that happen simultaneously or close to this as it is the case for example for KD and HH or SD and HH. Instead, it only transcribes SD at all ground truth onset positions. These findings are underlined in a more granular fashion by the confusion matrices shown in Figure 5.10.



(A) Results when trained on E-GMD-Clean-3 .



(B) Results when trained on RGDD-Stems-3 augmented.



(C) Results when trained on RGDD-7 augmented.

FIGURE 5.10: Summary of classification results of single instrument audio files of IDMT *WaveDrum02* (*IDMT-Stems*). The left part of each subfigure shows a confusion matrix indicating how many instruments were correctly classified and if not what instruments they were confused with. On the other hand, the right column shows a bar plot indicating the amount of missing (FN) or false predictions (FP) for a file that either has more or fewer instrument hits than finally predicted.

The proposed model that was trained on E-GMD-Clean-3 has major problems to distinguish between hi-hats (HH) and snare drums (SD). This is especially pronounced in case of HHs that are mostly misclassified as SDs. Overall this results in 1,820 false negatives (FN) for HH. On the other hand, although all existing SD hits are correctly classified, there are 26 additional HH classifications where there are in fact no SD hits. 287 false positives (FP) for SD indicate that for example instead of classifying 5 out of 5 SD hits the model predicts more than 5. In addition to that, there is confusion between KD and SD events showing that the model confuses KD as SD.

As opposed to that, a model that is trained on RGDD-Stems-3 augmented outperforms the aforementioned model by a big margin. It becomes clear that this model does not confuse HHs and SDs with other instruments at all while predicting almost all HHs and therefore reducing the number of FNs. At the same time, the number of FNs for SDs is reduced substantially. Even though the number of FNs for KD increases, the confusion between KD and SD is almost completely reduced.

Finally, the best performing model is the one trained on RGDD-7 augmented. Figure 5.10 shows that the amount of FNs for HHs and FPs for SDs are further reduced. Notwithstanding this improvement, there is a small amount of confusion between SD and HH. Generally speaking, this model is able to classify almost all instruments correctly while reducing instrument confusions, FPs and FNs.

5.2.1 Discussion

Table 5.3 showed the results of a qualitative analysis where the proposed model was trained on either mixes, mixes plus additional random sequences, random sequences only, or simply random stems to then automatically transcribe drums from audio files that only contain single instrument hits (stems).

The findings suggest that models trained on mixes only, that is files that feature multiple instruments, generally exhibit a decrease in performance which indicates that these models seem to have learned the context in which drum instruments occur and therefore struggle to generalize when there is only single instruments in a file. This problem can be counteracted to some degree by adding data that contain stems. In this case, adding audio files that feature random mixes and stems, here in the form of RGDD, can in fact improve a model's performance on stems. However, the model trained on E-GMD-Clean-7 + RGDD-7 augmented performs unexpectedly poorly. Similar to E-GMD-Clean-3 + RGDD-3 augmented, it is to be expected that augmenting the training data results in a better performance when compared to its unmodified counterpart which is not the case here. A possible explanation might be the statistical nature of training a DNN. Therefore, picking a single experiment from a range of experiments might result in varying performances in some cases. Another approach that improves the performance of the proposed model when evaluating on stems is to train models to classify more instruments than needed in the final setting. By increasing the number of instruments in the training process and therefore complicating the task at hand, the model can be regularized to some extent, albeit it is not always the case as shown by the model trained on E-GMD-Clean-7 + RGDD-7 augmented or trained on RGDD-Stems augmented. Nevertheless, it is a valid approach that might help increase the overall performance.

5.3 Full-mixes

In the following section, the results of the transcription from full-mixes are presented and compared to the results of Vogl, Widmer, and Knees (2018). The performance of the proposed model on full-mixes is examined for the sake of investigating its potential in a different setting than it was trained in. As the amount of labeled data in a full-mix scenario is limited and the creation and annotation of such a dataset is labor-intensive, it is explored how effective a DNN that is trained only on drums-only mixes can be in a full-mix scenario. The benchmark model that is used for comparison purposes is referred to as *Vogl-Drums*. In contrast to the drums-only experiments in Section 5.1, the test datasets now contain a wide range of other instruments like for example guitar, bass, vocals, and keyboard in addition to drums. It is important to note that the results cannot be directly compared to each other because during the training process of Vogl-Drums, the DNN has seen all types of files that compose the individual dataset which means that full-mixes were also part of the training dataset. This is due to the fact that Vogl, Widmer, and Knees (2018) employ a 3-fold cross-validation strategy which is different from the approach applied in this thesis where designated train, validation, and test splits exist so that every dataset has its unique purpose. As a result of this, the performance of Vogl-Drums on full-mixes is expected to be better when compared to the proposed model's performance which was solely trained on drums-only mixes. Notwithstanding this fact, the results give an impression of how capable the proposed model is to generalize on unseen data.

For each test dataset, two F-measures are presented in Table 5.4. In case of Vogl-Drums, two different types deep neural network are used as reference, the first of which refers to the result of the CNN displayed in Figure 4.3a whereas the second one indicates the performance of the CRNN shown in Figure 4.3b. This is done due to the reason that sometimes CNN yields better results than the CRNN. The main difference between these particular deep neural networks is the last hidden layer of the network that changes from dense layers in the CNN to bi-directional GRUs in the CRNN. In case of the proposed model, the two F-measures per row do not indicate two different types of architectures but rather the differentiation between a more strict tolerance window in the left column and a wider tolerance window when calculating the overall F-measure. The left column shows the results when applying a tolerance window of 20 ms which is identical to Vogl, Widmer, and Knees (2018). This column serves as a comparison to Vogl-Drums whereas the right column shows the results of a tolerance window of 50 ms which is identical to Callender, Hawthorne, and Engel (2020) and therefore to the drums-only results presented in Section 5.1, the latter of which are presented to get an impression of what impact a wider tolerance window has on the overall performance.

3-classes

As a first step, all drum events are grouped into the same 3 classes similar to the first part of the drums-only scenario in Section 5.1. Based on the findings in that section, only models that are either solely trained on E-GMD-Clean, RGDD, or a combination of both are included in this section. At first glance, it becomes obvious that no training setup of the proposed model can outscore the benchmark. Despite this fact, competitive results can be achieved.

Table 5.4 shows a benchmark performance of .770 for Vogl-Drums on ENST. Unlike the proposed model, however, this is a convolutional neural network. Note, that Vogl, Widmer, and Knees (2018) take an F-measure value of .784 as state-of-the-art when automatically transcribing drum from the full-mixes of ENST. This value is based on the work by Vogl et al. (2017). However, since their work only takes the annotations of KD, SD, and HH into consideration while neglecting all other instruments, they cannot be directly compared to the described setting in this thesis. As a result of this, the F-measure of .770 by Vogl, Widmer, and Knees (2018) is taken as reference.

TABLE 5.4: F-measures of the 3-class scenario tested on full-mixes. All instruments are grouped into either KD, SD, or HH. Models that were trained on a dataset with the "-7/25" suffix group their predictions for ENST, MDB, and RBMA13 as shown in Figure 5.1. The benchmark results from Vogl, Widmer, and Knees (2018) demonstrate the evaluation outcome based on the full-mixes of ENST, MDB, and RBMA13 in the same 3-class scenario for a tolerance window of 20 ms. For the sake of comparison, the left column of the proposed model shows the results for the same tolerance window. In contrast to that, the right column presents the results for a tolerance window of 50 ms. It is important to note that full-mixes were part of the training dataset for Vogl-Drums which is due to the fact that a 3-fold cross-validation was applied. This is an advantage over the proposed model that has neither seen any drums-only files of ENST, MDB, and RBMA13 nor any full-mixes.

				F-measures
Model	**Training Dataset**	**ENST**	**MDB**	**RBMA13**
Vogl-Drums CNN	ENST & MDB & RBMA13	**.770** / -	**.720** / -	.630 / -
CRNN	ENST & MDB & RBMA13	.760 / -	.700 / -	**.640** / -
Proposed CRNN	E-GMD-Clean-3	.628 / .667	.451 / .513	.390 / .431
	E-GMD-Clean-3 *augm.*	.589 / .663	.519 / .565	.429 / .470
	E-GMD-Clean-3 + RGDD-3	*.669* / *.706*	.552 / .619	.540 / .571
	E-GMD-Clean-3 + RGDD-3 *augm.*	.664 / *.706*	.573 / .635	.522 / .564
	RGDD-3	.651 / .698	.575 / .649	.560 / .590
	RGDD-3 *augm.*	.615 / .676	.574 / .640	.545 / .583
	E-GMD-Clean-7	.589 / .633	.413 / .480	.274 / .332
	E-GMD-Clean-7 *augm.*	.645 / .695	.538 / .608	.483 / .532
	E-GMD-Clean-7 + RGDD-7	.658 / .698	.557 / .622	.498 / .529
	E-GMD-Clean-7 + RGDD-7 *augm.*	.640 / .686	.531 / .606	.496 / .540
	RGDD-7	.646 / .687	.596 / .649	*.569* / .596
	RGDD-7 *augm.*	.619 / .680	*.604* / *.663*	*.569* / *.605*
	E-GMD-Clean-25	.626 / .672	.444 / .520	.370 / .412
	E-GMD-Clean-25 *augm.*	.652 / .703	.533 / .620	.468 / .523

In contrast to the results in the drums-only scenario, the performance of the model solely trained on E-GMD-Clean-25 augmented is better than the unmodified version of the training dataset when compared to the evaluation results of the model trained on E-GMD-Clean-3 where the unmodified version performs better than its augmented counterpart. Overall, the proposed model trained on E-GMD-Clean augmented performs better the more instruments it was initially trained to classify. This is similar to the findings in the drums-only scenario with the exception of E-GMD-Clean-25 augmented. Adding RGDD as additional training data to E-GMD-Clean can improve the performance of all models but the one trained on E-GMD-Clean-7 model when evaluating on ENST. Models that were trained only on RGDD in fact achieve comparable or even higher scores than models that were trained on E-GMD-Clean only or the combination of both. However, the best performing model with regard to automatically transcribing 3 classes from ENST is achieved with the proposed model trained on E-GMD-Clean-3 + RGDD-3 scoring an F-measure of .669 while its augmented version only scores .5 percentage points worse. When comparing the best results of the proposed model to the results of Vogl-Drums which includes ENST full-mixes among others in the training set, the difference amounts to roughly 10 percentage points.

With regard to individual instrument performance, Figure 5.11 shows that increasing the level of instrument complexity can improve the performance for KD and SD only for HH it decreases slightly. On top of that, adding RGDD as additional training data can also improve the performance when the instrument level complexity is set to 3 whereas for 7 instruments it decreases with respect to KD.

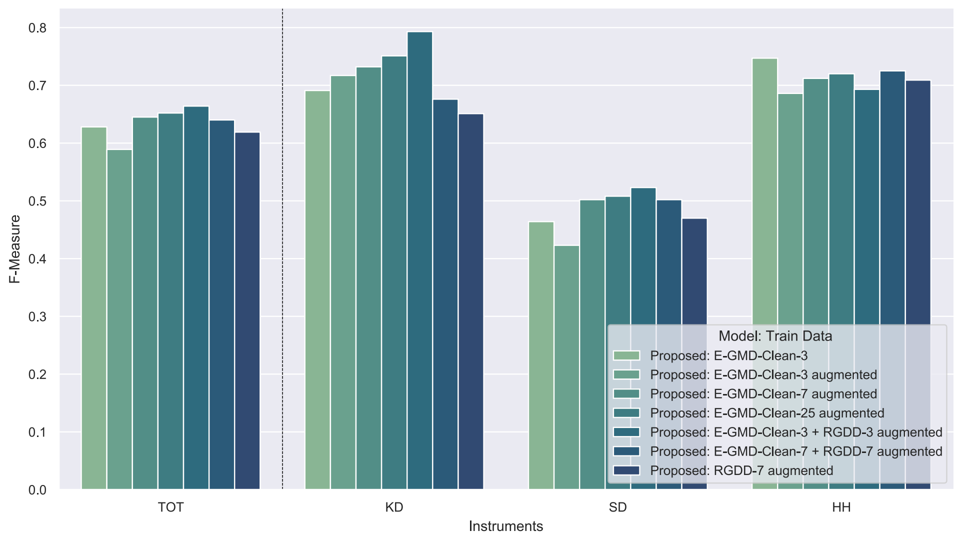


FIGURE 5.11: Overview of selected F-measures for the proposed model evaluated on ENST in a 3-class full-mix scenario.

In case of MDB, all results turn out to be worse when compared to the evaluation on ENST. This time, the best performing model is trained on RGDD-7 augmented resulting in an F-measure of .604. Compared to the benchmark, this corresponds to a difference of 11.6 percentage points. The worst results are achieved for models that were trained on the unmodified versions of E-GMD-Clean. However, the results on MDB suggest that augmenting the training dataset almost always yields an increase in performance. Only E-GMD-Clean-7 + RGDD-7 augmented performs around 2 percentage points worse than its unmodified counterpart. The same holds true for adding RGDD as additional training data. Figure 5.12 demonstrates the influence augmenting / adding data or increasing the level of instrument complexity can have on KD, SD, and HH F-measures.

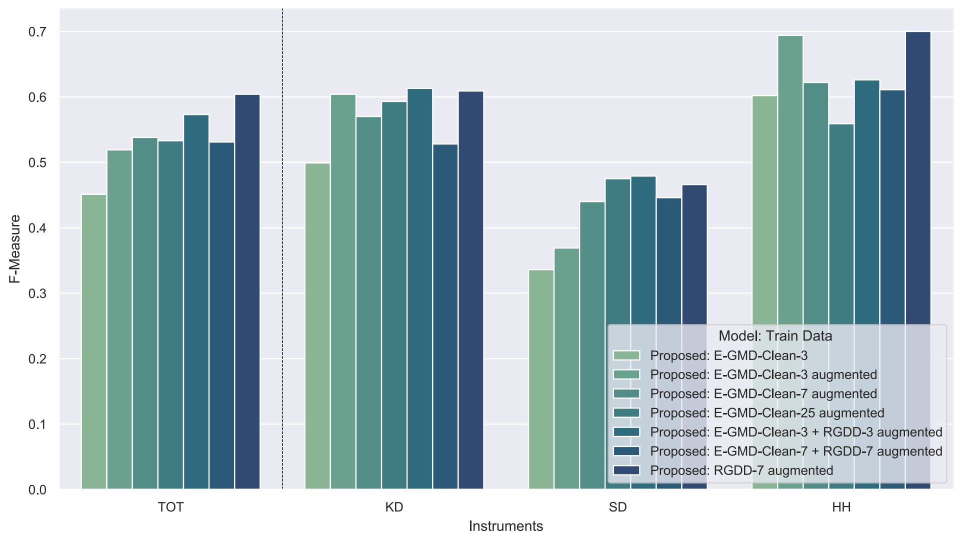


FIGURE 5.12: Overview of selected F-measures for the proposed model evaluated on MDB in a 3-class full-mix scenario.

Finally, the evaluation of the models on RBMA13 shows similar findings to the evaluation on ENST with the difference that the performance generally decreases which ranges from 1.2 percentage points for E-GMD-Clean-3 + RGDD-3 to 13.9 percentage points for E-GMD-Clean-7. The best performing model is again trained on RGDD-7 augmented yielding an F-measure of .569 which is matched by its unmodified version. A comparison to the benchmark shows a difference of 7.1 percentage points. Similar to ENST, the worst performing model is again trained on E-GMD-Clean-7 which only scores an F-measure of .274. This score is mainly based on the poor performance on SD and HH as shown in Figure 5.13 which indicates that augmentation helps improve these metrics drastically.

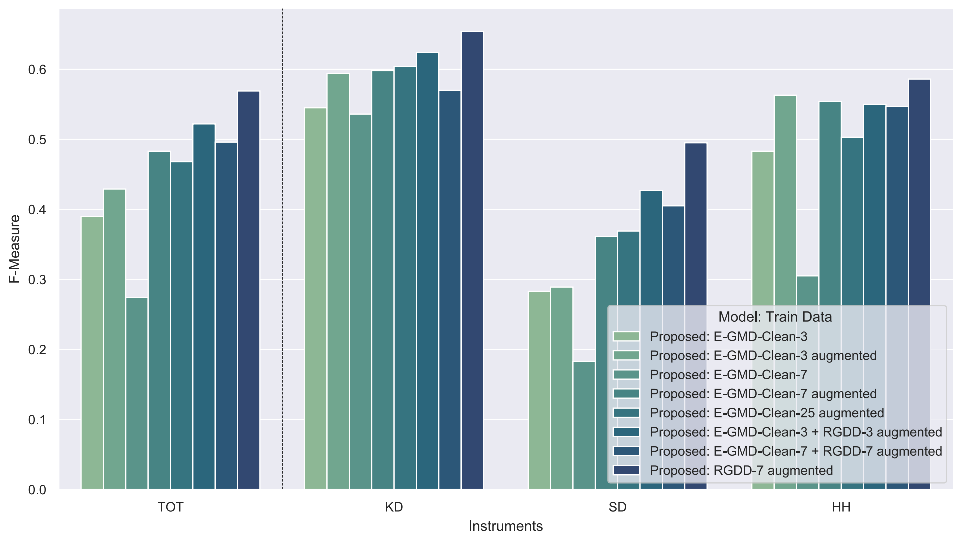


FIGURE 5.13: Overview of selected F-measures for the proposed model evaluated on RBMA13 in a 3-class full-mix scenario.

7-classes

Following the 3-class scenario is the evaluation on 7 classes, as opposed to the results from Vogl, Widmer, and Knees (2018) who group all instruments of ENST, MDB, and RBMA13 into 8 classes. Compared to the grouping of instruments presented in this thesis, they treat *clave/sticks* as an individual class. Therefore, the results cannot be directly compared with each other. Nevertheless, they give a good intuition about the performance of the proposed model since the difference in the number of instruments is marginal. On top of that, the overall number of occurrences of this instrument class in the full-mixes of ENST, MDB, and RBMA is either 0 or not among the most represented instruments as shown in Figures 3.12, 3.13, and 3.14.

Similar to the previously presented results of the 3-class scenario, the same tendency in the 7-class scenario can be seen where Vogl-Drums also performs better on ENST, MDB and RBMA13. Again, this is due to the fact that the model has seen full-mixes among others during training. Table 5.5 shows a result of .700 for Vogl-Drums when evaluated on ENST.

TABLE 5.5: F-measures of the 7-class scenario tested on full-mixes input signals. All instruments are grouped into either KD, SD, TT, HH, CY, RD, or BE. Models that were trained on a dataset with the "-25" suffix group their predictions for ENST, MDB, and RBMA13 as shown in Figure 5.1. The benchmark results from Vogl, Widmer, and Knees (2018) demonstrate the evaluation outcome based on the full-mixes of MDB in an 8-class scenario for a tolerance window of 20 ms. For the sake of comparison, the left column of the proposed model shows the results for the same tolerance window. In contrast to that, the right column presents the results for a tolerance window of 50 ms. It is important to note that full-mixes were part of the training dataset for Vogl-Drums which is due to the fact that a 3-fold cross-validation was applied. This is an advantage over the proposed model that has neither seen any drums-only files of ENST, MDB, and RBMA13 nor any full-mixes.

				F-measures
Model	**Training Dataset**	**ENST**	**MDB**	**RBMA13**
Vogl-Drums CNN	ENST & MDB & RBMA13	.630 / -	**.650** / -	.440 / -
CRNN	ENST & MDB & RBMA13	**.700** / -	.630 / -	**.500** / -
Proposed CRNN	E-GMD-Clean-7	.480 / .513	.337 / .393	.241 / .289
	E-GMD-Clean-7 *augm.*	.482 / .512	.383 / .434	.353 / .396
	E-GMD-Clean-7 + RGDD-7	.497 / .521	.378 / .423	.379 / .405
	E-GMD-Clean-7 + RGDD-7 *augm.*	.469 / .496	.381 / .437	.388 / .430
	RGDD-7	.438 / .462	.360 / .389	.411 / .431
	RGDD-7 *augm.*	.458 / .494	*.410* / .443	*.451* / *.481*
	E-GMD-Clean-25	.503 / .535	.388 / .455	.323 / .396
	E-GMD-Clean-25 *augm.*	*.545* / *.581*	.406 / *.478*	.347 / .360

The model that comes closest to this is trained on E-GMD-Clean-25 augmented scoring an F-measure of .545 which corresponds to a difference between the benchmark and the proposed model of 15.5 percentage points. All other models perform worse ranging from a 26.2 percentage point difference for RGDD-7 to 19.7 percentage points for E-GMD-Clean-25 which makes the models trained on RGDD the worst performing ones on ENST.

However, this changes with respect to MDB. Here, the model that was trained on RGDD-7 augmented performs the best with an F-measure of .410. When comparing this to the benchmark of .650, this yields a difference of 24 percentage points. Besides that, similar to the results of the evaluation on ENST, the proposed model trained on E-GMD-Clean-25 augmented scores the second best F-measure with a value of .406. In this evaluation scenario, the model trained on E-GMD-Clean-7 yields the worst performance which is similar to the findings in the 3-class scenario.

With regard to RBMA13, the proposed model trained on RGDD-7 augmented is in fact able to surpass the score of the CNN baseline from Vogl, Widmer, and Knees (2018) by 1.1 percentage points with an F-measure of .451. Nonetheless, this is not enough to surpass the CRNN baseline that scores an F-measure of .500. Contrary to the evaluation on ENST and MDB, the results for RBMA13 indicate that the models trained 25 instruments are not among the best ones. This time, adding RGDD as additional training data improves the overall performance especially regarding the training on E-GMD-Clean-7 which only scores an F-measure of .241 as compared to scoring an F-measure of .379 when RGDD-7 is added.

5.3.1 Discussion

In the full-mix scenario, the proposed model's ability to perform ADT on audio files that contain accompanying instruments besides drums was investigated in Section 5.3. For this purpose, RBMA13 was used as an additional test set because it features files that are commercially produced songs. In order to be able to gauge the performance of the proposed model, *Vogl-Drums* is used as a benchmark which was introduced by Vogl, Widmer, and Knees (2018). As described in Section 5.3, the comparison with the benchmark is not completely fair because it contains full-mixes in the training process and is therefore expected to outperform the proposed model which did not see any full-mixes during training. This expectation is indeed confirmed by the findings in both the 3- and 7-class scenario. As for the 3-class scenario, Vogl-Drums has an advantage ranging from 7.1 percentage points in case of RBMA13 to 11.6 percentage on MDB while performing from 4.9 percentage points better on RBMA13 up to 25 percentage points on MDB. One main reason for this is probably an overlap of instruments in the time and frequency domain in the full-mixes and therefore making it more difficult for models that were solely trained on drums-only mixes. Despite this fact, the results show that even though the proposed model was not trained on full-mixes, it can still achieve scores that seem promising. Hence, it can be assumed that when trained on full-mixes the overall performance can at least get in the same range as the benchmark.

Generally speaking, the results of the 3-class scenario show the best performances on ENST when the DNN is trained on the unmodified version of both E-GMD-Clean-3 and RGDD-3 while still performing comparably well in the 7 instrument setup of the same training data or when trained on E-GMD-Clean-25 augmented. In contrast to that, the evaluation on MDB and RBMA show an advantage of models that were trained on random drum sequences only, namely RGDD-7 augmented. The results of the evaluation on MBD suggest that for MDB the accompanying instruments might in fact mask the drums in the audio more than it is the case for ENST as the overall performance drops for all models when evaluated on MDB. A model that was initially trained to solve a more difficult task, that is being trained to transcribe more instruments than finally needed, can be favorable. When comparing the full-mixes of ENST and MDB sonically, it is easy to tell that the ones from MDB sound more natural and homogeneous since they are real recordings as opposed to the full-mixes of ENST that feature a lot of MIDI accompaniments which make the drums stand out of the mix. Therefore, one possible explanation for the better performance on ENST might be the more unnatural accompaniment as well as the volume balance between accompaniment and drums.

The same applies to RBMA13 which seems to be the most challenging ADT task in this setting. One major contributing factor is the overall complexity and variety of both drum set sounds featuring distorted samples among others as well as the accompanying instruments of the songs in RBMA13. They are commercially produced tracks mostly featuring synthesized sounds in the realm of electronic dance music (EDM) and techno which makes it difficult for the DNN to distinguish between e.g. a synthesizer playing a bass note or playing a synthetic kick drum because they can look similar in the frequency domain. This is probably the reason why the proposed model trained on RGDD-7 augmented performs best in both the 3- and 7-class scenario when evaluated on RBMA13 as it features a wide variety of sounds.

Notwithstanding the reasonably good results of the benchmark models, it remains a challenge to transcribe drums from full-mixes. Therefore, more labeled data of drums in a full-mix scenario are necessary to be able to further improve the performance of deep neural networks in ADT tasks. Otherwise, models that did not see full-mixes during training will not yield a competitive outcome.

Chapter 6

Conclusion and Outlook

In this thesis, deep neural networks were employed to automatically transcribe drums from audio files that contain either drums only or accompanying instruments in addition to drums. The proposed model for this ADT task was inspired by the models from Callender, Hawthorne, and Engel (2020) and Vogl, Widmer, and Knees (2018) which serve as benchmarks in the two aforementioned scenarios. In addition to that, a qualitative analysis on files that only contain one single drum event type at a time was done. The publicly available dataset called *E-GMD* introduced by Callender, Hawthorne, and Engel (2020) served as the fundamental training data.

A comprehensive analysis of E-GMD as training data has underlined the importance of clean data with respect to the performance on benchmark datasets. It has become obvious that when removing files from E-GMD with a deviation of $\geq 6\,\mathrm{ms}$ between audio and MIDI files the employed evaluation metric could be improved in almost all cases. Furthermore, it was shown, based on the results of the proposed model on IDMT-Mixes, that the suggested technique of augmenting audio files cannot compete with the *Shuffled mixup* approach presented by Callender, Hawthorne, and Engel (2020). However, the evaluation of the 7-event scenario on the drums-only mixes of ENST has shown that the proposed model is indeed capable of scoring comparable results even without applying augmentation to the training data.

Besides that, two additional methods of regularization were presented, the first of which is adding a newly created dataset consisting of random drum sequences — called RGDD — to E-GMD in order to increase the variance of the training dataset. It has been shown that in the majority of cases adding RGDD as additional training data improved the performance when evaluated on IDMT, ENST, MDB, or RBMA in comparison to the corresponding model that was not trained on RGDD as supplementary training data. Moreover, models trained solely on RGDD can indeed outperform models that were trained on E-GMD or the combination of both in a 3-event scenario when evaluated on IDMT, MDB, and RBMA13. A conclusion from this finding is that it is not necessary to have drum sequences played by humans for ADT as random data proved to be a good starting point as well. The second type of regularization method proposed in this thesis has been to train a model to classify more instruments than needed in the final ADT task and later group this finer level of predictions to a more coarse grouping. This approach has proven to be another valid method in many evaluations if no additional training data is available.

Finally, the evaluation results of the drum stems have shown that models which are trained on drums-only mixes struggle to generalize on data that is not represented in the training data by any means. Every model that was trained only on E-GMD(-Clean) has shown to be performing worse on IDMT-Stems than on IDMT-Mixes. Even though this finding is somewhat counter-intuitive as it might seem easier to transcribe drum events from files that only contain one single drum event, it confirms the inherent characteristic of the supervised machine learning approach where the DNN can only learn to generalize well when presented with a large variance in the data during training. The same holds true for models that were trained on stems only since they encounter problems of generalization when evaluated on drums-only mixes. While adding RGDD to E-GMD-Clean during training can already substantially improve the performance of a DNN on stems, training only on RGDD yielded the best results. This confirms again the finding from the ADT in the drums-only and full-mixes scenario where randomly generated data proved to be a legitimate approach to tackling ADT.

This work has shown that data is at the core of deep learning tasks. The data-centric approach presented here demonstrated that altering the training data or introducing further variance to the training data can result in substantial performance gains. The augmentation technique presented by Callender, Hawthorne, and Engel (2020) seems to be a promising starting point towards this direction while showing that additional data is needed for good generalization on publicly available datasets or (supposedly) easy ADT task on stems. Future work should also further focus on applying semi-supervised methods like few-shot learning to ADT tasks as presented by Wang et al. (2020) or completely unsupervised methods as shown by Choi and Cho (2019) as large labeled datasets are rare or not completely open to the community due to copyright issues like the one presented by Zehren, Alunno, and Bientinesi (2021). This becomes notably more relevant as the sonic complexity of drum sets especially with regard to electronic or synthetic drum sets in commercially produced music varies a lot.

Future work should also focus on improving the performance of ADT with regard to full-mixes by either training on more labeled data comprising full-mixes or to combine ADT with source separation. Défossez (2021) introduced a DNN called *Demucs* that performs especially well on separating drums from full-mixes which could be combined with a model that focuses on ADT to yield good results in a full-mix scenario.

In addition to that, it should be investigated if a DNN expresses a bias towards certain genres or tempos as E-GMD turns out to be unbalanced with regard to these two parameters.

Lastly, more work needs to be done in the direction of automatically transcribing the velocity of drum events as this is a key factor when it comes to natural sounding drum sequences. Creating synthetic datasets in the form of random drum sequences can be a viable approach to create large datasets that contain velocity annotations.

Appendix A

Odd E-GMD Files

A.1 Audio

TABLE A.1: Overview of the audio files from E-GMD that are empty. These files are not part of any version of E-GMD-Clean

#	File	#	File
1	drummer7/session3/25_hiphop_67_fill_4-4_23.wav	23	drummer7/session3/25_hiphop_67_fill_4-4_52.wav
2	drummer7/session3/25_hiphop_67_fill_4-4_21.wav	24	drummer7/session3/25_hiphop_67_fill_4-4_51.wav
3	drummer7/session3/25_hiphop_67_fill_4-4_25.wav	25	drummer7/session3/25_hiphop_67_fill_4-4_19.wav
4	drummer7/session3/25_hiphop_67_fill_4-4_36.wav	26	drummer7/session3/25_hiphop_67_fill_4-4_26.wav
5	drummer7/session3/25_hiphop_67_fill_4-4_18.wav	27	drummer7/session3/25_hiphop_67_fill_4-4_28.wav
6	drummer7/session3/25_hiphop_67_fill_4-4_1.wav	28	drummer7/session3/25_hiphop_67_fill_4-4_33.wav
7	drummer7/session3/25_hiphop_67_fill_4-4_8.wav	29	drummer7/session3/25_hiphop_67_fill_4-4_53.wav
8	drummer7/session3/25_hiphop_67_fill_4-4_54.wav	30	drummer7/session3/25_hiphop_67_fill_4-4_32.wav
9	drummer7/session3/25_hiphop_67_fill_4-4_16.wav	31	drummer7/session3/25_hiphop_67_fill_4-4_57.wav
10	drummer7/session3/25_hiphop_67_fill_4-4_42.wav	32	drummer7/session3/25_hiphop_67_fill_4-4_35.wav
11	drummer7/session3/25_hiphop_67_fill_4-4_5.wav	33	drummer7/session3/25_hiphop_67_fill_4-4_6.wav
12	drummer7/session3/25_hiphop_67_fill_4-4_43.wav	34	drummer7/session3/25_hiphop_67_fill_4-4_10.wav
13	drummer7/session3/25_hiphop_67_fill_4-4_12.wav	35	drummer7/session3/25_hiphop_67_fill_4-4_41.wav
14	drummer7/session3/25_hiphop_67_fill_4-4_24.wav	36	drummer7/session3/25_hiphop_67_fill_4-4_14.wav
15	drummer7/session3/25_hiphop_67_fill_4-4_56.wav	37	drummer7/session3/25_hiphop_67_fill_4-4_3.wav
16	drummer7/session3/25_hiphop_67_fill_4-4_11.wav	38	drummer7/session3/25_hiphop_67_fill_4-4_17.wav
17	drummer7/session3/25_hiphop_67_fill_4-4_55.wav	39	drummer7/session3/25_hiphop_67_fill_4-4_44.wav
18	drummer7/session3/25_hiphop_67_fill_4-4_34.wav	40	drummer7/session3/25_hiphop_67_fill_4-4_37.wav
19	drummer7/session3/25_hiphop_67_fill_4-4_31.wav	41	drummer7/session3/25_hiphop_67_fill_4-4_4.wav
20	drummer7/session3/25_hiphop_67_fill_4-4_13.wav	42	drummer7/session3/25_hiphop_67_fill_4-4_2.wav
21	drummer7/session3/25_hiphop_67_fill_4-4_15.wav	43	drummer7/session3/25_hiphop_67_fill_4-4_58.wav
22	drummer7/session3/25_hiphop_67_fill_4-4_22.wav		

A.2 MIDI

TABLE A.2: Overview of the files from E-GMD where MIDI events occur after their audio counterparts, Part 1/4.

#	File	#	File
1	drummer7/session3/146_soul_105_fill_4-4_52.midi	69	drummer7/session3/156_soul_98_fill_4-4_33.midi
2	drummer7/session3/63_funk_112_fill_4-4_10.midi	70	drummer3/session2/2_rock_100_beat_4-4_19.midi
3	drummer3/session2/2_rock_100_beat_4-4_41.midi	71	drummer7/session3/149_soul_105_fill_4-4_11.midi
4	drummer7/session2/81_country_78_fill_4-4_10.midi	72	drummer7/session3/146_soul_105_fill_4-4_55.midi
5	drummer1/session1/5_jazz-funk_116_beat_4-4_52.midi	73	drummer7/session3/156_soul_98_fill_4-4_15.midi
6	drummer1/session1/5_jazz-funk_116_beat_4-4_18.midi	74	drummer7/session2/81_country_78_fill_4-4_8.midi
7	drummer7/session3/109_rock_95_beat_4-4_43.midi	75	drummer1/session1/5_jazz-funk_116_beat_4-4_8.midi
8	drummer7/session3/25_hiphop_67_fill_4-4_28.midi	76	drummer7/session3/25_hiphop_67_fill_4-4_19.midi
9	drummer3/session2/2_rock_100_beat_4-4_2.midi	77	drummer7/session2/81_country_78_fill_4-4_51.midi
10	drummer3/session1/9_rock_105_beat_4-4_14.midi	78	drummer7/session3/149_soul_105_fill_4-4_5.midi
11	drummer7/session3/109_rock_95_beat_4-4_1.midi	79	drummer7/session3/109_rock_95_beat_4-4_33.midi
12	drummer7/session2/81_country_78_fill_4-4_26.midi	80	drummer7/session1/15_jazz_112_beat_4-4_3.midi
13	drummer7/session3/109_rock_95_beat_4-4_6.midi	81	drummer7/session3/156_soul_98_fill_4-4_58.midi
14	drummer7/session3/156_soul_98_fill_4-4_2.midi	82	drummer7/session3/63_funk_112_fill_4-4_17.midi
15	drummer3/session1/9_rock_105_beat_4-4_55.midi	83	drummer7/session3/63_funk_112_fill_4-4_33.midi
16	drummer3/session1/9_rock_105_beat_4-4_35.midi	84	drummer3/session1/9_rock_105_beat_4-4_24.midi
17	drummer3/session1/9_rock_105_beat_4-4_22.midi	85	drummer1/session1/5_jazz-funk_116_beat_4-4_56.midi
18	drummer7/session1/15_jazz_112_beat_4-4_25.midi	86	drummer10/session1/8_jazz-swing_215_beat_4-4_28.midi
19	drummer7/session1/15_jazz_112_beat_4-4_24.midi	87	drummer7/session3/149_soul_105_fill_4-4_33.midi
20	drummer7/session3/25_hiphop_67_fill_4-4_25.midi	88	drummer3/session2/2_rock_100_beat_4-4_12.midi
21	drummer7/session3/109_rock_95_beat_4-4_31.midi	89	drummer7/session3/25_hiphop_67_fill_4-4_21.midi
22	drummer7/session2/81_country_78_fill_4-4_43.midi	90	drummer3/session2/2_rock_100_beat_4-4_10.midi
23	drummer7/session3/25_hiphop_67_fill_4-4_16.midi	91	drummer7/session3/149_soul_105_fill_4-4_42.midi
24	drummer1/session1/5_jazz-funk_116_beat_4-4_13.midi	92	drummer7/session2/81_country_78_fill_4-4_31.midi
25	drummer7/session3/156_soul_98_fill_4-4_5.midi	93	drummer7/session2/81_country_78_fill_4-4_33.midi
26	drummer7/session1/15_jazz_112_beat_4-4_19.midi	94	drummer7/session3/149_soul_105_fill_4-4_23.midi
27	drummer3/session1/9_rock_105_beat_4-4_53.midi	95	drummer3/session2/2_rock_100_beat_4-4_33.midi
28	drummer3/session1/9_rock_105_beat_4-4_2.midi	96	drummer7/session3/63_funk_112_fill_4-4_37.midi
29	drummer7/session3/149_soul_105_fill_4-4_21.midi	97	drummer10/session1/8_jazz-swing_215_beat_4-4_42.midi
30	drummer3/session1/9_rock_105_beat_4-4_58.midi	98	drummer1/session1/5_jazz-funk_116_beat_4-4_51.midi
31	drummer7/session3/25_hiphop_67_fill_4-4_8.midi	99	drummer7/session3/63_funk_112_fill_4-4_21.midi
32	drummer7/session3/149_soul_105_fill_4-4_16.midi	100	drummer3/session1/9_rock_105_beat_4-4_43.midi
33	drummer7/session3/156_soul_98_fill_4-4_44.midi	101	drummer7/session3/149_soul_105_fill_4-4_35.midi
34	drummer7/session3/109_rock_95_beat_4-4_21.midi	102	drummer3/session1/9_rock_105_beat_4-4_33.midi
35	drummer3/session1/9_rock_105_beat_4-4_34.midi	103	drummer7/session3/156_soul_98_fill_4-4_35.midi
36	drummer7/session3/109_rock_95_beat_4-4_15.midi	104	drummer7/session1/15_jazz_112_beat_4-4_42.midi
37	drummer7/session2/81_country_78_fill_4-4_15.midi	105	drummer7/session3/63_funk_112_fill_4-4_24.midi
38	drummer3/session1/9_rock_105_beat_4-4_54.midi	106	drummer3/session1/9_rock_105_beat_4-4_44.midi
39	drummer10/session1/8_jazz-swing_215_beat_4-4_10.midi	107	drummer7/session3/25_hiphop_67_fill_4-4_5.midi
40	drummer7/session3/149_soul_105_fill_4-4_18.midi	108	drummer7/session3/146_soul_105_fill_4-4_58.midi
41	drummer1/session1/5_jazz-funk_116_beat_4-4_34.midi	109	drummer7/session2/81_country_78_fill_4-4_53.midi
42	drummer7/session3/25_hiphop_67_fill_4-4_56.midi	110	drummer7/session3/25_hiphop_67_fill_4-4_23.midi
43	drummer7/session3/109_rock_95_beat_4-4_32.midi	111	drummer1/session3/8_rock_135_beat_4-4_33.midi
44	drummer7/session3/149_soul_105_fill_4-4_52.midi	112	drummer7/session3/25_hiphop_67_fill_4-4_26.midi
45	drummer7/session3/25_hiphop_67_fill_4-4_43.midi	113	drummer7/session3/109_rock_95_beat_4-4_58.midi
46	drummer7/session3/146_soul_105_fill_4-4_11.midi	114	drummer7/session3/25_hiphop_67_fill_4-4_36.midi
47	drummer7/session3/146_soul_105_fill_4-4_16.midi	115	drummer1/session1/5_jazz-funk_116_beat_4-4_19.midi
48	drummer7/session1/15_jazz_112_beat_4-4_16.midi	116	drummer1/session1/5_jazz-funk_116_beat_4-4_12.midi
49	drummer7/session1/15_jazz_112_beat_4-4_51.midi	117	drummer7/session1/15_jazz_112_beat_4-4_52.midi
50	drummer7/session2/81_country_78_fill_4-4_1.midi	118	drummer7/session3/149_soul_105_fill_4-4_14.midi
51	drummer7/session2/81_country_78_fill_4-4_12.midi	119	drummer10/session1/8_jazz-swing_215_beat_4-4_23.midi
52	drummer3/session1/9_rock_105_beat_4-4_42.midi	120	drummer1/session1/5_jazz-funk_116_beat_4-4_15.midi
53	drummer7/session2/81_country_78_fill_4-4_25.midi	121	drummer7/session3/63_funk_112_fill_4-4_57.midi
54	drummer7/session3/149_soul_105_fill_4-4_28.midi	122	drummer7/session3/25_hiphop_67_fill_4-4_22.midi
55	drummer7/session3/56_funk_112_fill_4-4_19.midi	123	drummer3/session1/9_rock_105_beat_4-4_51.midi
56	drummer10/session1/8_jazz-swing_215_beat_4-4_2.midi	124	drummer7/session2/81_country_78_fill_4-4_56.midi
57	drummer7/session3/149_soul_105_fill_4-4_37.midi	125	drummer1/session1/5_jazz-funk_116_beat_4-4_25.midi
58	drummer7/session3/149_soul_105_fill_4-4_36.midi	126	drummer7/session3/156_soul_98_fill_4-4_41.midi
59	drummer7/session3/156_soul_98_fill_4-4_34.midi	127	drummer7/session3/149_soul_105_fill_4-4_2.midi
60	drummer7/session3/25_hiphop_67_fill_4-4_42.midi	128	drummer7/session1/15_jazz_112_beat_4-4_41.midi
61	drummer3/session2/2_rock_100_beat_4-4_56.midi	129	drummer3/session1/9_rock_105_beat_4-4_5.midi
62	drummer7/session3/63_funk_112_fill_4-4_28.midi	130	drummer3/session1/9_rock_105_beat_4-4_57.midi
63	drummer1/session1/5_jazz-funk_116_beat_4-4_10.midi	131	drummer3/session1/9_rock_105_beat_4-4_18.midi
64	drummer1/session1/5_jazz-funk_116_beat_4-4_26.midi	132	drummer3/session1/9_rock_105_beat_4-4_28.midi
65	drummer7/session3/109_rock_95_beat_4-4_57.midi	133	drummer7/session1/15_jazz_112_beat_4-4_35.midi
66	drummer7/session2/81_country_78_fill_4-4_17.midi	134	drummer7/session3/109_rock_95_beat_4-4_24.midi
67	drummer3/session1/9_rock_105_beat_4-4_8.midi	135	drummer7/session3/63_funk_112_fill_4-4_13.midi
68	drummer7/session1/15_jazz_112_beat_4-4_34.midi	136	drummer7/session2/81_country_78_fill_4-4_5.midi

TABLE A.3: Overview of the files from E-GMD where MIDI events occur after their audio counterparts, Part 2/4.

#	File	#	File
137	drummer7/session1/15_jazz_112_beat_4-4_55.midi	210	drummer7/session3/25_hiphop_67_fill_4-4_33.midi
138	drummer7/session3/149_soul_105_fill_4-4_34.midi	211	drummer7/session2/81_country_78_fill_4-4_32.midi
139	drummer7/session3/156_soul_98_fill_4-4_16.midi	212	drummer7/session3/63_funk_112_fill_4-4_41.midi
140	drummer3/session2/2_rock_100_beat_4-4_15.midi	213	drummer7/session2/81_country_78_fill_4-4_44.midi
141	drummer7/session3/109_rock_95_beat_4-4_8.midi	214	drummer7/session2/81_country_78_fill_4-4_4.midi
142	drummer3/session2/2_rock_100_beat_4-4_25.midi	215	drummer3/session2/2_rock_100_beat_4-4_42.midi
143	drummer7/session3/63_funk_112_fill_4-4_35.midi	216	drummer7/session3/146_soul_105_fill_4-4_37.midi
144	drummer7/session2/81_country_78_fill_4-4_13.midi	217	drummer7/session3/63_funk_112_fill_4-4_15.midi
145	drummer7/session3/146_soul_105_fill_4-4_4.midi	218	drummer7/session3/156_soul_98_fill_4-4_10.midi
146	drummer7/session3/146_soul_105_fill_4-4_44.midi	219	drummer7/session3/63_funk_112_fill_4-4_16.midi
147	drummer7/session3/109_rock_95_beat_4-4_22.midi	220	drummer7/session3/146_soul_105_fill_4-4_32.midi
148	drummer3/session2/2_rock_100_beat_4-4_1.midi	221	drummer7/session1/15_jazz_112_beat_4-4_31.midi
149	drummer1/session1/5_jazz-funk_116_beat_4-4_6.midi	222	drummer3/session1/9_rock_105_beat_4-4_6.midi
150	drummer3/session1/9_rock_105_beat_4-4_56.midi	223	drummer7/session1/15_jazz_112_beat_4-4_1.midi
151	drummer7/session1/15_jazz_112_beat_4-4_18.midi	224	drummer3/session1/9_rock_105_beat_4-4_10.midi
152	drummer3/session2/2_rock_100_beat_4-4_8.midi	225	drummer7/session3/63_funk_112_fill_4-4_18.midi
153	drummer3/session1/9_rock_105_beat_4-4_16.midi	226	drummer7/session3/63_funk_112_fill_4-4_12.midi
154	drummer7/session3/146_soul_105_fill_4-4_54.midi	227	drummer7/session3/63_funk_112_fill_4-4_54.midi
155	drummer7/session3/146_soul_105_fill_4-4_1.midi	228	drummer7/session2/81_country_78_fill_4-4_23.midi
156	drummer1/session1/5_jazz-funk_116_beat_4-4_28.midi	229	drummer7/session3/146_soul_105_fill_4-4_34.midi
157	drummer7/session3/63_funk_112_fill_4-4_11.midi	230	drummer3/session2/2_rock_100_beat_4-4_22.midi
158	drummer7/session3/156_soul_98_fill_4-4_43.midi	231	drummer7/session3/146_soul_105_fill_4-4_31.midi
159	drummer10/session1/8_jazz-swing_215_beat_4-4_13.midi	232	drummer7/session3/25_hiphop_67_fill_4-4_31.midi
160	drummer7/session3/146_soul_105_fill_4-4_13.midi	233	drummer3/session2/2_rock_100_beat_4-4_28.midi
161	drummer7/session3/149_soul_105_fill_4-4_12.midi	234	drummer7/session3/63_funk_112_fill_4-4_53.midi
162	drummer7/session3/109_rock_95_beat_4-4_34.midi	235	drummer7/session3/156_soul_98_fill_4-4_31.midi
163	drummer7/session3/63_funk_112_fill_4-4_14.midi	236	drummer7/session3/146_soul_105_fill_4-4_57.midi
164	drummer7/session3/149_soul_105_fill_4-4_24.midi	237	drummer7/session3/109_rock_95_beat_4-4_11.midi
165	drummer7/session3/25_hiphop_67_fill_4-4_3.midi	238	drummer3/session1/9_rock_105_beat_4-4_3.midi
166	drummer7/session3/146_soul_105_fill_4-4_56.midi	239	drummer3/session1/9_rock_105_beat_4-4_52.midi
167	drummer7/session3/146_soul_105_fill_4-4_8.midi	240	drummer7/session3/156_soul_98_fill_4-4_11.midi
168	drummer10/session1/8_jazz-swing_215_beat_4-4_16.midi	241	drummer7/session3/146_soul_105_fill_4-4_15.midi
169	drummer7/session3/109_rock_95_beat_4-4_56.midi	242	drummer7/session3/146_soul_105_fill_4-4_33.midi
170	drummer7/session2/81_country_78_fill_4-4_21.midi	243	drummer7/session3/109_rock_95_beat_4-4_51.midi
171	drummer7/session1/15_jazz_112_beat_4-4_44.midi	244	drummer3/session2/2_rock_100_beat_4-4_3.midi
172	drummer7/session3/138_soul_105_fill_4-4_33.midi	245	drummer7/session3/63_funk_112_fill_4-4_31.midi
173	drummer1/session1/5_jazz-funk_116_beat_4-4_43.midi	246	drummer7/session1/15_jazz_112_beat_4-4_14.midi
174	drummer7/session3/149_soul_105_fill_4-4_15.midi	247	drummer1/session1/5_jazz-funk_116_beat_4-4_32.midi
175	drummer1/session1/5_jazz-funk_116_beat_4-4_53.midi	248	drummer7/session3/146_soul_105_fill_4-4_6.midi
176	drummer7/session3/109_rock_95_beat_4-4_44.midi	249	drummer7/session3/146_soul_105_fill_4-4_26.midi
177	drummer7/session3/109_rock_95_beat_4-4_5.midi	250	drummer1/session1/5_jazz-funk_116_beat_4-4_4.midi
178	drummer7/session1/15_jazz_112_beat_4-4_13.midi	251	drummer3/session1/9_rock_105_beat_4-4_31.midi
179	drummer7/session1/15_jazz_112_beat_4-4_22.midi	252	drummer7/session1/15_jazz_112_beat_4-4_57.midi
180	drummer7/session3/146_soul_105_fill_4-4_3.midi	253	drummer7/session3/149_soul_105_fill_4-4_3.midi
181	drummer7/session1/15_jazz_112_beat_4-4_5.midi	254	drummer7/session3/149_soul_105_fill_4-4_1.midi
182	drummer7/session3/156_soul_98_fill_4-4_4.midi	255	drummer7/session2/81_country_78_fill_4-4_18.midi
183	drummer7/session3/149_soul_105_fill_4-4_51.midi	256	drummer7/session3/25_hiphop_67_fill_4-4_51.midi
184	drummer7/session1/15_jazz_112_beat_4-4_53.midi	257	drummer7/session2/81_country_78_fill_4-4_37.midi
185	drummer3/session1/9_rock_105_beat_4-4_13.midi	258	drummer7/session3/146_soul_105_fill_4-4_36.midi
186	drummer7/session2/81_country_78_fill_4-4_28.midi	259	drummer7/session3/109_rock_95_beat_4-4_2.midi
187	drummer3/session2/2_rock_100_beat_4-4_11.midi	260	drummer7/session3/138_soul_105_fill_4-4_56.midi
188	drummer7/session2/81_country_78_fill_4-4_14.midi	261	drummer7/session3/109_rock_95_beat_4-4_16.midi
189	drummer7/session3/109_rock_95_beat_4-4_3.midi	262	drummer7/session3/146_soul_105_fill_4-4_53.midi
190	drummer7/session2/81_country_78_fill_4-4_11.midi	263	drummer3/session2/2_rock_100_beat_4-4_13.midi
191	drummer7/session3/156_soul_98_fill_4-4_57.midi	264	drummer7/session2/81_country_78_fill_4-4_19.midi
192	drummer1/session1/5_jazz-funk_116_beat_4-4_5.midi	265	drummer7/session3/156_soul_98_fill_4-4_55.midi
193	drummer7/session1/15_jazz_112_beat_4-4_43.midi	266	drummer7/session3/149_soul_105_fill_4-4_22.midi
194	drummer1/session1/40_latin-samba_116_fill_4-4_2.midi	267	drummer3/session2/2_rock_100_beat_4-4_55.midi
195	drummer7/session3/146_soul_105_fill_4-4_23.midi	268	drummer10/session1/8_jazz-swing_215_beat_4-4_41.midi
196	drummer7/session3/146_soul_105_fill_4-4_24.midi	269	drummer10/session1/8_jazz-swing_215_beat_4-4_33.midi
197	drummer7/session3/109_rock_95_beat_4-4_10.midi	270	drummer7/session3/63_funk_112_fill_4-4_2.midi
198	drummer7/session3/156_soul_98_fill_4-4_19.midi	271	drummer3/session1/9_rock_105_beat_4-4_15.midi
199	drummer7/session3/109_rock_95_beat_4-4_14.midi	272	drummer1/session1/5_jazz-funk_116_beat_4-4_35.midi
200	drummer7/session3/149_soul_105_fill_4-4_53.midi	273	drummer7/session3/25_hiphop_67_fill_4-4_2.midi
201	drummer7/session3/149_soul_105_fill_4-4_25.midi	274	drummer7/session3/146_soul_105_fill_4-4_51.midi
202	drummer3/session1/9_rock_105_beat_4-4_21.midi	275	drummer7/session3/63_funk_112_fill_4-4_55.midi
203	drummer7/session3/109_rock_95_beat_4-4_28.midi	276	drummer10/session1/8_jazz-swing_215_beat_4-4_15.midi
204	drummer1/session1/5_jazz-funk_116_beat_4-4_44.midi	277	drummer10/session1/8_jazz-swing_215_beat_4-4_34.midi
205	drummer7/session3/63_funk_112_fill_4-4_1.midi	278	drummer7/session2/81_country_78_fill_4-4_36.midi
206	drummer7/session3/109_rock_95_beat_4-4_18.midi	279	drummer7/session3/25_hiphop_67_fill_4-4_34.midi
207	drummer7/session3/149_soul_105_fill_4-4_10.midi	280	drummer7/session3/146_soul_105_fill_4-4_25.midi
208	drummer7/session3/63_funk_112_fill_4-4_42.midi	281	drummer3/session2/2_rock_100_beat_4-4_32.midi
209	drummer7/session1/15_jazz_112_beat_4-4_11.midi	282	drummer1/session1/5_jazz-funk_116_beat_4-4_24.midi

TABLE A.4: Overview of the files from E-GMD where MIDI events occur after their audio counterparts, Part 3/4.

#	File	#	File
283	drummer7/session3/156_soul_98_fill_4-4_3.midi	356	drummer3/session2/2_rock_100_beat_4-4_6.midi
284	drummer3/session1/9_rock_105_beat_4-4_41.midi	357	drummer3/session1/9_rock_105_beat_4-4_32.midi
285	drummer7/session1/15_jazz_112_beat_4-4_33.midi	358	drummer7/session2/81_country_78_fill_4-4_42.midi
286	drummer7/session1/15_jazz_112_beat_4-4_15.midi	359	drummer7/session3/146_soul_105_fill_4-4_19.midi
287	drummer7/session3/25_hiphop_67_fill_4-4_13.midi	360	drummer7/session3/156_soul_98_fill_4-4_23.midi
288	drummer7/session1/15_jazz_112_beat_4-4_37.midi	361	drummer7/session3/156_soul_98_fill_4-4_25.midi
289	drummer7/session3/146_soul_105_fill_4-4_22.midi	362	drummer7/session3/149_soul_105_fill_4-4_57.midi
290	drummer7/session3/146_soul_105_fill_4-4_17.midi	363	drummer7/session2/81_country_78_fill_4-4_58.midi
291	drummer7/session3/25_hiphop_67_fill_4-4_35.midi	364	drummer3/session1/9_rock_105_beat_4-4_37.midi
292	drummer7/session3/63_funk_112_fill_4-4_32.midi	365	drummer7/session1/15_jazz_112_beat_4-4_54.midi
293	drummer7/session3/63_funk_112_fill_4-4_56.midi	366	drummer7/session3/156_soul_98_fill_4-4_37.midi
294	drummer10/session1/8_jazz-swing_215_beat_4-4_17.midi	367	drummer7/session3/146_soul_105_fill_4-4_10.midi
295	drummer7/session3/25_hiphop_67_fill_4-4_11.midi	368	drummer3/session2/2_rock_100_beat_4-4_16.midi
296	drummer7/session3/156_soul_98_fill_4-4_28.midi	369	drummer7/session3/25_hiphop_67_fill_4-4_15.midi
297	drummer1/session1/5_jazz-funk_116_beat_4-4_23.midi	370	drummer7/session3/146_soul_105_fill_4-4_18.midi
298	drummer7/session3/63_funk_112_fill_4-4_51.midi	371	drummer3/session2/2_rock_100_beat_4-4_43.midi
299	drummer7/session3/109_rock_95_beat_4-4_42.midi	372	drummer7/session1/15_jazz_112_beat_4-4_2.midi
300	drummer7/session3/156_soul_98_fill_4-4_8.midi	373	drummer1/session1/5_jazz-funk_116_beat_4-4_3.midi
301	drummer7/session3/149_soul_105_fill_4-4_8.midi	374	drummer7/session1/15_jazz_112_beat_4-4_23.midi
302	drummer3/session1/9_rock_105_beat_4-4_25.midi	375	drummer3/session1/9_rock_105_beat_4-4_26.midi
303	drummer7/session3/156_soul_98_fill_4-4_51.midi	376	drummer7/session2/81_country_78_fill_4-4_57.midi
304	drummer10/session1/8_jazz-swing_215_beat_4-4_31.midi	377	drummer7/session3/25_hiphop_67_fill_4-4_54.midi
305	drummer7/session3/146_soul_105_fill_4-4_2.midi	378	drummer7/session3/63_funk_112_fill_4-4_43.midi
306	drummer7/session1/15_jazz_112_beat_4-4_56.midi	379	drummer7/session3/109_rock_95_beat_4-4_26.midi
307	drummer7/session3/109_rock_95_beat_4-4_19.midi	380	drummer7/session2/81_country_78_fill_4-4_55.midi
308	drummer7/session3/109_rock_95_beat_4-4_12.midi	381	drummer7/session3/109_rock_95_beat_4-4_17.midi
309	drummer7/session1/15_jazz_112_beat_4-4_10.midi	382	drummer1/session1/5_jazz-funk_116_beat_4-4_37.midi
310	drummer7/session3/25_hiphop_67_fill_4-4_14.midi	383	drummer3/session2/2_rock_100_beat_4-4_37.midi
311	drummer7/session3/25_hiphop_67_fill_4-4_53.midi	384	drummer7/session3/25_hiphop_67_fill_4-4_58.midi
312	drummer7/session3/109_rock_95_beat_4-4_4.midi	385	drummer3/session2/2_rock_100_beat_4-4_23.midi
313	drummer7/session3/156_soul_98_fill_4-4_54.midi	386	drummer7/session3/25_hiphop_67_fill_4-4_41.midi
314	drummer7/session3/146_soul_105_fill_4-4_41.midi	387	drummer7/session2/81_country_78_fill_4-4_24.midi
315	drummer7/session3/146_soul_105_fill_4-4_42.midi	388	drummer7/session3/149_soul_105_fill_4-4_55.midi
316	drummer3/session2/2_rock_100_beat_4-4_36.midi	389	drummer7/session3/63_funk_112_fill_4-4_44.midi
317	drummer7/session3/149_soul_105_fill_4-4_43.midi	390	drummer3/session2/2_rock_100_beat_4-4_57.midi
318	drummer7/session3/156_soul_98_fill_4-4_36.midi	391	drummer7/session1/15_jazz_112_beat_4-4_4.midi
319	drummer7/session3/149_soul_105_fill_4-4_32.midi	392	drummer7/session3/109_rock_95_beat_4-4_35.midi
320	drummer1/session1/5_jazz-funk_116_beat_4-4_41.midi	393	drummer3/session1/9_rock_105_beat_4-4_1.midi
321	drummer7/session3/156_soul_98_fill_4-4_56.midi	394	drummer7/session1/15_jazz_112_beat_4-4_36.midi
322	drummer1/session1/5_jazz-funk_116_beat_4-4_57.midi	395	drummer7/session3/63_funk_112_fill_4-4_26.midi
323	drummer3/session2/2_rock_100_beat_4-4_26.midi	396	drummer7/session3/149_soul_105_fill_4-4_41.midi
324	drummer7/session3/146_soul_105_fill_4-4_14.midi	397	drummer7/session3/63_funk_112_fill_4-4_52.midi
325	drummer3/session1/9_rock_105_beat_4-4_4.midi	398	drummer7/session3/63_funk_112_fill_4-4_6.midi
326	drummer7/session3/149_soul_105_fill_4-4_6.midi	399	drummer3/session2/2_rock_100_beat_4-4_58.midi
327	drummer7/session1/15_jazz_112_beat_4-4_17.midi	400	drummer1/session1/5_jazz-funk_116_beat_4-4_31.midi
328	drummer1/session1/5_jazz-funk_116_beat_4-4_36.midi	401	drummer7/session3/63_funk_112_fill_4-4_23.midi
329	drummer7/session3/25_hiphop_67_fill_4-4_1.midi	402	drummer3/session2/2_rock_100_beat_4-4_4.midi
330	drummer7/session3/25_hiphop_67_fill_4-4_24.midi	403	drummer7/session3/156_soul_98_fill_4-4_17.midi
331	drummer7/session1/15_jazz_112_beat_4-4_8.midi	404	drummer7/session3/63_funk_112_fill_4-4_3.midi
332	drummer7/session3/109_rock_95_beat_4-4_52.midi	405	drummer7/session3/25_hiphop_67_fill_4-4_52.midi
333	drummer1/session1/5_jazz-funk_116_beat_4-4_11.midi	406	drummer7/session3/146_soul_105_fill_4-4_21.midi
334	drummer3/session1/9_rock_105_beat_4-4_11.midi	407	drummer1/session1/5_jazz-funk_116_beat_4-4_22.midi
335	drummer3/session1/9_rock_105_beat_4-4_23.midi	408	drummer7/session2/81_country_78_fill_4-4_52.midi
336	drummer3/session2/2_rock_100_beat_4-4_18.midi	409	drummer7/session3/149_soul_105_fill_4-4_58.midi
337	drummer7/session3/25_hiphop_67_fill_4-4_37.midi	410	drummer7/session3/63_funk_112_fill_4-4_36.midi
338	drummer1/session1/5_jazz-funk_116_beat_4-4_33.midi	411	drummer1/session1/5_jazz-funk_116_beat_4-4_2.midi
339	drummer7/session3/156_soul_98_fill_4-4_18.midi	412	drummer3/session2/2_rock_100_beat_4-4_44.midi
340	drummer1/session1/5_jazz-funk_116_beat_4-4_21.midi	413	drummer7/session3/146_soul_105_fill_4-4_5.midi
341	drummer7/session3/109_rock_95_beat_4-4_23.midi	414	drummer1/session1/5_jazz-funk_116_beat_4-4_14.midi
342	drummer7/session3/149_soul_105_fill_4-4_19.midi	415	drummer7/session3/109_rock_95_beat_4-4_41.midi
343	drummer7/session3/109_rock_95_beat_4-4_54.midi	416	drummer7/session1/15_jazz_112_beat_4-4_21.midi
344	drummer7/session3/156_soul_98_fill_4-4_12.midi	417	drummer7/session3/63_funk_112_fill_4-4_34.midi
345	drummer7/session3/138_soul_105_fill_4-4_12.midi	418	drummer7/session3/63_funk_112_fill_4-4_4.midi
346	drummer7/session3/15_pop-soft_83_fill_4-4_57.midi	419	drummer1/session1/5_jazz-funk_116_beat_4-4_54.midi
347	drummer7/session3/109_rock_95_beat_4-4_36.midi	420	drummer7/session3/63_funk_112_fill_4-4_5.midi
348	drummer7/session3/149_soul_105_fill_4-4_56.midi	421	drummer7/session3/146_soul_105_fill_4-4_28.midi
349	drummer7/session2/81_country_78_fill_4-4_3.midi	422	drummer7/session3/25_hiphop_67_fill_4-4_6.midi
350	drummer7/session3/149_soul_105_fill_4-4_44.midi	423	drummer1/session1/5_jazz-funk_116_beat_4-4_58.midi
351	drummer7/session3/156_soul_98_fill_4-4_21.midi	424	drummer1/session1/5_jazz-funk_116_beat_4-4_17.midi
352	drummer7/session3/25_hiphop_67_fill_4-4_12.midi	425	drummer3/session2/2_rock_100_beat_4-4_24.midi
353	drummer7/session3/109_rock_95_beat_4-4_37.midi	426	drummer7/session3/156_soul_98_fill_4-4_53.midi
354	drummer7/session3/146_soul_105_fill_4-4_43.midi	427	drummer10/session1/8_jazz-swing_215_beat_4-4_19.midi
355	drummer1/session1/5_jazz-funk_116_beat_4-4_16.midi	428	drummer7/session3/156_soul_98_fill_4-4_52.midi

TABLE A.5: Overview of the files from E-GMD where MIDI events occur after their audio counterparts, Part 4/4.

#	File	#	File
429	drummer7/session3/25_hiphop_67_fill_4-4_57.midi	463	drummer7/session2/81_country_78_fill_4-4_22.midi
430	drummer7/session3/109_rock_95_beat_4-4_13.midi	464	drummer1/session1/5_jazz-funk_116_beat_4-4_42.midi
431	drummer7/session1/15_jazz_112_beat_4-4_32.midi	465	drummer7/session3/25_hiphop_67_fill_4-4_10.midi
432	drummer7/session3/63_funk_112_fill_4-4_25.midi	466	drummer3/session2/2_rock_100_beat_4-4_54.midi
433	drummer7/session3/146_soul_105_fill_4-4_35.midi	467	drummer3/session1/9_rock_105_beat_4-4_17.midi
434	drummer3/session2/2_rock_100_beat_4-4_53.midi	468	drummer7/session2/81_country_78_fill_4-4_35.midi
435	drummer7/session3/156_soul_98_fill_4-4_26.midi	469	drummer7/session3/25_hiphop_67_fill_4-4_18.midi
436	drummer3/session1/9_rock_105_beat_4-4_19.midi	470	drummer7/session3/156_soul_98_fill_4-4_13.midi
437	drummer7/session3/109_rock_95_beat_4-4_25.midi	471	drummer3/session2/2_rock_100_beat_4-4_17.midi
438	drummer3/session1/9_rock_105_beat_4-4_36.midi	472	drummer7/session1/15_jazz_112_beat_4-4_12.midi
439	drummer7/session3/149_soul_105_fill_4-4_26.midi	473	drummer7/session3/63_funk_112_fill_4-4_8.midi
440	drummer7/session3/63_funk_112_fill_4-4_22.midi	474	drummer7/session3/156_soul_98_fill_4-4_32.midi
441	drummer3/session2/2_rock_100_beat_4-4_14.midi	475	drummer3/session2/2_rock_100_beat_4-4_35.midi
442	drummer7/session2/81_country_78_fill_4-4_34.midi	476	drummer7/session2/81_country_78_fill_4-4_6.midi
443	drummer7/session3/156_soul_98_fill_4-4_14.midi	477	drummer7/session1/15_jazz_112_beat_4-4_58.midi
444	drummer7/session3/63_funk_112_fill_4-4_58.midi	478	drummer7/session3/156_soul_98_fill_4-4_42.midi
445	drummer7/session3/146_soul_105_fill_4-4_12.midi	479	drummer10/session1/8_jazz-swing_215_beat_4-4_26.midi
446	drummer7/session3/149_soul_105_fill_4-4_17.midi	480	drummer7/session1/15_jazz_112_beat_4-4_28.midi
447	drummer7/session3/109_rock_95_beat_4-4_53.midi	481	drummer7/session1/15_jazz_112_beat_4-4_6.midi
448	drummer7/session3/149_soul_105_fill_4-4_54.midi	482	drummer7/session2/81_country_78_fill_4-4_16.midi
449	drummer7/session3/25_hiphop_67_fill_4-4_17.midi	483	drummer7/session3/25_hiphop_67_fill_4-4_55.midi
450	drummer7/session3/156_soul_98_fill_4-4_22.midi	484	drummer3/session2/2_rock_100_beat_4-4_21.midi
451	drummer3/session1/9_rock_105_beat_4-4_12.midi	485	drummer7/session3/156_soul_98_fill_4-4_1.midi
452	drummer7/session3/25_hiphop_67_fill_4-4_4.midi	486	drummer7/session3/156_soul_98_fill_4-4_24.midi
453	drummer3/session2/2_rock_100_beat_4-4_51.midi	487	drummer7/session3/63_funk_112_fill_4-4_19.midi
454	drummer7/session3/149_soul_105_fill_4-4_31.midi	488	drummer3/session2/2_rock_100_beat_4-4_5.midi
455	drummer7/session3/25_hiphop_67_fill_4-4_32.midi	489	drummer3/session2/2_rock_100_beat_4-4_52.midi
456	drummer3/session2/2_rock_100_beat_4-4_34.midi	490	drummer3/session2/2_rock_100_beat_4-4_31.midi
457	drummer7/session2/81_country_78_fill_4-4_41.midi	491	drummer7/session3/156_soul_98_fill_4-4_6.midi
458	drummer7/session3/149_soul_105_fill_4-4_4.midi	492	drummer7/session3/149_soul_105_fill_4-4_13.midi
459	drummer7/session2/81_country_78_fill_4-4_2.midi	493	drummer7/session3/25_hiphop_67_fill_4-4_44.midi
460	drummer10/session1/8_jazz-swing_215_beat_4-4_5.midi	494	drummer7/session2/81_country_78_fill_4-4_54.midi
461	drummer1/session1/5_jazz-funk_116_beat_4-4_55.midi	495	drummer1/session1/5_jazz-funk_116_beat_4-4_1.midi
462	drummer7/session3/109_rock_95_beat_4-4_55.midi	496	drummer7/session1/15_jazz_112_beat_4-4_26.midi

Appendix B

RGDD Creation

B.1 RGGD Drum Set Selection

TABLE B.1: Overview of the Logic Pro drum sets used for the creation of RGDD(-Stems). Drum sets 1-20 are acoustic drum set whereas the rest are electronic drum kits. Delay or reverb buses were turned off where necessary to ensure that the synthesized audio, and therefore the corresponding spectogram too, only contains intentional drum hits and that are also present in the ground truth. For the creation of RGDD-Stems, drum sets 1-17 and 21-52 are used. The remainder of RGDD comprises drum sets 1-20.

#	**Drum set**	#	**Drum set**	#	**Drum set**	#	**Drum set**
1	Blue Ridge	2	Bluebird	3	Brooklyn	4	Detroit Garage
5	East Bay	6	Four on the Floor	7	Heavy	8	Liverpool
9	Manchester	10	Motown Revisited	11	Neo Soul	12	Portland
13	Retro Rock	14	Roots	15	Scientific Method	16	Slow Jam
17	Smash	18	SoCal	19	Speakeasy	20	Sunset
21	808 Flex	22	Advanced Machines	23	After Party	24	Agogo Funk
25	Analog Circuits	26	Arcade Frenzy	27	Atlanta	28	Beat Machine
29	Control Voltage	30	Cream Soda	31	CR-78 Made Iconic	32	Crate Digger
33	Deep Tech	34	Depth Charge	35	Hybrid Knock	36	Ibiza
37	Infinity	38	Modern CR-8000	39	Modern TR-707	40	No Man's Joint
41	Pawn Shop 808	42	Roland CR-78	43	Roland TR-808	44	Roland TR-909
45	Silverlake	46	Snapback	47	Synthia	48	Tension
49	Titans of Bass	50	Trap Door	51	Uptown Flow	52	Video Star

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2015). *Neural Machine Translation by Jointly Learning to Align and Translate*. arXiv: 1409.0473 [cs.CL].

Benetos, Emmanouil et al. (2019). “Automatic Music Transcription: An Overview”. In: *IEEE Signal Processing Magazine* 36.1, pp. 20–30.

Benning, Martin and Martin Burger (2018). “Modern regularization methods for inverse problems”. In: *Acta Numerica* 27, 1–111.

Bittner, Rachel et al. (2014). “MedleyDB: A multitrack dataset for annotation-intensive MIR research”. In: *Proceedings of the 15th International Society for Music Information Retrieval Conference*. ISMIR, pp. 155–160.

Bryson, Arthur Earl (1961). “A gradient method for optimizing multi-stage allocation processes”. In: *Proc. Harvard Univ. Symposium on digital computers and their applications*. Vol. 72, p. 22.

Burred, Juan José and Alexander Lerch (2004). “Hierarchical Automatic Audio Signal Classification”. In: *Journal of the Audio Engineering Society (JAES)* 52, pp. 724–739.

Böck, Sebastian, Florian Krebs, and Gerhard Widmer (2016). “Joint Beat and Downbeat Tracking with Recurrent Neural Networks”. In: *Proceedings of the 17th International Society for Music Information Retrieval Conference*. ISMIR, pp. 255–261.

Callender, Lee, Curtis Hawthorne, and Jesse Engel (2020). *Improving Perceptual Quality of Drum Transcription with the Expanded Groove MIDI Dataset*. arXiv: 2004.00188 [cs.SD].

Cannam, Chris, Christian Landone, and Mark B. Sandler (2010). “Sonic visualiser: an open source application for viewing, analysing, and annotating music audio files”. In: *Proceedings of the 18th ACM international conference on Multimedia*.

Cartwright, Mark and Juan Pablo Bello (2018). “Increasing drum transcription vocabulary using data synthesis”. In: *Proceedings of the 21st International Conference on Digital Audio Effects (DAFx18)*, pp. 72–79.

Choi, Keunwoo and Kyunghyun Cho (2019). “Deep Unsupervised Drum Transcription”. In: *Proceedings of the 20th International Society for Music Information Retrieval Conference*. ISMIR, pp. 183–191.

Défossez, Alexandre (2021). “Hybrid Spectrogram and Waveform Source Separation”. In: *Proceedings of the ISMIR 2021 Workshop on Music Source Separation*.

Deisenroth, Marc Peter, A. Aldo Faisal, and Cheng Soon Ong (2020). *Mathematics for Machine Learning*. Cambridge University Press.

Dittmar, Christian and Daniel Gärtner (2014). “Real-Time Transcription and Separation of Drum Recordings Based on NMF Decomposition”. In: *Proceedings of the 17th International Conference on Digital Audio Effects (DAFx14)*.

Dittmar, Christian and Meinard Müller (2016). “Reverse Engineering the Amen Break — Score-Informed Separation and Restoration Applied to Drum Recordings”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24.9, pp. 1535–1547.

Duan, Zhiyao and Bryan Pardo (2011). “Soundprism: An Online System for Score-Informed Source Separation of Music Audio”. In: *IEEE Journal of Selected Topics in Signal Processing* 5.6, pp. 1205–1215.

Dumoulin, Vincent and Francesco Visin (2018). *A guide to convolution arithmetic for deep learning*. arXiv: 1603.07285 [stat.ML].

Esteva, Andre et al. (2021). “Deep learning-enabled medical computer vision”. In: *NPJ Digital Medicine* 4.

Fechner, Gustav Theodor (1860). *Elemente der Psychophysik*. Vol. 2. Leipzig: Breitkopf und Haertel.

Fernando, K. Ruwani M. and Chris P. Tsokos (2021). “Dynamically Weighted Balanced Loss: Class Imbalanced Learning and Confidence Calibration of Deep Neural Networks”. In: *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–12.

Ferri, Cèsar, Jose Hernández-Orallo, and R. Modroiu (2009). “An experimental comparison of performance measures for classification”. In: *Pattern Recognition Letters* 30.1, pp. 27–38. ISSN: 0167-8655.

Fitzgerald, Derry, Eugene Coyle, and Bob Lawlor (2002). “Sub-band independent subspace analysis for drum transcription”. In: *Proceedings of the 5th International Conference on Digital Audio Effects (DAFx02)*, pp. 65–69.

Fitzgerald, Derry, Robert Lawlor, and Eugene Coyle (2003). “Prior Subspace Analysis for Drum Transcription”. In: *Journal of The Audio Engineering Society*.

Gajhede, Nicolai, Oliver Beck, and Hendrik Purwins (2016). “Convolutional Neural Networks with Batch Normalization for Classifying Hi-Hat, Snare, and Bass Percussion Sound Samples”. In: *Proceedings of the Audio Mostly 2016*. New York, NY, USA: Association for Computing Machinery, 111–115.

Gholamalinezhad, Hossein and Hossein Khosravi (2020). *Pooling Methods in Deep Neural Networks, a Review*. arXiv: 2009.07485 [cs.CV].

Gillet, Olivier and Gaël Richard (2006). “ENST-Drums: an extensive audio-visual database for drum signals processing”. In: *Proceedings of the 7th International Conference on Music Information Retrieval*. ISMIR, pp. 156–159.

Gillick, Jon et al. (2019). “Learning to Groove with Inverse Sequence Transformations”. In: *International Conference on Machine Learning (ICML)*. arXiv: 1905.06118 [cs.SD].

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press.

Gouyon, Fabien, Francois Pachet, and Olivier Delerue (2000). “On the Use of Zero-Crossing Rate for an Application of Classification of Percussive Sounds”. In: *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx00)*.

Hawthorne, Curtis et al. (2018). *Onsets and Frames: Dual-Objective Piano Transcription*. arXiv: 1710.11153 [cs.SD].

Hebb, Donald O. (1949). *The organization of behavior: A neuropsychological theory*. New York: Wiley. ISBN: 0-8058-4300-0.

Hochreiter, Sepp (1998). “The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions”. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6, pp. 107–116.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). “Long Short-term Memory”. In: *Neural computation* 9, pp. 1735–80.

Hubel, David H. and Torsten N. Wiesel (1959). “Receptive fields of single neurones in the cat’s striate cortex”. In: *The Journal of physiology* 148.3, pp. 574–591.

— (1962). “Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex”. In: *The Journal of physiology* 160.1, pp. 106–154.

Ioffe, Sergey and Christian Szegedy (2015). *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. arXiv: `1502.03167 [cs.LG]`.

Ishizuka, Ryoto, Ryo Nishikimi, and Kazuyoshi Yoshii (2021). *Global Structure-Aware Drum Transcription Based on Self-Attention Mechanisms*. arXiv: `2105.05791 [cs.SD]`.

Jacques, Céline and Axel Roebel (2019). "Data Augmentation for Transcription with Convolutional Neural Networks". In: *2019 27th European Signal Processing Conference (EUSIPCO)*, pp. 1–5.

Jain, Arjit et al. (2021). *SpliceOut: A Simple and Efficient Audio Augmentation Method*. arXiv: `2110.00046 [cs.SD]`.

Jarrett, Kevin et al. (2009). "What is the best multi-stage architecture for object recognition?" In: *2009 IEEE 12th International Conference on Computer Vision*, pp. 2146–2153.

Johnson, Justin M. and Taghi M. Khoshgoftaar (2019). "Survey on deep learning with class imbalance". In: *Journal of Big Data* 6.1.

Kaliakatsos-Papakostas, Maximos et al. (2012). "Real-Time Drums Transcription with Characteristic Bandpass Filtering". In: *Proceedings of the 7th Audio Mostly Conference: A Conference on Interaction with Sound*. AM '12. Corfu, Greece: Association for Computing Machinery, 152–159.

Kelley, Henry J. (1960). "Gradient theory of optimal flight paths". In: *Ars Journal* 30.10, pp. 947–954.

Kingma, Diederik P. and Jimmy Ba (2015). "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015*.

Klambauer, Günter et al. (2017). *Self-Normalizing Neural Networks*. arXiv: `1706.02515 [cs.LG]`.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton (2012). "ImageNet classification with deep convolutional neural networks". In: *Communications of the ACM* 60, pp. 84 –90.

Kumar, Pradeep et al. (2021). "Classification of Imbalanced Data:Review of Methods and Applications". In: *IOP Conference Series: Materials Science and Engineering* 1099.1, p. 012077.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). "Deep Learning". In: *Nature* 521, pp. 436–44.

LeCun, Yann et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.

Li, Bochen and Zhiyao Duan (2016). "An Approach to Score Following for Piano Performances With the Sustained Effect". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24.12, pp. 2425–2438.

Litovsky, Ruth et al. (1999). "The precedence effect". In: *The Journal of the Acoustical Society of America* 106, pp. 1633–54.

McCulloch, Warren S. and Walter Pitts (1943). "A logical calculus of the ideas immanent in nervous activity". In: *The bulletin of mathematical biophysics* 5.4, pp. 115–133.

McFee, Brian et al. (2015). "librosa: Audio and Music Signal Analysis in Python". In: *Proceedings of the 14th python in science conference*, pp. 18–24.

Mohammed, Roweida, Jumanah Rawashdeh, and Malak Abdullah (2020). "Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results". In: *2020 11th International Conference on Information and Communication Systems (ICICS)*, pp. 243–248.

Niedermayer, Bernhard and Gerhard Widmer (2010). “A Multi-Pass Algorithm for Accurate Audio-to-Score Alignment”. In: *Proceedings of the 11th International Society for Music Information Retrieval Conference*. ISMIR, pp. 417–422.

O’Shaughnessy, Douglas (1987). *Speech communication: human and machine*. Addison-Wesley Publishing Company.

Paulus, Jouni and Anssi Klapuri (2009). “Drum Sound Detection in Polyphonic Music with Hidden Markov Models”. In: *EURASIP Journal on Audio, Speech, and Music Processing* 2009, pp. 1–9.

Paulus, Jouni and Tuomas Virtanen (2005). “Drum transcription with non-negative spectrogram factorisation”. In: *13th European Signal Processing Conference*, pp. 1–4.

Roelofs, Rebecca et al. (2019). “A Meta-Analysis of Overfitting in Machine Learning”. In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc.

Ruder, Sebastian (2016). *An overview of gradient descent optimization algorithms*. arXiv: `1609.04747 [cs.LG]`.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams (1986). “Learning representations by back-propagating errors”. In: *Nature* 323, pp. 533–536.

Salman, Shaeke and Xiuwen Liu (2019). *Overfitting Mechanism and Avoidance in Deep Neural Networks*. arXiv: `1901.06566 [cs.LG]`.

Schloss, W. Andrew (1985). “On the Automatic Transcription of Percussive Music - From Acoustic Signal to High-Level Analysis”. PhD thesis. Stanford, CA: Stanford University.

Schuster, Mike and Kuldip K. Paliwal (1997). “Bidirectional recurrent neural networks”. In: *IEEE Transactions on Signal Processing* 45.11, pp. 2673–2681.

Shaikh, Salahuddin, Liu Changan, and Maaz Rasheed Malik (2021). “An Empirical And Comparatively Research On Under-Sampling and Over- Sampling Defect-Prone Data-Sets Model In Light Of Machine Learning”. In: *International Journal of Advanced Networking and Applications* 12, pp. 4719–4724.

Smith, Leslie N. (2017). “Cyclical Learning Rates for Training Neural Networks”. In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 464–472.

Southall, Carl, Ryan Stables, and Jason Hockman (2016). “Automatic Drum Transcription Using Bi-Directional Recurrent Neural Networks”. In: *Proceedings of the 17th International Society for Music Information Retrieval Conference*. ISMIR, pp. 591–597.

Southall, Carl et al. (2017). “MDB Drums: An annotated subset of MedleyDB for automatic drum transcription”. In: *Proceedings of the 18th International Society for Music Information Retrieval Conference*. ISMIR.

Srivastava, Nitish et al. (2014). “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15.56, pp. 1929–1958.

Thakkar, Ankit and Ritika Lohiya (2021). “Analyzing fusion of regularization techniques in the deep learning-based intrusion detection system”. In: *International Journal of Intelligent Systems* 36.

Tompson, Jonathan et al. (2015). *Efficient Object Localization Using Convolutional Networks*. arXiv: `1411.4280 [cs.CV]`.

Tzanetakis, George, Ajay Kapur, and Richard I. McWalter (2005). “Subband-based Drum Transcription for Audio Signals”. In: *2005 IEEE 7th Workshop on Multimedia Signal Processing*, pp. 1–4.

Van Steelant, Dirk et al. (2004). “Classification of Percussive Sounds using Support Vector Machines”. In: *Proceedings of Annual Machine Learning Conference of Belgium and The Netherlands, BENELEARN 2004*, pp. 146–153.

Vogl, Richard (2018). "Deep Learning Methods for Drum Transcription and Drum Pattern Generation". PhD thesis.

Vogl, Richard, Matthias Dorfer, and Peter Knees (2016). "Recurrent Neural Networks for Drum Transcription". In: *Proceedings of the 17th International Society for Music Information Retrieval Conference*. ISMIR, pp. 730–736.

— (2017). "Drum transcription from polyphonic music with recurrent neural networks". In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 201–205.

Vogl, Richard, Gerhard Widmer, and Peter Knees (2018). "Towards multi-instrument drum transcription". In: *Proceedings of the 21st International Conference on Digital Audio Effects (DAFx18)*.

Vogl, Richard et al. (2017). "Drum Transcription via Joint Beat and Drum Modeling Using Convolutional Recurrent Neural Networks". In: *Proceedings of the 18th International Society for Music Information Retrieval Conference*. ISMIR, pp. 150–157.

Wang, Yu et al. (2020). "Few-Shot Drum Transcription in Polyphonic Music". In: *Proceedings of the 21st International Society for Music Information Retrieval Conference*. ISMIR, pp. 117–124.

Wei, I.-Chieh., Chih-Wei Wu, and Li Su (2021). "Improving automatic drum transcription using large-scale audio-to-midi aligned data". In: *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vol. 2021-June, pp. 246–250.

Wu, Chih-Wei and Alexander Lerch (2015). "Drum Transcription using Partially Fixed Non-Negative Matrix Factorization with Template Adaptation". In: *Proceedings of the 16th International Society for Music Information Retrieval Conference*. ISMIR, pp. 257–263.

— (2017). "Automatic Drum Transcription Using the Student- Teacher Learning Paradigm with Unlabeled Music Data." In: *Proceedings of the 18th International Society for Music Information Retrieval Conference*. ISMIR, pp. 613–620.

Wu, Chih-Wei et al. (2018). "A Review of Automatic Drum Transcription". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26.9, pp. 1457–1483.

Zehren, Mickaël, Marco Alunno, and Paolo Bientinesi (2021). "ADTOF: A large dataset of non-synthetic music for automatic drum transcription". In: *Proceedings of the 7th International Conference on Music Information Retrieval*. ISMIR, pp. 818–824.