

# **Activity Detection for Sound Events in Orchestral Music Recordings**

## **Aktivitätserkennung von Klangereignissen in Orchestermusikaufnahmen**

**Dissertation**

Der Technischen Fakultät  
der Friedrich-Alexander-Universität Erlangen-Nürnberg

zur

Erlangung des Doktorgrades  
Doktor der Ingenieurwissenschaften (Dr.-Ing.)

vorgelegt von

Michael Krause

Als Dissertation genehmigt  
von der Technischen Fakultät  
der Friedrich-Alexander-Universität Erlangen-Nürnberg

Tag der mündlichen Prüfung: 11.07.2023  
Gutachter: Prof. Dr. rer. nat. Meinard Müller  
Prof. Dasaem Jeong

# Abstract

Composers of music can express emotions and communicate with their audience in a multitude of ways. They decide on which voices or instruments to use, arrange notes into melodies, and develop recurring musical patterns. When a composition is performed and turned into sound, their decisions are realized acoustically as sound events. Despite being easily understood by human listeners, teaching a machine to perceive and process such musical sound events can be a challenging task. This thesis studies computational techniques for detecting the activity of sound events in a music recording, i. e., identifying the exact moments in time when a certain event occurs. We focus on orchestral and opera music, which are rarely considered in music processing research and particularly complex due to their high degree of polyphony. In this context, we cover four different types of musical sound events, namely singing, instrumental sounds, different pitches, and leitmotifs (special kinds of musical patterns used for storytelling in opera). To detect the activity of these events within a recording, we design, implement, and evaluate deep learning systems. In addition, we explore a range of techniques including hierarchical classification, differentiable sequence alignments, and representation learning. Beyond evaluating the accuracy of our detection systems, we aim at a deeper understanding of our models with regard to their robustness and sensitivity to confounding effects.

The main contributions of this thesis can be summarized as follows: First, we investigate signal processing and deep learning methods for detecting singing activity in opera recordings. Second, we extend this scenario towards simultaneously detecting singer gender and voice type. We compare several techniques for utilizing the hierarchical relationships between these classes and propose a novel loss formulation for ensuring consistency of detection results across different hierarchy levels. Third, we apply such a hierarchical technique to instrument activity detection. For this task, research progress is often limited by the cost of obtaining manually annotated audio examples for training. To address this issue, we demonstrate that hierarchical information reduces the need for fine-grained instrument annotations during training of our detection models. Fourth, we show how the structure of certain orchestral music datasets can be exploited to learn representations related to instrumentation, without requiring any instrument annotations at all. Fifth, we consider the problem of detecting pitch activity and show how differentiable sequence alignments can be used for learning from weak annotations. Finally, we perform classification and detection of leitmotifs. We present deep learning systems that successfully detect leitmotif activity and provide a detailed analysis of their generalization ability.





# Zusammenfassung

Musikkomponisten nutzen verschiedene Wege, um Emotionen auszudrücken und mit ihrem Publikum zu kommunizieren. Sie entscheiden, welche Stimmen oder Instrumente zu verwenden sind, arrangieren Noten zu Melodien, und entwickeln sich wiederholende musikalische Muster. Wenn eine Komposition aufgeführt und klanglich umgesetzt wird, spiegeln sich diese Entscheidungen akustisch als Klangereignisse wieder. Menschliche Zuhörer verstehen diese musikalischen Klangereignisse mühelos, für Maschinen ist das hingegen schwer. Diese Dissertation behandelt rechnergestützte Verfahren zur Aktivitätserkennung von Klangereignissen in Musikaufnahmen, d.h., zur Bestimmung der genauen Zeitpunkte, zu denen ein bestimmtes Ereignis vorkommt. Wir konzentrieren uns dabei auf Orchester- und Opernmusik, die in der Forschung zur Musikverarbeitung selten betrachtet werden und auf Grund ihres hohen Polyphoniegrades eine besondere Herausforderung darstellen. In diesem Zusammenhang behandeln wir vier verschiedene Typen von musikalischen Klangereignissen: Gesang, Instrumentenklänge, verschiedene Tonhöhen und Leitmotive (bestimmte musikalische Muster, die in Opern die Handlung untermalen). Wir entwerfen, implementieren und evaluieren Deep Learning Systeme zur Aktivitätserkennung. Darüber hinaus untersuchen wir weitere Techniken wie hierarchische Klassifikation, differenzierbare Sequenzalignierung und Repräsentationslernen. Wir evaluieren einerseits die Genauigkeit unserer Detektionssysteme und streben andererseits ein tieferes Verständnis ihrer Robustheit und Anfälligkeit für Störfaktoren an.

Die Hauptbeiträge dieser Dissertation sind wie folgt: Erstens untersuchen wir Signalverarbeitungs- und Deep Learning Verfahren zur Erkennung von Gesangsaktivität in Operaufnahmen. Zweitens erweitern wir dieses Szenario zur gleichzeitigen Erkennung von Geschlecht und Stimmlage der Sänger. Wir vergleichen mehrere Techniken, welche die hierarchischen Beziehungen zwischen diesen Klassen ausnutzen, und führen eine neuartige Kostenfunktion ein, welche die hierarchische Konsistenz der Detektionsergebnisse verbessert. Drittens wenden wir derartige hierarchische Techniken auf die Erkennung von Instrumentenaktivität an. Für diese Aufgabenstellung ist es besonders aufwändig, händisch annotierte Audiobeispiele zu sammeln. Wir zeigen daher auf, wie hierarchische Lernverfahren den Bedarf an Instrumentenannotationen verringern können. Viertens beschreiben wir, wie die Struktur bestimmter Orchesterdaten ausgenutzt werden kann, um Instrumentierung ohne jegliche händische Annotationen zu repräsentieren. Fünftens behandeln wir die Erkennung unterschiedlicher Tonhöhen und zeigen wiederum, wie differenzierbare Verfahren zur Sequenzalignierung den Annotationsaufwand reduzieren. Abschließend betrachten wir die Klassifikation und Detektion von Leitmotiven. Wir konstruieren Deep Learning Systeme, die Motifaktivität erfolgreich erkennen und führen eine detaillierte Analyse ihrer Generalisierungseigenschaften durch.



# Contents

<b>Abstract</b>	<b>iii</b>
<b>Zusammenfassung</b>	<b>v</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Structure and Main Contributions of this Thesis . . . . .	3
1.2 Publications Related to Ph.D. Thesis . . . . .	5
1.3 Additional Publications . . . . .	5
1.4 Acknowledgments . . . . .	6
<b>2 Fundamentals</b>	<b>9</b>
2.1 Audio Representations . . . . .	9
2.1.1 Waveforms . . . . .	9
2.1.2 Short-Time Fourier Transform . . . . .	10
2.1.3 Log-Frequency Representations . . . . .	12
2.2 Activity Detection . . . . .	14
2.3 Case Study: Detecting Singing Activity . . . . .	16
2.3.1 Feature-Engineering Approach . . . . .	16
2.3.2 Deep Learning Approach . . . . .	19
<b>3 Singing Activity Detection in Opera Recordings</b>	<b>23</b>
3.1 Introduction . . . . .	23
3.2 Related Work . . . . .	25
3.3 Singing Voice Detection Methods . . . . .	25
3.4 Dataset and Training Scenarios . . . . .	27
3.5 Experiments . . . . .	29
3.5.1 Training on Different Versions . . . . .	29
3.5.2 Training on Different Musical Material . . . . .	31
3.5.3 Training on Full Splits . . . . .	33
3.5.4 Impact of Dataset Size . . . . .	34

3.5.5	Transfer between Pop and Opera Datasets . . . . .	35
3.6	Conclusions . . . . .	35
<b>4</b>	<b>Hierarchical Approaches for Detecting Singing Activity, Gender, and Type</b>	<b>37</b>
4.1	Introduction . . . . .	37
4.2	Related Work . . . . .	38
4.3	Hierarchical Class Model . . . . .	39
4.4	Hierarchical Singing Detection . . . . .	40
4.5	Experiments . . . . .	41
4.5.1	Dataset . . . . .	42
4.5.2	Evaluation Measures . . . . .	43
4.5.3	Model . . . . .	43
4.5.4	Results . . . . .	43
4.6	Conclusion . . . . .	46
<b>5</b>	<b>Hierarchical Approaches for Instrument Activity Detection</b>	<b>47</b>
5.1	Introduction . . . . .	47
5.2	Related Work . . . . .	49
5.2.1	Instrument Detection . . . . .	49
5.2.2	Hierarchical Classification for Audio . . . . .	50
5.2.3	Orchestra and Opera in MIR . . . . .	51
5.3	Hierarchical Instrument Detection . . . . .	51
5.3.1	Hierarchical Class Model . . . . .	51
5.3.2	Classification Approach . . . . .	51
5.3.3	Evaluation Measures . . . . .	52
5.4	Orchestral Datasets . . . . .	53
5.4.1	Multi-Track Datasets . . . . .	53
5.4.2	Music Synchronization . . . . .	54
5.4.3	Dataset Overview and Split . . . . .	55
5.5	Model Architecture . . . . .	57
5.6	Main Results . . . . .	59
5.7	Consistency Losses . . . . .	62
5.8	Analysis of Confounding Factors . . . . .	64
5.9	Conclusion . . . . .	68
<b>6</b>	<b>A Cross-Version Approach to Representation Learning for Instrumentation</b>	<b>71</b>
6.1	Introduction . . . . .	71
6.2	Related Work . . . . .	73

6.3	Cross-Version Approach to Audio Representation Learning . . . . .	73
6.4	Experimental Setup . . . . .	75
6.4.1	Dataset and Splits . . . . .	75
6.4.2	Model . . . . .	76
6.4.3	Baselines . . . . .	76
6.5	Results . . . . .	77
6.5.1	Feature Analysis using Self-Similarity . . . . .	77
6.5.2	Qualitative Results . . . . .	78
6.5.3	Quantitative Results . . . . .	79
6.5.4	Feature Analysis Using Classification . . . . .	81
6.6	Conclusion . . . . .	82
<b>7</b>	<b>Soft Dynamic Time Warping for Pitch Activity Detection</b>	<b>83</b>
7.1	Introduction . . . . .	83
7.2	Weakly Aligned Training for MPE . . . . .	85
7.3	Soft Dynamic Time Warping . . . . .	86
7.4	Application to Multi-Pitch Estimation . . . . .	87
7.4.1	Implementation Details and Evaluation Metrics . . . . .	87
7.4.2	Comparison with MCTC . . . . .	88
7.4.3	Incorporating Note Durations . . . . .	89
7.4.4	Cross-Dataset Experiment . . . . .	89
7.5	Extension to Real-Valued Targets . . . . .	90
7.5.1	Pitch Estimation with Overtone Model . . . . .	91
7.5.2	Cross-Version Training . . . . .	91
7.6	Conclusion . . . . .	91
<b>8</b>	<b>Leitmotif Classification in Operas by Richard Wagner</b>	<b>93</b>
8.1	Introduction . . . . .	93
8.2	Scenario . . . . .	95
8.2.1	Leitmotifs in Wagner’s Ring . . . . .	95
8.2.2	Recorded Performances . . . . .	96
8.2.3	Leitmotif Classification Task . . . . .	97
8.3	Recurrent Neural Network for Leitmotif Classification . . . . .	98
8.4	Experiments . . . . .	99
8.4.1	Setup and Splits . . . . .	99
8.4.2	Evaluation Measures . . . . .	100
8.4.3	Results on the Performance Split . . . . .	100
8.4.4	Results on the Occurrence Split . . . . .	102

## Contents

8.4.5	Noise Class . . . . .	103
8.4.6	Random Labels . . . . .	103
8.5	Summary and Future Work . . . . .	104
<b>9</b>	<b>Leitmotif Activity Detection in Opera Recordings</b>	<b>105</b>
9.1	Introduction . . . . .	105
9.2	Musical Scenario and Task Specification . . . . .	107
9.2.1	Leitmotifs in Wagner’s Ring . . . . .	107
9.2.2	Cross-Performance Dataset . . . . .	109
9.2.3	Leitmotif Activity Detection . . . . .	110
9.3	Deep Learning-Based Leitmotif Activity Detection . . . . .	112
9.3.1	Methods . . . . .	112
9.3.2	Evaluation Measures . . . . .	115
9.3.3	Evaluation with Tolerance . . . . .	116
9.3.4	Experimental Results . . . . .	117
9.4	Robustness to Input Modifications . . . . .	118
9.4.1	Tempo Changes . . . . .	119
9.4.2	Pitch Shifts . . . . .	121
9.4.3	Noise . . . . .	122
9.4.4	Shuffling . . . . .	122
9.5	Towards Less Informed Scenarios . . . . .	124
9.6	Conclusion . . . . .	125
<b>10</b>	<b>Summary and Future Work</b>	<b>127</b>
	<b>Abbreviations</b>	<b>131</b>
	<b>Bibliography</b>	<b>133</b>







# 1 Introduction

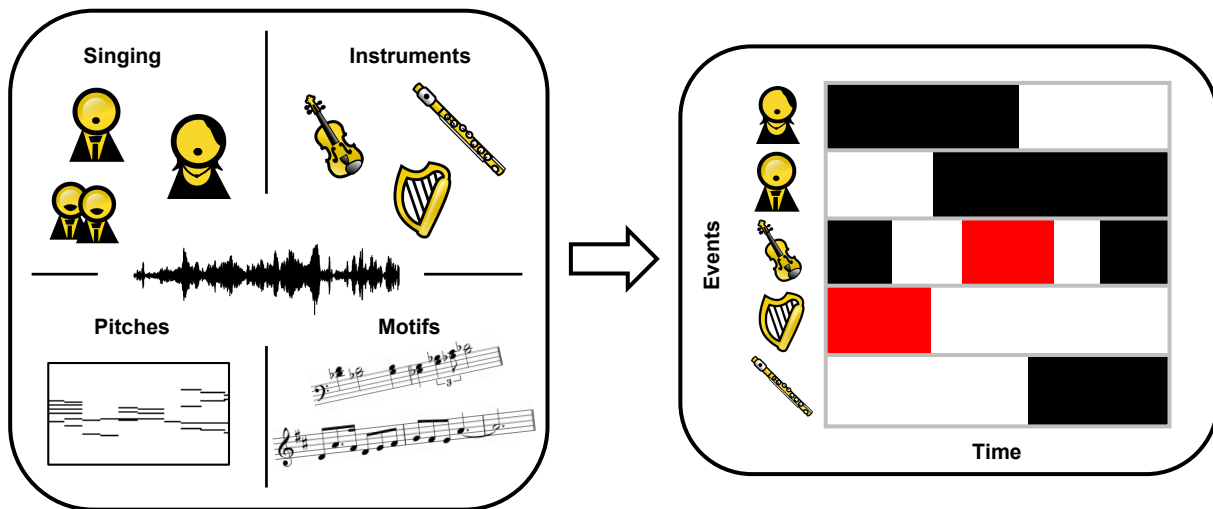
Music is commonly understood as a combination of sounds, arranged artistically to create beauty and express emotions.<sup>1</sup> When creating a new piece of music, composers may employ different instruments or human voices. They may choose the pitches to be played and organize them into melodies and harmonies. Often, they repeat certain ideas and patterns in order to create a coherent experience for the audience. When a musical composition is performed, it is transformed into acoustic waves (sounds) and these artistic choices are realized as sound events. Microphones may be used to record mixtures of sounds, turning them into audio signals to be listened to or to be processed by a machine. It usually requires no conscious effort on behalf of a human listener to perceive when a certain instrument or singer is active, or to hear different pitches within a music recording. For a machine, however, it can be hard to answer seemingly simple questions like “Which instrument is playing here?” or “Is this a female or a male voice?”

In this thesis, we study computational methods for automatically detecting the activity of different musical sound events within audio recordings. In other words, we build systems that can identify the time instances where certain events occur over the course of a recording. Such systems are fundamental building blocks for many applications in the field of music information retrieval (MIR), which examines computational methods for processing and understanding music. For example, a method for detecting different pitches might be used to analyze and evaluate the progress of a student who is learning to play a new instrument [119] or to learn a model of expressive performances [89]. Automated detection systems may also be helpful to enable navigating and searching in large music databases [17], or to enhance music listening experiences by providing additional information and effects alongside a performance [124].

From a technical perspective, our goal is related to automatic sound event detection (SED)—the task of detecting event activity in environmental audio recordings [137, 210]. Research on SED is typically concerned with the sounds of cars, sirens, or crying toddlers. In contrast, we aim at musical sound event detection, i. e., detecting the activity of musical sound classes within music recordings. In particular, we consider four different musical sound event types, also illustrated on the left-hand side of Figure 1.1: Singing activity, musical instrument activity, pitch activity, and activity of certain types of motifs. In all of these cases, our goal is to convert a music recording into a representation of event activity over time, as illustrated on the right-hand side of Figure 1.1.

---

<sup>1</sup> music, n. and adj. In *Oxford English Dictionary Online*. Oxford University Press, 2023.

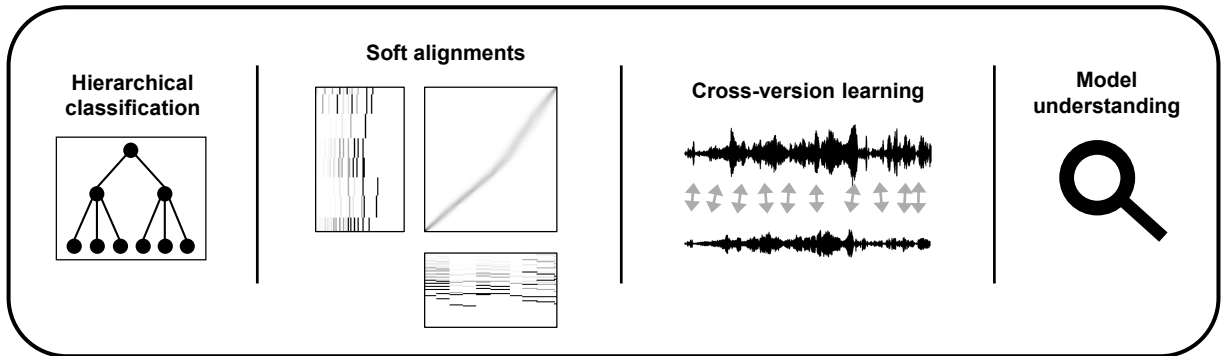


**Figure 1.1:** In this thesis, we consider different types of musical sound events within an opera or orchestral performance (left-hand side) and build computational methods to detect the activity of these events over time (right-hand side). Finally, we evaluate our automatic approaches against reference annotations to identify correct (black regions) and wrong (red regions) predictions, as illustrated on the right-hand side.

As our main application scenario, we consider orchestral and opera music. This style of music is rarely studied in the field of MIR, which often focuses on scenarios with greater commercial relevance, such as popular music. Orchestral sound mixtures feature a broad dynamic range, rhythmic and harmonic complexity, and a high degree of polyphony, i. e., a large number of instruments and singers are active simultaneously. In this context, composers often deliberately aim to make several sound sources appear as one coherent whole, further complicating the identification of individual sound classes. For these reasons, orchestral music represents a challenging scenario to identify the strengths and weaknesses of automatic detection methods.

As our main technical tool for activity detection, we use data-driven methods, especially approaches based on deep learning (DL). This has become the dominant paradigm in multimedia processing and is also being applied to various musical detection problems [8, 84, 114] and other tasks in MIR [26, 148]. In this thesis, we use several deep classification networks that learn to assign audio excerpts to different categories, based on a set of training examples with given reference annotations. Besides applying deep classification networks, which are a relatively common tool for SED, we also explore more advanced techniques, as illustrated in Figure 1.2. In the context of singing and instrument activity detection, we explore the use of hierarchical structures within DL-based systems. For pitch activity detection, we utilize approaches for differentiable or “soft” sequence alignment. We also demonstrate the potential of using cross-version datasets (i. e., datasets that contain multiple recordings/performances of the same musical piece) for music representation learning. Finally, we put a particular focus on understanding our detection systems, their robustness, and susceptibility to confounding effects.

In summary, this thesis covers a range of musical sound event detection tasks within a complex and seldom considered orchestral scenario. We approach singing, instrument, pitch, and leitmotif activity



**Figure 1.2:** Main deep learning techniques considered in this thesis.

detection using data-driven methods and explore several advanced deep learning techniques to improve and understand detection results. With this focus, the thesis lies at the intersection of MIR, signal processing, and machine learning. The next section gives an overview of our main contributions.

## 1.1 Structure and Main Contributions of this Thesis

This thesis is organized as follows. In Chapter 2, we cover fundamentals relevant to all parts of the thesis. These include a description of different audio representations, fundamentals of activity detection for musical and general sound events, as well as a case study of two representative approaches for a standard musical sound event detection task.

We then begin the main part of the thesis in Chapter 3, where we investigate singing voice detection (SVD), i. e., detecting the presence or absence of singing over the course of a music performance. We compare an approach based on traditional machine learning techniques (using hand-crafted features and a random forest classifier) with a DL-based approach using convolutional neural networks (CNNs). In our experiments, we put a particular focus on understanding the effect of dataset size and variability on the classifiers' performance. To this end, we utilize a cross-version dataset of 16 performances of Richard Wagner's tetralogy of operas *Der Ring des Nibelungen*, which is also used in subsequent chapters. We find that both the classical and the DL-based approach yield comparable results, although both show a tendency to overfit to the specific musical pieces in the training data.

We continue our exploration of SVD in Chapter 4, where we extend the scenario towards also detecting singer gender and voice type. We formalize this as a hierarchical classification problem and describe several approaches that utilize the class hierarchy for detection. In particular, we present additional loss terms for improving the hierarchical consistency of results. Our experiments demonstrate that a joint classification strategy using these additional loss terms is able to provide strong and consistent detection results with a single deep classifier.

In Chapter 5, we explore this hierarchical approach further and apply it to the even more challenging problem of instrument activity detection (IAD). In addition to considering a larger class hierarchy, we provide extensive additional analyses of the impact of hierarchical classification and our consistency losses. To perform our experiments, we collect a cross-version dataset of orchestral recordings with aligned instrument activity annotations. However, obtaining such reference annotations is a cumbersome and costly process. We show that utilizing hierarchical detection approaches allows us to learn from fewer fine-grained instrument annotations. We also analyze our model in detail and find that it exploits confounding effects between instrument classes that are often active at the same time.

In Chapter 6, we circumvent the need for having any sound event annotations at all and instead perform representation learning on orchestral music recordings. We propose a learning strategy that exploits the correspondences between different versions of a piece, as provided by cross-version datasets. Even though our strategy does not require instrument activity annotations (and can thus be trained on larger, unannotated datasets compared to our system in Chapter 5), we show that it is able to capture aspects of instrumentation and outperforms an alternative strategy that does not utilize cross-version information.

In Chapter 7, we consider pitches as our events of interest. We present an approach that can utilize weakly-aligned annotations for learning, i. e., annotations that are not perfectly in-sync with the music recording. The approach is based on a differentiable approximation of the classical dynamic time warping algorithm. We show that our method performs favorably compared to another, more complicated state-of-the-art algorithm for the same setting. We further demonstrate that it easily generalizes to alignment problems beyond pitch detection, thus opening up the possibility of learning from weakly-aligned data for various MIR tasks.

In the two final chapters, Chapter 8 and Chapter 9, we consider special kinds of musical patterns, called leitmotifs, as the events to be detected. Leitmotifs are musical ideas used for storytelling in opera or film soundtracks. They are particularly difficult to detect, because they may change significantly in key, rhythm, timbre and other musical aspects over the course of a piece. In Chapter 8, we consider a simplified scenario, where leitmotifs are classified based on pre-segmented excerpts. Subsequently, in Chapter 9, we extend this to a more realistic scenario where leitmotifs are detected throughout an entire performance, including simultaneous motif activity. In both cases, we show that our automated approaches are effective, provided the piece being analyzed has been included during training. However, we also find that our systems overfit to certain spectral statistics while ignoring aspects that a human listener would attend to. This limits their capability to generalize to unseen musical pieces.

Finally, we conclude this thesis in Chapter 10 with a summary and detailed discussion of promising directions for future work.

## 1.2 Publications Related to Ph.D. Thesis

The main chapters of this thesis are based on articles that have previously been published or accepted to appear in peer-reviewed journals and conference proceedings within the fields of audio signal processing and MIR. I am the main author of and contributor to all these publications.

- [103] Michael Krause and Meinard Müller. Hierarchical classification for instrument activity detection in orchestral music recordings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 31:2567–2578, 2023. doi: 10.1109/TASLP.2023.3291506
- [108] Michael Krause, Christof Weiß, and Meinard Müller. A cross-version approach to audio representation learning for orchestral music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Milano, Italy, 2023
- [109] Michael Krause, Christof Weiß, and Meinard Müller. Soft dynamic time warping for multi-pitch estimation and beyond. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Rhodes Island, Greece, 2023. doi: 10.1109/ICASSP49357.2023.10095907
- [102] Michael Krause and Meinard Müller. Hierarchical classification for singing activity, gender, and type in complex music recordings. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 406–410, Virtual and Singapore, 2022. doi: 10.1109/ICASSP43922.2022.9747690
- [106] Michael Krause, Meinard Müller, and Christof Weiß. Singing voice detection in opera recordings: A case study on robustness and generalization. *Electronics*, 10(10):1214:1–14, 2021. doi: 10.3390/electronics10101214
- [105] Michael Krause, Meinard Müller, and Christof Weiß. Towards leitmotif activity detection in opera recordings. *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 4(1):127–140, 2021. doi: 10.5334/tismir.116
- [104] Michael Krause, Frank Zalkow, Julia Zalkow, Christof Weiß, and Meinard Müller. Classifying leitmotifs in recordings of operas by Richard Wagner. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 473–480, Montréal, Canada, 2020. doi: 10.5281/zenodo.4245472

## 1.3 Additional Publications

Aside from the main articles that make up this thesis, I contributed to the following additional publications in the field of audio and music processing.

- [107] Michael Krause, Sebastian Strahl, and Meinard Müller. Weakly supervised multi-pitch estimation using cross-version alignment. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Milano, Italy, 2023
- [227] Christof Weiß, Vlora Arifi-Müller, Michael Krause, Frank Zalkow, Stephanie Klauk, Rainer Kleinertz, and Meinard Müller. Wagner Ring Dataset: A complex opera scenario for music processing and computational musicology. *Transactions of the International Society for Music Information (TISMIR)*, 2023
- [188] Simon Schwär, Michael Krause, Michael Fast, Sebastian Rosenzweig, Frank Scherbaum, and Meinard Müller. Singing voice reconstruction from larynx microphone signals. *Submitted for publication*, 2023

- [152] Yigitcan Özer, Michael Krause, and Meinard Müller. Using the sync toolbox for an experiment on high-resolution music alignment. In *Demos and Late Breaking News of the International Society for Music Information Retrieval Conference (ISMIR)*, Online, 2021. URL <https://archives.ismir.net/ismir2021/latebreaking/000025.pdf>
- [147] Meinard Müller, Yigitcan Özer, Michael Krause, Thomas Prätzlich, and Jonathan Driedger. Sync Toolbox: A Python package for efficient, robust, and accurate music synchronization. *Journal of Open Source Software (JOSS)*, 6(64): 3434:1–4, 2021. doi: 10.21105/joss.03434

## 1.4 Acknowledgments

I first learned about music computing through a course I took during a year abroad at the University of Edinburgh in 2016. That course—focusing primarily on symbolic music processing—had little to do with the subject of this PhD thesis, finished almost seven years later. It did motivate me, however, to keep on searching. I found my way to Chemnitz in September 2017, where Meinard Müller and his research group organized a tutorial and workshop on MIR for the Jahrestagung der Gesellschaft für Informatik. I vividly remember being endlessly fascinated by every single talk and poster presentation that took place during those two days, as well as being delighted about the approachable and friendly nature of Meinard and his colleagues. A year later, I took the opportunity to visit my first ISMIR conference in Paris and my excitement grew even more. After meeting Meinard again in Paris and following a visit to Erlangen in November 2018, I finally started my PhD at the AudioLabs in September 2019. Now, it is time to thank the colleagues and friends that accompanied me since.

First and foremost, I would like to say a very big “Thank you” to you, Meinard. I am extremely grateful for your time, advice and immense support during these years. The effort you extend to your PhD students is exceptional. Without you, I would never have started a PhD in MIR.

I am also deeply thankful to Christof Weiß, who co-supervised me on many of the papers that make up this thesis and enabled this research through his prior work. Thank you, Christof, for your detailed feedback and numerous ideas.

I would also like to thank Dasaem Jeong for his effort as reviewer of this thesis and for great discussions about opera.

Throughout my time at the AudioLabs, I had the pleasure and honor to work with and learn from many knowledgeable colleagues. This includes, of course, the members of “GroupMM”, namely (in alphabetical order): Jakob Abeßer, Vlora Arifi-Müller, Judith Bauer, Peter Meier, Yiğitcan Özer, Sebastian Rosenzweig, Simon Schwär, Hendrik Schreiber, Frank Zalkow, and Johannes Zeitler. I am incredibly thankful for your help, especially during the final stretches of thesis writing.

Aside from Meinard’s group, I was supported by many other people at the AudioLabs, including (again, alphabetically) Alexander Adami, Ahmad Aloradi, Carlotta Anemüller, Youssef El Baba, Leonie Bast, Soumitro Chakrabarty, Srikanth Raj Chetupalli, Pablo Delgado, Sascha Dick, Bernd Edler, Mohamed

Elminshawi, Esther Fee Feichtner, Richard Füg, Ünal Ege Gaznepoğlu, Sophie Grögler, Ning Guo, Philipp Götz, Emanuël Habets, Tracy Harris, Jürgen Herre, Adrian Herzog, Thorsten Kastner, Srikanth Korse, Kishor Kayyar Lakshminarayana, Thomas Lieb, Wolfgang Mack, David Maier, Goran Markovic, Daniele Mirabilii, Nils Peters, Day-See Riechmann, Thomas Robotham, Dipanjan Datta Roy, Eva Schmidt, Lorenz Schmidt, Konstantin Schmidt, Kerstin Schröppel, Neeraj Kumar Sharma, Martin Strauß, Frederike Stutika, Fajar Falah Supono, Matteo Torcoli, Stefan Turowski, Julian Wechsler, Frank Wefers, Elke Weiland, Nils Werner, Niklas Winter, and Zeyu Xu.

In these years, I also had the privilege to supervise several student jobs, projects and theses by (alphabetically) Halil Erdoğan, Carmelo Fascella, Michael Fast, Edgar Andrés Suárez Guarnizo, Seyed Mohammad Amin Heydarshahi, David Kopyto, Thomas Neher, Anna-Luisa Römling, Daniel Scheller, Sebastian Strahl, and Tim Zunner.

Few of the experiments and results presented in this thesis would have been possible without some enormous data collection and annotation efforts, which must not be overlooked. I am very grateful to everyone involved in the preparation of the datasets used in this thesis, most importantly, the Wagner Ring Dataset. A special thanks goes towards Julia Zalkow for undertaking the massive task of annotating leitmotif occurrences throughout the Ring.

During my PhD, I had the chance to meet many inspiring researchers from the MIR community. In particular, I would like to thank Ye Wang and his students at the Sound and Music Computing Lab in Singapore for their hospitality, as well as the participants of the Dagstuhl Seminar 22082 for enlightening discussions.

I am also thankful to all partners involved in the ISAD and CAS projects for their support and to the participants of the ALIGN reading group. I also want to gratefully acknowledge the compute resources and support provided by the Erlangen Regional Computing Center (RRZE).

Sometimes, when you try out one thing, you end up learning about something else.

Thank you to all the true friends I've met in the past years and those who have been with me for a long time. And thank you to my family for always being the home I need.

Danke, Mama. Danke, Papa. Danke, Melanie. Danke, Oma. Danke, Opa.

Opa, diese Arbeit ist für Dich. Ich hab Dich sehr lieb.





## 2 Fundamentals

Some excerpts of Section 2.2 have been adopted from [105], while some parts of Section 2.3 build upon [106]. The first author Michael Krause is the main contributor to these articles, which he wrote in collaboration with his supervisor Meinard Müller and Christof Weiß.

In this chapter, we introduce fundamental concepts that are relevant to all subsequent parts of the thesis. We begin by reviewing typical ways of digitally representing audio and music signals in Section 2.1. In particular, we describe (music) audio as waveforms before covering time-frequency representations such as the short-time Fourier transform, log-frequency spectrograms and constant-Q transforms. In Section 2.2, we discuss related work on activity detection for music and general audio signals. Finally, in Section 2.3, we focus on singing voice activity detection as a representative example of a common musical sound event detection problem. We briefly describe two existing approaches for this task, which also play a role in later chapters of the thesis.

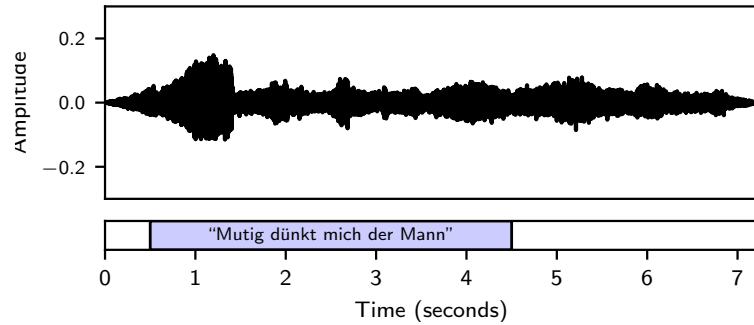
### 2.1 Audio Representations

We now introduce the most important representations of music audio considered in this thesis, providing a high-level and intuitive explanation. For more details, we refer to the textbook by Müller [145], whose notation we adopt.

#### 2.1.1 Waveforms

Human listeners typically perceive music (or any kind of sound) through their ears, which capture variations of air pressure caused by vibrating objects such as the strings of an instrument or the vocal folds of a singer. Plotting the air pressure change over time, one obtains a graphical representation of music audio, called the *waveform*. Mathematically, a waveform may be described as a function  $f : \mathbb{R} \rightarrow \mathbb{R}$ , mapping time in seconds  $t \in \mathbb{R}$  to amplitudes  $f(t) \in \mathbb{R}$ . Figure 2.1 shows the waveform of a short excerpt from an opera performance, which we will consider as our running example in this chapter. The excerpt lasts for around

**Figure 2.1:** An excerpt from an opera performance in waveform representation, serving as the running example in this chapter. Below the waveform plot, a blue bar indicates the activity and lyrics sung by a soprano singer in this excerpt.



seven seconds and contains a soprano singing a line in German (“Mutig dünkt mich der Mann”) from seconds 0.5 to 4.5, with soft string accompaniment throughout.

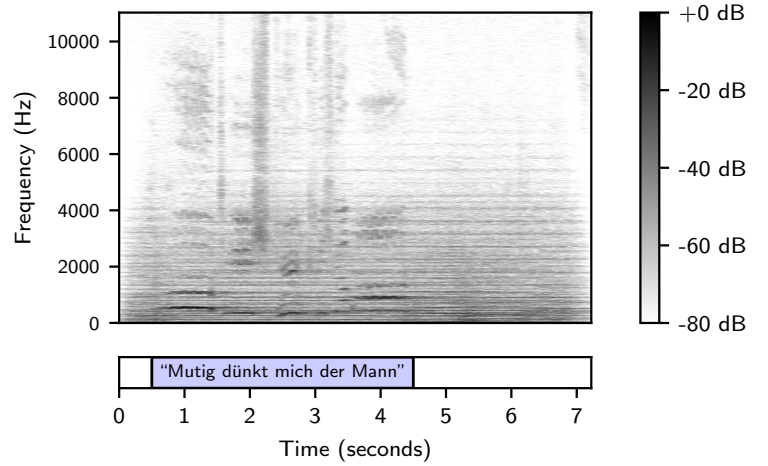
To process musical sounds on a computer, a microphone may be used to capture audible vibrations in a similar fashion. However, since digital computers are unable to directly store analog signals, one requires two additional steps to convert these signals into digital representations: *sampling* and *quantization*. Sampling refers to the process of only storing some measurements of  $f$  at certain points in time. This way,  $f$  is transformed into a finite sequence  $x : [1 : T] \rightarrow \mathbb{R}$  of amplitude values, where  $[1 : T] := \{1, 2, \dots, T\}$  are indices to the measurements taken. Quantization, on the other hand, refers to the process of mapping these real-valued amplitudes to a finite set of possible amplitudes  $\Gamma \subset \mathbb{R}$ , such that these values may be represented on a digital computer. Thus, a waveform is digitally described as a function  $x' : [1 : T] \rightarrow \Gamma$ .

We refer to [145] for a detailed description of possible sampling and quantization strategies. In this thesis, we always adopt a uniform sampling procedure, where samples from  $f$  are taken at regular time intervals, leading to a sampling rate of  $F_s$  samples per second of audio. The choice of  $F_s$  is important, as it determines the highest frequency oscillation from  $f$  that may be recovered from  $x$ . We typically choose  $F_s = 22\,050$  Hz, such that  $x$  captures frequencies of up to 11 025 Hz (according to the Nyquist theorem; enough to cover the main frequency content of all event classes we wish to detect). We usually do not further specify the exact quantization strategy used. In the following, to simplify notation, we omit the quantization step, extend  $x$  to be defined on the whole domain of  $\mathbb{Z}$  via zero-padding, and denote the resulting function again as  $x$ .

### 2.1.2 Short-Time Fourier Transform

As illustrated in Figure 2.1, the waveform representation provides only limited clues for determining the parts of a music recording where certain events (such as singing) are active. Because of this, we often consider time-frequency representations in this thesis. These reveal information about the frequency content at different time steps of a signal, enabling the localization of event activity throughout a recording. The most common such audio representation is the short-time Fourier transform (STFT).

**Figure 2.2:** The magnitude spectrogram of our running example. In the part of the excerpt containing singing, the frequency fluctuations produced by the singer are clearly visible. The remainder of the excerpt shows sounds produced by the string accompaniment. For this plot, a window size of  $N = 4096$ , a hop size of  $H = 512$  and a standard Hann window were used. Magnitudes are plotted on a logarithmic decibel scale, with +0 dB corresponding to the maximal magnitude in the excerpt.



To compute the discrete STFT, a waveform  $x$  is divided into overlapping analysis frames of length  $N \in \mathbb{N}$  using a hop size  $H \in [1 : N - 1]$  (typically  $H = \lfloor N/2 \rfloor$ ). In each of these frames, the waveform is weighted with some window function  $w : [0 : N - 1] \rightarrow \mathbb{R}$  (e. g., a standard Hann window) and correlated with complex exponentials at different frequency indices  $k \in [0 : K]$ , where  $K = N/2$  corresponds to the Nyquist frequency of  $F_s/2$ . Formally, the *discrete STFT*  $\mathcal{X} \in \mathbb{C}^{\mathbb{Z} \times K}$  of the waveform  $x$  is given by

$$\mathcal{X}(n, k) := \sum_{l=0}^{N-1} x(l + nH)w(l) \exp(-2\pi ikl/N) \quad (2.1)$$

for time indices  $n \in \mathbb{Z}$  and frequency indices  $k \in [0 : K]$ . For details, we refer to [145]. Notably, the hop size parameter  $H$  controls the number  $F_s/H$  of frames per second in  $\mathcal{X}$ , also called *frame rate*. Furthermore, the size  $N$  of the analysis window controls the trade-off between time and frequency resolution in  $\mathcal{X}$ . A smaller  $N$  allows for more exact localization of events in time, whereas a larger  $N$  provides more fine-grained information of the frequency content inside an analysis frame.

$\mathcal{X}$  is a complex-valued representation, encoding information about both magnitudes and phases for different time and frequency indices. For event detection, we usually consider only the *magnitude spectrogram*  $\mathcal{Y} \in \mathbb{R}^{\mathbb{Z} \times K}$  computed as

$$\mathcal{Y} := |\mathcal{X}|. \quad (2.2)$$

Finally, the magnitudes in  $\mathcal{Y}$  are often rescaled to account for the logarithmic perception of loudness by human listeners.

Figure 2.2 shows a magnitude spectrogram of the running example. The vertical axis corresponds to different frequencies, while the horizontal axis corresponds to the time axis of the recording. Levels of gray indicate magnitudes. We can observe rich harmonic structures (parallel horizontal lines, corresponding to integer multiples of a fundamental frequency) throughout the whole excerpt, produced by the string accompaniment. In addition, we can clearly identify the beginning and end of the singer's activity from

the characteristic vibrato patterns (wave-like structures in the spectrogram corresponding to expressive frequency modulations). Thus, the STFT is better suited than the raw waveform for identifying the activity of different events in the audio, but it does not directly solve the detection task. We will often use variants of the STFT as inputs to our detection systems.

### 2.1.3 Log-Frequency Representations

Throughout this thesis, we use variants of the STFT that account for the logarithmic perception of pitch by human listeners. These variants provide a view that is closer to human perception by using a logarithmic frequency axis.

One class of such alternative representations is the *log-frequency spectrogram*, which can be understood as a magnitude STFT with a logarithmic frequency axis. It may be computed from  $\mathcal{Y}$  in different ways, including bin re-assignment, application of a filterbank, or interpolation along the frequency axis. Here, we briefly discuss the latter option. For that, we rescale the linear frequency axis of  $\mathcal{Y}$  onto a logarithmic axis using interpolation (e. g., linear or cubic). We may, for example, choose the center frequencies of musical pitches as the targets for our rescaled axis. Dividing an octave into  $b$  bins and starting with a minimal frequency of  $f_{\min}$ , we can compute the center frequency for a bin  $d \in [0 : D - 1]$  on a rescaled axis with  $D$  bins as

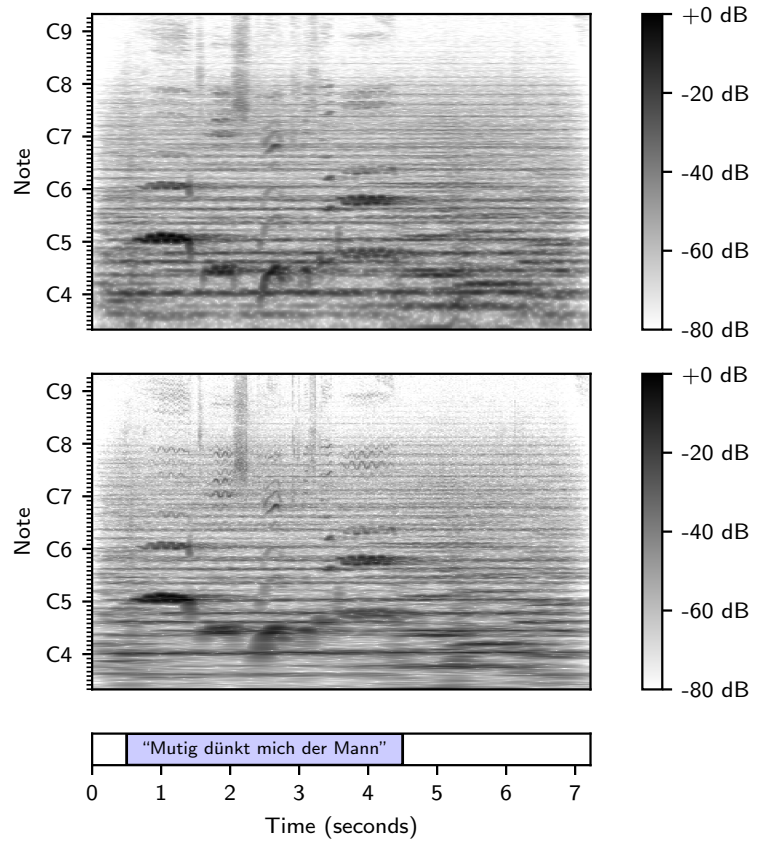
$$F(d) := 2^{d/b} \cdot f_{\min}. \quad (2.3)$$

The upper plot in Figure 2.3 gives an example of a log-frequency spectrogram computed in this fashion, where  $f_{\min} = 164$  (the center frequency of pitch E3),  $b = 120$  bins per octave are used (i. e., 10 bins for each semitone in an equal-tempered scale), and a total of  $D = 720$  bins are considered (up to the pitch E9). Compared to Figure 2.2, this visualization puts more emphasis on lower frequencies, making it easier to identify the fundamental frequency trajectory sung by the singer in this excerpt. Furthermore, octaves are now equidistant on the vertical axis, corresponding to human perception. However, as the magnitudes are obtained as interpolated values from  $\mathcal{Y}$ , this representation is blurry and imprecise, in particular for higher pitches.

Intuitively, one would like to use smaller window sizes to analyze upper frequencies (ensuring high temporal resolution) and larger window sizes to capture lower frequencies (providing high frequency resolution). This is realized by the *constant-Q transform (CQT)*, which ensures that the ratio between the center frequency  $F(d)$  and the bandwidth<sup>2</sup> for a bin  $d$  on the logarithmic frequency axis stays constant for all bins. The CQT has been proposed as a log-frequency representation specifically suited for music signals by Brown in [20]. In this thesis, we use a fast algorithm for computing the CQT described in [21, 184]. We refer to the original publications for detailed formulas and derivations. The CQT is parameterized by a hop size  $H$ , as well as  $b$ ,  $f_{\min}$ , and  $D$  as described above. The analysis window size  $N$  depends

<sup>2</sup> Roughly: The range of frequencies being covered by one bin.

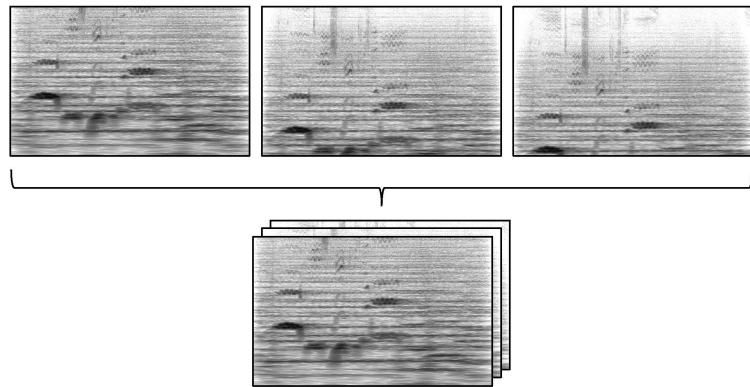
**Figure 2.3:** Different log-frequency representations of the running example. The upper plot shows a log-frequency spectrogram obtained from the magnitude spectrogram plotted in Figure 2.2 by linear rescaling of the frequency axis onto a pitch axis. The lower plot shows a CQT representation, computed with  $H = 512$ . In both plots, the lowest and highest pitches considered are E3 and E9, respectively, while 120 pitch bins are used per octave (10 per semitone).



on the bin  $d$  (details are provided in the original publications). The lower plot in Figure 2.3 shows a CQT representation of our running example. We can observe a finer-grained representation of the upper frequency structures compared to the interpolated spectrogram, making it easier to identify the singer’s vibrato in this example. On the other hand, the lower frequency structures are blurred out in time due to the use of a larger window size compared to  $\mathcal{Y}$ .

Finally, the *harmonic constant-Q transform* (*HCQT*) is a variant of the CQT representation. It was proposed as an appropriate input to neural networks for MIR tasks by Bittner et al. [13]. Here, multiple CQTs—computed using different  $f_{\min}$ —are stacked together along a third dimension, see also Figure 2.4. In practice, one usually chooses integer multiples or fractions of a reference  $f_{\min}$ , i. e.,  $1/2 \cdot f_{\min}$ ,  $f_{\min}$ ,  $2 \cdot f_{\min}$ ,  $3 \cdot f_{\min}$  and so on. In this way, magnitudes belonging to different harmonics and sub-harmonics of the same frequency are stacked on top of each other along the third axis. The resulting three-dimensional tensor is used as input for a learnable convolutional kernel. Since such kernels cover only a small receptive field along the time and frequency dimensions of the input, they are not usually able to capture different harmonics when applied to a standard CQT. When applied to an HCQT, however, the kernels can process the magnitudes of different (sub-)harmonics in addition to local time and frequency variations. As such, the HCQT is an appropriate input representation for DL systems that detect harmonic sound events. It has been used, e. g., for instrument recognition [84].

**Figure 2.4:** Conceptual overview of an HCQT representation for the running example. Several CQTs corresponding to different harmonics are computed and stacked together along a third dimension.



## 2.2 Activity Detection

As outlined in Chapter 1, this thesis is about detecting the activity of musical sound events within audio recordings. From a technical perspective, our aim has strong connections with sound event detection (SED), a task studied in the field of general audio processing. In this section, we will give an overview of research into SED and event detection in MIR, focusing on the most relevant aspects for this thesis.

SED is concerned with identifying the time instances throughout an audio recording where certain event classes occur. This is in contrast to related tasks like scene classification or audio tagging, where the aim is to assign one or several classes to an entire audio excerpt without temporal granularity. A typical application of SED is environmental sound scene analysis, where ambient sound events like car noise, sirens, or chatter must be recognized. Such systems may be employed, e. g., for robotic applications [151], noise monitoring in urban environments [176], or as an essential building block for smart home devices [55]. Other works consider bird call monitoring, which can be useful for environmental preservation initiatives [34, 141, 195].

Depending on the application scenario involved, one may require different levels of granularity from the output of an SED system. In some cases, only a single sound event is active at a time, whereas in others, several different event types may be active simultaneously, thus leading to a multi-label problem.<sup>3</sup> In addition, some systems model and evaluate event onsets and offsets, while others detect only the presence or absence of a class. Systems may also account for simultaneous activity of several events from the same class, e. g., by providing the count of active events per class as an additional output. The appropriate granularity for an SED system depends on the specific detection scenario as well as the annotations for available datasets, and affects the evaluation metrics that must be used. For a detailed introduction to SED, we refer to the tutorial by Mesaros et al. [137] and the textbook by Virtanen et al. [210, in particular Section 8.3]. A comprehensive review of different evaluation metrics for SED is provided in [136].

<sup>3</sup> In the literature on SED, these scenarios are called monophonic and polyphonic SED, respectively. To prevent confusion, we avoid those terms and always use the words “monophonic” and “polyphonic” in their original, musical sense.

Most SED systems follow a common processing pipeline that begins with data preparation and feature extraction. In this context, audio recordings are usually divided into frames and transformed into some feature domain, e. g., a log-frequency representation. See [210, Chapter 4] for an overview. These features are input to a recognition model that yields frame-wise event predictions. Finally, the model outputs may undergo a post-processing step (e. g., a median filter or hidden Markov model to remove outliers and incorporate temporal context) before being evaluated against reference annotations.

With regard to recognition models used, traditional approaches such as non-negative matrix factorization or spectrogram cross-correlation dominated the field of sound event detection until recently, as described in the survey by Stowell et al. [194]. As with many tasks in audio signal processing, recent years have seen deep neural networks (DNNs) become the dominant approaches for SED [137]. Usually, these are used in *supervised learning* schemes, where the networks are optimized on a training set of recordings with reference event annotations. In this context, annotations may be given per-frame (called strong annotations) or per-excerpt (called weak annotations). Weak annotations are easier to obtain but less accurate [137]. Network architectures that have been considered include basic fully connected networks, recurrent neural networks (RNNs), or convolutional neural networks (CNNs), as well as combinations of these [25]. More recent approaches make use of techniques such as dilated convolutions [121] or transformers [215]. We refer to [233] for an overview of neural networks for SED. Novel systems are proposed frequently and evaluated for standard (non-musical) SED scenarios at the yearly DCASE challenges.<sup>4</sup> Current research directions include extending the SED problem towards simultaneously detecting events and localizing their sound sources in a scene [161, 162], learning from weak rather than strong annotations [208], improving generalization when applying trained models to new environments or recording conditions [79, 138], and exploring augmentation strategies such as mixing of training examples [1, 240].

As for musical sound event detection, an exemplary task considered in the literature is singing voice activity detection, where regions containing singing activity must be identified, see also the following Section 2.3 and Chapter 3. Another task is beat tracking, where musical beats are considered as sound events. Here, only peak positions and no duration information must be predicted. Extending this task, Böck et al. [14] proposed an RNN that jointly detects beat and downbeat positions. Other works focus on detecting chords throughout music recordings, which can be useful for music analysis or automatic accompaniment [158]. One may also consider music transcription tasks as a variant of sound event detection. For instance, in drum transcription, individual drum hits are considered as sound events to be detected. For a comprehensive overview of recent drum transcription approaches, we refer to Wu et al. [230].

Works on musical sound event detection often focus on event presence or onsets, but do not account for offsets, since these are often ill-defined (e. g., in the case of note events fading out). A notable exception is [245], where onsets and offsets are explicitly modeled in the context of singing transcription. As another peculiarity, musical sound events are often highly correlated with background accompaniment or other

---

<sup>4</sup> <http://dcase.community/>

simultaneous events in a rhythmic or harmonic fashion (see also Chapter 9). This is in contrast to most environmental SED settings, where events typically occur independently from each other. Furthermore, musical scenarios are often inherently subjective and may allow for multiple correct answers. For example, when detecting local key in a music recording, several different detection results may be musically plausible [224]. Thus, when implementing and evaluating musical sound event detection systems, the properties of the underlying musical concepts must be kept in mind.

In this thesis, we will cover singing, musical instruments, pitches, and leitmotifs as the musical sound events to be detected. In most cases, we investigate multi-label scenarios where multiple events may be active at the same time. Our systems detect the presence or absence of different event classes and do not output onsets, offsets, or event counts. Our datasets provide strong annotations, though we also explore techniques to utilize different variants of weak annotations (see Chapter 7). From a technical perspective, most of our approaches are based on DNNs.

## 2.3 Case Study: Detecting Singing Activity

In this section, we focus on singing voice detection (SVD) as an exemplary musical sound event detection task. SVD is commonly formulated as a frame-wise, binary classification problem [114]. Thus, we are interested in identifying all frames where one or several singers are active, but we do not separately evaluate onsets or offsets, nor do we count the number of singers who are singing simultaneously in one frame.

From a technical perspective, one may generally distinguish two types of approaches for SVD. In the first, traditional machine learning classifiers are applied to hand-crafted features. In the second, neural networks learn to extract features from data. The engineering involved in the first approach is cumbersome, but may lead to more directly interpretable systems compared to the feature learning employed in neural networks. We will now describe representatives of both types on a conceptual level, while emphasizing the differences between the two paradigms. For this thesis, we have also reimplemented the approaches presented here. We provide more details on our reimplementations in Section 3.3.

### 2.3.1 Feature-Engineering Approach

Traditional systems for SVD [39, 170, 171] usually consist of two stages—the extraction of hand-crafted audio features and the supervised training of standard machine learning classifiers. Often, mel-frequency cepstral coefficients (MFCCs) are used as features, combined with classifiers such as support vector machines or decision trees [170, 209]. MFCCs are computed by performing an additional Fourier analysis step on the individual frames of a log-frequency spectrogram. The first few coefficients of the resulting vector yield a compact representation of the timbral characteristics of an audio frame and have proven

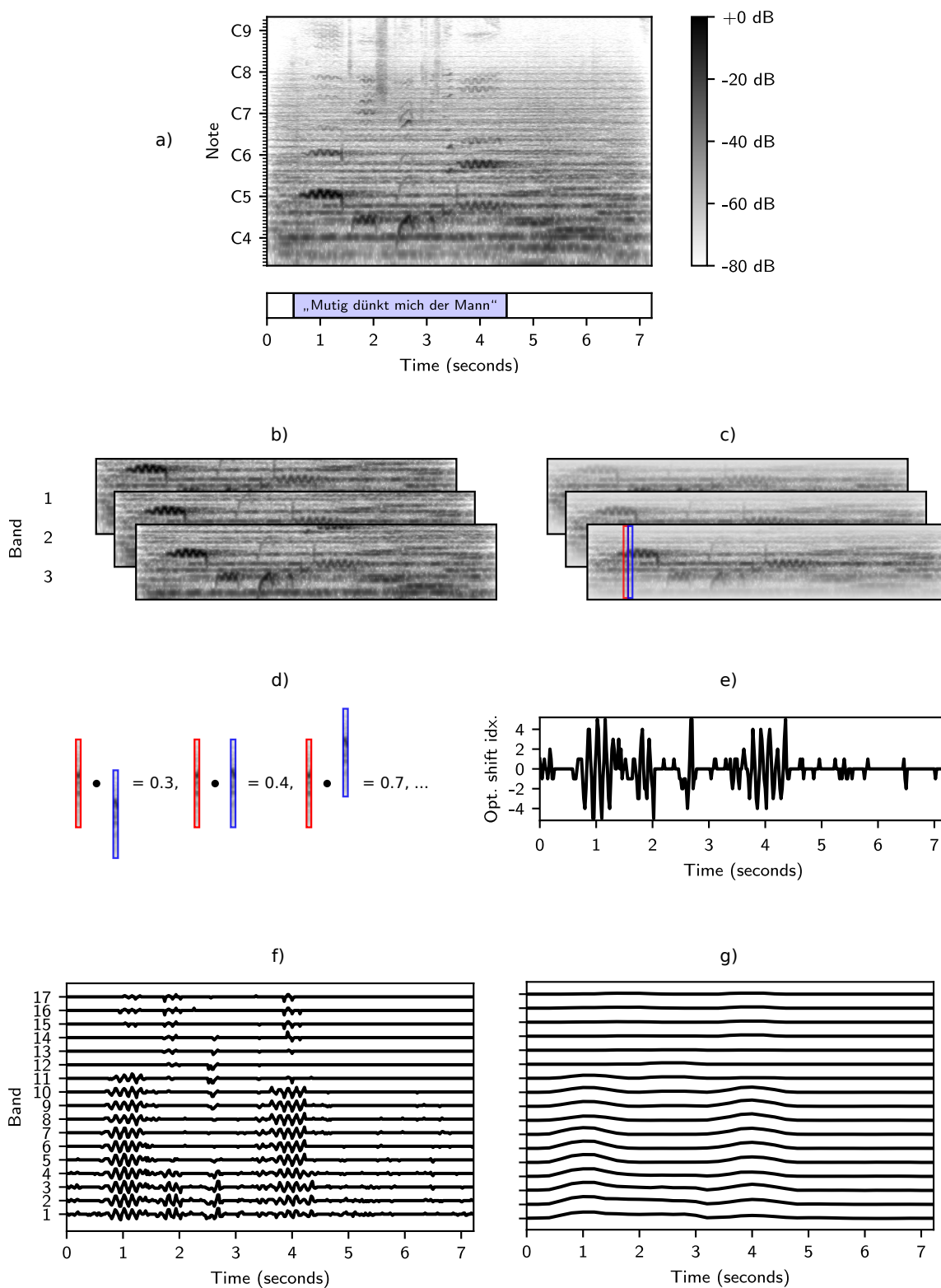


useful in speech processing applications. We refer to [36] for further information on MFCCs. Additional detail on machine learning classifiers is found in standard textbooks.

Lehner et al. [116] showed that considering further hand-crafted features can surpass the results obtained with MFCCs in isolation. In particular, they proposed a so-called *fluctogram* representation, which is well-suited for capturing vibrato-like modulations in different frequency bands, as commonly produced by professional singers (see the illustrations in Section 2.1). They further added features that describe, for each frequency band, the magnitude variance (called spectral flatness) and concentration (spectral contraction), and a feature sensitive to gradual spectral changes (vocal variance). These features serve as input to a random forest classifier [127].

As an illustrative example of traditional feature engineering, we now describe the computation of their fluctogram feature in greater detail, see also Figure 2.5. To capture the wave-like structures that characterize vibrato singing in a spectrogram, they begin by computing a log-frequency representation (Figure 2.5a) with  $f_{\min} = 164$ ,  $b = 120$ , and  $D = 720$ , based on a magnitude spectrogram computed with  $N = 2205$  and  $H = 441$  from a waveform sampled at  $F_s = 22\,050$  Hz (leading to a frame rate of 50 Hz). The authors proceed to divide the frequency axis into 17 overlapping bands. Concretely, they extract bands of width 240 bins each (corresponding to two octaves), with an overlap of 30 bins (three semitones), leading to 17 bands in total (see Figure 2.5b). The bins in each band are weighted with a Bartlett window, thus emphasizing the contribution of the central bins in each band (Figure 2.5c). Subsequently, for each band, the cross-correlation between every frame and its successor is computed, while shifting the successor frame up and down by up to 5 bins (half a semitone, using zero padding for the shift operation). The correlation values obtained using different shift indices are compared (Figure 2.5d) and the optimal shift indices are stored (Figure 2.5e). The resulting representation captures vibrato patterns within the different frequency bands (Figure 2.5f). Finally, a variance filter is applied to each band in time direction (in a centric fashion, taking the variance of 40 values at a time) and the feature sequence is downsampled by taking every 10th frame (Figure 2.5g). The resulting fluctogram variance feature consists of vectors with 17 variance values at a frame rate of 5 Hz, where a large variance indicates the presence of vibrato.

The fluctogram feature highlights both the advantages and limitations of hand-engineered features for event activity detection. On the one hand, this feature responds to clearly defined and musically relevant signal characteristics (vibrato patterns). Thus, one can be confident about the kinds of signals that this feature works well with (namely, singing recordings with heavy use of vibrato) and the kinds of signals for which it will fail (singing without vibrato, e. g., rap). On the other hand, new problem domains require careful engineering of appropriate features, which may be costly. As an alternative, we now discuss a DL-based approach that foregoes the need for hand-crafted features.



**Figure 2.5:** Computation of the fluctogram feature as proposed by Lehner et al. [116]. A log-frequency spectrogram (a) is divided into bands (b), which are weighted with a window function (c). Each frame inside each band is correlated with its successor at different shift indices (d) and the optimal shift index per frame and band is stored (e). The resulting representation captures vibrato (f). Finally, a variance filter is applied in time direction (g).

### 2.3.2 Deep Learning Approach

Recently, SVD systems relying on DNNs have become popular [115, 117, 118, 181, 219], with CNNs being among the most effective types of network architectures [139, 181, 182]. In general, a DNN is a function  $f_\theta$  with parameters  $\theta$  (also known as the *weights* of the neural network), which are optimized to meet some quality criterion on some dataset. Most neural networks for SVD are functions that map a spectrogram-like representation of an audio excerpt to a scalar in the interval  $[0, 1]$ , thus  $f_\theta: \mathbb{R}^{T \times K} \rightarrow [0, 1]$ , where  $T$  is the number of frames and  $K$  is the number of frequency bins in the input spectrograms. The output scalar usually corresponds to the predicted probability for singing activity in the center frame of the input. Finally, the quality criterion, often called the *loss function*, is some discrepancy between the output  $\hat{y} = f_\theta(\mathcal{Y}) \in [0, 1]$  of the network on some input  $\mathcal{Y} \in \mathbb{R}^{T \times K}$  and the correct *target (label)*  $y \in \{0, 1\}$  for  $\mathcal{Y}$  (as determined by a human annotator). A common choice is the *binary cross-entropy loss function*

$$\mathcal{L}_{\text{BCE}}(\hat{y}, y) = -(y \cdot \log(\hat{y}) + (1 - y) \cdot \log(1 - \hat{y})), \quad (2.4)$$

which is minimized when the prediction  $\hat{y}$  and target  $y$  are equal. During training of a neural network, this loss is minimized over a training dataset  $\mathcal{D} = (\mathcal{Y}_i, y_i)_{i \in [1:N]}$  containing  $N$  pairs of inputs  $\mathcal{Y}_i$  and targets  $y_i$ , leading to a total loss of

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^N \mathcal{L}_{\text{BCE}}(f_\theta(\mathcal{Y}_i), y_i). \quad (2.5)$$

In practice,  $f_\theta$  is usually constructed as a concatenation of differentiable functions<sup>5</sup> such that the gradients of  $\mathcal{L}$  with regard to the weights  $\theta$  may be computed using the back-propagation algorithm.  $\mathcal{L}$  can then be efficiently minimized using first-order methods like stochastic gradient descent. For a detailed introduction into these techniques, we refer to textbooks like [62].

As an illustrative example of DNNs for activity detection, we outline a state-of-the-art system for SVD proposed by Schlüter et al. [181, 182]. Their model follows the architectural paradigm of VGGNet [190], a popular CNN for computer vision applications like image classification. In this paradigm, several stacked convolutional and max-pooling layers process the network input and learn to automatically extract features—in contrast to the approach using hand-crafted features described above. These extracted features are subsequently passed through several dense layers that finally produce the prediction  $\hat{y}$ . Compared to the spectrogram, this network architecture is not designed to be sensitive to a particular signal characteristic. Instead, the network may extract arbitrary features that would otherwise need to be hand-crafted. Further extensions of this architecture may improve SVD results, such as a recently proposed approach by Zhang et al. [242] combining recurrent and convolutional layers.

<sup>5</sup> More exactly, sub-differentiable functions, since many standard DNN building blocks (e. g., max-pooling or the rectified linear unit (ReLU)) are not fully differentiable.

**Table 2.1:** Network architecture used for the singing activity detection system in [181, 182].

Layer (Kernel size), (Strides)	Output Shape	Parameters
Input	(115, 80)	
Conv2D (3, 3), (1, 1)	(113, 78, 64)	640
Batch normalization	(113, 78, 64)	256
Conv2D (3, 3), (1, 1)	(111, 76, 32)	18 464
Batch normalization	(111, 76, 32)	128
MaxPool2D (3, 3), (3, 3)	(37, 25, 32)	
Conv2D (3, 3), (1, 1)	(35, 23, 128)	36 992
Batch normalization	(35, 23, 128)	512
Conv2D (3, 3), (1, 1)	(33, 21, 64)	73 792
Batch normalization	(33, 21, 64)	256
Conv2D (3, 18), (1, 1)	(31, 4, 128)	442 496
Batch normalization	(31, 4, 128)	512
MaxPool2D (1, 4), (1, 4)	(31, 1, 128)	
Flatten	(3968)	
Dropout 0.5	(3968)	
Dense	(256)	1 016 064
Batch normalization	(256)	1024
Dropout 0.5	(256)	
Dense	(64)	16 448
Batch normalization	(64)	256
Dropout 0.5	(64)	
Dense	(1)	65
Output: Sigmoid	(1)	

The network architecture used by Schlüter et al. is given in Table 2.1. It takes log-frequency spectrogram patches<sup>6</sup> for excerpts of 1.64 s audio as input, computed at a frame rate of 70 Hz and with 80 bins reaching from  $f_{\min} = 27.5$  Hz to 8000 Hz. Thus, their input representation is a matrix of size (115, 80). The network consists of five *2D-convolutional layers* (Conv2D in Table 2.1), which locally correlate their input with learnable filter kernels that may extract arbitrary features. Each convolutional layer is followed by a custom non-linear activation function<sup>7</sup> and *batch normalization* [87], which normalizes the kernel outputs to conform to some (learned) mean and variance and has been shown to improve the convergence behavior of DNN training [178]. In addition, *max-pooling layers* are used that reduce the input dimension and introduce some degree of translation invariance. A detailed account of convolution and max-pooling operations in neural networks is provided in [43]. The result of the convolutional and max-pooling layers is subsequently *flattened*, i. e., concatenated into a single one dimensional feature vector. This vector is then processed by three *dense layers* (often also called fully-connected layers) with learnable weights, which further transform and reduce the dimension of the feature vector. The dense layers are interspersed with *dropout* operations, which randomly set a fraction of layer outputs (here, 50%) to zero during training. This has been shown to improve the generalization behavior of DNNs [192]. The scalar output of the final dense layer is processed by a *sigmoid* activation function, which produces predictions in the range [0, 1], as described above. In total, the network has around 1.6 million learnable weights.

<sup>6</sup> Note that Schlüter et al. use a mel-filterbank to obtain their input representation, as opposed to the interpolation-based method described in Section 2.1.

<sup>7</sup> Schlüter et al. use a custom variant of a leaky ReLU activation.

The neural network models discussed in later chapters of this thesis employ similar building blocks as well as some additional layer types which have not been explained above (such as dilated convolutions or recurrent layers). For detailed explanations of the neural network building blocks, we refer to standard textbooks on deep learning, such as [62].

To conclude, the DL system discussed in this section learns to automatically extract features, based on training data provided. This may have both advantages and disadvantages. In contrast to the feature engineering approach discussed before, no difficult and domain-specific feature design is necessary. However, since the neural network can learn to extract arbitrary features from its input data, it is hard to characterize the kinds of singing recordings that the trained system would work well for and those for which it will fail. In particular, the network may learn to respond to confounding factors that are correlated with singing activity in the training dataset, but which are not actual properties of the singing voice (e. g., loudness, see [182]). We will revisit such issues throughout all chapters of this thesis.



## 3 Singing Activity Detection in Opera Recordings

This chapter is based on [106]. The first author Michael Krause is the main contributor to this article. In collaboration with his supervisor Meinard Müller and Christof Weiß, he devised the ideas, designed the experiments, and wrote the paper. Furthermore, Michael Krause implemented all approaches and conducted the experiments.

Automatically detecting the presence of singing in music audio recordings is a central task within MIR. While recent machine learning systems produce high-quality results on this task, the reported experiments are usually limited to popular music and the trained systems often overfit to confounding factors. In this chapter, we aim to gain a deeper understanding of such machine learning methods and investigate their robustness in a challenging opera scenario. To this end, we compare two state-of-the-art methods for singing voice detection (SVD) based on supervised learning: A traditional approach relying on hand-crafted features with a random forest classifier, as well as a deep learning approach relying on CNNs (as described in Section 2.3). To evaluate these algorithms, we make use of a cross-version dataset comprising 16 recorded performances of Richard Wagner’s four-opera cycle *Der Ring des Nibelungen*. This scenario allows us to systematically investigate the ability to generalize to unseen versions, musical works, or both. In particular, we study the trained systems’ robustness depending on the acoustic and musical variety, as well as the overall size of the training dataset. Our experiments show that both systems can robustly detect singing voice in opera recordings even when trained on relatively small datasets with little variety.

### 3.1 Introduction

Singing constitutes a central component of many musical traditions. Identifying segments of singing activity—often denoted as singing voice detection (SVD)—therefore provides essential information about the content and structure of music recordings and may also serve as a pre-processing step for tasks such as lyrics alignment [110, 193] or lyrics transcription [65]. SVD has historically received a lot of attention within the field of MIR [9, 83]. The majority of SVD systems are designed for and evaluated on popular music [114, 115, 118, 150, 171]. However, Scholz et al. [183] showed that data-driven SVD methods

usually do not generalize well to other genres not seen during training, implying that the popular music focus limits the general applicability of SVD systems.

Particular differences exist between popular and classical music, which is due to the distinct singing techniques and instrumentations involved. Within classical music, opera recordings constitute challenging scenarios where singing voices are often embedded in a rich orchestral texture. As a peculiarity of classical music, several recorded *versions* (i. e., recordings, performances) of a musical work are usually available. Such cross-version scenarios provide great opportunities for testing the robustness of MIR algorithms in different tasks [52, 146] and their capability for generalizing across different versions [139, 224]. In particular, such cross-version analyses have shown that machine learning algorithms can overfit to characteristics of certain versions or musical works [224]. In the light of these observations, we want to investigate whether—and to what extent—SVD algorithms suffer from such overfitting effects. While one obtains good evaluation results with state-of-the-art SVD systems, previous investigations have shown that these systems sometimes over-adapt to confounding factors such as loudness [182] or singing style [183]. Therefore, we cannot generally expect these systems to generalize to unseen musical scenarios. Following these lines, our case study yields insights into the generalization behavior of machine learning systems, the aspects of training data that are relevant for building robust systems, and the benefits of deep learning against traditional machine learning approaches, which may be relevant beyond the SVD task. With this, our study may serve as an inspiration for similar research on other data-driven approaches to MIR, audio, and speech processing.

In this chapter, we aim to gain a deeper understanding of two state-of-the-art SVD methods, which represent two commonly used strategies: The traditional strategy based on hand-crafted features [39] and the strategy based on DL [182], respectively (previously described in Section 2.3). To analyze these systems, we make use of a cross-version scenario comprising the full cycle *Der Ring des Nibelungen* by Richard Wagner (four operas, 11 acts) in 16 different versions, thus leading to a novel dataset spanning more than 200 hours of audio in total. In this scenario, different versions vary with regard to singers' timbre and singing style, musical interpretation, and acoustic conditions, whereas different operas vary in singing registers and characters, lyrics, and orchestration. We exploit this scenario in a series of systematic experiments in order to analyze the robustness of the two algorithms depending on the musical and acoustic variety, as well as on the size of the training dataset. Our results indicate that both systems perform comparably well and are capable of generalizing across versions and operas—despite the complexity of the scenario and the variety of the data. This result shows that SVD systems based on traditional techniques may perform on par with DL-based approaches while having practical advantages such as lower computational costs and higher stability against random effects. Moreover, we find a small tendency for both systems to overfit to specific musical material, as well as a tendency for the DL-based system to benefit from large dataset sizes. With these general observations, our experimental results may inform the use of SVD algorithms in other musical scenarios beyond the opera context.



The chapter is organized as follows: Section 3.2 covers related work on singing voice detection in opera settings. In Section 3.3, we describe our re-implementations of the two SVD systems used for our experiments. Section 3.4 provides an overview of our cross-version dataset. Section 3.5 contains our experimental results and discusses their implications. Section 3.6 concludes the chapter.

## 3.2 Related Work

In this section, we discuss prior work on SVD for opera recordings. For an overview of approaches that have been proposed for SVD, we refer to Section 2.3.

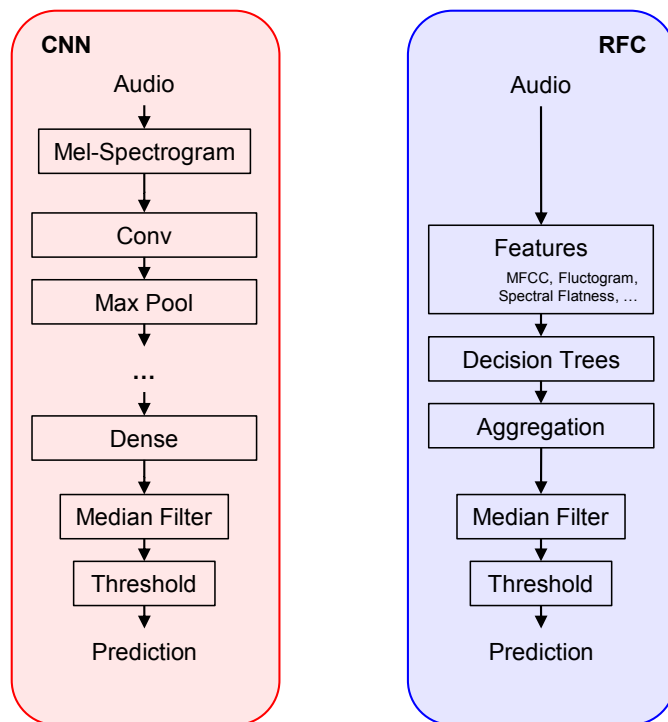
While most SVD approaches were developed and tested on popular music datasets, there is some previous work focussing on SVD for opera recordings. For example, Dittmar et al. [39] performed SVD experiments within an opera scenario comprising recordings of C. M. von Weber’s opera *Der Freischütz*. Using the feature set and classifier setup outlined in Section 2.3, they showed that bootstrap training [150] helps to overcome genre dependencies. In particular, they report frame-wise F-measures up to 0.95, which still constitutes the state of the art for SVD in opera recordings. They further showed that the existence of different versions can be exploited for improving SVD results by performing late fusion of the individual versions’ results. Mimitakis et al. [139] used a cross-version scenario comprising three versions of Richard Wagner’s opera *Die Walküre* (first act) for evaluating three SVD models based on deep learning. As one contribution of this chapter, we perform experiments using both a traditional and a DL-based system and in both cases outperform the results reported in [139]. Furthermore, we substantially extend the scenario of [139] to the full work cycle *Der Ring des Nibelungen* by adding the other acts and operas of the *Ring* cycle, as well as 13 further versions (see Section 3.4). We use this extended dataset to perform a series of systematic experiments, analyzing our two systems in depth.

## 3.3 Singing Voice Detection Methods

In this chapter, we consider two approaches to SVD, one based on traditional machine learning [39] and one based on DL [182], see also Section 2.3. In our re-implementations of these methods, we aimed to be as faithful to the original publications as possible. A conceptual overview of both methods is given in Figure 3.1.

Closely following [39], we first realize a traditional SVD system relying on hand-crafted features and a random forest classifier (RFC). In our re-implementation, we take special care in reproducing the exact feature set, comprising 110 feature dimensions with a feature rate of 5 Hz. Each feature vector incorporates information from 0.8 s of audio (with the exception of the vocal variance feature covering 2.2 s). For the RFC, we use 128 trees per forest as in [39] and use standard settings wherever possible [127, 159] (this leads to minor differences in sampling the training data per decision tree and the feature set per decision

**Figure 3.1:** Conceptual overview of the two methods for singing voice detection considered in this chapter. While the CNN-based approach operates directly on mel-spectrogram excerpts, the RFC requires an additional feature extraction step. Both approaches output continuous predictions which are post-processed using a median filter and thresholding. For technical details, we refer to Section 2.3 and [39, 182].



node compared to [39]). In order to test the validity of our re-implementation, we performed an experiment where we train and test on the public Jamendo<sup>8</sup> corpus, for which results were also reported in [39]. Jamendo is a dataset of over six hours of popular music recordings (published under creative commons licenses), which has been used for experiments on SVD and other tasks like music tagging [15]. In our experiment, we obtain a frame-wise accuracy of 0.887 and a frame-wise F-measure (with singing as the relevant class) of 0.882. This is close to the accuracy of 0.882 and F-measure of 0.887 as reported in [39] for the same scenario.

For our re-implementation of the DL-based system, we follow the description in [182]. We take special care in reproducing the input representation, the model architecture and the training scheme (since the convolution-activation-batch normalization order is not explicitly stated in [182], we use a potentially different order). To resolve ambiguities in [182], we also consulted a previous publication by the authors [181] and their public source code. As opposed to [182], we do not use any input augmentation in order to ensure comparability with the RFC approach where no such augmentations are used. The results in [182] are reported on an internal dataset. However, for the related method proposed in [181] the authors report an error rate of 9.4% (i. e. an accuracy of 0.906) when training and testing on the Jamendo corpus (no data augmentation). Our re-implementation achieves a comparable accuracy of 0.913 for the same scenario.

<sup>8</sup> <https://zenodo.org/record/2585988#.YDNvfmhKhaQ>

Both systems output continuous values between 0 and 1 (sigmoid probabilities for the CNN and the fraction of agreeing decision trees for the RFC). Inspired by several SVD approaches [39, 181, 182], we post-process the results of both the RFC and the CNN using a median filter. As suggested in [39], we use a filter length of 1.4 s and binarize the output with a fixed decision threshold of 0.5. The CNN system outputs predictions at a rate of 70 Hz. To ensure comparability to the RFC-based approach, we, therefore, downsample the CNN predictions to 5 Hz for comparison.

Since neither [39] nor [182] make use of a separate validation set for optimizing hyperparameters, we follow this convention. For the RFC-based system, we try to avoid overfitting by averaging over many trees, each of which is based on a different subsets of the training data. For the CNN-based system, we compute the training loss on mini-epochs of 1000 batches instead of the entire training set. Because of this, we can use early stopping on the training loss to try to prevent overfitting.

Both the traditional and the DL system involve random effects: For the CNN, this includes parameter initialization and random sampling of batches, whereas for the RFC, the choice of features at each split is randomized. Thus, each run of these algorithms produces slightly different results. In our experiments, we compensate for such random effects by averaging all results over multiple runs of the respective algorithm.

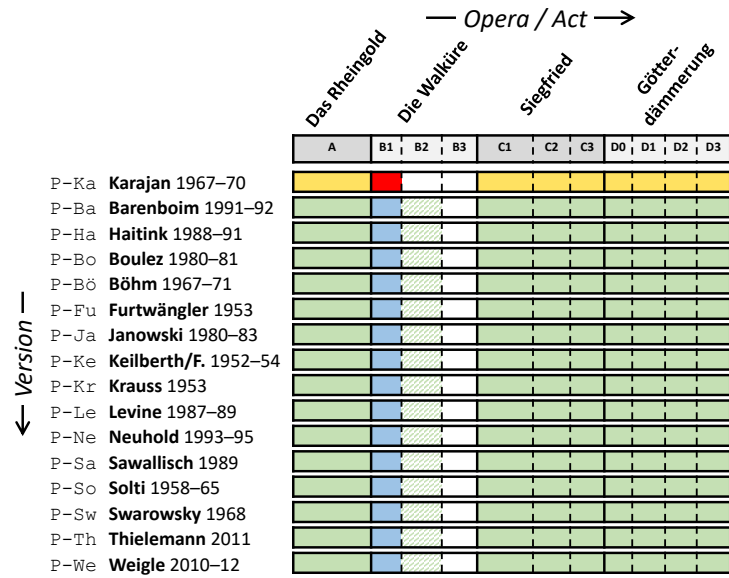
The two approaches differ in the computational resources required for training and testing. The computations for individual trees in the RFC can easily be parallelized and run on a standard CPU. The CNN requires a GPU or TPU for efficient training and testing. For example, when training both systems on the same training set of around 200 h of audio (excluding feature computation), the RFC finishes after requiring eight minutes runtime and 3.5 GB of RAM on a desktop computer, while the CNN requires around two hours of training time and 3 GB VRAM on a medium-sized cluster node. These numbers are highly implementation- and hardware-specific, but they demonstrate that the classical system requires less computation time than the deep learning approach. Inference can be parallelized for both approaches and takes less than a second for the RFC and about one minute for the CNN on roughly 70 min of test audio.

### 3.4 Dataset and Training Scenarios

In this section, we present our cross-version opera dataset, which we use for our systematic experiments in different training–test configurations. This dataset is also used in subsequent chapters of this thesis. Here, we will give an overview of the dataset and our procedure for obtaining singing activity annotations. In Chapter 5, we will explain how we obtained instrument activity annotations for a subset of the dataset and in Chapter 8, we will describe how we annotated leitmotif activity.

Within Western classical music, Richard Wagner’s tetralogy *Der Ring des Nibelungen* WWV 86 constitutes an outstanding work, not least because of its extraordinary length (see Figure 3.2). Spanning the four operas *Das Rheingold* (WWV 86 A), *Die Walküre* (WWV 86 B), *Siegfried* (WWV 86 C), and *Götterdämmerung* (WWV 86 D), the cycle unfolds an interwoven plot involving many different characters. The characters

**Figure 3.2:** Cross-version dataset comprising 16 versions of Wagner’s *Der Ring des Nibelungen* WWV 86. As test data, we use the first act of the second opera *Die Walküre* (WWV 86 B1) in the version conducted by Karajan (red). Training data stems either from the same version but other operas (opera split, yellow), from the same act in other versions (version split, blue), or from other operas in other versions (neither split, green). The hatched green cells (B2) indicate a variant of the neither split, where the training data stems from another act of the same opera .



are represented by different singers with the orchestra adding a rich texture of accompaniment, preludes, and interludes, thus making singing voice detection in recordings of the *Ring* a challenging task. For our experiments, we make use of a cross-version dataset comprising 16 recorded performances—denoted as versions—of the *Ring*, each consisting of 13:30 up to 15:30 h of audio data (see Figure 3.2 and [237] for an overview). All versions are structurally identical, i. e. there are no missing or repeated sections.

To enable comparability between versions, we produced manual annotations of musical measure positions for versions P-Ka, P-Ba, and P-Ha as listed in Figure 3.2 (see [222] for details). We transferred these measure annotations to the remaining versions using an automated alignment procedure [238]. We then used the resulting measure positions to generate audio-based singing voice annotations. To this end, we start from the libretto’s phrase segments and manually annotate the phrase boundaries as given by the score (in musical measures or beats). To transfer the boundaries to the individual versions, we rely on the measure annotations, refined to the beat level using score-to-audio synchronization [51] within each measure. We use these beat positions to transfer the singing voice segments from the musical time of the libretto to the physical time of the performances. The accuracy of the resulting annotations depends, on the one hand, on the accuracy of the measure annotations, which have typical deviations in the order of 100 ms for the manual measure annotations [222] and 200 ms for the transferred measure annotations [238]. On the other hand, score-to-audio synchronization within a measure may introduce further inaccuracies. This is an important consideration for putting any experimental results into context, e.g., for a feature rate of 5 Hz (200 ms) and an average length of a singing voice segment of, say, 4 s, an inaccuracy of one frame already results in a frame-wise error rate of 5%.

For the first act of *Die Walküre* (WWV 86 B1) in version P-Ka conducted by Karajan (DG 1998), we manually refined the phrase boundaries, thus accounting for both alignment errors and imprecision of

singers. We chose this act (B1) since our manual measure annotations are most reliable here, as described in [222]. Moreover, its content is roughly balanced between singing characters, with one female (Sieglinde) and two male singers (Siegmond and Hunding), and with singing activity covering about half its duration (37 of 67 min). In our experiments, we always use this recording and its more accurate annotations for testing (red box in Figure 3.2).

Inspired by [224], our novel dataset allows us to systematically test the generalization capabilities of our SVD systems in different training–test configurations. To this end, we split our dataset along different axes (Figure 3.2). In the *opera split*, we train our methods on other operas in the same version and, thus, need to generalize to different musical works (yellow cells in Figure 3.2). In the *version split*, we use the same act in other versions for training so that the methods need to generalize to a different musical interpretation, different singers, and different acoustic conditions (blue cells). In the *neither split*, neither the test opera nor the test version is seen during training so that the systems have to generalize across both dimensions. In our experiments, we consider different variants of these splits, utilizing, e. g., varying numbers of training versions, operas, or acts. Furthermore, we also exclude in some experiments the second and third act of *Die Walküre* (B2 & B3) since the individual singers (characters) from the first act (B1) re-appear in these acts. When considering all versions or all operas (except *Die Walküre* B1, B2, & B3) for training, we refer to this as a full split. Compared to our scenario, Mimitakis et al. [139] used the same test recording (B1 in version P–Ka), but considered only a version split with the two versions P–Ba and P–Ha (conducted by Barenboim and Haitink, respectively) used for training and validation. We extend this configuration in a systematic fashion in order to study individual aspects of generalization within the opera scenario.

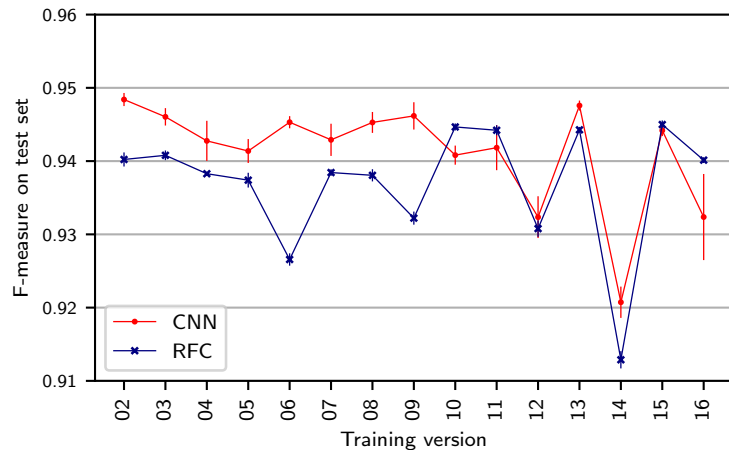
## 3.5 Experiments

In the following, we describe our experiments using the systems described in Section 3.3, taking advantage of the different split possibilities offered by our dataset as described in Section 3.4. We average all results over five runs in order to balance out effects of randomization during training, as discussed in Section 3.3. For comparability, we always use the first act of *Die Walküre* (B1) in the version by Karajan (P–Ka) as our test set as highlighted in Figure 3.2.

### 3.5.1 Training on Different Versions

We begin with a variant of the version split as used in [139], which only considers the first act of *Die Walküre*, WWV 86 B1. Here, the training set consists of version P–Ba (Barenboim) only. On the test set (version P–Ka, Karajan), Mimitakis et al. [139] reported a frame-wise F-measure of 0.80 (we only refer to the results of the zero-mean CNN evaluated in [139], which is most similar to our CNN approach). Using our CNN implementation within the same scenario, we achieve an F-measure of 0.948. The reasons for this substantial improvement remain unclear. With the RFC system, we obtain a comparable result of

**Figure 3.3:** Results for both systems when training on different (individual) versions of the test act (version split).

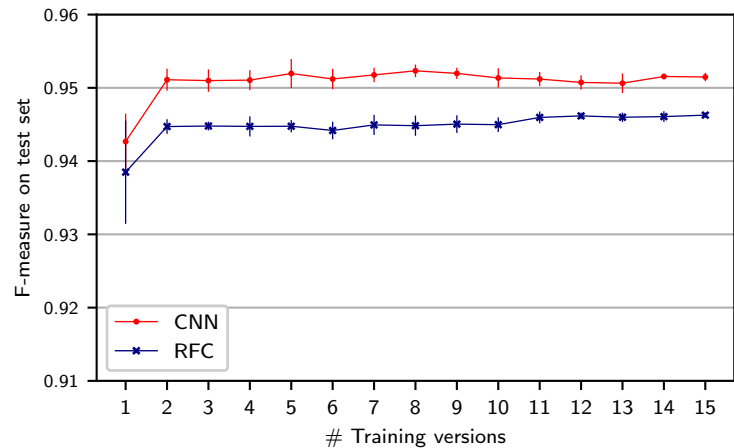


0.941, which is similar to the F-measures reported in [39] for the *Freischütz* opera scenario. From this experiment, we conclude that both a traditional and a DL-based system—when properly implemented and fairly compared—can achieve strong results that are roughly on par with each other.

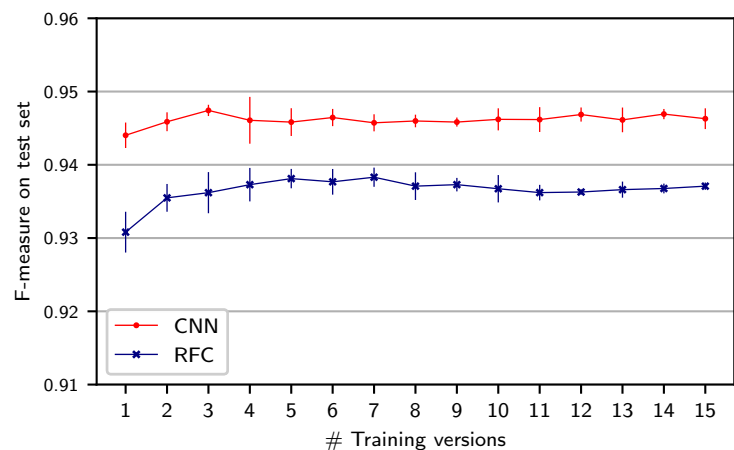
In the previous experiment, we chose version P-Ba (Barenboim) as the training version. To investigate the impact of this choice, we repeat the same experiment while changing the training version. Figure 3.3 shows results for both systems. Dots correspond to mean results averaged over five runs of the same experiment while vertical bars indicate the corresponding standard deviations over those runs. We observe that the choice of training versions has an impact on the test F-measure. The resulting F-measures range from 0.913 for the RFC (version P-Sw) to 0.948 for the CNN (version P-Ba). Furthermore, one can observe that the standard deviations over the runs for individual experiments are higher in the CNN than in the RFC case (see the blue and red vertical bars). In all scenarios, the results are above 0.91 F-measure, which shows that both traditional and deep learning approaches are capable of generalizing from one version to another version of the same work. Nevertheless, the choice of the training version affects test results, up to around 0.03 F-measure. From a practical point of view, such a difference may seem negligible at first, but when considering a full performance of the *Ring* lasting around 15 hours, a difference of 0.03 F-measure can affect roughly 27 min of audio.

The previous results raise the question whether our systems could benefit from increasing the acoustic and interpretation variety in the training set by training on multiple versions. Figure 3.4 shows results when systematically increasing the number of training versions of the same act (B1) used in the version split. In order to suppress the effect of the particular choice of versions, we repeat each experiment five times and, in each run, randomly sample (without replacement) from all possible versions to create a training set with the specified number of versions. For both classifiers, adding one additional training version leads to improved results. However, adding further training versions does not yield clear improvements. Moreover, these small differences have to be seen in light of the annotation accuracy as discussed in Section 3.4. Adding one additional training version seems to sufficiently prevent the systems from adapting to the

**Figure 3.4:** Results for both systems when training on varying numbers of versions of the test act (version split).



**Figure 3.5:** Results for both systems when training on varying numbers of versions of an act (B2) that is different from the test act (neither split).

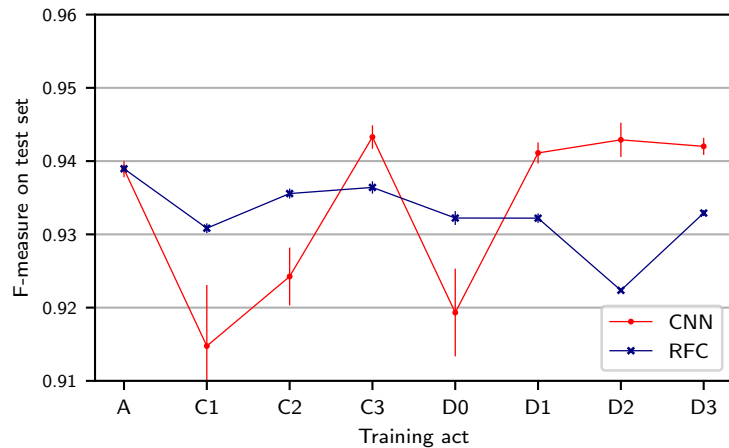


characteristics of individual versions. Additionally, the RFC seems to be more sensitive to the choice of training version: The standard deviation over runs with only one version is larger for the RFC (0.007 percentage points) than for the CNN (0.001), as indicated by the vertical bars around the left-most dots.

### 3.5.2 Training on Different Musical Material

In the experiments reported so far, we made use of the version split where training and test set consist of the same musical material (B1) in different versions. We now want to test generalization to a different musical content and, to this end, use a different act for training (second act of *Die Walküre*, B2) than for testing—a variant of the neither split (green hatched cells in Figure 3.2). As before, we successively increase the number of training versions used. The curves in Figure 3.5 indicate the results, which are worse in general compared to Figure 3.4. The RFC system now benefits slightly more from additional training versions, while the CNN seems unaffected by this. Although the CNN yields better results

**Figure 3.6:** Results for both systems when training on different (individual) acts from the test version (opera split).



than the RFC, neither system reaches its efficacy on the version split. A possible explanation for this may be that both systems might overfit to the musical material in the training act, adapting, e.g., to the instrumentation or the singing characters (high or low register, male or female voice) as appearing in the training data. The small but consistent gap between the curves of the same color in Figures 3.4 and 3.5 could be attributed to such work-related overfitting. We understand such small differences to illustrate a trend in the learning behaviors of our methods (though, as mentioned before, these differences may still be of practical relevance when considering an entire performance).

To investigate the impact of such work-related overfitting in more detail, we now examine specific variants of the opera split where we use the test version (P-Ka, Karajan) for training but take one act from another opera (A, C, or D—excluding B) as the training act, respectively. Compared to the previous experiment, the musical generalization is now harder (a different opera, rather than different acts from the same opera) but the acoustic generalization is less hard (same version). The results of this experiment (see Figure 3.6) are generally worse than for training on a different act of the same opera (cf. Figure 3.5). While the F-measure for the RFC depends only slightly on the particular choice of the training act, this effect is stronger for the CNN. Most prominently, we observe substantial drops when using C1 or C2 (first and second act of *Siegfried*) or D0 (Prologue to *Götterdämmerung*) for training the CNN. This provides insights into specific challenges of generalization: In C1, only male characters (Siegfried, Mime, Wanderer) are singing. In C2, this is similar, except for several short appearances of the character “Waldvogel” (soprano). In D0, in contrast, mainly female singers are singing. All these cases result in a more challenging generalization to B1, where both female and male characters appear. We also observe a drop for the RFC when training on act D2, which does not occur for the CNN. One reason for this difference could be the prominence of the men’s choir (Mannen) over large parts of D2, which is mostly absent from the rest of the work cycle. Choir singing could negatively affect, e.g., the flucrogram features used as input to the RFC (which are sensitive to vibrato) but could be accounted for by the automated feature extraction of the CNN. In general, the RFC-based system, which relies on hand-crafted features (capturing vibrato and



**Table 3.1:** F-measures on the test set for both systems, using the full variants of the split, respectively.

	Opera Split	Version Split	Neither Split
CNN	0.948	0.951	0.952
RFC	0.938	0.946	0.940

other singing characteristics), seems to be more robust to different singers and registers in training and test data. The CNN, in contrast, seems to generalize better to other musical content with the same characters and registers (as in Figure 3.5) and to choir singing. Furthermore, we can again observe that the standard deviation over CNN runs is higher than for the RFC (see vertical bars). Nevertheless, all results are well above an F-measure of 0.9, meaning that even without training examples for a certain gender of singers, both methods can robustly detect singing in unseen recordings.

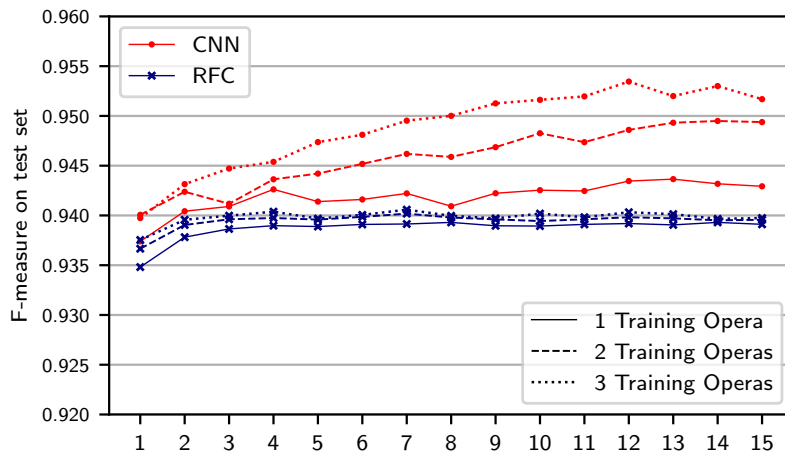
### 3.5.3 Training on Full Splits

We now extend these experiments to the full splits as shown in Figure 3.2 (solid cells). Table 3.1 shows the corresponding results. Let us first discuss the full opera split, where we train the systems on all acts from the three other operas (A, C, D) in the test version P-Ka, Karajan (yellow cells in Figure 3.2). We observe F-measures of 0.948 (CNN) and 0.938 (RFC), respectively. Next, we consider the full version split, where we train on all other versions for the test act B1 (blue cells in Figure 3.2). Here, we obtain results of 0.951 for the CNN and 0.946 for the RFC, which are slightly better than for the opera split. This confirms our observation that work-related overfitting effects (e.g., to singers’ register or gender) help to obtain better results in a version split compared to an opera split. For the neither split, we use all acts of A, C, and D in all versions except P-Ka for training (solid green cells in Figure 3.2). In this case, we observe F-measures of 0.952 (CNN) and 0.940 (RFC). Here, interestingly, the CNN performs similar as in the version split, though neither test act nor test version are seen during training. In contrast, the RFC yields an F-measure close to its result on the opera split. With more versions and operas available, the CNN seems to compensate for the missing test act in the training set. As before, all results are high in general, meaning that both systems work for all considered splits.

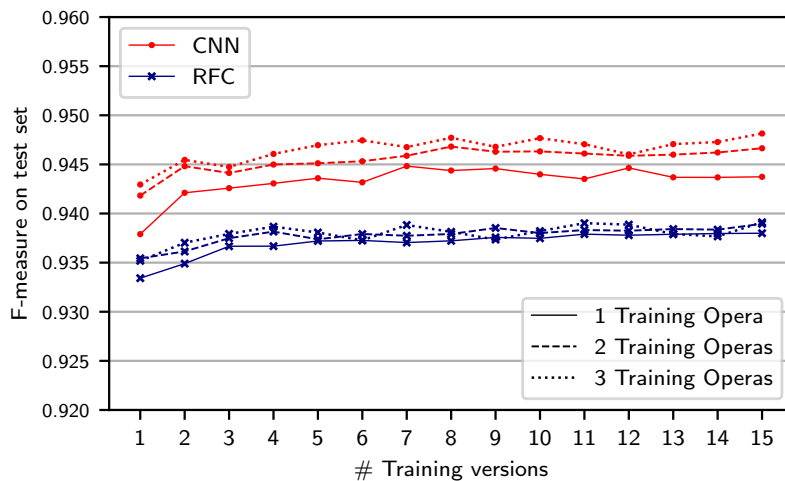
To better understand the important aspects of the training set—especially in the case that less versions and works are available—we now present an extension of the neither split where we successively increase the number of training versions and operas, always using all acts of an opera (Figure 3.7). In each of the five experiment runs, we randomly sample among the versions and operas used for the training set. In this visualization, we omit the vertical bars indicating standard deviations for better visibility. For the RFC (blue curves), we observe that the results are almost identical for different numbers of training operas (solid vs. dashed and dotted curves), but slightly improve for higher numbers of training versions. The CNN, in contrast, benefits from using more operas in the case that more training versions are available as well.

Summarizing these results, we find that the CNN has a slightly stronger tendency than the RFC to overfit to the musical material of the training set. We further see that the RFC-based system primarily exploits

**Figure 3.7:** Results for both systems when training on varying numbers of versions and operas (solid, dashed, and dotted curves) that are different from the test version and act (neither split), using the full data for training.



**Figure 3.8:** Results when training on varying numbers of versions and operas as in Figure 3.7 (neither split), using sub-sampling of the training datasets to the same size.



acoustic variety present in multiple training versions. In comparison, the CNN seems to also exploit variety in musical material and singing characters stemming from different operas. As a consequence, the CNN trained on the full training data of the neither split can generalize better to other operas and, thus, better compensate for the missing test act during training. We further find that the results for the CNN system vary more across runs, especially in the case that only few training versions and acts are used. As mentioned above, however, the results for all experiment settings are consistently high, meaning that both methods are feasible for singing voice detection in opera, even if only little training data variety is available.

### 3.5.4 Impact of Dataset Size

In our previous experiments, we systematically added training versions and operas, which led to an increase not only of the variety of training data, but also of the training dataset’s size. To separately study the two effects, we repeat the experiment from Figure 3.7 while randomly sub-sampling all training datasets to

have equal size (of about as many input patches as obtained from a single act of a single version). Results are shown in Figure 3.8. For the RFC (blue), results are almost identical as in Figure 3.7. For the CNN, we observe smaller improvements in Figure 3.8 than in Figure 3.7 for increasing the number of training versions and operas. Therefore, the improvement seen in Figure 3.7 appears to stem mainly from the training dataset’s size rather than from its variety. This suggests a fundamental difference between the two systems: While the CNN benefits from a larger amount of training data, the RFC is widely unaffected by this. For the RFC, a certain variety in training data seems to be sufficient for reaching an optimal efficacy.

### 3.5.5 Transfer between Pop and Opera Datasets

We finally compare system results when training on datasets from another genre of music. Specifically, we use the Jamendo dataset already discussed in Section 3.3. We expect generalization between the pop and opera styles to be poor, since SVD methods are known to be heavily genre specific [183] and, in particular, opera singers employ singing techniques that are distinct from those found in Western popular music. Consequently, when training on the Jamendo training corpus and evaluating on our own test set, we observe a low F-measure of 0.457 for the RFC (not better than random choice) and a medium result of  $F = 0.795$  for the CNN. When using the training set of the full neither split of our dataset (see Figure 3.2) and testing on the Jamendo test set instead, we obtain  $F = 0.693$  for the RFC and  $F = 0.692$  for the CNN. Thus, generalizing from opera to pop works better for the RFC and worse for the CNN, but genre-specific overfitting is evident in both scenarios.

## 3.6 Conclusions

Summarizing our experimental findings, we conclude that machine learning approaches, both relying on traditional techniques and deep learning, are useful for building well-performing SVD systems, which are capable of generalizing to unseen musical works, versions, or both at the same time. Both systems achieve F-measures in the order of 0.94 and their results do not drop below 0.91, even when considering training datasets with little musical or acoustic variety. While these are strong results for a challenging SVD scenario, we find a tendency of both systems to overfit to the specific musical material in the training set. Moreover, we observe that both systems benefit from a certain amount of acoustic variety in the training dataset. Nevertheless, overfitting to musical or acoustic characteristics does not lead to complete degradation of results in our scenario. For practical applications, the traditional approach based on random forest classifiers requires fewer resources and training time and is more robust to random effects of different training runs. In contrast, the CNN-based method leads to slightly better results in most scenarios, especially when a large training dataset is available.

In a manual analysis, we could trace back most of the remaining errors made by our systems to difficult situations where annotation errors and musical ambiguities play a major role. One source of such ambiguity

is the discrepancy between a phrase-based and a note-based consideration of singing voice segments, i.e., the question whether a short musical rest within a singing phrase should be considered as singing. Further ambiguities arise about whether to include breathing as singing or how to deal with choir passages. In the present study, breathing is mostly excluded from singing since our semi-automatic transfer relies on chroma features. Choir passages are included as singing but only occur in acts D2 and D3. However, as our annotations are based on phrases in the libretto, we could not differentiate, e.g., between silence and singing within sung phrases. Manual annotation could provide more accurate ground truth but is only feasible for smaller datasets. These and other challenges of annotation indicate that both systems are already close to a “glass ceiling” of SVD efficacy, where the definition of the task itself becomes problematic.

Such encouraging results close to the “glass ceiling” cannot generally be expected for other MIR tasks such as, e.g., the recognition of a specific register (soprano, mezzo, alto, tenor, baritone, or bass; see also Chapter 4) or even a specific singer or character (such as Siegfried or Sieglinde). The hand-crafted features used in the RFC-based system have been specifically designed to work well for SVD and may not perform equally well on such more advanced tasks. Thus, DL-based approaches may yet outperform classical systems in those contexts. Therefore, it may be promising to extend our studies to such further tasks and to more complex scenarios (including other composers and genres). More generally, the experimental procedures presented in this chapter can be applied to various domains of audio and music processing where different data splits are possible. In future work, these procedures may contribute to understanding the benefits of deep learning against traditional machine learning and identifying the aspects of training data that are relevant for building robust systems.

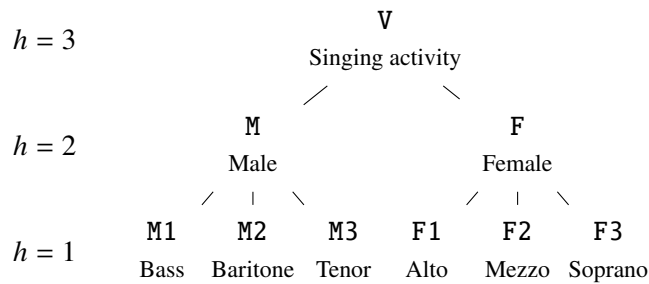
## 4 Hierarchical Approaches for Detecting Singing Activity, Gender, and Type

This chapter is based on [102]. The first author Michael Krause is the main contributor to this article. In collaboration with his supervisor Meinard Müller, he devised the ideas, developed the formalization, and wrote the paper. Furthermore, Michael Krause implemented all approaches and conducted the experiments.

Traditionally, work on singing voice detection (SVD) has focused on identifying singing activity in music recordings, as was also our goal in the previous Chapter 3. In this chapter, our aim is to extend this task towards simultaneously detecting the presence of singing voice as well as determining the singer’s gender and voice type. We describe and compare four strategies for utilizing the hierarchical relationships between these levels. In particular, we introduce a novel loss term that promotes consistency across hierarchy levels. We evaluate the strategies on our *Ring* dataset containing over 200 hours of complex opera recordings with various singers of different genders and voice types, with a particular focus on hierarchical consistency. Our experiments show that by adding our loss term, a joint classification strategy using a single neural network achieves slightly improved evaluation scores and far more consistent results.

### 4.1 Introduction

SVD has been a long standing task in the field of MIR [83]. Prior work has focused on increasing the accuracy for detecting singing activity in music recordings [114, 116, 181], see also the previous Chapter 3. Aside from mere activity, however, singing voice can be classified with regard to a multitude of aspects related to singing styles and techniques. Western opera, for example, is performed by singers with certain voice types (e. g., baritone, tenor, soprano) who may sing individually or simultaneously, creating a complex sound mixture of singing and orchestral music. In this context, we propose to simultaneously detect singing activity, singer gender, and voice type in music recordings. A system for detecting gender and voice type may be useful for annotating recorded and live performances in order to enhance audience



**Figure 4.1:** Class hierarchy as considered in this chapter (number of hierarchy levels  $H = 3$ ).

experience or to aid in navigating music collections. It may also be useful as a pre-processing step for tasks such as singer identification.

In our scenario, the classes under consideration form hierarchical relationships, as illustrated in Figure 4.1. As a main contribution of this chapter, we describe four different strategies for classification of singing voice that incorporate hierarchical information in different ways. In particular, we compare these strategies with regard to consistency of their predictions across hierarchy levels and discuss a joint classification strategy requiring only a single neural network which uses a novel loss to promote consistency. The strategies are comprehensively evaluated using our large dataset of over 200 hours of audio recordings from Richard Wagner’s cycle of operas *Der Ring des Nibelungen*.

We make the following contributions: First, we introduce a novel hierarchical classification problem by extending singing voice detection towards singer gender and voice types. Second, motivated by this problem, we formalize an appropriate hierarchical class model and provide evaluation and hierarchical consistency measures (Section 4.3). Third, we describe four strategies to approach this problem that utilize the hierarchical relationships between classes in different ways (Section 4.4). In particular, we propose a novel loss term that promotes consistent predictions. Finally, we evaluate these strategies in the context of a dataset containing over 200 hours of complex opera music (Section 4.5). We show that the joint strategy using our additional loss achieves strong and consistent results.

## 4.2 Related Work

Historically, SVD has received a lot of attention from the MIR community and numerous approaches have been proposed over the years, see also Section 3.2. Few works have considered finer grained classes, such as the classification of singing gender [228]. For applications beyond music processing, some researchers have considered the automatic classification of singers according to gender or voice type [144, 163], but this is usually constrained to short, isolated excerpts of singing. Here, we want to detect and classify singing activity across entire music recordings.

Several works have investigated singing voice detection for opera [39, 139], without considering finer grained classes. Other works have focused on identifying emotion [155] or melody [201] from opera recordings.

Hierarchical classification has been explored for tasks such as bird call classification [34], singing transcription [245] or general sound event detection [88, 234]. It has also been considered in the wider machine learning literature [11, 220]. Often, however, these works rule out simultaneous class activity or do not evaluate results with regard to hierarchical inconsistencies. We refer to [189] for a comprehensive overview of pre-DL hierarchical classification approaches.

### 4.3 Hierarchical Class Model

We begin by formalizing our class hierarchy and detection task. Let  $\mathbf{C} = \{\mathbf{V}, \mathbf{M}, \mathbf{F}, \mathbf{M1}, \mathbf{M2}, \mathbf{M3}, \mathbf{F1}, \mathbf{F2}, \mathbf{F3}\}$  be the set of all classes in our scenario. These classes are organized in a hierarchy, where  $H$  is the total number of hierarchy levels and  $\mathbf{C}^h$  are the classes at hierarchy level  $h \in [1 : H]$ . The lowest level  $h = 1$  corresponds to the most specific classes, whereas the highest level  $h = H$  corresponds to the most general one. We assume that  $(\mathbf{C}^h)_{h \in [1:H]}$  forms a partition of  $\mathbf{C}$ . In our case,  $H = 3$  and the hierarchy levels correspond to voice type ( $h = 1$ ), singer gender ( $h = 2$ ), and singing activity ( $h = 3$ ), respectively. In particular, we have  $\mathbf{C}^1 = \{\mathbf{M1}, \mathbf{M2}, \mathbf{M3}, \mathbf{F1}, \mathbf{F2}, \mathbf{F3}\}$ ,  $\mathbf{C}^2 = \{\mathbf{M}, \mathbf{F}\}$ , and  $\mathbf{C}^3 = \{\mathbf{V}\}$ .

For a class  $c \in \mathbf{C}$  we write  $c \uparrow$  for the immediate parent of  $c$  in the class hierarchy (e. g.,  $\mathbf{F2} \uparrow = \mathbf{F}$ ). Additionally, we write  $c \downarrow$  for the set of immediate children of  $c$  (e. g.,  $\mathbf{M} \downarrow = \{\mathbf{M1}, \mathbf{M2}, \mathbf{M3}\}$ ).

We formulate our singing detection task as a frame-wise, multi-label classification problem. Formally, let  $\mathcal{I}$  be the set of items under consideration. In our case, the elements of  $\mathcal{I}$  are audio frames. We describe our reference annotations as well as predictions made by some detection model as families  $(\mathcal{I}_c)_{c \in \mathbf{C}}$  of subsets  $\mathcal{I}_c \subseteq \mathcal{I}$ . For  $i \in \mathcal{I}_c$  we say that class  $c$  is active for frame  $i$ . Note that the sets  $\mathcal{I}_c$  for different  $c$  need not be disjoint (i. e., multiple singers with different genders and voice types may be active for the same audio frame), and there may be items  $i \in \mathcal{I}$  that are not contained in any set  $\mathcal{I}_c$  (i. e., there may be audio frames without any singing). In this way, we account for our multi-label scenario.<sup>9</sup>

Generally, we would like the  $\mathcal{I}_c$  to be in some sense consistent with the hierarchical structure of  $\mathbf{C}$ . For example, an item  $i \in \mathcal{I}_c$  should also be an element of  $\mathcal{I}_{c \uparrow}$ . We will refer to this requirement as bottom-up consistency. Likewise, if  $i \in \mathcal{I}_c$ , then there should be some child  $c' \in c \downarrow$  such that  $i \in \mathcal{I}_{c'}$ . We will call that top-down consistency. We now introduce three measures that capture the degree of bottom-up consistency ( $\gamma_c^\uparrow$ ), top-down consistency ( $\gamma_c^\downarrow$ ), or both ( $\gamma_c$ ) for the set  $\mathcal{I}_c$ .

<sup>9</sup> In the terminology adopted by [189], we are dealing with a hierarchically multi-label problem on a tree with full depth labeling.

First, for a subset  $\mathbf{C}' \subseteq \mathbf{C}$ , we introduce the notation

$$\mathcal{I}_{\mathbf{C}'} = \bigcup_{c \in \mathbf{C}'} \mathcal{I}_c. \quad (4.1)$$

Now, for any  $h > 1$  and  $c \in \mathbf{C}^h$ , we define the following consistency measures with values in the range  $[0, 1]$ :

$$\gamma_c = \frac{|\mathcal{I}_c \cap \mathcal{I}_{c\downarrow}|}{|\mathcal{I}_c \cup \mathcal{I}_{c\downarrow}|}, \quad \gamma_c^\downarrow = \frac{|\mathcal{I}_c \cap \mathcal{I}_{c\downarrow}|}{|\mathcal{I}_c|}, \quad \gamma_c^\uparrow = \frac{|\mathcal{I}_c \cap \mathcal{I}_{c\downarrow}|}{|\mathcal{I}_{c\downarrow}|}. \quad (4.2)$$

Intuitively, these measures capture the amount of agreement between  $\mathcal{I}_c$  and  $\mathcal{I}_{c\downarrow}$ . If all  $i \in \mathcal{I}_c$  are also contained in  $\mathcal{I}_{c'}$  for some  $c' \in c\downarrow$ , then  $\gamma_c^\downarrow = 1$ . If for all  $i \in \mathcal{I}_{c\downarrow}$  it holds that  $i \in \mathcal{I}_c$ , then  $\gamma_c^\uparrow = 1$ . Finally,  $\gamma_c = 1$  if and only if  $\mathcal{I}_c = \mathcal{I}_{c\downarrow}$ .  $\gamma_c$  is also called intersection-over-union or Jaccard index.

## 4.4 Hierarchical Singing Detection

In this section, we introduce various strategies for hierarchical singing detection. For now, we assume a given classification model  $\mathcal{M}$  that can be trained on subsets of items  $\mathcal{I}' \subseteq \mathcal{I}$  to yield predictions for subsets of classes  $\mathbf{C}' \subseteq \mathbf{C}$ . After training, we can use such a model to get probabilities  $p_c$  per class  $c \in \mathbf{C}'$  for unseen inputs  $i \in \mathcal{I} \setminus \mathcal{I}'$ . For the classification, we threshold these probabilities at 0.5 to obtain  $\mathcal{I}_c^{\text{Est}}$ , the set of items that have been predicted for a certain class  $c \in \mathbf{C}$ . Details on  $\mathcal{M}$  are given in Section 4.5. Next, we describe our detection strategies.<sup>10</sup>

**Strategy A: Independent Decisions.** In this first strategy, we train one independent model for each hierarchy level  $h \in [1 : H]$ . Thus, predictions are made separately at each hierarchy level and no consistency between predictions is enforced.

**Strategy B: Bottom-Up Aggregation.** Here, we train only one model for hierarchy level  $h = 1$ . We then obtain predictions for higher levels by iteratively aggregating results from lower levels, setting  $\mathcal{I}_c^{\text{Est}} = \mathcal{I}_{c\downarrow}^{\text{Est}}$  first for all  $c \in \mathbf{C}^2$  and then for all  $c \in \mathbf{C}^3$ . By design, this ensures that  $\gamma_c = \gamma_c^\downarrow = \gamma_c^\uparrow = 1$  for all  $c$ . However, classification errors made at lower levels are propagated upwards.

**Strategy C: Top-Down Divide-and-Conquer.** Here, we begin with a model for hierarchy level  $h = H$  and divide items into subsets  $\mathcal{I}_c$  for  $c \in \mathbf{C}^h$  according to the classification results. We then iterate that process, proceeding with separate classification models for the subsets  $\mathcal{I}_c$ , where for  $\mathcal{I}_c$  one considers the classes in  $c\downarrow \subseteq \mathbf{C}^{h-1}$ . In other words, only one model is trained on the entire dataset and operates at the highest hierarchy level. Subsequent models differentiate between more specific classes and are trained and evaluated only on frames for which the parent class is active. By design, this ensures that  $\gamma_c^\uparrow = 1$  for all  $c$ . However, since each model considers a multi-label classification problem, there may be frames for which

<sup>10</sup> In the terminology adopted by [189], Strategy D would be considered a “global” approach, whereas B corresponds to a “flat” approach. Strategies A and C are both “local”, with A consisting of local classifiers per layer (LCL) and Strategy C employing local classifiers per parent node (LCPN).



some class is predicted at a higher hierarchy level, but subsequent models predict none of its child classes as active, leading to  $\gamma_c^\downarrow < 1$ . Furthermore, errors made at higher levels are propagated downwards and many separate models need to be trained.

**Strategy D: Joint Classification.** Finally, we consider a strategy with a single model for all classes  $\mathbf{C}$ . To this end, we utilize a multi-task model that performs singing activity detection, gender recognition, and voice type classification at the same time. This model makes predictions jointly at all hierarchy levels (as opposed to the independent decisions in Strategy A), but may violate consistency properties. To address this, we introduce two losses that encourage consistent predictions in a soft way. Intuitively, in order to promote bottom-up consistency and improve  $\gamma_c^\uparrow$ , a loss should encourage the predictions for a parent class  $c$  to be at least as high as the prediction for any of its child classes  $c'$ . Writing  $p_c$  for the probability output by the model for class  $c$ , this is realized by the loss

$$\mathcal{L}_\uparrow = \frac{1}{|\mathbf{C} \setminus \mathbf{C}^H|} \sum_{h=2}^H \sum_{c \in \mathbf{C}^h} \sum_{c' \in c^\downarrow} \max\{0, p_{c'} - p_c\}^2, \quad (4.3)$$

which contains penalties for every  $p_{c'} > p_c$ . This loss formulation has been proposed in [220]. The normalization factor ensures that the loss is in the range  $[0, 1]$ .

Similarly, to promote top-down consistency and improve  $\gamma_c^\downarrow$ , a loss should penalize predictions for a parent class  $c$  that are above the highest probability predicted for a child class  $c'$ . Thus, we propose a novel loss by defining

$$\mathcal{L}_\downarrow = \frac{1}{|\mathbf{C} \setminus \mathbf{C}^1|} \sum_{h=2}^H \sum_{c \in \mathbf{C}^h} \max\{0, p_c - \max_{c' \in c^\downarrow} p_{c'}\}^2. \quad (4.4)$$

The final loss for the model is obtained as

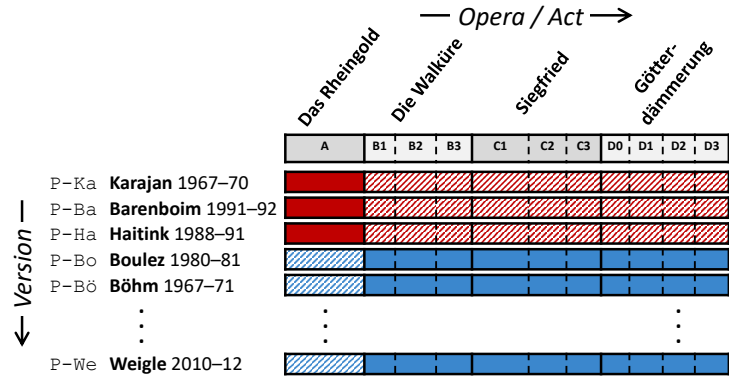
$$\mathcal{L} = \mathcal{L}_{\text{BCE}} + \alpha \mathcal{L}_\downarrow + \beta \mathcal{L}_\uparrow,$$

where  $\alpha, \beta \in \mathbb{R}$  are weights associated with each consistency loss and  $\mathcal{L}_{\text{BCE}}$  is the standard binary cross-entropy loss applied at each output of the network. We will refer to the variants of Strategy D without or including additional consistency losses as strategies  $\text{D}^{0,0}$  and  $\text{D}^{\alpha,\beta}$ , respectively.

## 4.5 Experiments

In this section, we introduce the dataset and evaluation measures used for comparing our hierarchical detection strategies and describe the specific classification model used. Finally, we discuss results.

**Figure 4.2:** Structure of Richard Wagner’s *Ring* cycle and overview of 16 recorded versions, see Section 3.4 for details. P-Ka, P-Ba, and P-Ha (red) are used for testing, while the remaining versions (blue) are used for training. The solid cells indicate one run of cross-validation, where a certain act is removed from training and used for testing.



### 4.5.1 Dataset

To compare the effectiveness of the strategies outlined in Section 4.4, we make use of a dataset containing a multitude of singers with different genders and voice types, as well as complex orchestral accompaniment. Specifically, we consider 16 recorded versions of *Der Ring des Nibelungen* by Richard Wagner, previously used in Chapter 3. In total, our dataset consists of over 200 hours of opera music. An overview of the dataset as it is used in this chapter is given in Figure 4.2. Reference annotations have been obtained with a semi-automatic procedure using score-to-audio synchronization, see Section 3.4 for details.

We reserve three versions for testing and take the rest for training, in line with Chapter 3. The reference annotations for the training and test sets can be represented as families  $(\mathcal{I}_c^{\text{Ref}})_{c \in \mathcal{C}}$ , where  $\mathcal{I}_c^{\text{Ref}} \subseteq \mathcal{I}$  contains items labeled as class  $c$ . These families naturally fulfill all consistency properties, i. e.,  $\gamma_c = \gamma_c^\downarrow = \gamma_c^\uparrow = 1$  for all  $c$ .

Writing  $\delta_c = |\mathcal{I}_c|/|\mathcal{I}| \in [0, 1]$  for the fraction of items where class  $c$  is active, we make several observations on the distribution of classes in  $\mathcal{I}^{\text{Ref}}$  for our test set: around half of all audio frames in our dataset contain singing ( $\delta_V = 0.55$ ) and male voices are more common ( $\delta_M = 0.36$ ) than female voices ( $\delta_F = 0.196$ ). Some voice types occur more often ( $\delta_{M3} = 0.175$ ) than others ( $\delta_{F1} = 0.033$ ). Around 2% of frames contain activity for more than one voice type.

In addition to splitting our dataset across versions, we perform cross-validation over opera acts in the test set. Figure 4.2 illustrates one run of cross-validation (solid cells). As a consequence, our approaches need to generalize both to unseen versions (containing different singers and acoustic conditions) and unseen acts (i. e., different musical compositions).<sup>11</sup>

<sup>11</sup> In Chapter 3, this is referred to as a “neither split”.

### 4.5.2 Evaluation Measures

As described in Section 4.3, we formulate our detection task as frame-wise classification on full recordings. The detection strategies described in Section 4.4 yield predictions for all classes at all hierarchy levels for each frame. The performance of the detection strategies can be evaluated using standard measures from information retrieval such as class-wise<sup>12</sup> precision, recall, and F-measure. Formally:

$$P_c = \frac{|\mathcal{I}_c^{\text{Ref}} \cap \mathcal{I}_c^{\text{Est}}|}{|\mathcal{I}_c^{\text{Est}}|}, R_c = \frac{|\mathcal{I}_c^{\text{Ref}} \cap \mathcal{I}_c^{\text{Est}}|}{|\mathcal{I}_c^{\text{Ref}}|}, F_c = \frac{2 \cdot P_c \cdot R_c}{P_c + R_c}. \quad (4.5)$$

Intuitively,  $P_c$  is the fraction of frames that are correctly predicted as belonging to class  $c$ ,  $R_c$  refers to the fraction of ground truth frames of class  $c$  that are correctly identified, and  $F_c$  is an average of the two.

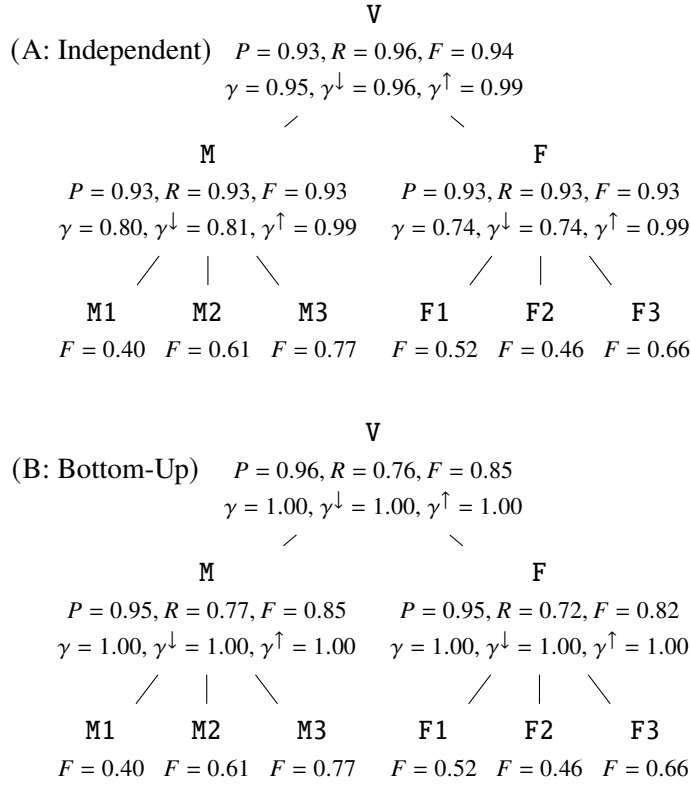
### 4.5.3 Model

The detection strategies described in Section 4.4 depend on a classification model  $\mathcal{M}$  that can be trained to classify audio frames. In our experiments, we use a state-of-the-art model for singing activity detection, as also used in Chapter 3. This system is a VGGNet-inspired CNN applied to log-mel spectrograms patches (of length 1.64 s) with a single sigmoid output. The network is trained to predict singing activity for the center frame of the input patch. For details on the architecture and the specific reimplementation used, we refer to Section 3.3. We only slightly modify this system by increasing the number of sigmoid outputs at the final layer depending on the number of classes we wish to predict (for example, the network used in Strategy B has  $|\mathbf{C}^1| = 6$  outputs, while the network for Strategy  $\text{D}^{\alpha,\beta}$  has  $|\mathbf{C}| = 9$ ). For Strategy  $\text{D}^{\alpha,\beta}$ , we use the additional losses described in Section 4.4. In our experiments we set  $\alpha = \beta = 0.1$ . We determined these values empirically such that all terms in  $\mathcal{L}$  have a similar magnitude. As some classes in our dataset occur less frequently than others, we resample the training set (with replacement) such that each class occurs the same number of times. For post-processing, we follow Chapter 3 by applying a median filter of length 1.4 seconds and then downsampling the predictions to a frame rate of 5 Hz.

### 4.5.4 Results

Figures 4.3 and 4.4 show results for the four detection strategies on our test set. The results for Strategy A demonstrate that, using models trained independently per hierarchy level, one can achieve high evaluation scores for the upper two levels ( $F_V = 0.94$ ,  $F_M = F_F = 0.93$ ) and lower results for the finest level (e. g.,  $F_{M1} = 0.40$  or  $F_{M3} = 0.77$ ). Bottom-up consistency is high ( $\gamma_c^\uparrow = 0.99$  for all  $c$ ), even though the strategy does not enforce this. Therefore, models at higher levels can identify all frames that are predicted as

<sup>12</sup> Many works on hierarchical classification use evaluation measures that aggregate across classes (see [100] for an overview). In contrast, we use class-wise measures to analyze the behavior of our systems with regard to the specific musical challenges associated with different genders and voice-types.



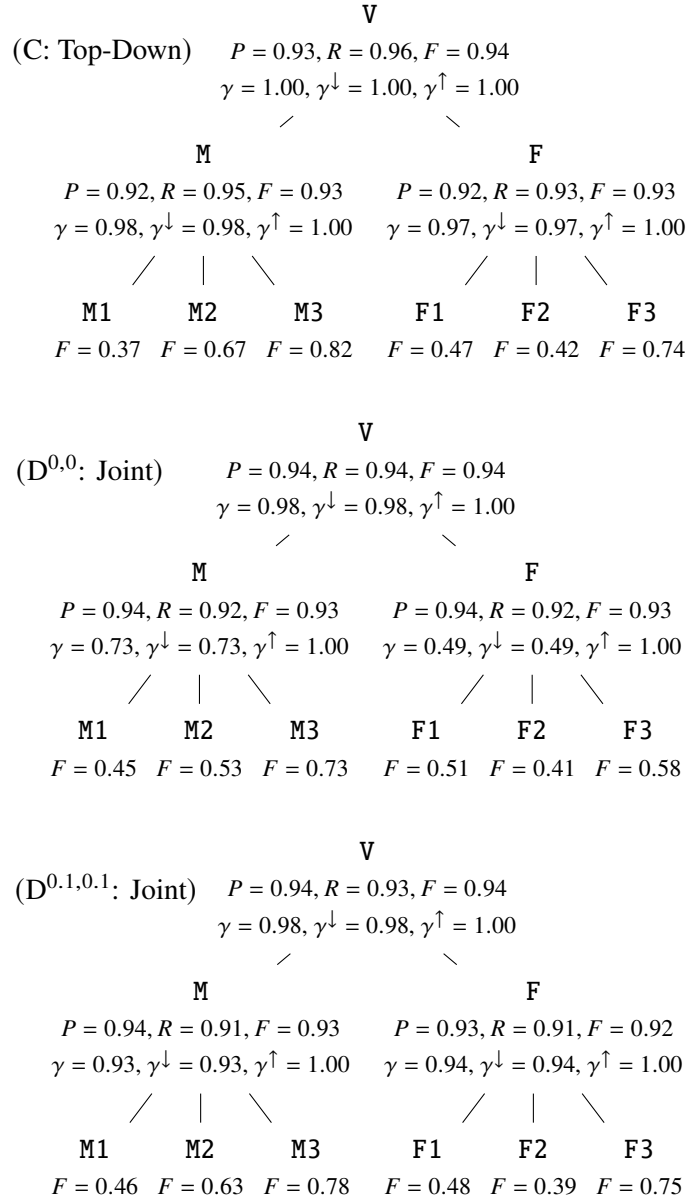
**Figure 4.3:** Results for our detection strategies on the full test set. Subscripts (such as in  $P_{M2}$ ) are omitted for readability. See also Figure 4.4.

active at lower levels. The opposite does not hold, as evident in the low top-down consistency values. For example,  $\gamma_F^\downarrow = 0.74$  implies that some frames for which female singing is being predicted at level  $h = 2$  were misclassified as nonactive or as a male voice type at level  $h = 1$ .

Strategy B involves the same model for level  $h = 1$  as Strategy A, but results for higher levels are obtained through bottom-up aggregation. As such, no inconsistencies arise for Strategy B, but evaluation results are much worse on higher levels (e. g.,  $F_V = 0.85$  as opposed to  $F_V = 0.94$  for Strategy A) owing to frames incorrectly classified as non-singing (see e. g.  $R_V = 0.76$ ).

For Strategy C, we observe the same F-measures as for Strategy A at levels  $h = 3$  (by design) and  $h = 2$ . In addition, predictions are mostly top-down consistent, even though the strategy does not enforce this. On the finest level, there are large improvements for some classes (e. g.,  $F_{F3} = 0.74$  as opposed to  $F_{F3} = 0.66$  for strategies A and B) but also degradations for some results (e. g.,  $F_{F1} = 0.47$  as opposed to  $F_{F1} = 0.52$  for strategies A and B).

Employing the joint classification strategy without additional loss terms  $D^{0,0}$ , we obtain less accurate predictions for most of the classes on the lowest level (e. g.,  $F_{F3} = 0.58$  as opposed to  $F_{F3} = 0.66$  for strategies A and B) and also a large amount of top-down inconsistencies (e. g.,  $\gamma_F^\downarrow = 0.49$ ). We are able to improve this using the additional loss terms of Strategy  $D^{0.1,0.1}$ . In particular, our proposed loss term  $\mathcal{L}_\downarrow$



**Figure 4.4:** Results for our detection strategies on the full test set. See also Figure 4.3.

leads to high top-down consistency scores (e. g.,  $\gamma_F^\downarrow = 0.94$  compared to  $\gamma_F^\downarrow = 0.49$  for D<sup>0,0</sup> and  $\gamma_F^\downarrow = 0.74$  for Strategy A). Strategy D<sup>0.1,0.1</sup> also improves F-measures for some classes on the finest level (notably  $F_{F3} = 0.75$ ). As another advantage, unlike strategies A and C, this joint strategy requires only a single model.

From a musical point of view, our results indicate that singing activity and singer gender can be identified reliably (as evident by the high evaluation results for these classes across all strategies, except Strategy B). This is particularly important for Strategy C, where mistakes made at higher level are propagated downwards. Thus, Strategy C may perform worse in settings where classes at higher levels are more

difficult to separate. Our evaluation results also indicate that, in contrast to higher hierarchy levels, differentiating between different voice types is much more challenging. In addition, by looking more closely at the results for Strategy  $D^{0.1,0.1}$ , we found that most false negative predictions for a certain voice type co-occur with a false positive prediction for another voice type from the same gender. Confusions often occur between baritone and bass as well as between soprano and other female voice types. For example, around half of all frames annotated as mezzo are incorrectly predicted to contain soprano instead of mezzo singing.

## 4.6 Conclusion

In this chapter, we have formalized a hierarchical extension of singing voice detection towards singer gender and voice type. We evaluated four possible strategies for solving our task in the context of a large dataset of opera recordings. We compared these strategies with regard to evaluation scores and consistency of predictions. In particular, we proposed a novel loss term for promoting consistent predictions across hierarchy levels. We showed that a joint classification strategy with our additional loss achieves high results and consistency. The singing scenario considered in this chapter indicates the potential of our hierarchical modeling and loss term. These may also be helpful in other use cases. For example, in the following Chapter 5, we focus on the more complex class hierarchies encountered in musical instrument recognition.

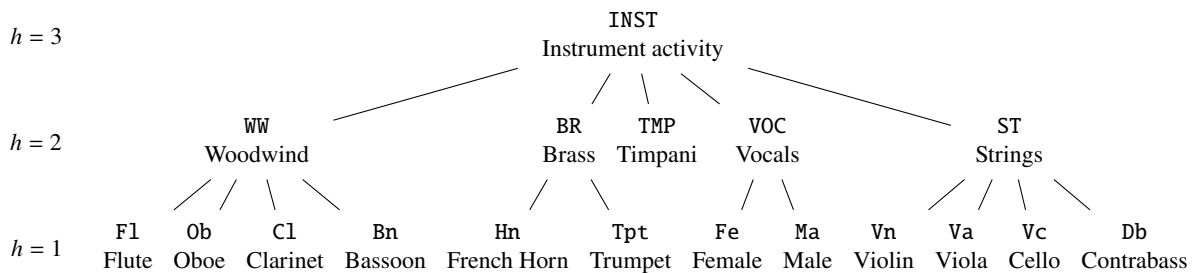
# 5 Hierarchical Approaches for Instrument Activity Detection

This chapter is based on [103]. The first author Michael Krause is the main contributor to this article. In collaboration with his supervisor Meinard Müller, he devised the ideas, developed the formalization, and wrote the paper. Furthermore, Michael Krause collected the dataset, implemented all approaches and conducted the experiments.

Instrument activity detection is a fundamental task in MIR, serving as a basis for many applications, such as music recommendation, music tagging, or remixing. Most published works on this task cover popular music and music for smaller ensembles. In this chapter, we embrace orchestral and opera music recordings as a rarely considered scenario for automated instrument activity detection. Orchestral music is particularly challenging since it consists of intricate polyphonic and polytimbral sound mixtures where multiple instruments are playing simultaneously. Orchestral instruments can naturally be arranged in hierarchical taxonomies according to instrument families. Here, we extend our investigation from the previous Chapter 4 towards this much larger hierarchy of musical instruments. As the main contribution of this chapter, we show that a hierarchical classification approach can be used to detect instrument activity in our scenario, even if only few fine-grained, instrument-level annotations are available. We further consider additional loss terms for improving the hierarchical consistency of predictions. For our experiments, we collect a dataset containing 14 hours of orchestral music recordings with aligned instrument activity annotations. Finally, we perform an analysis of the behavior of our proposed approach with regard to potential confounding errors.

## 5.1 Introduction

Instrument recognition is a long-studied task in the field of MIR, which aims at identifying the musical instruments that are playing in an audio excerpt. It is a difficult task, since many instruments produce sounds in overlapping pitch ranges and may exhibit similar timbral characteristics, especially those from the same instrument family. Furthermore, in real music recordings, multiple instruments may be active simultaneously (also called polyphonic instrument recognition). The task is closely related to instrument



**Figure 5.1:** Class hierarchy as used in this chapter. The number of hierarchy levels is  $H = 3$ . Level  $h = 2$  corresponds to instrument families, while level  $h = 1$  corresponds to fine-grained instrument classes.

activity detection (IAD), where the aim is to identify the active instruments in a frame-wise fashion, over the course of an entire music recording. IAD can be useful to inform music recommendation or auto tagging systems, as well as aid in music editing and remixing.

In this chapter, we are concerned with IAD in complex orchestral and opera recordings. Orchestral music in general constitutes a very challenging scenario for instrument detection, as many individual instruments are playing simultaneously, creating a highly complex sound mixture. In addition to the high degree of polyphony, orchestral music is also polytimbral, i. e., sounds from various instrument groups merge to create a single texture of sound. To approach this scenario, we utilize the hierarchical relationships that exist between orchestral instruments and instrument families, as illustrated in Figure 5.1. In this context, we show that hierarchical classification improves results of a deep neural network for IAD, especially when fine-grained, instrument-level annotations are unavailable, but coarse, family-level annotations exist. Moreover, we investigate the consistency of predictions across hierarchy levels and demonstrate how additional training losses (as introduced in Chapter 4) can promote consistent predictions, while preserving detection quality.

In contrast to popular music settings, public datasets with instrument activity annotations for orchestral or opera music rarely exist. For our experiments, we thus collect a dataset based on a combination of existing multi-track datasets and semi-automatically annotated commercial recordings. In total, our dataset consists of 14 hours of real-life orchestral recordings and covers 18 different classes. We make the instruments annotations for these recordings publicly available.<sup>13</sup>

A common pitfall of music classification systems is over-reliance on confounding factors in training and test data, which may lead to poor generalization ability. A system for genre classification may, for example, make decisions based on inaudible artifacts rather than musical content [196]. Such confounding effects may also arise for our IAD system by, e. g., affecting predictions for classes that are often active simultaneously (such as brass and woodwinds). To explore the impact of such effects on our system, we perform an analysis of model predictions with regard to classes that composers often use in conjunction.

<sup>13</sup> <https://www.audiolabs-erlangen.de/resources/MIR/2023-TASLP-HierarchicalInstrumentClass>



We now summarize the main contributions of this chapter. First, we introduce a challenging new setting for IAD, for which no standard datasets exist. Second, we show how one can improve detection results and reduce the need for instrument-level annotations by exploiting the hierarchical class structure of our scenario. Third, we show how the consistency of predictions made by our model can be improved through additional loss terms. Fourth, we perform an analysis of our model’s behavior and uncover confounding effects for certain instrument classes.

In the previous chapter Chapter 4, we explored hierarchical classification for detecting singing activity, singer gender and voice type in orchestral recordings. Here, we go beyond Chapter 4 by considering an instrument detection scenario, involving a different and larger class hierarchy compared to Chapter 4. Furthermore, we present additional technical details, use different datasets and models, and provide extensive additional experiments and analyses.

The remainder of the chapter is organized as follows: Section 5.2 discusses related work on instrument recognition, hierarchical classification, and analysis of orchestra recordings. In Section 5.3, we formalize our problem statement, outline our main classification approach, and describe evaluation measures used. Sections 5.4 and 5.5 cover our dataset and model architecture, respectively. Section 5.6 contains the main experimental results. In Section 5.7, we describe and evaluate losses for improving the consistency of predictions. In Section 5.8, we analyze our model with regard to confounding effects among instrument classes. Finally, Section 5.9 concludes the chapter with an outlook on possible future work.

## 5.2 Related Work

This chapter draws upon related work from several fields, including the vast field of music instrument classification. In our review of these fields, we focus on key references relevant to the present chapter and relate our contributions to the state of the art.

### 5.2.1 Instrument Detection

Early work on automatic musical instrument classification dealt with recordings of isolated note events and used classical machine learning techniques [49, 63, 206]. Other works considered real music recordings, but restricted themselves to a single instrument playing [50, 91, 157], a scenario called monophonic instrument recognition. In contrast, polyphonic instrument recognition attempts to recognize instruments within mixtures where several instruments are playing simultaneously. This has been approached with classical machine learning techniques [45, 57, 70, 78, 95, 125, 157] and, more recently, with deep learning [66, 67, 71, 84, 85, 101].

Works on instrument recognition can also be categorized according to whether only the predominant instrument in a mixture (e. g., [57, 71]) or all active instruments (e. g., [84]) are to be recognized.

Furthermore, some works classify activity for an entire audio excerpt lasting several seconds (e. g., [57, 66, 67, 71, 78, 82, 101]) whereas more fine-grained approaches yield predictions on a frame-level (e. g., [84, 85]). Such frame-level outputs can be used to obtain instrument predictions for every time step in a music recording—also called instrument activity detection (IAD). The scenario considered in this chapter is polyphonic IAD and considers all instruments playing. In contrast to prior work on this scenario, which usually examines popular music or works for small ensembles, we consider complex orchestral and opera music.

### 5.2.2 Hierarchical Classification for Audio

Some previous works have used hierarchical class structures for classification of audio data. For example, the authors in [34] propose a specialized network architecture for classifying bird calls, based on bird taxonomies. They perform classification on an excerpt- and not on a frame-level. In [234], a network for sound event detection is iteratively pretrained on successive hierarchy levels. Some papers [88, 243] employ tree hierarchies for audio representation learning (but not for classification). These works also usually do not take into account audio inputs where several classes may be active at the same time.

Fewer works use hierarchical structures for music audio classification. In [245], hierarchical classification is used in the context of singing transcription. Essid et al. [50] use hierarchies for instrument classification, but their scenario involves only synthetic audio data and their system does not yield predictions on a frame-level (which are necessary for IAD). In a recent contribution, Zhong et al. [244] utilized hierarchical techniques for polyphonic instrument classification on popular music. They propose two attention-based approaches for learning bottom-up aggregation rules in a data-driven way and compare them to our Strategy  $D^{\alpha,\beta}$  proposed in Chapter 4. Their approaches and our strategy are shown to perform on par.

Hierarchical structures have also been exploited for other tasks in MIR, including musical instrument separation [131]. Garcia et al. [58] use instrument hierarchies for few-shot detection, where the model requires examples of the target class at test time (see also [218]). In contrast, we use a classification approach with a fixed set of classes, since the instruments in orchestral recordings are known a-priori. Nolasco and Stowell [149] employ instrument hierarchies for audio representation learning (not for classification).

As mentioned in Section 4.2, incorporating hierarchical structures in machine classifiers has also been a topic in the wider machine learning literature, see, e. g., [11, 28, 61, 189, 220]. These works typically evaluate their proposed methods on small and artificial datasets. In contrast, we perform hierarchical classification on a dataset of real orchestra and opera recordings.

### 5.2.3 Orchestra and Opera in MIR

Only few works in MIR have focused on opera and orchestral music (see also Sections 3.2 and 4.2). Among these are works on singing detection [39, 139], emotion identification [155], predominant melody estimation [201], as well as source separation informed by multichannel recordings [140] or by score information [72]. Taenzer et al. [200] performed instrument family classification on classical music recordings, but they only considered monotimbral pieces (i. e., works for ensembles consisting of one family only). To the best of our knowledge, there rarely is work in MIR on IAD in real-world, polyphonic and polytimbral recordings of orchestra and opera.

## 5.3 Hierarchical Instrument Detection

In this chapter, we aim to utilize the hierarchical relationships among orchestral instruments. To this end, we now formalize the hierarchical class model, classification approach, and evaluation measures used throughout this chapter. In this section, we follow Chapter 4, where we introduced the concepts and notation.

### 5.3.1 Hierarchical Class Model

As introduced in Section 4.3, we write  $\mathbf{C}$  for the set of all classes in our detection problem and partition these classes into hierarchy levels. In our setting, we consider 18 different classes, corresponding to the nodes of the tree illustrated in Figure 5.1. We use  $H = 3$  hierarchy levels, with the lowest level  $h = 1$  corresponding to fine-grained instrument classes, level  $h = 2$  containing coarse instrument families, and the highest level  $h = 3$  signifying any kind of instrument activity (as opposed to silence or noise). Thus,  $\mathbf{C}^1 = \{\text{Fl}, \text{Ob}, \text{Cl}, \dots\}$ ,  $\mathbf{C}^2 = \{\text{WW}, \text{BR}, \dots\}$ , and  $\mathbf{C}^3 = \{\text{INST}\}$ . It should be noted that the hierarchy could also be constructed in alternative ways. Our choice here is motivated by practical considerations, i. e., simplicity and the availability of sufficient data for each class  $c \in \mathbf{C}$ . Generally, constructing appropriate instrument hierarchies can be a challenging problem, especially for electronic or non-standard instruments [96, 129].

### 5.3.2 Classification Approach

To approach IAD using a deep network, we pose the problem as a frame-wise, multi-label classification task. Thus, several instrument classes may be active simultaneously and we want to produce predictions for every frame in a music recording. As in Section 4.3, we model both reference annotations and predictions as families of subsets of frames. As a reminder: We write  $\mathcal{I}$  for the set of all audio frames in our test recordings. Now, our instrument annotations are given as families  $(\mathcal{I}_c^{\text{Ref}})_{c \in \mathbf{C}}$  of subsets  $\mathcal{I}_c^{\text{Ref}} \subseteq \mathcal{I}$ , with

**Figure 5.2:** An illustration of (a) bottom-up and (b) top-down inconsistencies. Filled and empty circles correspond to classes predicted as active or inactive, respectively. Red circles indicate inconsistencies.



$\mathcal{I}_c^{\text{Ref}}$  containing all frames where class  $c \in \mathbf{C}$  is active. Note that the sets  $\mathcal{I}_c^{\text{Ref}}$  are generally not disjoint, e. g., in cases where instruments are playing at the same time. Frames with silence or noise are not contained in any  $\mathcal{I}_c^{\text{Ref}}$ .

In a similar fashion, we model the estimates made by our IAD system as families of sets  $(\mathcal{I}_c^{\text{Est}})_{c \in \mathbf{C}}$ . These outputs are obtained from a deep network that takes an audio excerpt as input and yields predictions for the center frame of that excerpt. These estimates are values in  $[0, 1]$  for every class  $c \in \mathbf{C}$ , which are subsequently thresholded to obtain the sets  $\mathcal{I}_c^{\text{Est}}$ . Since this approach jointly considers all hierarchy levels  $1 \leq h \leq H$ , we will refer to it as **HC** (hierarchical classification).<sup>14</sup> More details on the network architecture used are provided in Section 5.5.

### 5.3.3 Evaluation Measures

With our formulation above, we can define frame-wise precision, recall, and F-measure for each class  $c \in \mathbf{C}$ , see Equation (4.5). Note that the three measures may be overly optimistic if  $c$  is a very common class that is active in most frames (as is the case, e. g., for  $c = \text{INST}$ ). In this case, it is important to additionally consider the specificity for class  $c$ , i. e., the recall of non-active frames, defined as

$$S_c = \frac{|(\mathcal{I} \setminus \mathcal{I}_c^{\text{Ref}}) \cap (\mathcal{I} \setminus \mathcal{I}_c^{\text{Est}})|}{|\mathcal{I} \setminus \mathcal{I}_c^{\text{Ref}}|}, \quad (5.1)$$

where  $\setminus$  denotes the set difference operator.

A classification approach that is aware of class hierarchies should not produce predictions that are hierarchically inconsistent, as explained in Section 4.3. For example, a frame  $i$  may be classified as trumpet (Tpt), so  $i \in \mathcal{I}_{\text{Tpt}}^{\text{Est}}$ , but at the same time may not be classified as brass (BR), so  $i \notin \mathcal{I}_{\text{BR}}^{\text{Est}}$ . We will refer to this as a bottom-up inconsistency. Such an inconsistency makes the output of a detection system difficult to interpret, since it is unclear whether the frame was erroneously classified as trumpet or whether it does indeed contain brass, but the system failed to identify BR. Similarly, we may have a frame  $i \in \mathcal{I}_{\text{BR}}^{\text{Est}}$  that is classified as brass but at the same time is neither classified as horn nor trumpet, thus  $i \notin \mathcal{I}_{\text{Hn}}^{\text{Est}} \cup \mathcal{I}_{\text{Tpt}}^{\text{Est}}$ . We call this a top-down inconsistency. Figure 5.2 gives an illustration of these inconsistencies. Ideally, a detection system produces no inconsistencies, making its output straightforward to interpret. To capture different kinds of inconsistencies, we use the  $\gamma$  metrics introduced in Section 4.3.

<sup>14</sup> This approach is called Strategy  $D^{0,0}$  in Chapter 4.

Note that the consistency measures can trivially be maximized by setting, e. g.,  $\mathcal{I}_c^{\text{Est}} = \mathcal{I}$  or  $\mathcal{I}_c^{\text{Est}} = \emptyset$  for all  $c \in \mathbf{C}$ . However, obtaining good detection results (e. g., in terms of F-measure) while simultaneously preserving consistency is non-trivial.

## 5.4 Orchestral Datasets

As mentioned in Section 5.2, orchestral and opera music are seldom explored in MIR. In particular, no standard datasets for IAD on real orchestral recordings exist. The instrument classes commonly considered in IAD datasets for popular music (e. g., [12], which contains guitar, drums, base etc.) do not usually appear in orchestral music. We thus assembled our own datasets for training and evaluating our system, based on existing datasets and our own annotation efforts.

For effectively training our deep learning system, we require a dataset of several hours length. For such a size, manually annotating instrument activity for all 18 classes in our hierarchy would be prohibitively expensive. We thus consider two ways of obtaining orchestral recordings with aligned instrument annotations:

1. Use multi-track recordings of orchestral pieces, where activity annotations can easily be obtained from the individual tracks.
2. Use music synchronization techniques to align a score representation to an audio recording of a piece. Instrument activity in the recording can then be transferred from the aligned score.

A third possible option would be to use artificial recordings of pieces based on synthesized score representations. Recently, Sarkar et al. [179] released a dataset of synthesized, multi-track recordings of classical pieces. However, their dataset contains, for the most part, chamber music rather than full orchestra pieces. Their work also demonstrates that synthesizing convincing renditions of classical music is a challenging task in itself. Here, rather than creating artificial recordings of orchestral scores, we instead collect multiple real recordings per score for synchronization (option 2).

### 5.4.1 Multi-Track Datasets

Due to the challenges of recording orchestra pieces in a multi-track fashion<sup>15</sup>, only a few such datasets have been released. One of these is Phenix Anechoic [140], which contains clean multi-track recordings of four orchestral excerpts by different composers with note on- and offsets manually annotated for each track. We derive instrument activity from these annotations. Böhm et al. [24] provide multi-track recordings for three movements of Beethoven’s Symphony No. 8, resulting in the longest multi-track dataset we use. The

---

<sup>15</sup> Orchestra musicians usually perform within the same room and in close proximity, making it very difficult to obtain clean tracks without cross-talk coming from other instruments playing simultaneously.

individual tracks are mostly free of cross-talk. We therefore use a simple energy thresholding procedure on the tracks to obtain instrument activity annotations.

Since both datasets are annotated based on clean multi-track recordings, we expect the derived activity labels for Phenix and Beethoven Anechoic to be highly reliable. Nevertheless, there remains some ambiguity in defining note on- and offsets, especially for strings and woodwind instruments [122].

Prätzlich et al. [166] provide multi-track audio for three numbers from Carl-Maria von Weber’s opera “Der Freischütz”. However, due to their recording setup, the individual tracks incur a large amount of cross-talk coming from other sources. We obtained annotations for this dataset using music synchronization, see below.

### 5.4.2 Music Synchronization

Audio-to-score alignment techniques are used to temporally align a recorded music performance with the corresponding musical score. Once an alignment is obtained, information about instruments or pitches played can be transferred from the aligned score to the recorded performance. Here, we use audio-to-score alignment to transfer instrument activity from a symbolic score to several recorded performances of a piece. A popular dataset annotated in this fashion is MusicNet [203], which contains chamber rather than orchestra pieces. Note that automatic audio-to-score alignments may introduce annotation errors. We thus expect the resulting activity labels to be less reliable compared to those obtained from multi-track data.

Even though a large amount of MIDI files for classical music pieces can be found online, only few correspond to full orchestral scores with separate MIDI tracks per instrument. The lack of available score-data represents a big bottleneck for this approach to obtaining instrument annotations. For this work, we manually encoded a score representation of the first act of Richard Wagner’s opera “Die Walküre” (requiring several months of work for musically trained annotators). For some other musical works—namely, several movements of Beethoven’s Symphony No. 3, Dvorak’s Symphony No. 9, and Tschaikowsky’s Violin Concerto—we obtained clean orchestral scores from the Mutopia project.<sup>16</sup> We choose these works because they each contain many instruments from the hierarchy we employ. Furthermore, they belong to the classical and romantic periods and are thus stylistically similar to the remaining pieces we use. We then obtained six orchestral audio versions (i. e., performances, recordings) of these pieces from commercial CD releases and created instrument activity annotations using a state-of-the-art score-to-audio synchronization pipeline [147, 167]. Care had to be taken to verify that there are no structural differences between score and recorded versions, which would corrupt the alignment results. Thus, the annotation process can be considered as a semi-automatic approach, where one must expect alignment errors in the order of 0.2 s [222]. Such errors need to be kept in mind when interpreting evaluation results.

---

<sup>16</sup> <https://www.mutopiaproject.org/>

Subset/Composer	Work	# Versions	Dur. (min)
<b>Ours</b>			
Wagner	Die Walküre, Act 1	6	389
Beethoven	Symphony No. 3, Mvmt. 1	1	17
	Mvmt. 2	5	72
	Mvmt. 3	5	29
	Mvmt. 4	5	57
Dvorak	Symphony No. 9, Mvmt. 1	5	44
	Mvmt. 2	5	56
	Mvmt. 4	1	11
Tschaikowsky	Violin Concerto, Mvmt. 1	5	91
	Mvmt. 2	5	32
	Mvmt. 3	1	10
<b>Freischütz Digital [166]</b>			
Weber	Der Freischütz, No. 6	1	5
	No. 8	1	9
	No. 9	1	7
<b>Phenix Anechoic [140]</b>			
Mozart	Aria from Don Giovanni	1	4
Beethoven	Symphony No. 7, Mvmt. 1 (Excerpt)	1	3
Bruckner	Symphony No. 8, Mvmt. 2 (Excerpt)	1	1
Mahler	Symphony No. 1, Mvmt. 4 (Excerpt)	1	2
<b>Beethoven Anechoic [24]</b>			
Beethoven	Symphony No. 8, Mvmt. 1	1	8
	Mvmt. 2	1	4
	Mvmt. 4	1	7
			858

**Table 5.1:** Recordings of orchestral and opera works used in this chapter. For some pieces, multiple versions are available. The last column gives the total duration of all versions for a piece. Freischütz Digital, Phenix Anechoic, and Beethoven Anechoic are existing multi-track datasets.

### 5.4.3 Dataset Overview and Split

An overview of all recordings used in this chapter is given in Table 5.1. The multi-track datasets we use each contain one recording per piece and contribute a total duration of around 50 minutes of orchestral music. For the pieces annotated via music synchronization, we have several versions per piece. In total, our dataset contains roughly 14 hours of orchestral music, making it amenable to deep learning. We make all instrument activity annotations publicly available on our accompanying website.<sup>17</sup>

Note that not all instrument classes are present in every recording and that there is a large imbalance among activity of different classes. We define the fraction  $\delta_c$  of frames where class  $c \in \mathbf{C}$  is active as

$$\delta_c = \frac{|\mathcal{I}_c^{\text{Ref}}|}{|\mathcal{I}^{\text{Ref}}|} \in [0, 1]. \quad (5.2)$$

<sup>17</sup> Link provided in Section 5.1.

**Figure 5.3:** Activity ( $\delta_c$ , upper matrix) and multi-labeledness ( $\lambda_c$ , lower matrix) of instrument classes for subsets of our dataset, relative to the total length of recordings in that subset.

$\delta_c$ :		INST	Ww	Fl	Ob	Cl	Bn	BR	Hn	Tp <sup>c</sup>	TMP	VOC	Fe	Ma	ST	Vn	Va	Vc	Db
Walküre	.97	.38	.12	.22	.26	.28	.39	.33	.05	.05	.51	.17	.34	.72	.44	.47	.61	.35	
Beethoven 3	.96	.63	.38	.48	.44	.48	.49	.49	.16	.17	.00	.00	.00	.87	.82	.70	.71	.60	
Dvorak	.97	.70	.36	.45	.44	.37	.32	.28	.16	.13	.00	.00	.00	.86	.78	.71	.69	.51	
Tschaikowsky	.98	.37	.17	.18	.28	.26	.21	.21	.06	.05	.00	.00	.00	.92	.91	.50	.47	.37	
Freischütz	.94	.48	.20	.05	.34	.31	.26	.26	.00	.00	.70	.62	.21	.80	.74	.70	.50	.54	
Phenicx	.96	.70	.40	.42	.54	.52	.45	.40	.21	.00	.00	.00	.00	.91	.81	.69	.63	.59	
Beethoven 8	.94	.67	.41	.53	.49	.54	.48	.46	.24	.22	.00	.00	.00	.88	.78	.71	.62	.68	
$\lambda_c$ :		INST	Ww	Fl	Ob	Cl	Bn	BR	Hn	Tp <sup>c</sup>	TMP	VOC	Fe	Ma	ST	Vn	Va	Vc	Db
Walküre	.35	.12	.22	.25	.28	.37	.32	.05	.04	.42	.15	.27	.53	.44	.47	.58	.35		
Beethoven 3	.60	.38	.48	.44	.48	.47	.46	.16	.17	.00	.00	.00	.62	.78	.69	.70	.59		
Dvorak	.66	.36	.45	.44	.37	.31	.27	.16	.12	.00	.00	.00	.65	.75	.71	.69	.51		
Tschaikowsky	.34	.17	.18	.28	.26	.21	.21	.06	.05	.00	.00	.00	.35	.63	.50	.47	.37		
Freischütz	.46	.20	.05	.34	.30	.25	.25	.00	.00	.63	.56	.21	.71	.72	.70	.50	.54		
Phenicx	.66	.40	.42	.54	.52	.45	.40	.21	.00	.00	.00	.00	.69	.79	.69	.63	.59		
Beethoven 8	.66	.41	.53	.49	.54	.48	.46	.24	.22	.00	.00	.00	.67	.76	.70	.62	.67		

Additionally, for any  $h < H$  and  $c \in \mathbf{C}^h$ , we denote the fraction of items where both  $c$  and some other class  $c' \in \mathbf{C}^h$  of the same hierarchy level are active by

$$\lambda_c = \frac{1}{|\mathcal{I}^{\text{Ref}}|} \left| \bigcup_{c' \in \mathbf{C}^h \setminus \{c\}} \mathcal{I}_c^{\text{Ref}} \cap \mathcal{I}_{c'}^{\text{Ref}} \right|. \quad (5.3)$$

Note that  $0 \leq \lambda_c \leq \delta_c$ . Intuitively,  $\lambda_c$  captures the amount of “multi-labeledness” for class  $c$ . Figure 5.3 shows the values of  $\delta_c$  and  $\lambda_c$  for each class  $c$  in the different subsets of our dataset. We can observe that strings and woodwind instruments are the most common classes. Only *Die Walküre* and Freischütz Digital contain singing. For most classes and pieces, there is a large amount of joint activity among classes on the same hierarchy level, indicated by  $\lambda_c$  being close to  $\delta_c$ . A notable exception is Vn in the Tschaikowsky recordings ( $\delta_c = 0.92$ ,  $\lambda_c = 0.35$ ), due to the many solo parts of the violin in this concerto.

To train and evaluate our IAD system, we split our dataset into train and test recordings. We put different movements into train and test in order to investigate whether our models are capable of generalizing to new musical content or whether they overfit to specific compositions (see also [224]). We also choose different versions in train and test to control for varying recording characteristics and aspects of interpretation (reducing the impact of the so-called album-effect [53]). Among the multi-track datasets, we select No. 6 of Freischütz Digital, the second movement of Beethoven’s Symphony No. 8, and all recordings in Phenix Anechoic for our test set. From the remaining pieces, we choose the first movement of Beethoven’s Symphony No. 3, the fourth movement of Dvorak’s Symphony No. 9 and the third movement of Tschaikowsky’s Violin Concerto for testing. Since we do not have multiple opera works that could be distributed into train and test sets, we choose an excerpt of the Wagner opera act (measures 697 to 955, corresponding to around twelve minutes of music), omit this excerpt during training, and use it for testing.



**Table 5.2:** Network architecture used for our IAD system.

Layer (Kernel size), (Strides)	Output Shape	Parameters
Input	(201, 252, 5)	
Conv2D (15, 15), (1, 1)	(201, 252, 64)	72 000
Batch normalization	(201, 252, 64)	256
Conv2D (1, 3), (1, 3)	(201, 84, 64)	12 288
Batch normalization	(201, 84, 64)	256
Conv2D (3, 3), (1, 1)	(199, 82, 64)	36 864
Batch normalization	(199, 82, 64)	256
Conv2D (3, 3), (1, 1)	(197, 80, 64)	36 864
Batch normalization	(197, 80, 64)	256
MaxPool2D (3, 3), (3, 3)	(65, 26, 64)	
Conv2D (3, 3), (1, 1)	(63, 24, 128)	73 728
Batch normalization	(63, 24, 128)	512
Conv2D (3, 3), (1, 1)	(61, 22, 128)	147 456
Batch normalization	(61, 22, 128)	512
MaxPool2D (3, 3), (3, 3)	(20, 7, 128)	
Conv2D (3, 3), (1, 1)	(18, 5, 256)	294 912
Batch normalization	(18, 5, 256)	1024
Conv2D (3, 3), (1, 1)	(16, 3, 256)	589 824
Batch normalization	(16, 3, 256)	1024
MaxPool2D (3, 3), (3, 3)	(5, 1, 256)	
Conv2D (5, 1), (1, 1)	(1, 1, 512)	655 360
Batch normalization	(1, 1, 512)	2048
Squeeze	(512)	
Dropout 0.5	(512)	
Dense	(256)	131 328
Batch normalization	(256)	1024
Dropout 0.5	(256)	
Dense	(128)	32 896
Batch normalization	(128)	512
Dropout 0.5	(128)	
Dense	(18)	2322
Output: Sigmoid	(18)	

In all cases, we choose one version for testing and use the remaining five versions for training.<sup>18</sup> In other words, we follow the neither split described in Section 3.4.

## 5.5 Model Architecture

In this section, we give details on the architecture of our model for hierarchical classification. Note that the main technical focus of our work is on hierarchical structures and consistency losses rather than a particular architecture and alternative architectures (e. g., based on ResNets [75]) could also be used here. We employ a CNN inspired by standard VGG-like architectures [190]. The architecture is illustrated in

<sup>18</sup> For the act from *Die Walküre*, we choose P-Ne, P-Le, P-Bö, P-Ke, and P-Bo for training as well as P-Ka for testing, see also Figure 3.2.

Table 5.2. The network takes a HCQT of an audio excerpt as input and outputs a vector of 18 values in  $[0, 1]$ , corresponding to activity of the 18 classes in  $\mathbf{C}$  predicted for the center frame of the input excerpt.

The HCQT input consists of 201 frames (roughly 4.7 seconds), computed using a hop-size of 512 on recordings sampled at 22 050 Hz (i. e., frame rate of 43 Hz). The constant-Q spectrum ranges from C1 to B7 with three bins per pitch, meaning 252 bins in total. For the harmonic CQT, five harmonic representations (including one subharmonic) are stacked in channel dimension. The final input tensor has a size of (201, 252, 5).

The network consists of three stages, separated by doubled lines in the table. Inspired by [164, 226], we first process each input tensor with a large pre-filtering kernel of size (15, 15) and strides (1, 1), followed by a kernel of size and stride (1, 3), i. e., the kernel is applied in pitch direction only. The resulting intermediate feature map has a pitch axis with a single bin per pitch. The second stage of our network applies three conv-conv-pool processing blocks, as in [182, 190]. In the final stage, we use a convolutional filter of size (5, 1) and stride (1, 1) to aggregate temporal context (as in [226]) and apply several dense layers to obtain the final output. We further use batch normalization after each learnable layer and apply dropout before dense layers. All layers are followed by a leaky ReLU activation, except for the final dense layer, which is followed by a sigmoid.

We train our network by minimizing a binary cross-entropy loss for a maximum of 1000 epochs (with 320 batches per epoch, each containing 32 input excerpts) using the Adam optimizer with a learning rate of 0.002. We additionally use early stopping by terminating training after the validation loss (evaluated on a randomly selected subset of the training set) has not decreased for 15 epochs. We further half the learning rate after 10 epochs without improvement of the validation loss. We use label smoothing inside the cross-entropy loss as regularization (thus, 0-labels are replaced with 0.02 and 1-labels are replaced by 0.98).

Each of the 252 bins in the input excerpts is individually normalized to be zero-mean and unit-variance (for this, mean and standard deviation per bin are estimated on the training set). As is common practice [2, 156, 181, 200], we augment training excerpts by randomly applying time warping, pitch shifting, masking of time-frequency bins, adding random noise, or applying random equalization. Training the model on our dataset takes around 14 hours on an RTX 2080 Ti

At test time, we evaluate our network for every frame in the test recordings, apply a median filter of length 0.5 seconds on the sequence of predictions for each class, and then downsample to a feature rate of 5 Hz. This is common practice in musical activity detection systems (e. g., [116, 181]) and makes our evaluation results more robust to annotation errors introduced by music synchronization (see Section 5.4). We finally binarize these predictions using a threshold of 0.5. Inference requires 3.5 seconds per minute of input audio (given pre-computed HCQT representations).

**Table 5.3:** Results for our **HC** (hierarchical classification) approach to IAD on our orchestral datasets.

	<b>P</b>	<b>R</b>	<b>F</b>	<b>S</b>	$\gamma^\downarrow$	$\gamma^\uparrow$	$\delta$
INST	0.99	1.00	0.99	0.67	0.99	1.00	0.96
WW	0.86	0.87	0.87	0.80	0.92	1.00	0.59
Fl	0.76	0.62	0.69	0.90			0.35
Ob	0.75	0.73	0.74	0.85			0.38
Cl	0.74	0.70	0.72	0.82			0.42
Bn	0.74	0.76	0.75	0.80			0.42
BR	0.79	0.70	0.74	0.85	0.94	0.99	0.45
Hn	0.75	0.68	0.71	0.84			0.41
Tpt	0.76	0.50	0.60	0.97			0.16
TMP	0.79	0.57	0.66	0.98			0.11
VOC	0.93	0.87	0.90	0.99	0.96	0.99	0.12
Fe	0.96	0.81	0.88	1.00			0.06
Ma	0.90	0.87	0.88	0.99			0.06
ST	0.95	0.95	0.95	0.65	0.99	1.00	0.87
Vn	0.90	0.93	0.91	0.66			0.77
Va	0.81	0.88	0.84	0.64			0.64
Vc	0.85	0.88	0.87	0.74			0.64
Db	0.86	0.87	0.87	0.83			0.56
Avg. ( <b>C</b> )	0.84	0.79	0.81	0.83	0.96	1.00	
Avg. ( <b>C</b> <sup>2</sup> )	0.86	0.79	0.82	0.85			
Avg. ( <b>C</b> <sup>1</sup> )	0.82	0.77	0.79	0.84			

## 5.6 Main Results

In this section, we present the main evaluation results for our IAD system and, additionally, demonstrate that hierarchy information reduces the need for fine-grained labels during training.

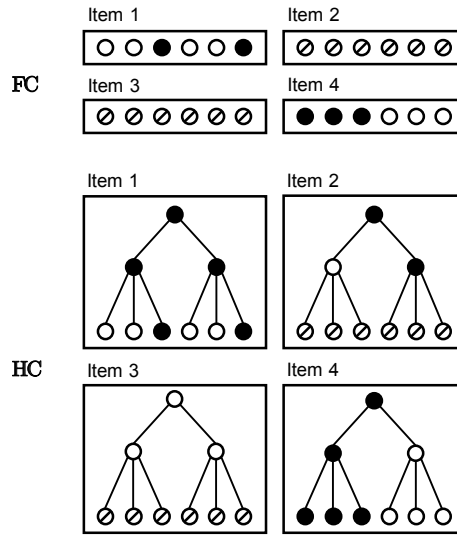
We begin with our main experiment. We train the model described in Section 5.5 on the training subset of our dataset and subsequently evaluate it on the test set (according to the split described in Section 5.4). Recall that we choose a joint classification approach **HC** (short for hierarchical classification), i. e., the network outputs predictions for all 18 classes in **C**, see also Section 5.3.

Evaluation results on the test set are shown in Table 5.3. Columns contain different metrics, computed over the entire test set (note that the consistency metrics  $\gamma$  are only defined for classes in **C**<sup>*h*</sup> for *h* > 1). Rows correspond to different classes in **C**, and the last three rows show averages over classes. In particular, we report averages for families (**C**<sup>2</sup>) and instruments (**C**<sup>1</sup>).<sup>19</sup>

Overall, evaluation results are moderately high with an average F-measure of 0.81 and specificity of 0.83. For comparison, Hung and Yang [84] achieve an average F-measure of 0.89 for IAD on recordings of small classical ensembles with seven different instrument classes. However, our results vary across classes. We can observe that classifying instrument families works better on average ( $F = 0.82$ ) than classifying fine-grained classes ( $F = 0.79$ ). For example, for woodwinds (**WW**), the family F-measure of 0.87 is much higher than the detection results obtained for the individual woodwind instruments (e. g., for

<sup>19</sup> These are macro averages, i. e., computed as the arithmetic mean of the results for the individual classes. As such, both common and uncommon classes contribute equally to these averages.

**Figure 5.4:** Experiment setup for reducing the amount of instrument-level labels available for training. Boxes correspond to training items and circles indicate class annotations. Crossed-out circles correspond to missing annotations. For the **FC** (flat classification) approach, we can only use training items for which fine-grained labels are given. For **HC** (hierarchical classification), we can utilize higher-level information even if instrument labels are unavailable.



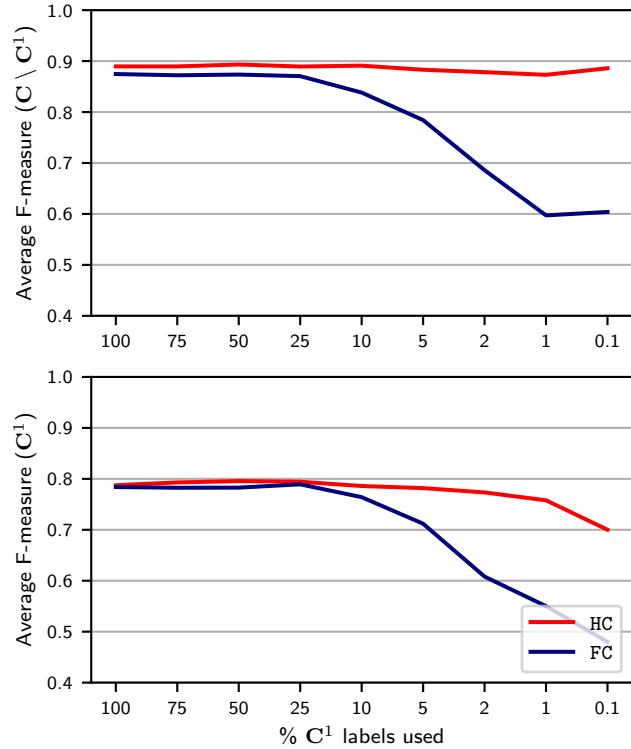
clarinets:  $F_{CL} = 0.72$ ). Some very common classes yield high F-measures, but low specificity (e. g., for strings:  $\delta_{ST} = 0.87$ ,  $F_{ST} = 0.95$ , and  $S_{ST} = 0.65$ ), indicating that our system produces many false positive predictions for these classes. In Section 5.8, we will conduct some additional analyses to better understand these detection results with regard to possible confounding effects.

For singing activity (VOC), we obtain a family F-measure of 0.90 and the difference to the results for individual vocal classes female (Fe) and male (Ma, for both:  $F = 0.88$ ) is small. For reference, one obtains accuracies of around 0.91 for singing voice detection on popular music [181, 182]. With regards to consistency,  $\gamma^\uparrow$  is always high. Therefore, predictions on a fine-grained class are almost always accompanied by a prediction for the parent class. However,  $\gamma^\downarrow$  is lower for some classes such as WW ( $\gamma_{WW}^\downarrow = 0.92$ ). Thus, for about eight percent of frames where the woodwind family is predicted as active, neither of the woodwind instruments is identified. In Section 5.7, we will consider loss terms for improving these consistency issues.

To demonstrate the impact of our hierarchical classification approach **HC**, we now analyze whether utilizing the instrument hierarchy during training can reduce the amount of fine-grained labels required. To this end, we compare **HC** with a flat classification baseline **FC**. There, our model is trained to only produce predictions for classes in  $\mathbf{C}^1$  (i. e., instrument-level). At test time, we obtain predictions for classes in  $\mathbf{C} \setminus \mathbf{C}^1$  by aggregating predictions from lower levels in a bottom-up fashion.<sup>20</sup> For example, VOC is predicted if and only if the model has classified the input as Fe or Ma. By construction, the predictions obtained in this way are always consistent, so  $\gamma_c = \gamma_c^\uparrow = \gamma_c^\downarrow = 1$  for all classes  $c$ . The **FC** baseline allows us to determine the detection quality that can be achieved without informing the model about the class hierarchy.

<sup>20</sup> This baseline is referred to as Strategy *B* in Chapter 4. Note that we cannot obtain predictions for timpani (TMP) using this baseline and thus omit TMP from consideration in the following discussion.

**Figure 5.5:** Results for reducing the amount of instrument-level (i. e.,  $C^1$ ) labels available for training. Average F-measures are plotted separately for instrument classes (lower plot) and higher-level classes ( $C \setminus C^1$ , upper plot). Lines correspond to hierarchical classification (**HC**) and flat classification (**FC**), respectively.



We now reduce the amount of fine-grained instrument labels that are available during training. For **FC** (flat classification baseline), we do so by reducing the size of the training set, since this approach does not utilize class labels at higher levels for training. For **HC** (hierarchical classification), we disable the cross-entropy loss associated with classes in  $C^1$  on a portion of the training items, but we still use the instrument family and activity labels for these items.<sup>21</sup> This experiment setup is illustrated in Figure 5.4.

Figure 5.5 shows the results of this experiment. The upper plot shows results for higher-level classes (families and instrument activity), whilst the lower plot contains results for fine-grained classes. When utilizing all fine-grained labels of the training dataset, we observe almost identical average F-measures for  $C^1$  with both **FC** and **HC** (lower plot, leftmost point). For  $C \setminus C^1$  (upper plot), **HC** yields slightly higher average F-measures at 0.89 (compared to  $F = 0.87$  for **FC**). When reducing the amount of  $C^1$  labels used, **HC** outperforms **FC** on both fine-grained and higher-level classes. For example, at 10% of labels used, we obtain an average F-measure of 0.84 on  $C \setminus C^1$  for **FC**, which drops to  $F = 0.60$  for 0.1% of labels. Meanwhile, the results for **HC** on  $C \setminus C^1$  stay roughly constant at around  $F = 0.88$ . **HC** also yields higher F-measures for  $C^1$ , with  $F = 0.76$  at 1% of labels used as opposed to  $F = 0.55$  for **FC**.

We have seen that, by utilizing higher-level structure when fine-grained labels are scarce, our hierarchical classification approach **HC** can still yield good results, even for small amounts of instrument-level labels. This opens up the possibility of incorporating partially labeled data for training, where the instrument

<sup>21</sup> Disabling the cross-entropy loss for learning from partial labels was suggested in [44]. Gururani and Lerch [66] used this technique in the context of polyphonic instrument classification. They did not consider hierarchical information.

**Table 5.4:** Results for the  $\mathbf{HC}^{\alpha,\beta}$  strategy that includes hierarchy information and consistency losses during training. Here, we set  $\alpha = \beta = 10$ .

	<b>P</b>	<b>R</b>	<b>F</b>	<b>S</b>	$\gamma^\downarrow$	$\gamma^\uparrow$	$\delta$
INST	0.99	0.99	0.99	0.73	1.00	1.00	0.96
WW	0.88	0.85	0.87	0.83	0.98	0.99	0.59
F1	0.76	0.65	0.70	0.89			0.35
Ob	0.76	0.74	0.75	0.86			0.38
Cl	0.73	0.77	0.75	0.79			0.42
Bn	0.73	0.82	0.77	0.78			0.42
BR	0.79	0.70	0.74	0.85	1.00	1.00	0.45
Hn	0.73	0.70	0.71	0.82			0.41
Tpt	0.80	0.54	0.65	0.97			0.16
TMP	0.78	0.59	0.67	0.98			0.11
VOC	0.93	0.87	0.90	0.99	1.00	0.99	0.12
Fe	0.93	0.83	0.88	1.00			0.06
Ma	0.90	0.88	0.89	0.99			0.06
ST	0.94	0.96	0.95	0.60	1.00	1.00	0.87
Vn	0.88	0.95	0.91	0.58			0.77
Va	0.81	0.88	0.84	0.63			0.64
Vc	0.85	0.91	0.88	0.71			0.64
Db	0.86	0.87	0.87	0.82			0.56
Avg. (C)	0.84	0.81	0.82	0.82	0.99	1.00	
Avg. (C <sup>2</sup> )	0.87	0.79	0.83	0.85			
Avg. (C <sup>1</sup> )	0.81	0.80	0.80	0.82			

family is known, but fine-grained labels are unavailable (e. g., monotimbral recordings of brass or string ensembles).

## 5.7 Consistency Losses

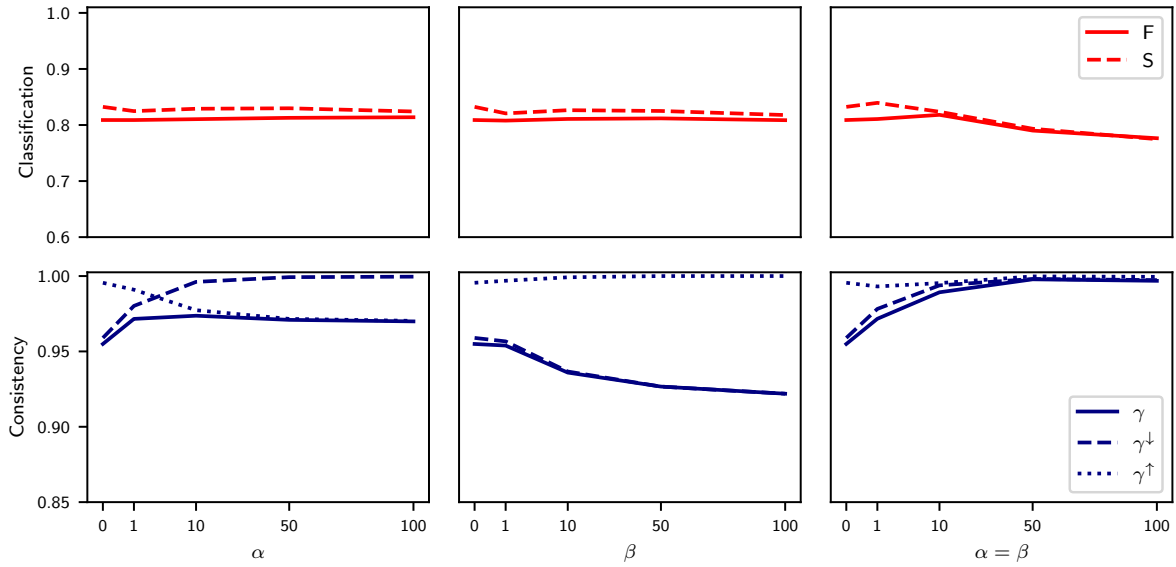
As discussed in Section 5.6, our hierarchical IAD approach  $\mathbf{HC}$  outperforms the flat classification baseline  $\mathbf{FC}$  in terms of F-measures. However, the predictions of  $\mathbf{FC}$  are always consistent, making the output of that system easier to understand compared to  $\mathbf{HC}$ , which may produce inconsistent outputs. In this section, we will investigate additional loss terms for  $\mathbf{HC}$  that can address this shortcoming.

In the previous Chapter 4, we describe the  $\mathcal{L}_\uparrow$  loss term for improving bottom-up consistency and the  $\mathcal{L}_\downarrow$  loss for addressing top-down consistency of predictions. These losses are combined with the standard cross entropy loss  $\mathcal{L}_{\text{BCE}}$  using weights  $\alpha, \beta \in \mathbb{R}$ , yielding the final loss

$$\mathcal{L} = \mathcal{L}_{\text{BCE}} + \alpha \mathcal{L}_\downarrow + \beta \mathcal{L}_\uparrow \quad (5.4)$$

for our model. We will denote the hierarchical classification approach trained with these additional losses as  $\mathbf{HC}^{\alpha,\beta}$

Results for training with these additional consistency losses are shown in Table 5.4. Here, we set  $\alpha = \beta = 10$  (as in Chapter 4). With regard to the detection evaluation measures like F-measure and specificity, we obtain similar results as in our previous experiments that did not employ consistency losses



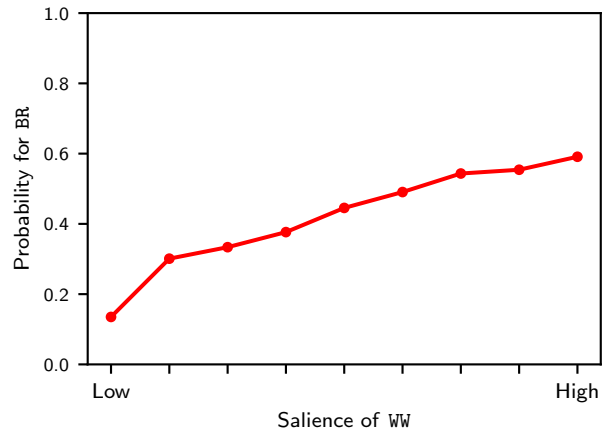
**Figure 5.6:** Results for different choices of  $\alpha$  (coefficient for the top-down loss  $\mathcal{L}_\downarrow$ ) and  $\beta$  (coefficient for the bottom-up loss  $\mathcal{L}_\uparrow$ ). The upper row shows classification results in terms of F-measure (F) and specificity (S). The lower row shows measures of top-down ( $\gamma^\downarrow$ ), bottom-up ( $\gamma^\uparrow$ ) and overall consistency ( $\gamma$ ). In the first column  $\beta = 0$ , and in the second column  $\alpha = 0$ . Measures are averaged over all classes.

(see in Table 5.3). For example, we get an average  $F = 0.82$  and  $S = 0.82$  for  $\mathbf{HC}^{\alpha,\beta}$  compared to  $F = 0.81$  and  $S = 0.83$  for  $\mathbf{HC}$ . However, the top-down consistency scores  $\gamma^\downarrow$  have improved (e. g.,  $\gamma_{\text{BR}}^\downarrow = 1.00$  and  $\gamma_{\text{WW}}^\downarrow = 0.98$  for  $\mathbf{HC}^{\alpha,\beta}$  compared to  $\gamma_{\text{BR}}^\downarrow = 0.94$  and  $\gamma_{\text{WW}}^\downarrow = 0.92$  for  $\mathbf{HC}$ ). Thus, the additional loss terms can improve consistency while retaining the overall quality of results.

A more detailed analysis of the impact of the loss weights  $\alpha$  and  $\beta$  is provided in Figure 5.6. Here, we either use solely  $\mathcal{L}_\downarrow$  (by setting  $\beta = 0$  and increasing  $\alpha$ , first column), use solely  $\mathcal{L}_\uparrow$  (setting  $\alpha = 0$  and increasing  $\beta$ , second column), or use both losses simultaneously ( $\alpha = \beta$ , third column). The upper row shows classification results in terms of average F-measure (F) and specificity (S), while the lower row shows average consistency scores. As expected, by using solely  $\mathcal{L}_\downarrow$ , we are able to improve  $\gamma^\downarrow$ . However,  $\gamma^\uparrow$  decreases for large values of  $\alpha$ . The opposite behavior can be observed for using only  $\mathcal{L}_\uparrow$ . In both cases,  $\gamma$  decreases while F-measure and specificity remain roughly constant. By utilizing both  $\mathcal{L}_\downarrow$  and  $\mathcal{L}_\uparrow$ , we are able to improve  $\gamma$ . However,  $F$  and  $S$  are reduced for large values of  $\alpha = \beta$ . We conclude that both  $\mathcal{L}_\downarrow$  and  $\mathcal{L}_\uparrow$  are required to increase consistency, while  $\alpha = \beta$  should be chosen small enough in order not to deteriorate detection results.

Overall, our results suggest that, while consistency is a necessary condition for interpretable system outputs, it is not sufficient to achieve good classification results. Our losses can be used to induce more consistent detection outputs for our model, but high consistency needs to be balanced with preserving classification results.

**Figure 5.7:** Average probabilities output by our model for brass depending on the salience of woodwinds within the orchestral mixture. Here we only consider frames where no brass is active at all, i. e., the ideal probability output would be 0.



## 5.8 Analysis of Confounding Factors

In this section, we aim at a deeper understanding of the behavior of our model. In particular, we analyze how the detection of an instrument class is affected by the presence of other instruments playing simultaneously. To this end, we systematically evaluate our model on audio inputs for which we can calculate the relative salience of different classes within the overall mixture. In this way, we can reach conclusions about confounding effects exploited by our model.

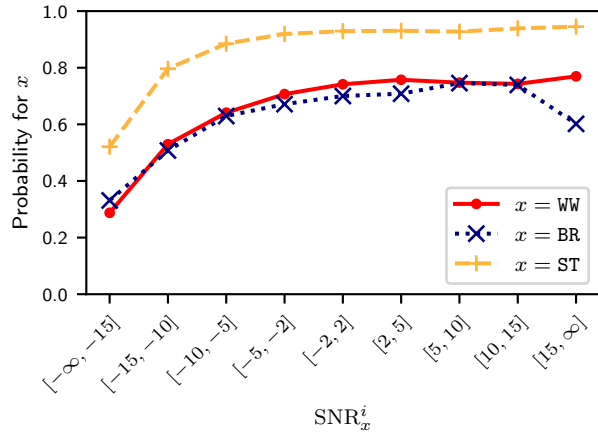
We begin with a representative example of an analysis result, shown in Figure 5.7. Technical details are provided below. In this example, we are interested in the interactions between brass (BR) and woodwind (WW) instruments. In orchestra music, brass and woodwinds are often active simultaneously. For this analysis, we select frames for which no brass (BR) is active and then observe the predictions for BR depending on the salience of woodwind (WW) instruments. The vertical axis shows average probabilities predicted by our model for BR. The probabilities are averaged over all frames where woodwinds have a certain salience within the orchestral mixture (horizontal axis). Here, salience is measured as a signal-to-noise ratio, with woodwinds considered as signal and all other instrument sounds considered as noise. We observe that the model outputs low probabilities of around 0.1 for BR if WW is inactive. However, these outputs gradually increase as woodwinds become more salient in the input. We conclude that brass detection is highly sensitive to woodwind instruments, even if no brass instrument is active in the mixture. This is a confounding effect.

We can reach a number of similar conclusions by performing systematic analyses:

1. The model also exploits the presence of woodwind instruments for detecting brass if brass instruments are present in the mixture.
2. Our model is biased towards predicting string activity, even when no strings are active in the input. Thus, the model exploits the fact that strings are active in most frames of the recordings in our dataset (see also Figure 5.3).



**Figure 5.8:** Average probabilities output by our model for different instrument families (colored lines) as functions of the signal-to-noise ratio for that family within the orchestral mixture.



3. Predictions for strings are not affected much by the presence of other instrument families.
4. Looking at the model behavior for woodwind instrument classes, we find that detection of flutes is improved by the simultaneous activity of other woodwind instruments, similar to the behavior observed for BR.

In the following, we provide details on how we obtained our conclusions. We utilize multi-track orchestral datasets (see Section 5.4) that allow us to measure the signal-to-noise ratio (SNR)<sup>22</sup> for different instruments and instrument families in the recordings. In practice, we only use the Phenix and Beethoven Anechoic datasets for this analysis, because the individual tracks in Freischütz Digital contain a lot of interference from other sources.<sup>23</sup> To compute the SNRs, we split both the individual track for class  $c \in \mathcal{C}$  and the corresponding mixture into frames<sup>24</sup>, compute the signals' power in each frame and finally obtain the SNR on a decibel scale as

$$\text{SNR}_c^i = 10 \log_{10} \left( \frac{P_c^i}{P_{\text{Mix}}^i} \right), \quad (5.5)$$

where  $P_c^i$  and  $P_{\text{Mix}}^i$  correspond to the power in frame  $i$  of the track for class  $c$  and the full mixture, respectively. We exclude frames without any instrument activity from our analysis, since their SNR values are meaningless.

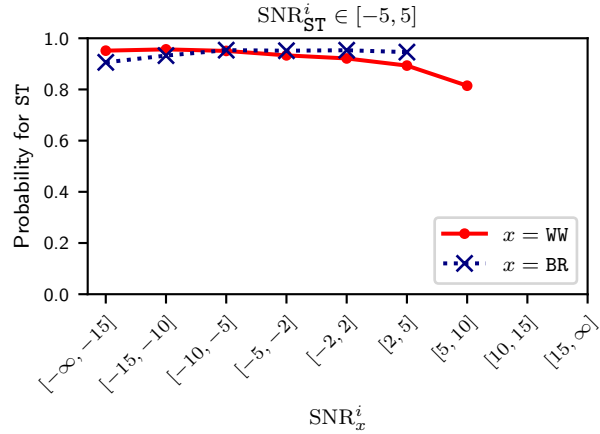
We now utilize  $\text{SNR}_c^i$  to partition the frames in our test recordings by SNR and then analyze the outputs of our model in these frames. We begin by analyzing the probabilities predicted by our model for different instrument families, depending on the SNR for that family in the mixture. Figure 5.8 shows the probabilities predicted by our model for an instrument family  $x$  (vertical axis), averaged over all frames  $i$  where  $x$  has a certain  $\text{SNR}_c^i$  (horizontal axis). Colored lines correspond to different families  $x$ . As

<sup>22</sup> Here, “signal” refers to the sound produced by the instrument in question and “noise” refers to all other sounds playing simultaneously, i. e., other instruments as well as non-musical noise.

<sup>23</sup> In order to obtain more input mixtures for our analysis, we also expand our test dataset by creating additional mixtures from the multi-track datasets, where we boost or reduce the contributions of different instruments.

<sup>24</sup> We use a window size of 2048 and a hop size of 512 at a sample rate of 22 050 Hz, leading to a frame rate of 43 Hz.

**Figure 5.9:** Average probabilities for strings depending on the signal-to-noise ratio of other instrument families (colored lines). Only frames where strings are jointly active with another family (i. e.,  $\text{SNR}_{\text{ST}}^i \in [-5, 5]$ ) are considered.



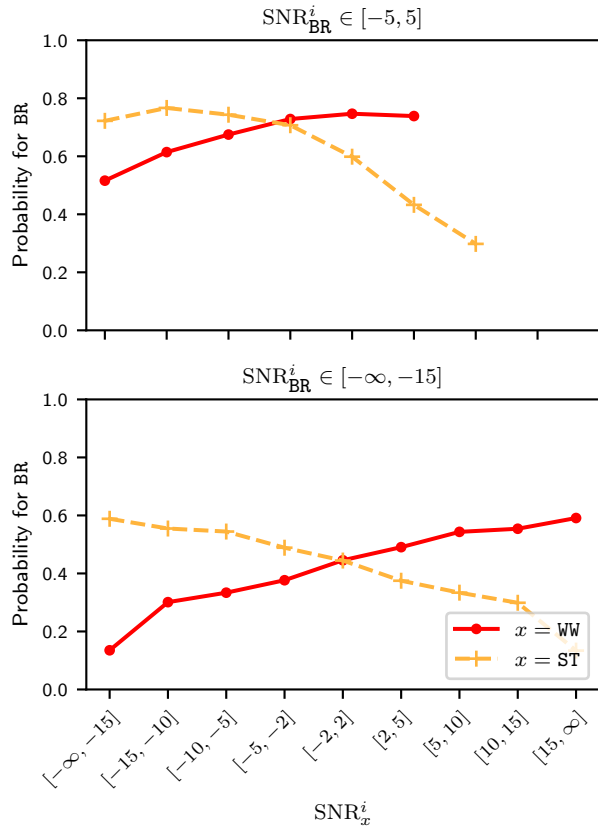
one may expect, higher SNRs for an instrument family generally result in higher probabilities for that family. Thus, our model is more confident about predicting an instrument family when that family is more salient in the mixture. However, for BR, frames with  $\text{SNR}_{\text{BR}}^i \in [15, \infty]$  yield lower predictions, i. e., our model is less confident about detecting brass when brass is the only family being active. Furthermore, the average probability for ST never falls below 0.5, even if strings are not present in the mixture (i. e.,  $\text{SNR}_{\text{ST}}^i \in [-\infty, -15]$ ). In other words, our model is biased towards predicting string activity, exploiting the fact that strings are very common in our dataset (see also Figure 5.3). Because of this, we obtain low specificity for string classes in Table 5.3.

We now look at the interactions between different instrument families and investigate how the presence of one family may influence predictions for other families. Figure 5.9 shows how probabilities predicted for strings change based on the SNR of other instrument families in the mixture. Note that we only consider frames here where strings are playing jointly with other instrument families (i. e.,  $\text{SNR}_{\text{ST}}^i \in [-5, 5]$ ). We observe that predicted probabilities change only little depending on the presence of BR. There is a slight trend towards lower predictions when woodwinds are active, with the average probability dropping to 0.8 for  $\text{SNR}_{\text{WW}}^i \in [5, 10]$ . Overall, Figure 5.9 demonstrates that predictions for strings are not affected much by the presence of other families.

We repeat this analysis for brass probabilities in the upper row of Figure 5.10. In contrast to strings, there is a large impact of other instrument families on predictions for BR. We observe that probabilities for BR are higher when  $\text{SNR}_{\text{WW}}^i$  increases. Probabilities are much lower at around 0.3 for frames  $i$  with  $\text{SNR}_{\text{ST}}^i \in [5, 10]$ . Both observations suggest that our model exploits the presence of woodwind instruments for brass detection. This is an undesirable confounding effect.

Analogously, we consider frames where no brass is active in the lower row of Figure 5.10. We observe that predicted probabilities for BR are high if no strings are active and then gradually drop to around 0.1 if only strings are active. Conversely, we get low probabilities for BR if WW is inactive but high predictions

**Figure 5.10:** Average probabilities for brass depending on the signal-to-noise ratio of other instrument families (colored lines). In the upper row, we consider frames where brass is jointly active with other instruments. In the lower row, we consider frames where no brass is active at all.

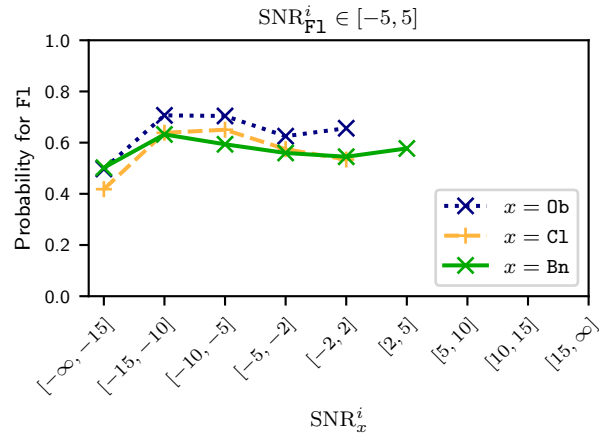


for high  $\text{SNR}_{\text{WW}}^i$ . We conclude that our brass detection is highly sensitive to woodwind instruments, even if no brass instrument is active in the mixture.

We can use similar techniques to understand the model behavior for woodwind instrument classes. Figure 5.11 shows predicted probabilities for flutes, depending on SNRs of other woodwind instruments. Here, we again only consider frames where flutes are playing jointly with other instruments ( $\text{SNR}_{\text{F1}}^i \in [-5, 5]$ ). Predictions for F1 increase if other woodwind instruments have  $\text{SNR}_{\text{C}}^i \in [-15, 5]$ , compared to these instruments being inactive. In other words, detection of flutes depends on the simultaneous activity of other woodwind instruments, similar to the behavior observed for brass and other instrument families.

Our analysis reveals confounding factors exploited by our model, arising from the large amount of joint instrument activity (see also Figure 5.3). It is important to note that it may indeed be desirable for our model to use these confounding factors for detection, since these are strong cues for IAD on many orchestral works from the classical and romantic periods (as present in our dataset). Yet, use of these confounding factors may also limit the generalization ability of our model to music with other instrument statistics, e. g., works with many brass-only sections. To reduce the impact of these effects, one may collect additional training data (which is cumbersome, see Section 5.4). Entirely removing confounding effects from our system may be impossible, however, as we also discuss in the next section.

**Figure 5.11:** Average probabilities for flutes depending on the signal-to-noise ratio of other woodwind instruments (colored lines). Only frames where flutes are jointly active with other instruments (i. e.,  $\text{SNR}_{F1}^i \in [-5, 5]$ ) are considered.



## 5.9 Conclusion

In this chapter, we investigated instrument activity detection in the context of complex orchestral music recordings. We showed that utilizing information about hierarchical relationships between instruments is helpful, especially when only few fine-grained instrument-level labels are available. Furthermore, we demonstrated how one can increase the consistency of predictions across hierarchy levels using additional consistency losses, while preserving detection quality. To perform these experiments, we collected a large dataset of real-world opera and orchestra recordings with aligned instrument activity annotations. Finally, we analyzed the behavior of our detection system and identified confounding effects exploited by our model.

Future work may make use of more complex instrument hierarchies (e. g., hierarchies that incorporate knowledge about different sound production techniques), train more complex detection models, collect additional data for training and testing (e. g., using music synchronization or synthesizers), or extend the scenario considered here towards orchestral music from other epochs (e. g., Baroque).

However, even when collecting additional data, it is likely that our model will continue to exploit confounding effects arising from the training data, as shown in our analysis in Section 5.8. This may be desirable (in case that training and test conditions are very similar) or undesirable (when generalizing to music from different styles). Our analysis results mirror those obtained for other systems for music classification. For example, Kelz and Widmer [92] found that a neural network for piano transcription trained on combinations of notes struggles with correctly classifying unknown note combinations. The system performs transcription for certain notes by considering other, unrelated notes as well. The confounding effects exploited in MIR systems can often be even less intuitive. In [198], for example, a system for recognizing Latin music styles (which are usually characterized by their rhythms) was shown to exploit tempo information. Similarly, the authors in [172] found that a system for genre recognition is sensitive to sounds outside the human hearing range. In Chapter 9, a deep learning system for leitmotif

detection in opera recordings is introduced. It is shown to rely heavily on spectral statistics as opposed to melody or rhythm. Such problems are amplified when evaluating a system on music styles not seen during training. For example, the authors in [123] showed that off-the-shelf music classifiers perform poorly on a large and diverse music database and are highly sensitive to encoding artifacts.

In order to tackle this challenge for IAD, one may consider using source separation as pre-processing, thereby reducing the impact of unrelated instruments on detection outputs. However, due to the small amount of multi-track orchestral data available, no off-the-shelf systems for source separation in orchestra recordings are currently available, leaving this as an avenue for future work.



# 6 A Cross-Version Approach to Representation Learning for Instrumentation

This chapter is based on [108]. The first author Michael Krause is the main contributor to this article. In collaboration with his supervisor Meinard Müller and Christof Weiß, he devised the ideas, developed the formalization, designed the experiments, and wrote the paper. Furthermore, Michael Krause collected the dataset, implemented all approaches, and conducted the experiments.

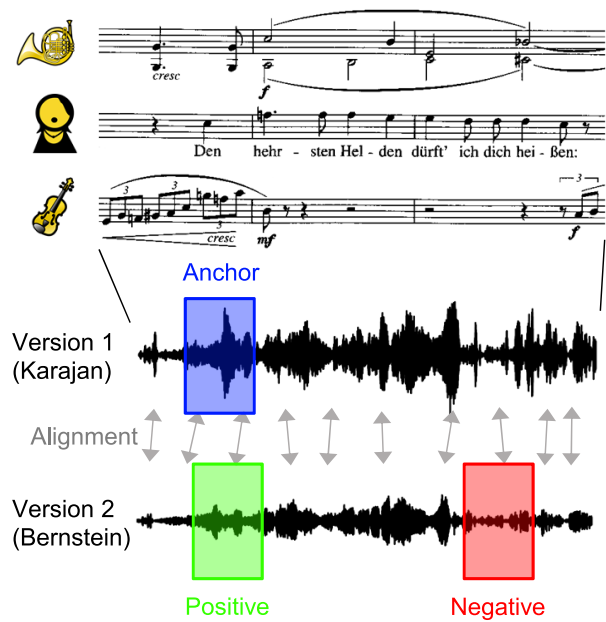
Deep learning systems often require large amounts of labeled data for supervised training, which can be very costly to obtain (see, e. g., the effort required for obtaining instrument activity annotations as described in Section 5.4). To alleviate this problem, recent papers on learning music audio representations employ alternative training strategies that utilize unannotated data. In this chapter, we introduce a novel cross-version approach to audio representation learning that can be used with music datasets containing several versions of a musical work. Our method exploits the correspondences that exist between two versions of the same musical section. We evaluate our proposed cross-version approach qualitatively and quantitatively on complex orchestral music recordings and show that it can better capture aspects of instrumentation compared to techniques that do not use cross-version information.

## 6.1 Introduction

Deep learning has become a common tool for approaching diverse tasks in MIR. These approaches usually follow a supervised learning scheme, where a neural network is trained on the annotations of some dataset. For many MIR tasks, however, such annotations are costly to obtain. Recent work has investigated alternatives that require little or no annotations and enable training on large, unannotated datasets.

For certain music genres, there are datasets that contain several versions of a musical work, see also Section 3.4. For example, the same classical symphony or concerto can be performed by different orchestras, and several commercial recordings are often available. On such datasets, automatic music

**Figure 6.1:** Visualization of our cross-version approach to representation learning for orchestral music. An anchor (blue) excerpt is selected from a music recording. The positive (green) and negative (red) excerpts are chosen from a different version of the same musical piece. For this, an alignment between versions is needed (gray arrows).



synchronization techniques can be used to find alignments between different versions of a work, requiring minimal annotation effort [145, 147].

In this chapter, we introduce a conceptually novel approach to audio representation learning that exploits cross-version datasets, thus requiring only alignments between versions and no further human annotations. Our approach aims at learning embeddings of audio excerpts such that musically corresponding excerpts in different versions are mapped to close points in the embedding space (Figure 6.1).

There are several musical aspects that stay roughly constant across most versions, e. g., pitches, harmonies or rhythm. For orchestral music, aspects of instrumentation (i. e., active instruments or instrument families) are another such property. Instrumentation represents a challenging MIR scenario given the complexity of instrument taxonomies and the difficulty of annotating instrument activity in orchestral music. In our experiments on a dataset of complex orchestral music, we show qualitatively and quantitatively that—by utilizing the correspondences between different versions of a musical section—our proposed representation learning technique is better at capturing aspects of instrumentation and instrument texture compared to approaches that do not exploit cross-version information.

The remainder of the chapter is structured as follows: Section 6.2 covers related work on music audio representation learning, cross-version analysis, and instrumentation in orchestral music. In Section 6.3, we introduce our proposed approach. In Section 6.4, we describe our experimental setup, including datasets, our model architecture, and baselines. Section 6.5 contains qualitative and quantitative results and Section 6.6 concludes the chapter with a discussion of possible future work.



## 6.2 Related Work

Several recent contributions have explored so-called self-supervised strategies for learning representations from unannotated music recordings. Often, in these studies, excerpts from a music recording that are in close proximity are considered as positive pairs (i. e., should be mapped to similar representations) whereas excerpts that are further apart (or from other recordings) are negative pairs (i. e., should be mapped to dissimilar representations). This idea is also illustrated in Figure 6.2. McCallum [133] originally considered this with the aim of learning features for music structure analysis. Wang et al. [216] had a similar use case but used a supervised learning approach. Several authors employed such a strategy for learning more general purpose representations [23, 134, 175, 191, 205, 211], often applying additional augmentations. Apart from using temporal proximity, other papers on music representation learning exploit audio-visual or audio-text correspondences [120, 130], use classical features as training targets [231], exploit metadata [6], or investigate music generation models [27].

The approach proposed in this paper is conceptually different since we utilize cross-version datasets, rather than relying on temporal proximity alone. Such datasets contain several recorded versions of a musical work, which may vary in aspects related to musical interpretation, recording conditions, and timbral characteristics of the instruments used. These datasets have been exploited for expressive performance rendering [241] or improved harmonic analysis [52]. Cross-version datasets also allow for investigating model biases and overfitting effects in MIR models through different dataset splits [185]. To our knowledge, the only other work utilizing cross-version information for embedding learning is by Zalkow et al. [236], whose aim was to compress chromagram excerpts for efficient music retrieval. In contrast, we propose to learn representations based on spectrogram-like input features and investigate them for capturing instrument texture.

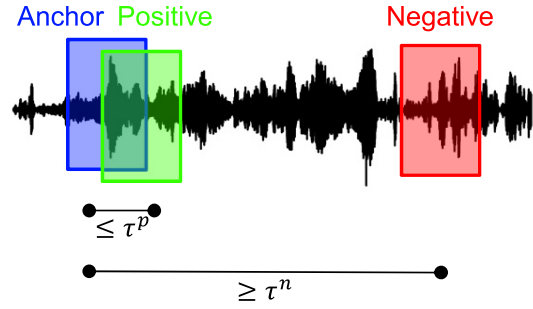
In the wider machine learning literature, representations are often learned by masking a part of an input and predicting the masked content [76, 81]. Other strategies utilize multi-modal datasets, e. g., containing text-image [169] or audio-text pairs [68].

As previously summarized in Section 5.2, orchestral music has been explored in the context of source separation [140] or melody extraction [201]. The authors in [200] considered instrument family recognition for classical, monotimbral recordings using a supervised learning approach. Other recent papers on instrument activity detection in music recordings [67, 71, 84] have also considered DL-based, supervised learning approaches, but not within orchestral scenarios.

## 6.3 Cross-Version Approach to Audio Representation Learning

In this section, we formalize our proposed cross-version approach to representation learning. The key idea is to utilize correspondences between different versions (i. e., recorded performances played by different

**Figure 6.2:** When forming triplets of audio excerpts, the anchor and positive/negative excerpts are chosen according to a maximum/minimum distance  $\tau^p/\tau^n$ .



orchestras) of the same musical work. We aim to learn embeddings of audio excerpts such that the same musical section in different versions is represented by neighboring points in the embedding space and audio excerpts for unrelated musical sections are mapped to distant points in the embedding space. To this end, inspired by [236], we sample triplets of audio excerpts as in Figure 6.1, and apply a triplet loss for learning. Musical characteristics that stay roughly constant across different versions of an orchestral work include pitches and harmonies, as well as instrumentation. In later sections, we will analyze to what extent our approach captures pitches or aspects of instrumentation.

**Single-Version Approach (SV).** We begin by formalizing a common approach to music representation learning that only utilizes temporal proximity inside a single version, see also Section 6.2 and Figure 6.2. Let  $\mathcal{W}$  be a set of musical works and let  $V_w$  be the set of available versions for a work  $w \in \mathcal{W}$ . We first randomly select a work  $w \in \mathcal{W}$  and some version of this work  $v \in V_w$ . Let  $T$  denote the length of  $v$  in seconds. We choose an anchor excerpt by uniformly sampling an anchor position  $a \in [0, T]$  and extracting the excerpt  $\mathbf{x}^a$  of  $v$  that is centered around  $a$ . To obtain the positive and negative excerpts, we choose a position  $p \in [0, T]$  for the positive excerpt  $\mathbf{x}^p$  of  $v$  such that  $|a - p| \leq \tau^p$ . Thus, the positive excerpt is in temporal proximity of the anchor excerpt—up to a threshold of  $\tau^p$  seconds—and is likely to correspond to a musically similar section. In the same way, we choose a position  $n \in [0, T]$  for the negative excerpt  $\mathbf{x}^n$  of  $v$  such that  $|a - n| \geq \tau^n$ . The negative excerpt is therefore a certain minimum distance of  $\tau^n$  seconds away from the anchor position, likely corresponding to a musically dissimilar section.<sup>25</sup>

**Embedding Learning.** We obtain embeddings by passing these excerpts through a neural network (described in Section 6.4.2), i. e.:

$$\mathbf{Y} = (\mathbf{y}^a, \mathbf{y}^p, \mathbf{y}^n) = (f(\mathbf{x}^a), f(\mathbf{x}^p), f(\mathbf{x}^n)), \quad (6.1)$$

where  $f$  is a neural network that embeds an audio excerpt  $\mathbf{x}$  into an embedding vector  $\mathbf{y}$ . Using this triplet, we can apply a standard triplet loss [186] such as:

$$\mathcal{L}(\mathbf{Y}) = \max\left(0, \|\mathbf{y}^a - \mathbf{y}^p\|_2^2 - \|\mathbf{y}^a - \mathbf{y}^n\|_2^2 + \alpha\right), \quad (6.2)$$

<sup>25</sup> Due to repetitions and other structural similarities, there may in fact be some musically related sections that are far apart temporally. In the majority of cases, however, the assumption underlying positive and negative sampling will hold [133].

Composer	Work	Versions	
		Num.	Avg. Duration
Wagner	Die Walküre, Act 1	8	1 h
Beethoven	Symph. 3, Mvmts. 1–4	6	45 min
Dvorak	Symph. 9, Mvmts. 1, 2, 4	6	40 min
Tschaikowsky	Violin Concerto, Mvmts. 1–3	6	35 min
Total duration			20 h

**Table 6.1:** Our cross-version dataset containing several commercial recordings of different orchestral and opera compositions.

where  $\alpha \in \mathbb{R}_{\geq 0}$  describes the desired minimum margin between the distance of embeddings for anchor and positive versus the distance of embeddings for anchor and negative.

**Cross-Version Approach (CV).** For our proposed cross-version approach, we sample triplets in a different fashion. Since we utilize multiple versions per work, we now require  $|V_w| \geq 2$ . To form a triplet of excerpts, we randomly select some version  $v_1 \in V_w$  of a work  $w \in \mathcal{W}$ . We then sample an anchor position  $a_1 \in [0, T_1]$ , where  $T_1$  is the length of  $v_1$  in seconds, and extract the corresponding excerpt  $\mathbf{x}^a$  of  $v_1$ . To obtain the positive and negative excerpts, we randomly select another version  $v_2 \in V_w \setminus \{v_1\}$  of  $w$ . As before, let  $T_2$  denote the length of  $v_2$  in seconds. We can find the position  $a_2 \in [0, T_2]$  in  $v_2$  corresponding to the same musical position as the anchor  $a_1$  in  $v_1$  using music alignment techniques. With this, we choose a position  $p \in [0, T_2]$  for the positive excerpt  $\mathbf{x}^p$  of  $v_2$  such that  $|a_2 - p| \leq \tau^p$ . Thus, the positive excerpt corresponds to the same musical section as the anchor, up to some tolerance of  $\tau^p$  seconds (in addition to alignment inaccuracies). Similarly, we sample  $n \in [0, T_2]$  (with  $|a_2 - n| \geq \tau^n$ ) and extract  $\mathbf{x}^n$ . Note that only  $\mathbf{x}^a$  is an excerpt of the first version  $v_1$ , whereas both  $\mathbf{x}^p$  and  $\mathbf{x}^n$  are excerpts of the second version  $v_2$ . As before, we construct a triplet  $\mathbf{Y}$  using these excerpts and apply a standard triplet loss.

## 6.4 Experimental Setup

### 6.4.1 Dataset and Splits

To show the potential of our representation learning technique, we construct a cross-version dataset of commercial symphonic and opera music recordings, illustrated in Table 6.1. This dataset is closely related with the one described in Section 5.4. Here, we used two additional versions of the first act of *Die Walküre*.<sup>26</sup> In line with Section 5.4, we choose the first movement of the Beethoven Symphony, the fourth movement of the Dvorak Symphony and the third movement of the Tschaikowsky Concerto for testing. We further choose an excerpt of the act from *Die Walküre*, omit this excerpt during training, and use it for testing. We further ensure that the train and test set contain different versions. By splitting our dataset in

<sup>26</sup> Overall, we use P-Ne, P-Le, P-Bö, P-Ke, and P-Bo for training as well as P-Ka, P-Ba, and P-Ha for testing. See also Figure 3.2.

this fashion, we aim to avoid overfitting to specific musical compositions or recording conditions (the latter is also referred to as “album effect” [53]).

For the cross-version approach CV, we obtain an alignment between versions of the same work using state-of-the-art music synchronization techniques involving chroma onset features and multi-scale alignment [147]. For some experiments, we also require pitch-class and instrument activity annotations for our dataset. Again, we use music synchronization techniques to align score to audio and create the annotations. We refer to Section 5.4 for details.

### 6.4.2 Model

We implement all representation learning approaches using a CNN that takes a HCQT [13] of an audio excerpt as input and outputs a corresponding embedding vector. The model is identical to the one described in Section 5.5 (except for the final sigmoid activation, which we omit here, and the size of the final output layer, which is increased). For convenience, we shortly summarize the model in the following: The HCQT input consists of 201 frames (at a frame rate of 43 Hz, i. e., roughly 4.7 seconds), three bins per semitone from C1 to B7 (leading to 252 bins), and five harmonics (with frequency multiples of [0.5, 1, 2, 3, 4]). The model architecture is adapted from [226] and receives an HCQT input patch, processes it through several convolution and max-pooling layers, and outputs a single  $\ell_2$ -normalized vector (length 128) per input. We take this output as the embedding vector for the center frame of the input patch. In total, the architecture has roughly 1.5 million learnable parameters. We train our network for 200 epochs (with 16 000 triples randomly sampled per epoch) using the Adam optimizer with a learning rate of 0.002. In the interest of reproducibility, we release code and trained models for our approach.<sup>27</sup>

In line with previous studies on audio representation learning [175, 191, 205], we apply a number of augmentations to excerpts during training, including time scaling, pitch shifting, random masking, adding noise and applying random equalization. For all experiments, we set  $\tau^p = 0.2$  s. With this, the maximal distance between anchor and positive excerpt is in the same order of magnitude as the typical alignment inaccuracy between versions. We further set  $\tau^n = 10.0$  s and  $\alpha = 1.0$ . We found that results are stable for a broad range of settings of these parameters.

### 6.4.3 Baselines

To investigate the musical properties captured by the representation learning approaches CV and SV, we compare them to several optimistic baselines: First, we extract traditional music audio features. We use mel-frequency cepstral coefficients (MFCC), which are known to capture aspects of instrumentation [202], and Chroma features, which contain the dominant pitch-classes in the recording. Here, our goal is not to outperform MFCC or Chroma, but to compare them to our learned representations. If our learning

---

<sup>27</sup> <https://www.audiolabs-erlangen.de/resources/MIR/2023-ISMIR-CrossVersionLearning>

approaches capture instrumentation, we expect them to behave similar to MFCCs. Likewise, in case they contain pitch-class information, we expect them to perform like Chroma features.

Second, we consider a supervised learning approach `Sup` where we train a model on instrument activity annotations and use its hidden representations as features. For this, we utilize the same model architecture as for `CV` and `SV` and only add a final dense layer with a number of outputs equal to the number of instruments to detect. Rather than using the triplet loss from Section 6.3, we train this approach by applying a sigmoid activation and binary cross-entropy loss. Note that in contrast to `CV` and `SV`, the `Sup` approach requires instrument activity annotations for the recordings in the training set.

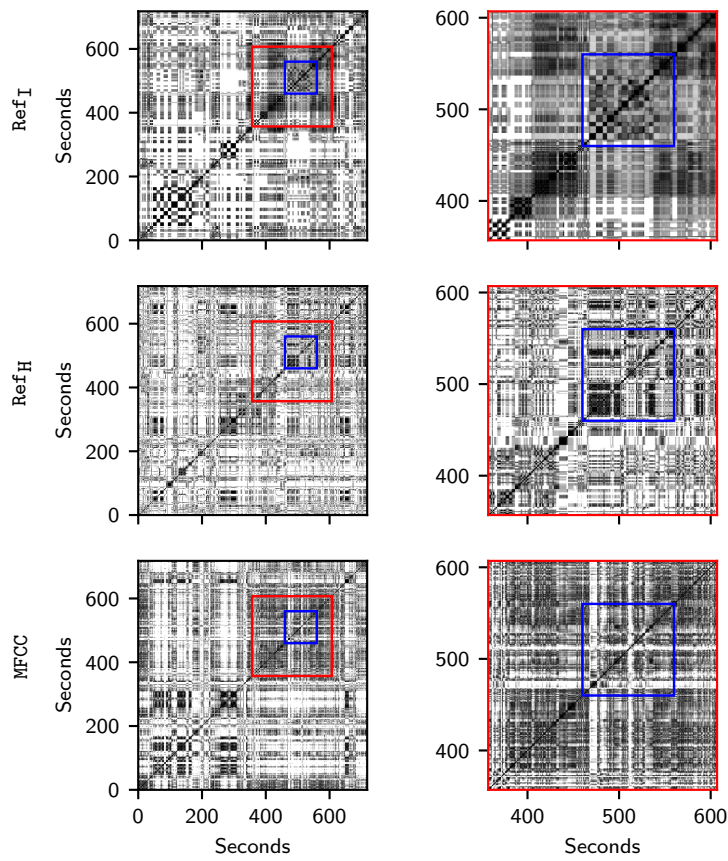
## 6.5 Results

### 6.5.1 Feature Analysis using Self-Similarity

In order to visualize and compare the representations learned by different techniques, we employ self-similarity matrices. Such matrices are commonly used for music structure analysis and allow for visualizing structures based on repetition and homogeneity in feature sequences [145]. Here, we use them to analyze our learned representations without the need for additional fine-tuning. This also allows us to directly compare approaches trained with a fixed instrument vocabulary (`Sup`) to others that are not informed about instruments. We provide an alternative evaluation in Section 6.5.4.

Given a sequence  $X = (x_1, \dots, x_N)$  of (learned) representations of  $N$  audio frames, we construct the corresponding self-similarity matrix  $S \in \mathbb{R}^{N \times N}$  as follows. We first normalize all representations with respect to the  $\ell_2$ -norm, yielding  $\tilde{X} = (\tilde{x}_1, \dots, \tilde{x}_N)$ . We then compute  $S(n, m) := \langle \tilde{x}_n, \tilde{x}_m \rangle$  for  $n, m \in [1 : N]$ . Thus,  $S$  contains the cosine similarities between elements of  $X$ , and all its entries lie in the interval  $[-1, 1]$ . By definition, all entries on the diagonal of  $S$  are equal to 1. In addition, repeated subsequences appear as path-like structures and homogeneous segments appear as block-like structures, see also [145].

We compare the self-similarity matrices obtained from learned representations to matrices created using reference annotations. First, we represent an instrument activity annotation as a sequence of multi-hot binary vectors (indicating the presence of instruments in different frames). By normalizing and computing the dot product as before, we obtain a matrix corresponding to instrument texture, where blocks indicate segments with similar instrumentation. We will refer to this matrix using the shorthand  $\text{Ref}_I$ . For example, the start of the middle measure in Figure ?? would be encoded as a vector  $(1, 1, 1)^\top$ , i. e., all instruments are active, and the end of that measure would be encoded as  $(1, 1, 0)^\top$ , i. e., only horn and soprano are active. After normalization, the dot product of these vectors is 0.82, indicating similar instrumentation. Analogously, we construct another matrix  $\text{Ref}_H$  from a sequence of pitch-class annotations. This matrix captures regions with similar harmonies and pitches.

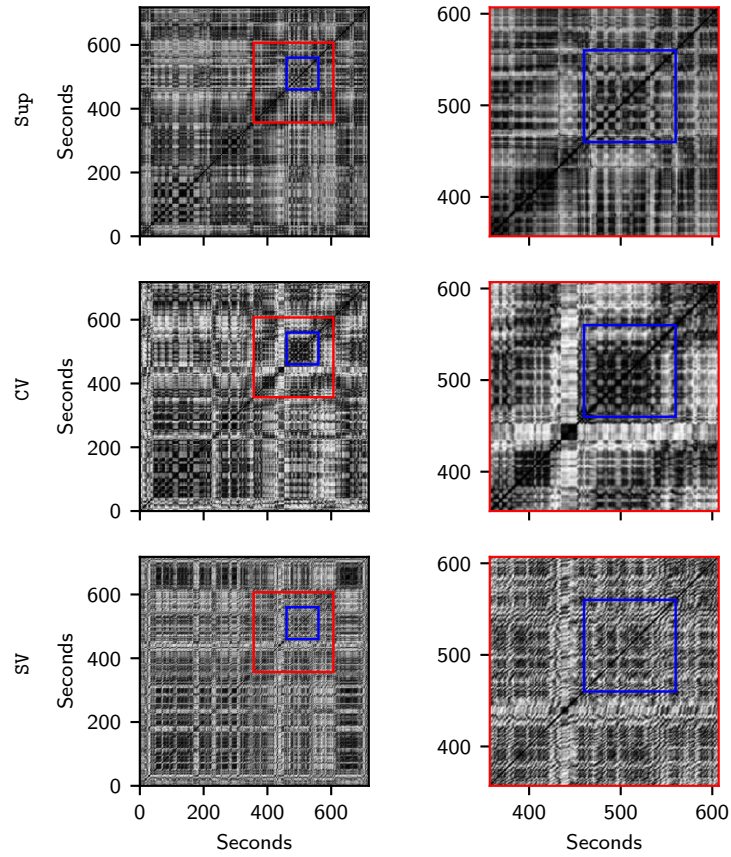


**Figure 6.3:** Self-similarity matrices constructed from instrument annotations ( $\text{Ref}_I$ ), pitch-class annotations ( $\text{Ref}_H$ ), or computed using MFCCs. The right column shows the sections highlighted in red on the left.

## 6.5.2 Qualitative Results

Figures 6.3 and 6.4 show several self-similarity matrices obtained through reference annotations or by different representation learning approaches. The excerpt shown in the left column is the test excerpt from “Die Walküre” (similar results are obtained on other inputs). The right column shows magnified sections from above. Darker color indicates higher similarity. In the  $\text{Ref}_I$  matrix, arising from instrument annotations as explained in Section 6.5.1, one can observe many block and checkerboard-like structures. For example, from seconds 460 to 560, different combinations of woodwind instruments are playing together, creating block and checkerboard-like patterns (highlighted in blue). White areas indicate  $S(n, m) = 0$ , i. e., no common instruments are playing. The matrix  $\text{Ref}_H$ , on the other hand, indicates harmonic similarities which are mostly distinct from the instrument similarities in  $\text{Ref}_I$ .

For the Sup system, many of the patterns in  $\text{Ref}_I$  are replicated, albeit with less detail. This is expected, since this system has been trained on the same kind of annotations that have been used to create  $\text{Ref}_I$ . Interestingly, many of the patterns present in the  $\text{Ref}_I$  and Sup matrices also appear for the proposed



**Figure 6.4:** Self-similarity matrices obtained with a supervised learning system (Sup), the proposed cross-version approach (CV), and a baseline that does not incorporate cross-version information (SV).

approach CV, which has not been trained using instrument annotations. In particular, the checkerboard pattern starting at second 460 is captured by CV, as well as many block structures.

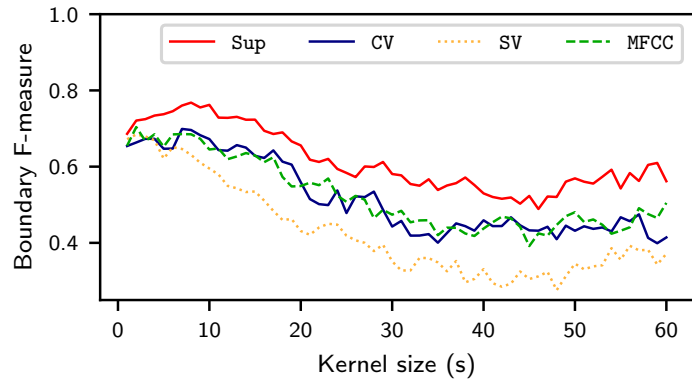
There are fewer similarities between CV and  $\text{Ref}_H$ , indicating that the CV representations are more likely to capture instrumentation rather than pitch-class content. This behavior is encouraged by our augmentation strategy, where we randomly pitch-shift the anchor, positive and negative excerpts.

The matrix obtained through the SV approach is blurry and, unlike the results for CV, fails to capture many of the checkerboard-like patterns present in  $\text{Ref}_I$ . The example suggests that exploiting cross-version information during training is important for capturing aspects of instrumentation in learned representations.

### 6.5.3 Quantitative Results

In order to quantify the correlation between our learned representations and instrument texture, we now apply a procedure for detecting the boundaries of block-like structures in self-similarity matrices. We then

**Figure 6.5:** Results for different representation learning approaches when comparing estimated structure boundaries to boundaries from  $\text{Ref}_I$ .



compare block boundaries estimated on  $\text{Ref}_I$  with boundaries from all other matrices. Such procedures are often used in the context of music structure analysis [54, 145].

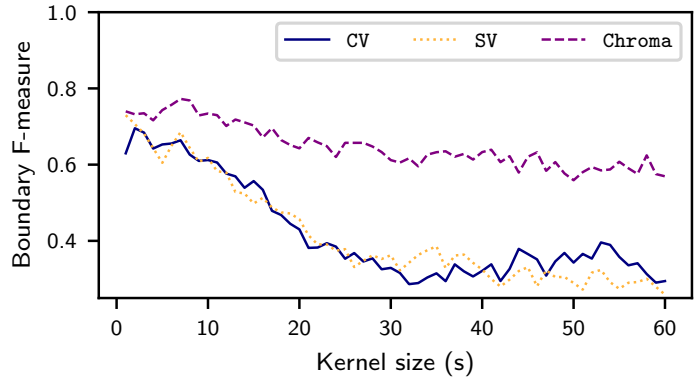
To detect block boundaries, we first correlate a self-similarity matrix with a checkerboard kernel along the main diagonal, as proposed in [54]. From this, we obtain a novelty curve. We then apply a peak picking procedure using local thresholding on this novelty curve, yielding sparse positions of detected block structures. We do this for all approaches and reference matrices. We finally compare—with a tolerance of up to three seconds—the detected boundaries for all approaches to those of  $\text{Ref}_I$ , yielding a boundary F-measure. By adjusting the size of the checkerboard kernel in this procedure, we can identify changes of instrument texture on short or larger time scales. For more details on the boundary detection, peak picking, and evaluation procedure, we refer to [145].

Figure 6.5 shows the results of our quantitative evaluation for different sizes of the checkerboard kernel. The F-measures given are averaged over all recordings in the test dataset. We observe that the supervised approach is best at capturing instrument texture (as encoded by  $\text{Ref}_I$ ) compared to all others, with the highest F-measure of 0.77 for a kernel of eight seconds. CV and MFCC perform roughly on par. This is surprising, since CV is trained without any instrument annotations, while MFCC is known to capture instrumentation. Results for SV deteriorate with larger kernel sizes, dropping to as low as 0.28 F-measure for a kernel of 48 seconds. The proposed approach CV is better at capturing instrument texture than the alternative SV that does not utilize cross-version information.

To examine whether our representations capture information related with harmonies and pitches played, we perform the same evaluation procedure with boundaries from  $\text{Ref}_H$  (see Figure 6.6). We obtain low F-measures for both CV and SV (dropping below 0.4 for kernel sizes above 20 seconds for both approaches). In particular, while we observe an advantage of CV over SV for capturing instrumentation, there is no such advantage with regard to pitch-classes. Additionally, standard Chroma features are clearly superior at capturing the structures in  $\text{Ref}_H$ . We conclude that the representations learned by our proposed approach CV indeed contain information about instrument texture rather than pitch-classes and harmonies.



**Figure 6.6:** Results for comparing with  $\text{Ref}_H$ .



Scenario	AP	AUC	Micro Avg.		Macro Avg.	
			F1	S	F1	S
MFCC	0.777	0.780	0.600	0.890	0.450	0.847
SV	0.708	0.735	0.590	0.871	0.407	0.820
CV	0.753	0.795	0.657	0.872	0.514	0.835
Sup	0.838	0.881	0.772	0.894	0.714	0.874

**Table 6.2:** Results for different representation learning approaches when performing instrument classification.

#### 6.5.4 Feature Analysis Using Classification

To gain further insights into the information captured by our learned representations, we also perform an indirect evaluation as typically done in representation learning. Previous studies often rely on training small classifiers on top of learned representations to investigate their usefulness for different downstream tasks [27, 191]. In this section, we complement our self-similarity-based analysis with such a classification-based evaluation strategy.

To this end, we pass individual representation vectors through a small network of dense layers with 128, 64, and 32 hidden units followed by leaky ReLU activations, respectively. The final layer produces outputs for every instrument annotated in our dataset, followed by a sigmoid activation. For each representation learning technique, we train and evaluate such a network using the dataset split as described in Section 6.4.1. Concretely, we minimize the mean binary cross-entropy loss over all instrument classes on the training set, using stochastic gradient descent with a learning rate of  $10^{-4}$  for 10 epochs. We finally evaluate the classification results on the test set using standard metrics, including ranking-based average precision (AP), mean area under the ROC curves (AUC), F-measure (F1), and specificity (S). For F1 and S, we threshold the predicted probabilities at 0.5 and compute both micro and macro averages of the evaluation scores, where the macro average is not affected by imbalance among instrument classes.

The results of this experiment are shown in Table 6.2. We observe similar trends as in our self-similarity-based evaluation. As expected, the supervised baseline again yields best results. Our proposed cross-version approach CV clearly outperforms the traditional SV across all metrics (e. g.,  $\text{AP}=0.753$  as opposed to 0.708

Scenario			Micro Avg.		Macro Avg.	
	AP	AUC	F1	S	F1	S
Chroma	0.802	0.854	0.591	0.964	0.586	0.963
SV	0.427	0.568	0.001	1.000	0.001	1.000
CV	0.430	0.584	0.021	0.994	0.018	0.994
Sup	0.457	0.612	0.137	0.959	0.122	0.958

**Table 6.3:** Results for pitch-class classification using the learned representations.

for SV). Furthermore, CV even improves upon the optimistic MFCC baseline in terms of AUC and F-measure (e. g., micro F1 = 0.657 instead of 0.600 for MFCC). Finally, SV performs worse than MFCC. Overall, the representations learned by our proposed approach CV are more effective for instrument classification compared to the standard SV approach that does not utilize cross-version information.

We repeat this experiment using pitch-classes as the classification targets instead of instruments. Table 6.3 shows the results of the modified experiment, which are inline with our conclusions from previous sections. Standard Chroma features strongly outperform all learned representations on this task. We conclude that our proposed approach captures instrumentation rather than pitches.

## 6.6 Conclusion

In this chapter, we described a novel audio representation learning approach for cross-version music data and investigated its application to orchestral music. Our approach utilizes the correspondences between different versions of the same musical work. We showed qualitatively and quantitatively that the representations learned by our approach capture aspects of instrumentation. We outperform a standard training strategy that relies on temporal proximity alone.

In contrast to the models presented in previous chapters, the approach proposed here does not require aligned activity annotations for training and relies solely on correspondences between versions.

Our approach can be applied to any kind of cross-version music dataset where alignments between versions can be obtained using standard music synchronization techniques. Future work may apply our approach to other musical scenarios and larger datasets, explore more complex feature extraction networks, investigate alternatives to our triplet loss formulation, or apply the learned representations in the context of different downstream tasks (such as structure analysis). One may also study the impact of design choices such as  $\tau^p$  and  $\tau^n$ , the pitch shifting augmentation, or the number of versions used for training.

## 7 Soft Dynamic Time Warping for Pitch Activity Detection

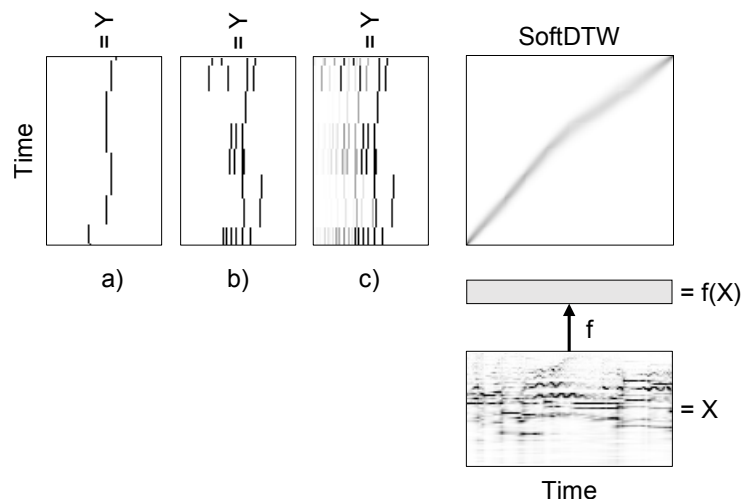
This chapter is based on [109]. The first author Michael Krause is the main contributor to this article. In collaboration with his supervisor Meinard Müller and Christof Weiß, he devised the ideas, developed the formalization, designed the experiments, and wrote the paper. Furthermore, Michael Krause implemented all approaches, using a code base by Christof Weiß, and conducted the experiments.

Many tasks in MIR involve weakly aligned data, where exact temporal correspondences are unknown. The connectionist temporal classification (CTC) loss is a standard technique to learn feature representations based on weakly aligned training data. However, CTC is limited to discrete-valued target sequences and can be difficult to extend to multi-label problems. In this chapter, we show how SoftDTW, a differentiable variant of classical dynamic time warping (DTW), can be used as an alternative to CTC. Using multi-pitch activity detection—commonly known as multi-pitch estimation (MPE)—as an example scenario, we show that SoftDTW yields results on par with a state-of-the-art multi-label extension of CTC. In addition to being more elegant in terms of its algorithmic formulation, SoftDTW naturally extends to real-valued target sequences.

### 7.1 Introduction

Many applications in MIR require alignments between sequences of music data. Often, the sequences given are only weakly aligned. For example, in audio-to-score transcription, pairs of audio and score excerpts are easy to find but exact temporal correspondences between these pairs are hard to establish [221]. Furthermore, music data sequences may involve different levels of complexity. For instance, given a single-instrument monophonic music recording, monophonic pitch estimation [16] aims at detecting a single pitch value per time step (thus, the task may also be described as “pitch activity detection”, see also Figure 7.1a). Other scenarios with discrete, single-label targets include lyrics transcription or lyrics alignment for songs with a single singer [187, 217]. More complex sequences appear in MPE, where multiple pitches may be active simultaneously (Figure 7.1b). Finally, some scenarios involve alignment

**Figure 7.1:** Illustration of SoftDTW for aligning a learned feature sequence  $f(X)$  and a target sequence  $Y$ , where one may consider (a) single-label, (b) multi-label, or (c) real-valued targets.



between real-valued sequences (Figure 7.1c), e. g., audio–audio synchronization [40, 51] or multi-modal alignment problems such as synchronizing dance videos with music [207].

The CTC [64] loss, a fully differentiable loss function initially developed for speech recognition, is commonly used for learning features from weakly aligned data when the targets are sequences over a finite alphabet of labels. Recently, CTC was extended to handle multi-label learning problems [229], where the main idea was to locally transform the multi-label into the single-label case. However, in addition to its complicated algorithmic formulation, this approach is unsuitable for target sequences that do not originate from a discrete vocabulary.

A common technique used in MIR for finding an optimal alignment between weakly aligned sequences is DTW in combination with hand-crafted features [145]. Such a pipeline can provide good alignment results for tasks like audio–audio synchronization [51], but the standard DTW-based cost function is not fully differentiable, which prevents its use in an end-to-end deep learning context. To resolve this issue, Cuturi and Blondel [35] proposed a differentiable variant of DTW, called SoftDTW, that approximates the original DTW cost. In recent work, SoftDTW and related techniques have been successfully used in computer vision applications such as action alignment [29, 69]. To our knowledge, the only prior work applying SoftDTW in an MIR context is by Agrawal et al. [4].

Our contributions are as follows: We demonstrate the use of SoftDTW for MPE. In particular, we show that SoftDTW performs on par with a multi-label extension of CTC, while being conceptually simpler. Furthermore, we show that the SoftDTW approach naturally generalizes to real-valued target sequences, as illustrated in Figure 7.1, making it applicable for a wide range of alignment tasks.

The remainder of the chapter is structured as follows: In Section 7.2, we review the current state of the art for multi-pitch estimation from weakly aligned data with CTC. In Section 7.3, we formalize SoftDTW for general sequences and, in Section 7.4, apply it for MPE. Section 7.5 demonstrates the potential of

SoftDTW for learning with real-valued targets. Finally, Section 7.6 concludes the chapter with an outlook towards future applications.

## 7.2 Weakly Aligned Training for MPE

In recent years, automated music transcription has become a central topic in MIR research, with deep learning techniques achieving state-of-the-art results [30, 73, 93]. We here focus on MPE as a sub-problem of automated music transcription, where the goal is to transform an input music recording  $X$  into a piano-roll representation  $Y$  of pitches played. In particular, multiple pitches may be active at the same time. Most learning-based approaches for MPE require strongly aligned data for training, i. e., pitches are annotated for each audio frame of the input recording. Since annotating data in such a frame-wise fashion is very time consuming, most MPE datasets have been generated (semi-)automatically, e. g., by using MIDI pianos or by applying score–audio synchronization techniques (which may introduce labeling errors). Techniques that allow learning from pairs of  $X$  and  $Y$  that are not temporally aligned are therefore highly desirable.

As discussed in the introduction, a common technique for dealing with weakly aligned learning problems is CTC [64]. Here, the target sequences  $Y$  consist of symbols from a discrete alphabet  $L$ , including a special blank symbol necessary for distinguishing repetitions of symbols. For each frame in the input sequence  $X$ , a neural network outputs a probability distribution over  $L$ . The CTC loss then corresponds to the likelihood of  $Y$  given these network outputs, taking into account all possible alignments between  $X$  and  $Y$ . Note that CTC is agnostic about the durations of symbols in  $Y$ , i. e., even if information about symbol durations is available, CTC is unable to exploit this for alignment. An efficient dynamic programming (DP) algorithm for computing the CTC loss exists (with time complexity  $\mathcal{O}(|L|^2 \cdot N)$ , where  $N$  is the length of  $X$ ), but it requires special care in handling the blank symbol [64].

A naive extension of CTC towards multi-label target sequences would introduce unique network outputs for all possible symbol combinations, which leads to a combinatorial explosion. Instead, the authors in [229] propose to locally reduce the multi-label to the single-label case by only considering those symbol combinations that occur within a single training batch (called multi-label CTC, i. e., MCTC). This defines a “batch-dependent alphabet,” avoiding the combinatorial explosion. The technical details of this process are tricky and special care needs to be taken for handling the blank symbol. In [221], this idea is adapted for MPE by considering pitches as symbols and multi-pitch annotations as combinations of symbols. This formulation allows them to train networks for MPE on pairs of  $X$  and  $Y$  that are only weakly aligned, e. g., where  $X$  is a music recording and  $Y$  is a MIDI representation derived from the corresponding score. In this chapter, using MPE from [221] as an example application, we show how the technically intricate MCTC can be replaced by a conceptually more elegant SoftDTW approach. SoftDTW does not involve the need for a blank symbol, which may be well-motivated in text applications but can be unnatural in MIR problems such as MPE.

### 7.3 Soft Dynamic Time Warping

The objective of DTW is to find an optimal temporal alignment between two sequences. SoftDTW [35] is a differentiable approximation of DTW that allows for propagating gradients through the alignment procedure, making SoftDTW applicable for deep learning. Like classical DTW, SoftDTW admits an efficient DP recursion for computing the optimal alignment cost. Furthermore, there also exists a DP-algorithm for efficiently computing the gradient of that cost. In this section, we briefly summarize the problem statement and DP recursion of SoftDTW for general sequences. We then apply this to our music scenarios in later sections.

Consider two sequences  $X = (x_1, x_2, \dots, x_N)$  and  $Y = (y_1, y_2, \dots, y_M)$  of lengths  $N, M \in \mathbb{N}$  with elements coming from some feature spaces  $\mathcal{F}_1, \mathcal{F}_2$  (i. e.,  $x_n \in \mathcal{F}_1, y_m \in \mathcal{F}_2$  for all  $n \in [1 : N], m \in [1 : M]$ ). Given some differentiable cost function  $c : \mathcal{F}_1 \times \mathcal{F}_2 \rightarrow \mathbb{R}$  defined on these feature spaces, we can construct a matrix  $C \in \mathbb{R}^{N \times M}$  of local costs where each entry

$$C(n, m) = c(x_n, y_m)$$

contains the cost of locally aligning  $x_n$  with  $y_m$ . To determine an optimal global alignment<sup>28</sup> between the sequences  $X$  and  $Y$  one computes an accumulated cost matrix  $D^\gamma \in \mathbb{R}^{N \times M}$  using the recursion

$$\begin{aligned} D^\gamma(1, 1) &= C(1, 1), \\ D^\gamma(1, m) &= \sum_{k=1}^m C(1, k), \text{ for } m \in [1 : M], \\ D^\gamma(n, 1) &= \sum_{k=1}^n C(k, 1), \text{ for } n \in [1 : N], \\ D^\gamma(n, m) &= C(n, m) + \mu^\gamma(\{D^\gamma(n-1, m-1), \\ &\quad D^\gamma(n-1, m), D^\gamma(n, m-1)\}), \end{aligned}$$

for  $n \in [2 : N], m \in [2 : M]$ . Here,  $\mu^\gamma$  refers to a differentiable approximation of the minimum function given by

$$\mu^\gamma(S) = -\gamma \log \sum_{s \in S} \exp\left(-\frac{s}{\gamma}\right),$$

where  $S$  is some finite set of real numbers and  $\gamma \in \mathbb{R}^{>0}$  is a temperature parameter that determines the “softness” of the approximation. One can show that  $\mu^\gamma$  is a lower bound of the minimum function [69] and converges towards the true minimum for  $\gamma \rightarrow 0$ . As a consequence,  $D^\gamma$  becomes the accumulated cost matrix from classical DTW for  $\gamma \rightarrow 0$ . Thus, SoftDTW becomes DTW in the limit case.

<sup>28</sup> Subject to some constraints, namely, the first and last elements of both sequences are aligned to each other (boundary constraint), no element is skipped (step-size constraint), and the alignment is monotonous (monotonicity constraint).

After evaluating the SoftDTW recursion, the entry  $\text{DTW}^\gamma(C) = D^\gamma(N, M)$  contains the approximate minimal cost of aligning the sequences  $X$  and  $Y$ , given the local costs  $C$ . A similar recursion exists for computing the gradient of  $\text{DTW}^\gamma(C)$  with regard to any matrix coefficient  $C(n, m)$  for  $n \in [1 : N]$  and  $m \in [1 : M]$  [35, Algorithm 2]. The time and space complexity of the SoftDTW recursion as well as the gradient computation is both  $\mathcal{O}(N \cdot M)$ , which is sufficiently fast for use in deep learning.

Note that SoftDTW requires no prior knowledge of the alignment between  $X$  and  $Y$ , which enables the use of  $\text{DTW}^\gamma(C)$  as a loss function for learning problems with weakly aligned data. Furthermore,  $X$  and  $Y$  can come from arbitrary feature spaces, as long as an appropriate cost function  $c$  can be defined.

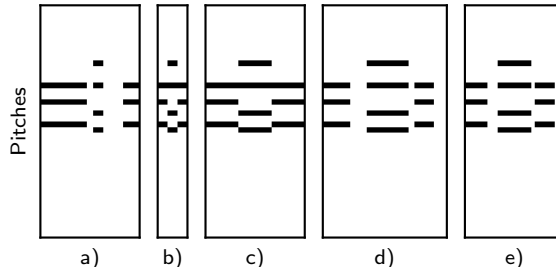
## 7.4 Application to Multi-Pitch Estimation

We now apply SoftDTW to multi-pitch estimation. For a given piece of music, the sequence  $X$  corresponds to some representation of an input recording, while  $Y$  corresponds to a multi-hot encoding of pitches played. Note that  $Y$  does not need to be temporally aligned with  $X$  and could arise, e. g., from a score representation of the musical piece. An element  $y_m$  of the sequence  $Y$  is encoded as a vector  $y_m \in \{0, 1\}^{72}$  and the entries of  $y_m$  correspond to the 72 pitches from C1 to B6. In our experiments, rather than directly aligning  $Y$  with some fixed representation  $X$ , we use a neural network  $f$  that takes  $X$  as input and outputs a feature vector per frame in  $X$ . Thus, we obtain a sequence  $f(X) = (z_1, \dots, z_N)$  with the same length  $N$  as  $X$ . We construct  $f$  such that  $z_n \in \mathbb{R}^{72}$  for the elements  $z_n$  of  $f(X)$ . Thus, both sequences  $Y$  and  $f(X)$  contain elements from the features space  $\mathcal{F}_1 = \mathcal{F}_2 = \mathbb{R}^{72}$ . We then align  $f(X)$  and  $Y$ , as illustrated in Figure 7.1.

To our knowledge, SoftDTW has not previously been used for MPE and is seldom explored in MIR. The authors in [187] used the classical, non-differentiable DTW recursion inside an attention mechanism for lyrics alignment, which led to training instabilities. The work by Agrawal et al. [4] constitutes the first use of SoftDTW for an MIR application. They successfully employ a variant of SoftDTW to train a system for score-audio synchronization. In their scenario, SoftDTW is applied to discrete-valued, one-dimensional, and strongly aligned sequences. In contrast, we employ SoftDTW for multi-dimensional sequences in weakly aligned settings.

### 7.4.1 Implementation Details and Evaluation Metrics

Since the focus of our work is on evaluating the efficacy of SoftDTW for MIR tasks and in order to maintain comparability with the results presented in [221], we adopt the same training setup and network architecture. Thus, we use HCQT [13] excerpts of roughly ten second lengths as input and pass them through a five-layer CNN to obtain a sequence of per-frame representations  $f(X)$  (see [221] for details on the network architecture and HCQT representation).



**Figure 7.2:** (a) Strongly aligned pitch annotations for an audio excerpt, (b) Annotations without note durations (as used by MCTC), (c) Annotations without note durations, stretched to excerpt length, (d) Score representation, not aligned to the audio excerpt, (e) Score representation, stretched to excerpt length

Scenario	F-measure	CS	AP	Acc.
CE [221]	0.70	0.759	0.764	0.546
MCTC [221]	0.69	0.744	0.734	0.532
SoftDTW <sub>W1</sub>	0.00	0.465	0.297	0.002
SoftDTW <sub>W2</sub>	0.69	0.736	0.737	0.529

**Table 7.1:** Results for multi-pitch estimation on the Schubert Winterreise Dataset for SoftDTW compared with MCTC.

We train our networks by minimizing the soft alignment cost  $\text{DTW}^\gamma(C)$ .<sup>29</sup> In all experiments, we use the squared Euclidean distance for  $c$  and set  $\gamma = 10.0$ . We did not see improvements for alternative choices of  $c$  and obtained similar results for a wide range of values for  $\gamma \in [0.5, 20.0]$ . Furthermore, we use a fast GPU implementation of the SoftDTW recursion and gradient computation which was implemented in [128].

To compare network predictions with the strongly aligned pitch annotations of the test sets, we use common evaluation measures for MPE, including cosine similarity between predictions and annotations (CS), area under the precision-recall curve (also called average precision, AP), as well as F-measure and accuracy (Acc., introduced in [160]) at a threshold of 0.4 (which is a common choice in MPE systems, see also [203]).

#### 7.4.2 Comparison with MCTC

We begin by comparing our results with the main results reported in [221], which are obtained on the Schubert Winterreise Dataset (SWD) [225]. SWD provides strongly aligned annotations for all recordings. Due to this, one can consider a baseline trained on the aligned annotations with a per-frame cross-entropy loss (CE). The first line of Table 7.1 shows results for such an optimistic baseline (reprinted from [221]), which yields an F-measure of 0.70 and  $\text{AP} = 0.764$ . To train a network using MCTC instead, one must remove all information about note durations from the label sequence  $Y$  (see Figure 7.2b). The results obtained this way are just slightly lower at  $\text{AP} = 0.734$ , even though only weakly aligned labels are

<sup>29</sup> Note that we normalize  $\text{DTW}^\gamma(C)$  by its value for the first training batch. Thus, the loss is exactly 1 for the first batch and its value range remains similar across training configurations, regardless of the sequence lengths  $N$  and  $M$  or other factors.



Scenario	F-measure	CS	AP	Acc.
SoftDTW <sub>W3</sub>	0.71	0.756	0.755	0.552
SoftDTW <sub>W4</sub>	0.71	0.757	0.750	0.555
SoftDTW <sub>S</sub>	0.72	0.761	0.769	0.563

**Table 7.2:** Results on the Schubert Winterreise Dataset for incorporating note durations with SoftDTW.

used. When performing the same experiment using SoftDTW (denoted by SoftDTW<sub>W1</sub>), we obtain much weaker results with an F-measure of 0.00 and AP = 0.297.<sup>30</sup> In this experiment, the label sequence  $Y$  may be significantly shorter than the learned sequence  $f(X)$ .<sup>31</sup> We repeat the experiment by temporally stretching the sequence  $Y$  to match the number of frames in  $f(X)$  (illustrated in Figure 7.2c). When applying SoftDTW together with this trick (denoted by SoftDTW<sub>W2</sub>), results are again very similar to MCTC (AP = 0.737). Thus, SoftDTW may be used to replace MCTC in this scenario.

### 7.4.3 Incorporating Note Durations

In contrast to MCTC, SoftDTW is able to incorporate (approximate) note durations during training. SWD, for example, contains non-aligned score representations of the pieces performed. We now use these score representations as target sequences  $Y$  (denoted by SoftDTW<sub>W3</sub>, see Figure 7.2d for an illustration). Table 7.2 shows evaluation results, which are slightly improved compared to training without note durations (F-measure of 0.71 compared to 0.69 and CS = 0.756 compared to 0.736 for SoftDTW<sub>W2</sub>). Here, there is only a moderate difference between the lengths of excerpt and label sequence and stretching the label sequence to the length of the input yields nearly identical results (denoted by SoftDTW<sub>W4</sub>, see Figure 7.2e). Finally, we may also use SoftDTW using strongly aligned label sequences (denoted by SoftDTW<sub>S</sub>). In this very optimistic scenario, no alignment is necessary, but SoftDTW may compensate for inaccuracies introduced by the dataset annotation procedures. Indeed, this scenario yields best results (F-measure of 0.72 and AP = 0.769), even slightly improving upon the cross-entropy baseline in Table 7.1.

### 7.4.4 Cross-Dataset Experiment

We also perform a cross-dataset experiment (again following the setup in [221]), where we train on the popular MAESTRO [74] and MusicNet [203] datasets. Both contain strongly aligned pitch annotations for the training recordings, but they do not provide non-aligned score representations of the pieces, so SoftDTW<sub>W3</sub> and SoftDTW<sub>W4</sub> are not applicable here. We then evaluate on the four smaller datasets SWD, Bach10 [42], TRIOS [56] and Phenix Anechoic [140]. Note that the latter three datasets each contain less than ten minutes of audio. This is a difficult scenario since some styles and instruments in the test

<sup>30</sup> Note that the F-measure and Accuracy scores can be improved to 0.32 and 0.20, respectively, by choosing a more suitable detection threshold. Still, these scores are notably worse compared to the results for MCTC.

<sup>31</sup> A large discrepancy in sequence lengths is well known to cause problems for classical DTW. Further investigation is needed to understand how this affects the training process with SoftDTW.

Scenario	AP			
	SWD	Bach10	TRIOS	Phenicx
<b>Default network architecture</b>				
CE [221]	0.684	0.864	0.825	0.829
MCTC [221]	0.666	0.861	0.824	0.833
SoftDTW <sub>w2</sub>	0.665	0.835	0.812	0.788
<b>Larger network architecture</b>				
CE [221]	0.701	0.886	0.863	0.846
MCTC [221]	0.677	0.871	0.849	0.850
SoftDTW <sub>w2</sub>	0.682	0.896	0.864	0.838

**Table 7.3:** Results for multi-pitch estimation in a cross-dataset experiment. Here, MAESTRO and MusicNet have been used for training while four different smaller datasets are used for testing.

datasets are not present during training. For example, Phenicx Anechoic contains orchestral instruments, while MAESTRO and MusicNet contain piano and chamber music.

The results of this experiment are given in Table 7.3. Here, MCTC and a cross-entropy baseline perform roughly on par. SoftDTW yields slightly lower results, especially on Phenicx (AP = 0.788 compared to 0.833 for MCTC). Given that this evaluation scenario is harder and the training datasets are larger, we also repeat this experiment with a larger network architecture (increasing the number of channels for all convolutional layers in the network). The resulting architecture has roughly 600 000 parameters, compared to 50 000 parameters in the default architecture. Results are shown in the lower half of Table 7.3. Average precision scores improve consistently across all methods and datasets, e. g., AP = 0.896 for SoftDTW on Bach10 compared to 0.835 using the smaller architecture. In particular, SoftDTW now outperforms MCTC on all test datasets except for Phenicx, where the performance gap is now much smaller (AP = 0.838 compared to 0.850 for MCTC).

All in all, we conclude that the results for MCTC and SoftDTW are roughly comparable, even in a challenging cross-dataset evaluation. Thus, MCTC may be replaced with SoftDTW without sacrificing alignment quality. In addition, SoftDTW can generalize to other kinds of target sequences, as discussed in the next section.

## 7.5 Extension to Real-Valued Targets

As explained in Section 7.3, the two sequences  $X$  and  $Y$  that are used as input to SoftDTW may come from arbitrary feature spaces. In order to illustrate the potential of using SoftDTW for learning from arbitrary sequences, we now perform two experiments with real-valued targets, i. e.,  $y_n \in \mathbb{R}^{72}$  for the elements  $y_n$  of  $Y$ . Note that MCTC is unable to handle such a setting.

### 7.5.1 Pitch Estimation with Overtone Model

First, we consider a straightforward extension of MPE, where we transform the binary, multi-hot target vectors of MPE to real-valued vectors by adding energy according to a simple overtone model, see Figure 7.1c. Here, we consider 10 overtones for each active pitch, with amplitude  $(1/3)^n$  for the  $n$ -th overtone. As a baseline utilizing strongly aligned labels, we compare with a model trained using an  $\ell_2$  regression loss at each frame (similar to the cross-entropy baseline in Section 7.4). To evaluate, we use the cosine similarity CS between network outputs and annotations. Note that other MPE evaluation metrics are not applicable for real-valued vectors.

When performing this experiment on the SWD dataset, we obtain  $CS = 0.794$  for per-frame training with strongly aligned labels, which is higher than for MPE on SWD (cf. Table 7.1). Training without strongly aligned labels using  $\text{SoftDTW}_{W_2}$  yields only slightly lower cosine similarities at 0.770. This illustrates that  $\text{SoftDTW}$  also works for settings with real-valued target sequences.

### 7.5.2 Cross-Version Training

Second, as a scenario with more realistic target sequences, we choose  $Y$  to be the CQT representation of another version of the piece played in  $X$ . In this case, the two sequences  $f(X)$  and  $Y$  will not correspond temporally, but  $\text{SoftDTW}$  can be used to find an appropriate alignment during training. We perform this experiment using SWD, which provides multiple versions of the same musical pieces. In particular, we choose one version (OL06) as the target version and train our network using  $\text{SoftDTW}$  to align input excerpts from other versions to excerpts from OL06. Finally, we pass versions unseen during training through the trained network and evaluate against excerpts from OL06 using cosine similarity. As a learning-free baseline, we also compute CS between the original CQT representations of the test recordings and the OL06 representations. To compute the cosine similarities during testing, we use the ground truth alignments between OL06 and all other versions provided by the dataset, but we do not need ground truth alignments during training.

Directly comparing the CQT representations of input version and target yields an average cosine similarity of 0.576. Training (using  $\text{SoftDTW}_{W_3}$ ) yields much higher results at  $CS = 0.720$ . Thus, the network trained using  $\text{SoftDTW}$  is able to produce real-valued outputs that are similar to the target version.

## 7.6 Conclusion

In this chapter, we have considered  $\text{SoftDTW}$  as a tool for dealing with weakly aligned learning problems in MIR, in particular, multi-pitch estimation. We showed that a network trained with  $\text{SoftDTW}$  performs on par with the same network trained using a state-of-the-art multi-label CTC loss. We further demonstrated

that SoftDTW can be used to learn features when the target sequences have real-valued entries—something not possible with CTC.

In future work, SoftDTW may be applied to more diverse MIR tasks, such as lyrics alignment, audio–audio synchronization, or cross-modal learning from unaligned video–audio pairs. Furthermore, one may explore the possibility of combining both strongly aligned and non-aligned data within the same training. All these options are supported by the same algorithmic framework.

## 8 Leitmotif Classification in Operas by Richard Wagner

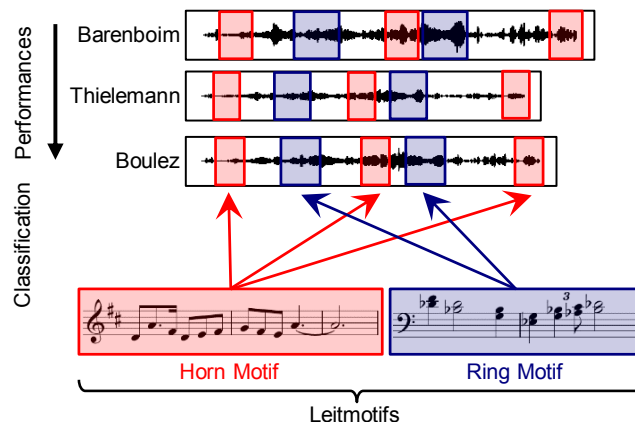
This chapter is based on [104]. The first author Michael Krause is the main contributor to this article. In collaboration with his supervisor Meinard Müller and Frank Zalkow, he devised the ideas, designed the experiments, and wrote the paper. Furthermore, Michael Krause implemented all approaches and conducted the experiments. Julia Zalkow annotated the leitmotif occurrences. Frank Zalkow and Christof Weiß contributed to the dataset preparation.

From the 19th century on, several composers of Western opera made use of *leitmotifs*. As explained in Chapter 1, leitmotifs are short musical patterns used for guiding the audience through the plot and illustrating the events on stage. As such, they refer to semantic entities like characters, places, items, or feelings. A prime example of this compositional technique is Richard Wagner’s four-opera cycle *Der Ring des Nibelungen*. Across its different occurrences in the score, a leitmotif may undergo considerable musical variations. Additionally, the concrete leitmotif instances in an audio recording are subject to acoustic variability. This chapter approaches the task of classifying such leitmotif instances in isolated excerpts of audio recordings (the task of leitmotif activity detection throughout full recordings will be dealt with in the next Chapter 9). As our main contribution, we conduct a case study on our *Ring* dataset with annotations of ten central leitmotifs, leading to 2403 occurrences and 38448 instances in total. We build a neural network classification model and evaluate its ability to generalize across different performances and leitmotif occurrences. Our findings demonstrate the possibilities and limitations of leitmotif classification in audio recordings and pave the way towards the fully automated detection of leitmotifs in music recordings.

### 8.1 Introduction

Music has long been used to accompany storytelling, from Renaissance madrigals to contemporary movie soundtracks. A central compositional method is the association of a certain character, place, item, or feeling with its own musical idea. This technique culminated in 19th century opera where these ideas are

**Figure 8.1:** Illustration of example leitmotifs (red for the Horn motif, blue for the Ring motif) occurring several times in the Ring cycle and across different performances.



denoted as *leitmotifs* [18, 19]. A major example for the use of leitmotifs is Richard Wagner’s tetralogy *Der Ring des Nibelungen*, a cycle of four operas<sup>32</sup> with exceptional duration (a performance lasts up to 15 hours) and a continuous plot spanning all four operas. As many characters or concepts recur throughout the cycle, so do their corresponding leitmotifs. This allows the audience to identify these concepts not only through text or visuals, but also in a musical way. While all these different *occurrences* of a leitmotif in the score share a characteristic musical idea, they can appear in different musical contexts and may vary substantially in compositional aspects such as melody, harmony, key, tempo, rhythm, or instrumentation. When considering recorded *performances*<sup>33</sup> of the *Ring*, another level of variability is introduced due to acoustic conditions and aspects of interpretation such as tempo, timbre, or intonation. In the following, we denote the concrete realization of a leitmotif in an audio recording as an *instance* of the motif. This chapter approaches the problem of classifying such leitmotif instances in audio recordings, as illustrated in Figure 8.1. In particular, we study generalization across occurrences and performances.

Cross-version studies on multiple performances have been conducted regarding the harmonic analysis of Beethoven sonatas [97] or Schubert songs [185], but also for the *Ring* [223, 237]. Beyond harmonic aspects, the *Ring* scenario was considered for capturing audience experience using body sensors and a live annotation procedure [153] or for studying the reliability of measure annotations [222, 238]. Regarding leitmotifs, several works have focused on the human ability to identify motifs [5, 7, 142]. In particular, [143] found that distance of chroma features correlates with difficulty for listeners in identifying leitmotifs. In [237], Zalkow et al. presented a framework for exploring relationships between leitmotif usage and tonal characteristics of the *Ring*.

From a technical perspective, our scenario entails the task of automatically detecting leitmotifs within an audio recording. This chapter represents a first step towards this goal by considering a simplified

<sup>32</sup> While Wagner referred to his works as *music dramas* instead of operas, we choose the more commonly used latter term.

<sup>33</sup> In the current and the following Chapter 9, we use the term “performance” to refer to different recorded versions of the piece. We do not use the term “version” from the previous chapters, because it may lead to confusion when also considering occurrences and instances of motifs.

classification scenario with pre-segmented instances (see Figure 8.1). In the next Chapter 9, we cover leitmotif activity detection over the course of full recordings.

Due to the multiple sources of variability described above, we opt for a data-driven approach. Neural networks have emerged as the dominant classification models. In particular, RNNs are able to handle input sequences of varying length. Our study shows that despite the difficulties of the scenario, an RNN classifier is surprisingly effective in dealing with the variability across occurrences and performances.

The main contributions of our work are as follows: We conduct a case study on classifying leitmotif instances in audio recordings of the *Ring*. To this end, we describe the task of leitmotif classification and provide a dataset of more than 38000 annotated instances within 16 performances of the *Ring* (Section 8.2). We further build an RNN model for classifying leitmotifs in audio recordings (Section 8.3). We carefully evaluate our model with respect to variabilities across performances and leitmotif occurrences over the course of the *Ring*. Moreover, we investigate the effect of adding temporal context and critically discuss the potential limitations and generalization capabilities of our classifier (Section 8.4). Finally, we suggest new research directions that may continue our work (Section 8.5).

## 8.2 Scenario











We now discuss the dataset and leitmotif classification scenario underlying our experiments.

### 8.2.1 Leitmotifs in Wagner’s Ring

While Wagner mentioned the importance of motifs for his compositional process [213], he did not explicitly specify the concrete leitmotifs appearing in the *Ring*. Whether a recurring musical idea constitutes a leitmotif—and how to name it—is a topic of debate even among musicologists, see, e. g., [41] where differences in leitmotif reception are discussed. In line with [237], we follow Julius Burghold’s specification of more than 130 leitmotifs in the *Ring* [214].

For our experiments, we selected ten central motifs frequently occurring throughout the *Ring* (see Table 8.1 for an overview including the number of occurrences per motif). These motifs constitute the classes of our classification task. The selection comprises motifs associated with an item such as the sword (L-Sc), with characters such as the dwarf Mime (L-Mi), or with emotions such as love (L-Ge). All occurrences of these motifs were annotated by a musicologist using a vocal score of the *Ring* as a reference, resulting in 2403 occurrences.

As discussed in Section 8.1, a leitmotif may occur in different shapes over the course of a drama. These musical variations may be necessary to fit the musical context in which the occurrences appear and, thus, be adjusted to the current key, meter, or tempo. Moreover, occurrences of leitmotifs may appear in different registers, musical voices, or instruments. In addition to this, motifs can also occur in abridged or extended

Name (English)	ID	Score	# Occ.	Length	
				Measures	Seconds
Nibelungen (Nibelungs)	L-Ni		536	0.96 ± 0.23	1.72 ± 0.50
Ring (Ring)	L-Ri		286	1.49 ± 0.65	3.64 ± 2.30
Mime (Mime)	L-Mi		242	0.83 ± 0.25	0.87 ± 0.24
Nibelungenhass (Nibelungs' hate)	L-NH		237	0.96 ± 0.17	3.10 ± 1.11
Ritt (Ride)	L-RT		228	0.66 ± 0.17	1.24 ± 0.37
Waldweben (Forest murmurs)	L-Wa		223	1.10 ± 0.30	2.70 ± 0.76
Waberlohe (Swirling blaze)	L-WL		190	1.21 ± 0.39	4.39 ± 1.60
Horn (Horn)	L-Ho		172	1.38 ± 1.05	2.44 ± 1.57
Geschwisterliebe (Siblings' love)	L-Ge		155	1.31 ± 0.83	3.03 ± 2.55
Schwert (Sword)	L-Sc		134	1.89 ± 0.55	3.68 ± 1.88

**Table 8.1:** Overview of the leitmotifs used in this study. Lengths are given as mean and standard deviations over all annotated occurrences (in measures) or instances (in seconds) from all performances given in Figure 8.2.

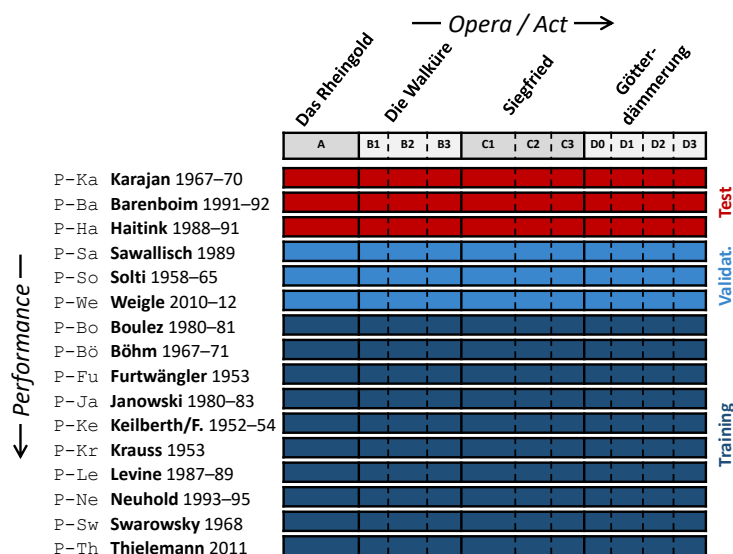
shape, with parts of the motif being repeated, altered, or left out. Despite these diverse musical variations across occurrences, listeners can often identify motifs easily when listening to a performance. This is in line with Wagner's intention of using the motifs as a guideline, thus forming the musical surface of the *Ring* [212].

### 8.2.2 Recorded Performances

As mentioned in the introduction, we do not attempt to classify leitmotifs within a score representation but on the basis of a performance given as an audio recording. To be more concrete, our work relies on 16 recorded performances of the *Ring* that have been used throughout earlier chapters of this thesis, see also



**Figure 8.2:** Structure of Richard Wagner’s *Ring* cycle and overview of 16 recorded performances, see also Section 3.4. Measure positions have been annotated manually for the topmost three performances (P-Ka, P-Ba, and P-Ha), which also constitute the test set in our performance split. The three middle performances (P-Sa, P-So, and P-We) constitute the validation set. All other performances are used for training.



Section 3.4. For three of these performances, the positions of measures from the score were manually annotated in the audio [222]. For the remaining 13 performances, the measure positions were transferred from the manually annotated performances using automatic audio-to-audio synchronization [238]. For convenience, we specify the performances again in Figure 8.2. We automatically located the 2403 leitmotif occurrence regions from the score in each of the 16 recorded performances using linear interpolation between measure positions. This way, we obtained the 38448 instances used for our experiments. The occurrence and instance positions are made publicly available as a dataset for further research.<sup>34</sup>

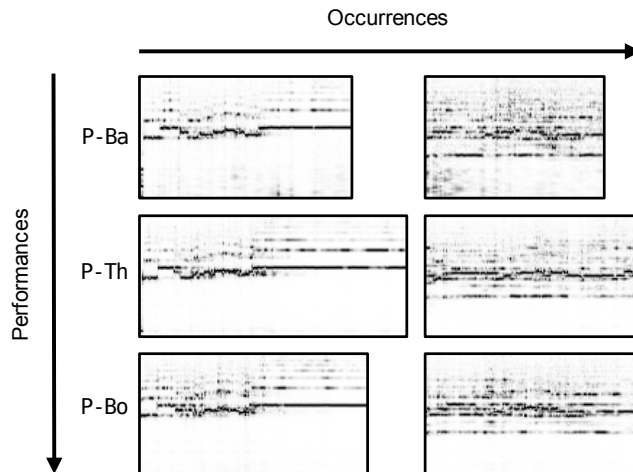
### 8.2.3 Leitmotif Classification Task

In this chapter, we consider the task of leitmotif classification. We define this as the problem of assigning a given audio excerpt to a class according to the occurring leitmotif. Here, we consider ten classes corresponding to the motifs in Table 8.1. We further make the simplifying assumption that only a single leitmotif is played at a time. Thus, we omit excerpts where multiple motifs occur simultaneously (this constraint will be relaxed in the next Chapter 9). Our classification task therefore becomes a multi-class, single-label problem.

Our dataset allows us to approach the leitmotif classification task from two perspectives, each of which incorporates its own types of variabilities. First, the *performance* perspective concerns variabilities across different performances, resulting from different instrumental timbres, tempi, or other decisions made by the artists. Furthermore, this perspective encompasses technical properties such as acoustic, recording, and mastering conditions, which can lead to the so-called album-effect [154]. Second, the compositional or *occurrence* perspective concerns diverse musical variabilities of leitmotif occurrences in the score

<sup>34</sup> <https://www.audiolabs-erlangen.de/resources/MIR/2020-ISMIR-LeitmotifClassification>

**Figure 8.3:** Variability of L-Ho across occurrences and performances. Six instances (two occurrences for three performances) are shown in a CQT representation, which is also used as input to our classification model.



(as discussed in Section 8.2.1). Figure 8.3 shows the Horn motif L-Ho for different performances and occurrences. The variability is evident in different durations of the instances as well as different energy distributions due to other musical sound events being active simultaneously. These variabilities make our classification task a challenging problem. In our experiments, we investigate the generalization across these two perspectives, similar to the study in [185].

### 8.3 Recurrent Neural Network for Leitmotif Classification

Neural networks have previously proven to be useful for classification tasks in the music domain, see, e. g., [90, 99, 165]. As we are dealing with variable length inputs (leitmotif instances may last from less than one to over ten seconds in a performance), RNNs are a natural choice for our scenario.

As input to our system, we take audio excerpts containing leitmotif instances from our 16 performances of the *Ring*, sampled at 22 050 Hz. These excerpts are processed by a CQT [20, 184] with semitone resolution over six octaves and a hop length of 512 samples, where we adjust for tuning deviations (estimated automatically per performance and opera act). These steps are implemented using *librosa*. Finally, all CQT frames are normalized using the max-norm and the resulting representations serve as inputs to our network.

Table 8.2 gives an overview of the network structure. We use an RNN-variant, the long short-term memory (LSTM) proposed in [80]. We stack multiple LSTM layers and, after the final LSTM output, append batch normalization [87] as well as a single fully connected classification layer to obtain leitmotif predictions. We set the number of LSTM layers and the size of their internal representation to 3 and 128, respectively. We train this network for 50 epochs by minimizing the cross-entropy loss between predictions and correct classes using the Adam optimizer [94] with a learning rate of 0.001 on mini-batches of 32 excerpts. Since the excerpts in a batch may have different lengths, we need to zero-pad them to the

Layer	Output Shape	Parameters
Input	(V, 84)	
LSTM	(V, 84)	109056
LSTM	(V, 128)	131584
LSTM	(V, 128)	131584
Take last	(128)	
Batch normalization	(128)	512
Dense	(10)	1290
Output: Softmax	(10)	

**Table 8.2:** Architecture of our RNN for leitmotif classification. V indicates variable length.

maximum number of frames among excerpts in that batch. During computation, we then use masking to ignore zeros added to shorter inputs. We further avoid overfitting by selecting the weights of the epoch that yields the highest mean F-measure on the validation set (as described in Section 8.4.2). The network is implemented in Python using Tensorflow.

## 8.4 Experiments

### 8.4.1 Setup and Splits

We follow the common machine learning approach of partitioning our dataset into training, validation, and test subsets to train, tune hyperparameters, and estimate the results on unseen samples, respectively. In contrast to standard procedures, we partition the data according to musical aspects as motivated in Section 8.2.3. We will consider two splits: the performance and occurrence splits.

For the performance split<sup>35</sup>, we select the three recordings with manually annotated measure positions (P-Ba, P-Ha and P-Ka, see Figure 8.2) for the test set and three performances with automatically transferred measure positions for the validation set (P-Sa, P-So and P-We). The remaining ten performances are used for training. In this split, all subsets comprise all occurrences of all motifs. Results on the performance split are given in Section 8.4.3.

In contrast, for the occurrence split, we randomly choose 80% of the occurrences for training and 10% each for the validation and test set.<sup>36</sup> We further ensure that the proportions of occurrences for each motif is the same in all subsets. In this split, each subset contains all instances of the occurrences in that subset. Results on the occurrence split are given in Section 8.4.4.

<sup>35</sup> Such a split is referred to as version split in Section 3.4.

<sup>36</sup> The same occurrences are chosen in all experiments for comparability.

Context	<i>Strict</i>			<i>Variable</i>			<i>Fixed (10 sec.)</i>		
	P	R	F	P	R	F	P	R	F
L-Ni	0.94	0.95	0.94	0.90	0.95	0.92	0.93	0.93	0.93
L-Ri	0.93	0.92	0.93	0.84	0.93	0.88	0.86	0.89	0.87
L-Mi	0.96	0.95	0.96	0.95	0.93	0.94	0.92	0.98	0.95
L-NH	0.94	0.92	0.93	0.96	0.88	0.92	0.97	0.87	0.92
L-RT	0.95	0.94	0.95	0.94	0.90	0.92	0.96	0.95	0.96
L-Wa	0.94	0.98	0.96	0.98	0.96	0.97	0.96	0.99	0.98
L-WL	0.98	0.93	0.96	0.93	0.93	0.93	0.95	0.94	0.94
L-Ho	0.90	0.89	0.89	0.93	0.85	0.89	0.92	0.91	0.91
L-Ge	0.94	0.94	0.94	0.93	0.91	0.92	0.97	0.94	0.96
L-Sc	0.91	0.96	0.93	0.94	0.89	0.92	0.84	0.86	0.85
Mean	0.94	0.94	0.94	0.93	0.91	0.92	0.93	0.92	0.93

**Table 8.3:** Main results of our method on the test set of the performance split for different strategies of using temporal context.

### 8.4.2 Evaluation Measures

We adopt standard measures from information retrieval for evaluating our models. For a given class (i. e., motif), we treat the classification problem as a retrieval problem, yielding class-dependent precision (P), recall (R), and F-measure (F) as usual, see, e. g., [145].

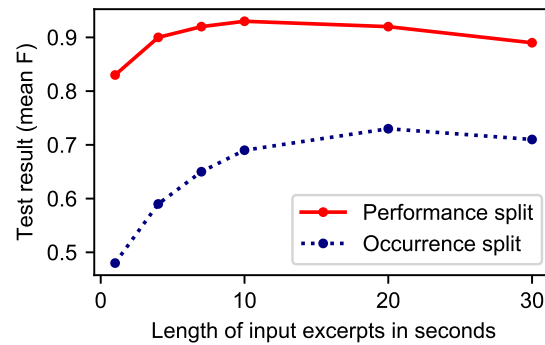
We also report the mean precision, recall, and F-measure over all classes. This gives a general impression of the classification quality. Note that these averages are not affected by class imbalance. Therefore, low results on an infrequent class will influence the mean results as much as low results on a frequent class.

### 8.4.3 Results on the Performance Split

**Basic Experiment.** The left block in Table 8.3 (*Strict*) summarizes results for our model on the test subset of the performance split. We obtain high classification results with a mean F-measure of 0.94. Results are similar across motifs. Highest precision (P = 0.98) is obtained for L-WL, while highest recall (R = 0.98) is reached for L-Wa. Recall and precision per motif are often similar. We conclude that it is indeed possible to classify leitmotif instances in previously unseen performances, provided that all occurrences were seen before in other performances. In the following, we expand on this result by considering other classification and split scenarios.

**Temporal Context.** In our basic experiment, we considered isolated leitmotif instances as input to our classification model, i. e., the audio excerpts to be classified start and end strictly at instance boundaries. We therefore call this the *Strict* scenario. Identifying leitmotifs when instance boundaries are not known in advance could pose an additional challenge. However, the temporal context before and after the instance boundaries might also be helpful in identifying the class of an excerpt. Next, we analyze the effect of temporal context on the leitmotif classification results.

**Figure 8.4:** Mean F-measures for our model when using different input lengths in the *Fixed* scenario.



To this end, we compare the *Strict* scenario with an alternative, called *Variable*, where we add a randomly chosen amount of temporal context to the input excerpts. Context may be added before and after the motif instance. More specifically, the excerpt length is at most doubled and the instance in question is not constrained to be in the excerpt center. Such use of context also prevents our model from relying on length and boundary properties of the leitmotif instances. The center block in Table 8.3 gives the results for this scenario. Compared to the *Strict* case, the mean F-measure decreases slightly to 0.92.

We also perform experiments on fixed input lengths, which we call the *Fixed* scenario. Here, we randomly take subsections of an instance if it is longer than the fixed input length or add context before and after in case it is shorter. Mean F-measure values for different fixed input lengths are shown in Figure 8.4 (solid red line). The plot indicates that results decrease for lengths that are shorter than most instances,<sup>37</sup> e. g., one second. When a fixed length of ten seconds is chosen, which encompasses almost all instances in the dataset, results are comparable to the *Strict* case (see also the right block in Table 8.3). Longer inputs again yield lower results, which may be attributed to the difficulty posed by additional context. However, one should note that for such large durations, input excerpts are no longer guaranteed to contain instances of a single motif only and thus, our initial assumption on a single label per input may be violated.

In Section 8.5, we discuss how the results for different amounts of temporal context may be interpreted in the context of a leitmotif detection scenario.

**Potential for Overfitting.** Deep learning models often rely on features of the input that would be deemed task-irrelevant by human experts, see, e. g., [86, 197]. In our case, the correct class for each input may be inferred not only from musically relevant aspects of leitmotifs such as melody or rhythm (as given in Table 8.1), but also from confounding features of the excerpts such as instrument activity or volume. This is especially true for the performance split, where a classification model may predict correct outputs on the test set by merely memorizing all occurrences during training instead of distinguishing musically relevant features of the leitmotifs (we will revisit this possibility in Section 8.4.6). In contrast, for the

<sup>37</sup> Statistics on instance lengths are given in Table 8.1 (rightmost column).

Context	<i>Strict</i>			<i>Variable</i>			<i>Fixed (10 sec.)</i>		
	P	R	F	P	R	F	P	R	F
L-Ni	0.67	0.80	0.73	0.67	0.86	0.75	0.80	0.91	0.85
L-Ri	0.36	0.41	0.38	0.44	0.43	0.43	0.49	0.67	0.56
L-Mi	0.79	0.87	0.83	0.82	0.80	0.81	0.97	0.96	0.97
L-NH	0.72	0.20	0.31	0.62	0.25	0.36	0.92	0.32	0.47
L-RT	0.57	0.65	0.61	0.60	0.77	0.68	0.71	0.91	0.80
L-Wa	0.87	0.80	0.84	0.81	0.88	0.84	0.95	0.95	0.95
L-WL	0.25	0.21	0.23	0.23	0.17	0.20	0.52	0.20	0.28
L-Ho	0.46	0.57	0.51	0.52	0.57	0.54	0.61	0.91	0.73
L-Ge	0.28	0.30	0.29	0.38	0.43	0.40	0.58	0.68	0.63
L-Sc	0.52	0.50	0.51	0.64	0.53	0.58	0.76	0.58	0.66
Mean	0.55	0.53	0.52	0.57	0.57	0.56	0.73	0.71	0.69

**Table 8.4:** Main results of our method on the test set of the occurrence split for different strategies of using temporal context.

occurrence split, the model needs to generalize to previously unseen realizations of the leitmotif classes and therefore needs to rely on their common musical characteristics.

#### 8.4.4 Results on the Occurrence Split

Table 8.4 presents results for the occurrence split with different strategies for adding temporal context. Overall results are lower than for the performance split. In the *Strict* scenario, the obtained mean F-measure of 0.52 is substantially lower than for the performance split, but still well above chance (which corresponds to 0.1 mean F-measure). Results vary considerably among motifs, with F-measures ranging from 0.23 for L-WL to 0.84 for L-Wa. In addition, the differences between precision and recall per motif can be large as in the case of L-NH ( $P = 0.72$  and  $R = 0.20$ ). We conclude that classifying leitmotif instances for unknown occurrences is challenging but possible.

We further observe that—in contrast to the performance split—context is beneficial in the occurrence split. Mean F-measures of the *Variable* and *Fixed* scenarios increase to 0.56 and 0.69, respectively. Figure 8.4 shows F-measures for different amounts of context in the occurrence split (dotted blue line). Results increase for excerpt lengths up to ten seconds and then stabilize. We see two potential reasons for this. Firstly, by training with temporal context, the classifier may learn to identify features that indicate instance starts and ends, which could be helpful for identifying instances in the test set. Secondly, however, longer temporal context also means that instances from the training set may occur in the context added to validation and test instances. Indeed, we observed that for a context length of 10 seconds, 67% of test excerpts overlap with a training instance of the same class, while 8% overlap with a training instance of another class. Predicting the class of known training occurrences would therefore yield good results on the test set. The results for adding temporal context may thus partly be explained by overfitting to the training set.

Split	Performance			Occurrence		
	P	R	F	P	R	F
Noise	0.90	0.87	0.89	0.32	0.36	0.34
L-Ni	0.90	0.95	0.93	0.63	0.74	0.68
L-Ri	0.89	0.89	0.89	0.28	0.32	0.30
L-Mi	0.94	0.93	0.94	0.78	0.75	0.76
L-NH	0.95	0.88	0.91	0.52	0.28	0.37
L-RT	0.93	0.93	0.93	0.54	0.73	0.63
L-Wa	0.93	0.96	0.94	0.79	0.79	0.79
L-WL	0.94	0.93	0.94	0.17	0.12	0.14
L-Ho	0.89	0.87	0.88	0.40	0.45	0.42
L-Ge	0.91	0.91	0.91	0.20	0.18	0.19
L-Sc	0.90	0.95	0.93	0.68	0.38	0.49
Mean	0.92	0.92	0.92	0.48	0.46	0.46

**Table 8.5:** Results of our method when incorporating a noise class in the performance and the occurrence split. No temporal context is added (*Strict* scenario).

### 8.4.5 Noise Class

So far, we only considered excerpts that contain one of ten leitmotifs. However, the *Ring* also contains regions with other or with no leitmotifs at all. Because of this, we also perform experiments with an additional Noise class, denoting excerpts where none of the leitmotifs in our selection are being played. We evaluate whether our model is able to correctly classify our selection of leitmotifs in the presence of this noise class, both for the performance and the occurrence split. To do so, we randomly select 400 Noise occurrences from the *Ring*, leading to 6400 Noise instances. The model described in Section 8.3 remains unchanged except for the final classification layer, which now has eleven outputs.

Results are given in Table 8.5. For the performance split, the additional noise class does not change results by much. Leitmotif classes obtain somewhat lower results (e. g.,  $P = 0.90$  for L-Ni compared to  $P = 0.94$  in Table 8.3) while the noise class yields an F-measure lower than most leitmotif classes ( $F = 0.89$ ). For the occurrence split, results for the leitmotif classes again decrease slightly (e. g.  $P = 0.63$  for L-Ni compared to  $P = 0.67$  in Table 8.4), while the noise class itself is especially hard to distinguish ( $F = 0.34$ ). In both splits, the noise class does not lead to a complete deterioration of results. Section 8.5 discusses the implications of this for the task of leitmotif detection.

### 8.4.6 Random Labels

In all experiments, our model has consistently obtained higher results on the performance than on the occurrence split. As discussed at the end of Section 8.4.3, the latter split requires generalizing to new musical realizations of a motif. In contrast, the performance split could be tackled by memorizing all leitmotif occurrences, which is not possible on the occurrence split.

To further investigate the gap in results between performance and occurrence split, we now evaluate our model’s capability to memorize input features on the performance split. To do so, we create a variant of the performance split where we assign a random class label from one to ten to each occurrence. Thus, while occurrences are labeled consistently across performances, their classes no longer correspond to leitmotifs. In this variant of the performance split, the class of a test excerpt can only be obtained by memorizing classes for occurrences during training and not by learning common properties of all occurrences for a motif. This random-labeling experiment is inspired by [239].

When training our model on this variant, we obtain a mean F-measure of 0.54 on the test set after 50 epochs, which is much lower than the 0.94 obtained for the original labels (see Table 8.3). We observed that training for this experiment had not converged after 50 epochs and trained for an additional 75 epochs, leading to an F-measure of 0.57. The faster convergence and higher results on the original labels suggest that our model does learn some relevant characteristics of leitmotifs. Our experiment shows, however, that memorizing excerpts may also contribute to the results.

## 8.5 Summary and Future Work

In this chapter, we evaluated the capability of a neural network classification model for identifying leitmotifs in audio excerpts. Despite the complex musical variabilities in this scenario, our RNN-based classification model is able to differentiate between a fixed set of motifs and to distinguish them from non-motif excerpts. Generalization is strong across performances and—to a lesser extent—across occurrences. Using temporal context is helpful in the latter case, although the improvement may partly be the result of overfitting.

Our results encourage the development of a system for automated detection of motif instances in full performances, which we approach in the next Chapter 9. Unlike the classification task, no pre-segmented instance boundaries are available for detection. We therefore expect this to be a more challenging scenario. Additionally, a model used for automated leitmotif detection from audio will also need to handle input excerpts that do not contain any leitmotifs at all. Our experiments with a noise class suggest that this may lead to slightly deteriorated but still useful predictions.

As an even more advanced scenario, one may consider an informed detection setting in which instances of a previously unseen motif must be identified given only a few exemplary instances of that motif.



## 9 Leitmotif Activity Detection in Opera Recordings

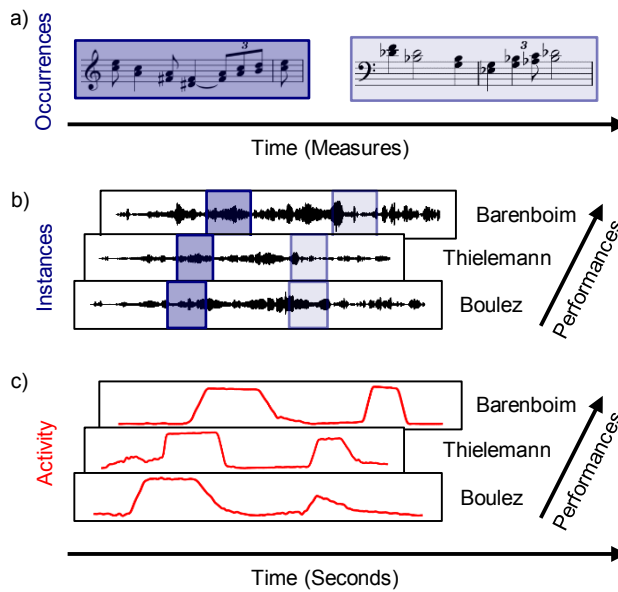
This chapter is based on [105]. The first author Michael Krause is the main contributor to this article. In collaboration with his supervisor Meinard Müller and Christof Weiß, he devised the ideas, developed the formalization, and wrote the paper. Furthermore, Michael Krause implemented all approaches and conducted the experiments.

Building upon the results presented for leitmotif classification in the previous Chapter 8, this chapter approaches the automatic detection of musical patterns in audio recordings with a focus on leitmotifs. The detection of such leitmotifs is a particularly challenging example of a musical sound event detection task, since their appearance can change substantially over the course of a musical work. In our case study, we continue to investigate the *Ring* scenario from previous chapters. Within this scenario, we introduce and formalize the novel task of leitmotif activity detection. Based on our dataset of 200 hours of audio with over 50 000 annotated leitmotif instances, we explore the benefits and limitations of deep learning techniques for detecting leitmotifs. To this end, we adapt two common deep learning strategies based on recurrent and convolutional neural networks, respectively. To investigate the robustness of the trained systems, we test their sensitivity to different modifications of the input. We find that our deep learning systems work well in general but capture confounding factors, such as pitch distributions in leitmotif regions, instead of characteristic musical properties, such as rhythm and melody. Thus, our in-depth analysis demonstrates some challenges that may arise from applying deep learning approaches for detecting complex musical patterns in audio recordings.

### 9.1 Introduction

Within MIR, detecting musical patterns in audio recordings is a fundamental task. These patterns can be characterized by any musical property, including rhythmic phrases, melodic shapes, or harmonic progressions. Across different occurrences, a pattern may vary considerably both in musical aspects and acoustic realization and may appear within different accompanying parts and other musical voices, thus

**Figure 9.1:** Illustration of a leitmotif (here the *Ring* motif L-R1) and its manifestations as (a) leitmotif occurrences in the score, (b) leitmotif instances in several recorded performances (audio), (c) continuous leitmotif activity output by a detection system.



being embedded in varying sound mixtures. In Western music tradition, such patterns play a crucial role for the narration, interpretation, and enrichment of dramatic plots in many genres—from Renaissance madrigals to movie soundtracks. In this context, composers have found creative ways of associating certain characters, places, items, or feelings with specific musical ideas, thus guiding their audience through the story. The use of such compositional techniques culminated in 19th century opera where these ideas became known as *leitmotifs* [18], later adopted by movie soundtracks. A central role is attributed to Richard Wagner’s operas with their extensive usage of leitmotifs. In his theoretical writings, Wagner intended these motifs to be particularly memorable and to guide the listeners through the work [213]. Knowing, rediscovering, and understanding the usage of leitmotifs may therefore enrich the experience of an audience [7] and help musicologists analyze the compositional structure of the works [237]. In this context, automated methods for detecting leitmotifs over the course of an opera (as illustrated by Figure 9.1) are of high interest for various applications such as the augmentation of recorded, virtual, and live performances and may serve commercial, didactic, and musicological research purposes. For instance, an automated leitmotif detection procedure may be used to display leitmotif names alongside a recorded performance of the work, thus enhancing the audience’s experience of the composition.

In this chapter, we study leitmotif detection in the context of Richard Wagner’s four-opera cycle *Der Ring des Nibelungen*, for which a typical performance lasts about 15 hours. To the best of our knowledge, this is the first work dealing with automated leitmotif detection. We explore this task using our novel dataset of the *Ring* involving over 50 000 annotated leitmotif instances. We design two typical deep learning systems for detecting the activity of several leitmotifs in recordings of the *Ring* and investigate their robustness under different modifications of the input, thus simulating different types of musical variability. We find evidence that despite achieving good numerical results on a held-out test set, our models capture confounding factors rather than relying on characteristic musical properties. By analyzing our systems in this complex leitmotif scenario, we aim for a deeper understanding of their properties and explore some of

the challenges that may arise from applying standard deep learning systems for detecting musical patterns in audio recordings.

As discussed in Chapter 8, a leitmotif may be subject to several musical variations across its different *occurrences* in the musical score (see Figure 9.1a), such as transposition, tempo changes, abridgment, prolongation, as well as melodic, harmonic, or rhythmic changes. Due to this variety, systems generally need to be informed about the specific leitmotifs to detect. Possible application scenarios may have different degrees of such side information. In the main scenario considered in this chapter, we have annotations of all *instances* of the relevant leitmotifs (see Figure 9.1b) for a specific recording. Based on this input, a system needs to detect the leitmotifs in other performances.

The remainder of the chapter is organized as follows: In Section 9.2, we introduce the musical scenario of the *Ring*, outline our cross-performance dataset, and formalize the leitmotif activity detection task. In Section 9.3, we summarize related work, outline our deep learning approaches and evaluation procedure, and present first results. In Section 9.4, we analyze our models with regard to different input modifications. Section 9.5 presents an outlook to less-informed scenarios. Section 9.6 summarizes our findings.

## 9.2 Musical Scenario and Task Specification





















This section outlines our musical scenario consisting of Wagner’s *Ring* cycle and its specific use of leitmotifs. We present an overview of our cross-performance dataset and provide a formalization of the leitmotif activity detection task.

### 9.2.1 Leitmotifs in Wagner’s Ring

The scenario of our case study is centered around Richard Wagner’s tetralogy *Der Ring des Nibelungen*, a musical work of extraordinary dimensions. As indicated by Figure 8.2, the *Ring* consists of the four operas *Das Rheingold*, *Die Walküre*, *Siegfried*, and *Götterdämmerung*, spanning a continuous plot. Comprising 21 941 measures, this large work has been considered for several tasks within MIR such as audio-based harmony analysis [237], symbolic pattern search [98], or meta-analyses of audience experience [153]. For organizing this comprehensive material, we consider eleven parts of the *Ring* (first row in Figure 8.2), which usually correspond to acts of individual operas (thus hereafter denoted as *acts*) with continuous measure count in the score.

The *Ring* cycle is well-known for its frequent use of leitmotifs, as we previously discussed in Chapter 8. Most motifs are characterized by their melodic and rhythmic shape but are interwoven into the compositional structure. Therefore, a leitmotif may appear in different musical contexts, thereby varying in compositional aspects (such as melody, harmony, or rhythm) in order to fit the current key, meter, or tempo. [237] explored relationships between leitmotif usage and tonal characteristics of the *Ring*. Beyond that, leitmotifs

Chapter 9. Leitmotif Activity Detection in Opera Recordings

Name (English)	ID	Score	# Occ.	Length	
				Measures	Seconds
Nibelungen (Nibelungs)	L-Ni		562	0.95 ± 0.24	1.72 ± 0.50
Ring (Ring)	L-Ri		297	1.50 ± 0.66	3.77 ± 2.46
Nibelungenhass (Nibelungs' hate)	L-NH		252	0.96 ± 0.17	3.22 ± 1.20
Mime (Mime)	L-Mi		243	0.83 ± 0.25	0.84 ± 0.20
Ritt (Ride)	L-RT		228	0.66 ± 0.17	1.26 ± 0.38
Waldweben (Forest murmurs)	L-Wa		228	1.10 ± 0.30	2.65 ± 0.73
Waberlohe (Swirling blaze)	L-WL		194	1.21 ± 0.39	4.59 ± 1.70
Horn (Horn)	L-Ho		195	1.30 ± 1.02	2.34 ± 1.51
Geschwisterliebe (Siblings' love)	L-Ge		158	1.32 ± 0.84	3.13 ± 2.65
Schwert (Sword)	L-Sc		148	1.88 ± 0.63	3.73 ± 1.99
Jugendkraft (Youthful vigor)	L-Ju		146	1.23 ± 0.57	0.96 ± 0.38
Walhall-b (Valhalla-b)	L-WH		143	1.10 ± 0.47	3.53 ± 2.14
Riesen (Giants)	L-RS		136	0.95 ± 0.39	2.83 ± 1.96
Feuerzauber (Magic fire)	L-Fe		112	1.18 ± 0.40	3.57 ± 1.09
Schicksal (Fate)	L-SK		94	2.02 ± 0.47	8.11 ± 2.64
Unmuth (Upset)	L-Un		92	1.87 ± 0.70	5.85 ± 3.21
Liebe (Love)	L-Li		89	1.78 ± 0.51	5.54 ± 2.47
Siegfried (Siegfried)	L-Si		86	2.88 ± 1.60	8.03 ± 5.46
Mannen (Men)	L-Ma		83	1.15 ± 0.50	1.37 ± 0.70
Vertrag (Contract)	L-Ve		83	2.29 ± 0.65	5.72 ± 2.12

**Table 9.1:** Overview of the 20 leitmotifs used in this study (the first ten of these motifs were used in Chapter 8). Score examples shown are adapted from Wagner [214]. Lengths are given as means and standard deviations over all annotated occurrences (in measures) or instances (in seconds) from all performances given in Figure 8.2. Counts and lengths differ from Chapter 8, because we allow for concurrent motif activity in this study.

may occur in different registers, voices, or instruments, and in abridged or extended versions with parts of the motif being repeated, altered, or left out. Despite these musical variations, listeners can often identify motifs when listening to a performance. This is in line with Wagner’s intention of using the motifs as a guideline and, thus, employing them in a clearly perceivable way [213]. This human ability to identify motifs has been analyzed from a psychological perspective [5, 7, 142].

While Wagner mentioned the importance of such motifs for his compositional process [213], there is no explicit specification of concrete leitmotifs by the composer. Whether a recurring musical idea constitutes a leitmotif or not is topic of debate among musicologists [41]. In line with Chapter 8, we follow the specification of 130 leitmotifs in the *Ring* by Julius Burghold [214]. A musicologist annotated the score-based segments (in measures/beats) for all occurrences of these motifs in the *Ring*. Contiguous repetitions of motifs are considered as individual segments, and abridged, extended, or varied occurrences are also included (with our annotator deciding on the amount of variation that can be considered as the same motif). Since many leitmotifs occur rarely or are musically ambiguous, we pursue a pragmatic approach, restricting ourselves to 20 characteristic and frequent motifs, which are specified in Table 9.1. The motif L-Ho, for example, is associated with the hero Siegfried and is often used as a narrative device. It appears in its full heroic form when Siegfried is first introduced, changes to a diminished chord as the hero is fighting a great beast and is played again as other characters remember him following his demise. In total, our annotations comprise 3569 occurrences of these 20 motifs.

### 9.2.2 Cross-Performance Dataset

In this chapter, we make use of the cross-performance dataset of the *Ring* introduced in Section 3.4. As a reminder, this dataset comprises 16 audio recordings (both live and studio) listed in Figure 8.2. Their duration varies between 13.5 and 15.5 hours. As explained previously, the measure positions were manually annotated in the audio recordings for the performances P-Ka, P-Ba, and P-Ha [222]. For the remaining 13 performances, we made use of an automated transfer of measure positions from the manually annotated performances relying on highly accurate audio–audio synchronization methods [238]. As an indicator for this high accuracy, we analyzed measure positions obtained for one performance (P-Ba) using this transfer procedure and found that they deviate only marginally from the manually annotated measure positions (by 0.137 seconds on average).

Relying on these measure positions, we transferred the 3569 leitmotif occurrence regions from the score to the 16 recorded performances. For leitmotif boundaries not lying on measure boundaries, we used linear interpolation between measure positions. The resulting 57 104 *leitmotif instance regions* in the different recordings (see Figure 9.1) represent the reference annotations for our detection task. We provide our annotations of occurrence and instance positions as a publicly available dataset.<sup>38</sup>

<sup>38</sup> <https://www.audiolabs-erlangen.de/resources/MIR/2021-TISMIR-TowardsLeitmotifDetection>

In the previous Chapter 8, we used these instances (for ten selected motifs) for evaluating a leitmotif classification task, where presegmentation of relevant audio excerpts (containing a leitmotif) is assumed to be given. In this chapter, we aim for detecting the *activity* of the leitmotifs in a continuous fashion (Figure 9.1c) without assuming any presegmentation. In consequence, our detection problem is substantially harder than the classification problem studied in Chapter 8. Moreover, we extend the task to 20 leitmotifs in total.

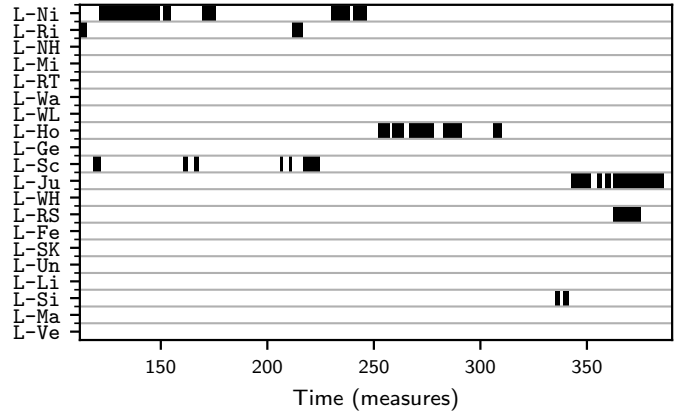
As explained in Chapter 3, musical datasets can be split across different dimensions to systematically test the generalization capabilities of MIR systems. For example, [185] observed differences between systems for detecting local key when generalizing to unknown performances versus unknown songs. For most experiments in this chapter, we make use of a *performance split* (see Figure 8.2 and Section 8.4), using the three recordings with manually annotated measure positions (P-Ka, P-Ba, P-Ha) for testing. The synchronization-based measure transfer may introduce small deviations for the other performances, which may be unproblematic for training but quite relevant for testing purposes. As in the previous chapter, the validation set comprises the performances P-Sa, P-So, and P-We. The remaining ten performances are used for training. In Section 9.5, we report preliminary results for detecting leitmotifs in unknown musical material using an *opera split*.

### 9.2.3 Leitmotif Activity Detection

We now want to formalize the leitmotif activity detection task motivated in the introduction. To this end, we consider a set of leitmotifs  $\mathcal{L}$  that is indexed by  $\ell \in [1 : L] := \{1, 2, \dots, L\}$  with  $L = |\mathcal{L}|$ . In our dataset described in Section 9.2.2, we have  $\mathcal{L} = \{\text{L-Ni}, \text{L-Ho}, \dots\}$  with  $L = 20$ , see Table 9.1. We further consider an audio recording with a discretized time axis given by the index set  $[1 : N]$ . Due to variations in tempo, the time axis  $[1 : N]$  is performance-specific and the value of  $N$  varies between performances of the same act. Then, a leitmotif activity function  $\varphi_\ell$  outputs probabilities for motif  $\ell$  being active at each frame  $n \in [1 : N]$  of a specific performance, thus  $\varphi_\ell : [1 : N] \rightarrow [0, 1]$ .

In our dataset, we consider audio recordings from 16 performances of the eleven acts in the *Ring* (see Figure 8.2). As described in Section 9.2.2, the reference leitmotif annotations are given on a musical time axis specified in measures. For an act with  $S$  measures, we represent our reference annotations as a binary matrix  $\mathcal{A}^{\text{Ref}} \in \mathbb{B}^{L \times M}$ , for  $\mathbb{B} = \{0, 1\}$  and  $M = S \cdot B$  (see Figure 9.2 for an illustration of an excerpt of such a matrix). Here,  $B$  is a discretization factor. Setting  $B = 1$ , we evaluate on the level of whole measures. Setting  $B = 16$ , we subdivide each measure into 16 equidistant sub-segments and evaluate on sixteenth of a measure (e. g., in a 4/4 time signature, each sub-segment would correspond to a 16th note).  $M$  is then the total number of such measure sub-segments in the act and  $m \in [1 : M]$  are indices on our musical time axis. We set  $B = 16$  for all our experiments.  $\mathcal{A}^{\text{Ref}}$  can now be constructed from the annotations by assigning  $\mathcal{A}_{\ell m}^{\text{Ref}} = 1$  if and only if an occurrence of motif  $\ell$  covers measure sub-segment  $m$ .

**Figure 9.2:** Illustration of our ground truth occurrence annotations. Measures 112 to 390 from the first act of *Siegfried* are shown. For instance, L-Ni is active around measure 150, whereas L-SK is never active throughout this excerpt.



In contrast to our reference annotations  $\mathcal{A}^{\text{Ref}}$ , which are defined on the musical time axis  $[1 : M]$  of an act, we define our leitmotif activity functions  $\varphi_\ell$  on the physical time axis  $[1 : N]$  of an audio recording. Therefore, to evaluate a leitmotif activity function, we first transfer its outputs onto a musical time axis by taking the maximum over all outputs for a measure sub-segment. Here, the correspondence between physical and musical time axes is given by our measure annotations refined with linear interpolation. Since  $\varphi_\ell$  has a continuous output, we then use a thresholding procedure (described in Section 9.3.1) to also obtain a binary matrix  $\mathcal{A}^{\text{Est}} \in \mathbb{B}^{L \times M}$ . This matrix can be evaluated against  $\mathcal{A}^{\text{Ref}}$  using standard measures such as precision, recall, and F-measure (see Section 9.3.2).

Evaluating detection results on a musical time axis has two advantages: first, it allows us to quantitatively compare results obtained on different performances (for which the physical time axes might differ, but the musical time axis does not). Second, by defining our evaluation metrics in terms of measure sub-segments, we are able to relate evaluation scores to musical material rather than physical duration (thus, e. g., equally considering faster and slower sections) and to introduce a musically informed tolerance parameter in our evaluation (see Section 9.3.3).

Conceptually, our leitmotif activity detection task can be considered as a special case of sound event detection (SED) as illustrated by Virtanen et al. [210, Fig 8.1d] (see also Section 2.2 for an introduction into SED). For example, the task of environmental sound detection consists of detecting the activity of multiple parallel sound sources within an environmental sound scene. Similarly, multiple different leitmotifs may be active at the same time. However, the activity functions of different environmental sounds are typically independent from each other, i. e., uncorrelated, and from any other sound in the mixture. As opposed to this, we can expect correlations between motif activities.<sup>39</sup> Furthermore, our leitmotifs are not independent of other musical parts (such as accompaniment or other motifs), since all musical parts have to fit into the larger harmonic context. These characteristics distinguish our task from other, more general SED scenarios.

<sup>39</sup> An example of motifs whose occurrences are possibly correlated are the motif for the horn of the hero Siegfried (L-Ho) and the motif for the character himself (L-Si).

Concerning a coarsely related problem, the Music Information Retrieval Evaluation eXchange (MIREX)<sup>40</sup> has run a task on *Discovery of Repeated Themes and Sections*, but this was limited to synthesized audio and prominent themes with little variation. In contrast, we are concerned with real-world orchestral recordings and our leitmotifs may vary considerably or appear within the background accompaniment.

## 9.3 Deep Learning-Based Leitmotif Activity Detection

In this section, we present two approaches to leitmotif activity detection based on neural networks, introduce the evaluation measures used and report first results using our models. For a discussion of existing approaches to sound event detection, we refer to Chapter 2. There, for most of the mentioned tasks, sound events are usually short and only depend on very local context. In contrast, the leitmotif instances considered in this chapter can last several seconds and a detection system must be able to process the appropriate amount of temporal context to identify them. Therefore, to implement a leitmotif activity detection function, an RNN-based approach can be considered appropriate. Such an architecture can, at least in theory, detect entities of arbitrary lengths (such as our leitmotif instances). As described in Section 2.2, convolutional architectures are commonly used for sound event detection, too. As a second approach, we therefore consider a CNN-based system, paying special attention to the appropriate receptive field in time.

### 9.3.1 Methods

We begin by extracting audio excerpts of ten seconds' length (containing leitmotif instances, but also excerpts where none of our motifs occur) from the ten training performances of the *Ring* described in Section 9.2.2. Here, ten second excerpts are long enough to completely cover the full leitmotif instance for nearly all instances in our dataset. For the 3569 leitmotif occurrence regions, we randomly add context before and after the instance in case the motif is shorter than ten seconds or randomly remove parts of the beginning and end of the instance in case it is longer. We further include 4000 examples where no motif occurs.

The audio excerpts are sampled at 22 050 Hz and converted to mono. Subsequently, we process the excerpts by a CQT with twelve semitones per octave from C1 to B7 and a hop length of 512 samples, adjusted for tuning deviations (estimated automatically per performance and opera act). These steps are implemented using *librosa*.<sup>41</sup> We only take the magnitude of the CQT. The resulting CQT frames with a frame rate of 43.1 Hz are then max-normalized individually (in order to obtain normalized network input and achieve some degree of loudness invariance) and used as input to our networks. Both networks process CQT frames and output frame-wise predictions per leitmotif.

---

<sup>40</sup> [https://www.music-ir.org/mirex/wiki/2017:Discovery\\_of\\_Repeated\\_Themes\\_%26\\_Sections](https://www.music-ir.org/mirex/wiki/2017:Discovery_of_Repeated_Themes_%26_Sections)

<sup>41</sup> <https://librosa.org/>



Layer	Output Shape	Parameters
Input	(431, 84)	
LSTM	(431, 128)	109 056
LSTM	(431, 128)	131 584
LSTM	(431, 128)	131 584
Batch normalization	(431, 128)	512
Dense (per frame)	(431, 21)	2 709
Output: Sigmoid	(431, 21)	

**Table 9.2:** Network architecture used for our RNN-based leitmotif activity detection system (adapted from Chapter 8).

### 9.3.1.1 RNN-Based Approach

For our experiments, adapting the approach from Chapter 8, we use the network architecture as specified in Table 9.2. The input consists of 431 CQT frames (obtained from a ten-second audio excerpt), every frame being a vector of 84 CQT bins (one for each semitone in seven octaves), resulting in the input shape (431, 84). The input is processed by three stacked LSTM layers, which are variants of RNN layers designed to be easily trainable [62]. Each LSTM uses 128 units for its internal operations. The third LSTM layer is followed by batch normalization and a dense layer (applied at each frame individually), which outputs one prediction per motif as well as an additional output indicating no motif activity (leading to 21 outputs in total). These predictions (logits) are converted to probabilities through a standard sigmoid activation. Based on these frame-wise outputs, our network models leitmotif activity functions  $\varphi_\ell$  for each motif  $\ell \in \mathcal{L}$ . Since this corresponds to a frame-wise multi-label classification problem, multiple outputs may be activated for the same frame (corresponding to simultaneous motif activity). Moreover, the procedure is causal, meaning that the output at any frame depends only on this frame and the preceding frames. We did not observe improvements for increasing the number of stacked LSTM layers, increasing their number of units, replacing them with gated recurrent unit (GRU) layers, or applying regularization such as weight decay or dropout.

### 9.3.1.2 CNN-Based Approach

As our second network, we consider a convolutional architecture as illustrated in Table 9.3. The input of shape (431, 84) is identical to the RNN input. The subsequent architecture follows the paradigm of stacking convolution and max-pooling operations [62] and is inspired by the network used by [182] for SVD (see Chapter 3). In order to obtain a frame-wise output and a receptive field of appropriate size, we made two adjustments: first, all max-pooling operations have a stride of one in time such that the final output consists of 431 frames (same as the input). Consequently, all layers following the max pooling operations have appropriate dilation factors in time. Second, after the pitch axis has been pooled out, we add one-dimensional convolutions to increase the receptive field in time. Ultimately, the network has a receptive field covering the full pitch axis (all 84 CQT bins) and around 5.5 seconds on the time axis

Layer (Kernel size, (Strides), (Dilations))	Output Shape	Parameters
Input	(431, 84)	
Expand	(431, 84, 1)	
Conv2D (3, 3), (1, 1), (1, 1)	(431, 84, 128)	1 152
Batch normalization	(431, 84, 128)	512
Conv2D (3, 3), (1, 1), (1, 1)	(431, 84, 64)	73 728
Batch normalization	(431, 84, 64)	256
MaxPool2D (3, 3), (1, 3), (1, 1)	(431, 29, 64)	
Conv2D (3, 3), (1, 1), (3, 1)	(431, 29, 128)	73 728
Batch normalization	(431, 29, 128)	512
Conv2D (3, 3), (1, 1), (3, 1)	(431, 29, 64)	73 728
Batch normalization	(431, 29, 64)	256
MaxPool2D (3, 3), (1, 3), (3, 1)	(431, 10, 64)	
Conv2D (3, 3), (1, 1), (9, 1)	(431, 10, 128)	73 728
Batch normalization	(431, 10, 128)	512
Conv2D (3, 3), (1, 1), (9, 1)	(431, 10, 64)	73 728
Batch normalization	(431, 10, 64)	256
MaxPool2D (3, 3), (1, 3), (9, 1)	(431, 4, 64)	
Conv2D (1, 4), (1, 1), (1, 1)	(431, 1, 64)	16 384
Batch normalization	(431, 1, 64)	256
Squeeze	(431, 64)	
Conv1D (3), (1), (27)	(431, 128)	24 576
Batch normalization	(431, 128)	512
Conv1D (3), (1), (27)	(431, 64)	24 576
Batch normalization	(431, 64)	256
MaxPool1D (3), (1), (27)	(431, 64)	
Dense (per frame)	(431, 21)	1 365
Output: Sigmoid	(431, 21)	

**Table 9.3:** Network architecture used for our CNN-based leitmotif activity detection system (inspired by [182]). Note that all operations have stride one in time and pitch, except for MaxPool2D, which has stride three in the pitch direction. Dilation rates in time increase after each max-pooling operation.

(encompassing most motif instances in our dataset, see Table 9.1). All convolutional layers use a leaky rectified linear unit (ReLU) activation function with  $\alpha = 0.2$ . After the final convolution and max-pooling stage, we apply a dense layer at each frame and obtain leitmotif activity functions  $\varphi_\ell$  in the same fashion as the RNN system. Unlike the RNN, however, this system is not causal but operates in a centric fashion, so the output at any frame depends on the frame itself and an equal number of preceding and subsequent frames.

### 9.3.1.3 Training and Post-Processing

We consider both networks as representatives for their respective architectural paradigms (recurrent vs. convolutional). Thus, we abstain from proposing complicated improvement strategies to either model. For the same reason, we take care to keep the number of parameters in the same order of magnitude (375 445

for the RNN and 440 021 for the CNN). This allows us to attribute any differences in network behavior to the architectural paradigms rather than the network size.

We train the networks by minimizing the average binary cross-entropy loss between predicted probabilities and correct labels at all frames using the Adam optimizer with a learning rate of 0.002 on mini-batches of 32 excerpts. We use the validation loss as a monitor for early stopping. After 30 epochs without decreasing loss, we reset the weights to the optimal epoch. These operations are implemented in Python using Tensorflow 2.<sup>42</sup>

After training, we obtain leitmotif activity predictions by pre-processing the test recordings and passing the resulting CQT frames through the model (from start to finish, i. e., including parts not containing leitmotifs). Essentially, the network layers are operating on entire test recordings, without restrictions due to their input shape (431, 84). For the RNN-based model, this is achieved by passing on the internal LSTM states from frame to frame. Regarding the CNN-based model, we apply it on overlapping chunks of the test recordings with the overlap equal to its receptive field in time. This way, we can obtain predictions that are not affected by zero padding at the input edges. This yields the frame-wise activity functions  $\varphi_\ell$  for each  $\ell$ . Then, we post-process  $\varphi_\ell$  using a median filter of length 0.5 seconds (applied in a centric fashion). Median filtering removes outliers (such as gaps and spikes) from  $\varphi_\ell$  that are much shorter than the typical length of a leitmotif instance (see Table 9.1). Such a post-processing step is common for other detection procedures, e. g., for detecting singing voice [182]. Finally, we apply binarization with an individual binarization threshold per motif (tuned to maximize motif F-measure on the validation set using grid search). We proceed with the post-processed network outputs as described in Section 9.2.3 (transferring predictions from a physical to a musical time axis) to obtain  $\mathcal{A}^{\text{RNN}}$  and  $\mathcal{A}^{\text{CNN}}$ .

### 9.3.2 Evaluation Measures

After this conversion to a musical time axis, it is straightforward to use the resulting matrix  $\mathcal{A}^{\text{Est}}$  (i. e.,  $\mathcal{A}^{\text{RNN}}$ ,  $\mathcal{A}^{\text{CNN}}$ , or any other model output) and the reference  $\mathcal{A}^{\text{Ref}}$  for computing the number of true positive, false positive, and false negative predictions for a motif  $\ell \in [1 : L]$ :

$$\text{TP}_\ell = \sum_{m=1}^M \mathcal{A}_{\ell m}^{\text{Ref}} \mathcal{A}_{\ell m}^{\text{Est}} \quad (9.1)$$

$$\text{FP}_\ell = \sum_{m=1}^M (1 - \mathcal{A}_{\ell m}^{\text{Ref}}) \mathcal{A}_{\ell m}^{\text{Est}} \quad (9.2)$$

$$\text{FN}_\ell = \sum_{m=1}^M \mathcal{A}_{\ell m}^{\text{Ref}} (1 - \mathcal{A}_{\ell m}^{\text{Est}}) \quad (9.3)$$

<sup>42</sup> <https://www.tensorflow.org/>

Based on these numbers, we derive standard metrics such as precision (P), recall (R), and F-measure (F) for motif  $\ell$ . Finally, we take the mean over these values for all motifs in order to obtain what we call the *class mean* evaluation measures. Thus, for these mean values, all classes (i. e., motifs) are counted equally, regardless of the amount of leitmotif activity per class.

Furthermore, we also compute

$$\text{TP} = \sum_{\ell \in [1:L]} \text{TP}_\ell \quad (9.4)$$

(likewise for FP, FN) and then obtain precision, recall, and F-measure based on TP, FP, and FN, instead. Since we aggregate values from the whole matrices here (regardless of class), we call these the *matrix mean* evaluation measures. These values are subject to class imbalance on the level of measure sub-segments: motifs with more (and longer) activity regions affect the result more than rare (and short) motifs. For these values, all leitmotif activity is counted equally, regardless of class.

The metrics described here correspond to segment-based precision, recall, and F-measure in their class-based (macro-averaged) and instance-based (micro-averaged) variant [136], respectively.

### 9.3.3 Evaluation with Tolerance

Many applications of leitmotif activity detection may not require a very fine temporal granularity. For example, indicating a leitmotif one measure in advance may be sufficient for an application that draws a listener's attention to a forthcoming leitmotif. Furthermore, our automated annotation transfer with linear interpolation described in Section 9.2.2 may have introduced small errors, which should be accounted for in the evaluation. Motivated by such requirements, we introduce an additional tolerance parameter  $K$  in our evaluation. When comparing  $\mathcal{A}^{\text{Ref}}$  and  $\mathcal{A}^{\text{Est}}$ , we filter both matrices prior to thresholding using a moving maximum filter of length  $K$  for each motif. In the subsequent experiments, we set  $K=B$  so that the filter length corresponds to one measure. Thus, short interruption of a motif's activity (less than a measure long) are considered as the motif still being active. As another consequence, each false positive sub-segment leads to a minimal penalty in the evaluation, since the maximum filter enlarges false positive predictions to a duration of at least one measure (even if they are shorter). The same applies to false negative sub-segments, since any leitmotif activity in  $\mathcal{A}^{\text{Ref}}$  is also enlarged to a duration of at least one measure. In a similar fashion, each true positive prediction is enlarged to a duration of at least one measure, which can be thought of as a minimal reward for true positives. In this context, it is important to note that the median filter applied to the model outputs already eliminates very short positive predictions (of less than roughly 0.25 seconds).

	RNN			CNN		
	P	R	F	P	R	F
L-Ni	0.87	0.76	0.81	0.85	0.79	0.82
L-Ri	0.80	0.73	0.76	0.82	0.76	0.79
L-NH	0.89	0.78	0.83	0.91	0.82	0.86
L-Mi	0.86	0.86	0.86	0.87	0.79	0.83
L-RT	0.85	0.86	0.85	0.80	0.83	0.82
L-Wa	0.94	0.90	0.92	0.93	0.95	0.94
L-WL	0.86	0.85	0.85	0.83	0.85	0.84
L-Ho	0.80	0.76	0.78	0.82	0.80	0.81
L-Ge	0.89	0.81	0.85	0.85	0.81	0.83
L-Sc	0.74	0.72	0.73	0.83	0.72	0.77
L-Ju	0.82	0.68	0.74	0.87	0.78	0.82
L-WH	0.79	0.77	0.78	0.78	0.76	0.77
L-RS	0.87	0.84	0.86	0.86	0.81	0.84
L-Fe	0.87	0.88	0.88	0.93	0.86	0.89
L-SK	0.75	0.72	0.74	0.81	0.75	0.78
L-Un	0.79	0.75	0.77	0.84	0.81	0.83
L-Li	0.89	0.81	0.85	0.82	0.84	0.83
L-Si	0.78	0.75	0.76	0.83	0.80	0.81
L-Ma	0.79	0.81	0.80	0.87	0.79	0.83
L-Ve	0.84	0.73	0.78	0.83	0.83	0.83
Class mean	0.83	0.79	0.81	0.85	0.81	0.83
Matrix mean	0.83	0.78	0.80	0.85	0.80	0.82

**Table 9.4:** Results for our DL-based leitmotif activity detection systems on the test set.

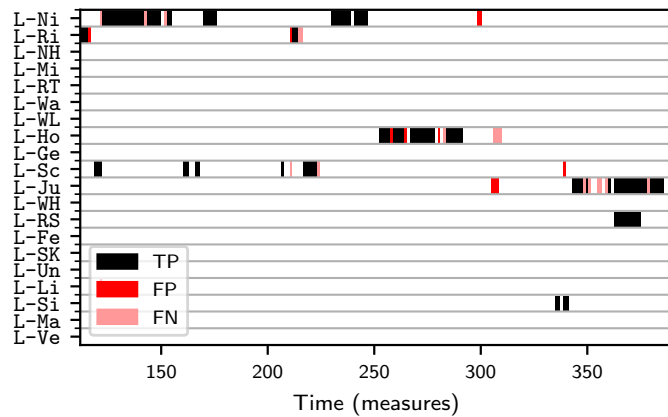
### 9.3.4 Experimental Results

We evaluate the trained models on the three test performances (see Figure 8.2), post-process the output, and apply the evaluation procedure and metrics as described above. For the RNN-based system, we obtain the results given in the left block of Table 9.4. Precision, recall, and F-measure are given for each motif, e. g., for L-RT,  $P=0.85$ ,  $R=0.86$ , and  $F=0.85$ . In this experiment based on the RNN model, precision values are usually higher than recall values, especially for L- Ju, where  $P=0.82$  and  $R=0.68$ . The effect is also evident in the class mean, where  $P=0.83$  and  $R=0.79$ , implying that our model has more difficulties with false negatives than false positives.

We obtain the highest F-measure for L-Wa with  $F=0.92$ , while the lowest is  $F=0.73$  for L-Sc. The class mean F-measure ( $F=0.81$ ) and the matrix mean F-measure ( $F=0.80$ ) are close to each other, which indicates that results for frequent and infrequent motifs (in terms of active measure sub-segments per motif) are similar. Overall, evaluation metrics for our RNN-based system for all motifs are above 0.7, with the mean results at around 0.8 for all evaluation metrics.

In Figure 9.3, we visualize results for our RNN-based model on an excerpt of the first act of *Siegfried*. Here, black regions correspond to true positive predictions of our model (after thresholding), while light and dark red regions indicate false negative and false positives, respectively. White color indicates true negative predictions. In the excerpt in Figure 9.3, most regions of leitmotif activity (and inactivity) are

**Figure 9.3:** Illustration of results for our RNN-based leitmotif activity detection system (shown for measures 112 to 390 from the first act of *Siegfried* in P-Ba).



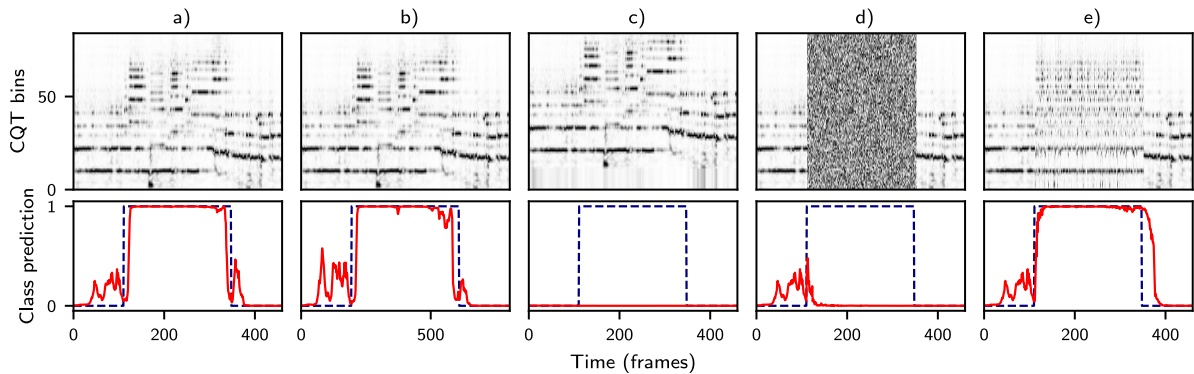
predicted correctly (black and white regions). Sometimes, only parts of a leitmotif instance are predicted as active (see, e. g., for L-Ri around measure 220). There are also some clear outliers such as the false positive predictions for L-Ni at measure 300 and L-Ju around measure 310. Overall, the correctly predicted regions dominate the visualization.

The right block of Table 9.4 shows our results obtained with the CNN-based system. Overall, results are slightly better than for the RNN (see, e. g., the class mean F-measure  $F=0.83$  compared to  $F=0.81$  for the RNN). Aside from this, we observe similar behavior as for the RNN. For example, L-Wa again yields the highest F-measure among motifs with  $F=0.94$ . We conclude that it is unlikely that either architecture is strongly superior to the other in terms of evaluation scores on the test set.

## 9.4 Robustness to Input Modifications

We now want to gain a deeper understanding of the properties learned by our neural network-based models. To do so, we systematically modify the input to our models in different ways and investigate the impact this has on the model outputs.

Figure 9.4, upper row, gives a qualitative overview of the modifications we consider in this section. Besides the unmodified model input (a), these modifications encompass (b) tempo changes, (c) pitch shifts, (d) replacement of leitmotif frames by noise, and (e) shuffling of leitmotif frames. The lower row of Figure 9.4 illustrates the activity functions resulting from the RNN for an example (solid red line), together with the reference annotation (dashed blue line). From a musical point of view, we would expect our activity detection approach to be robust against tempo changes and pitch shifts, while it should be sensitive to shuffling and noise replacement of frames. Strikingly, however, we see that tempo change and shuffling do not seem to change the results much, while pitch shifting and noise affect them strongly. We can also observe that our model anticipates the motif instance before it actually begins (Figure 9.4a). Very similar



**Figure 9.4:** Results for our RNN-based leitmotif activity detection system on measures 117 to 123.5 of the first act of *Siegfried* in P-Ba (see also Figure 9.2 and Figure 9.3; outputs of the CNN-based model are similar). A prominent instance of L-Sc is being played in the higher registers, accompanied by low-frequency tremolo. The model input is shown in the upper row. The respective output activations for the L-Sc class are plotted underneath in red (solid line). The dashed blue line corresponds to the ground truth annotations for L-Sc. The input is given to the network (a) unchanged, (b) slowed down to 175% of the original length, (c) with a pitch shift of eleven semitones, (d) with motif frames replaced by noise, and (e) with motif frames shuffled along the time axis.

behavior can be observed for the CNN (not shown here in the interest of space), although the CNN does not anticipate the motif instance in this example.

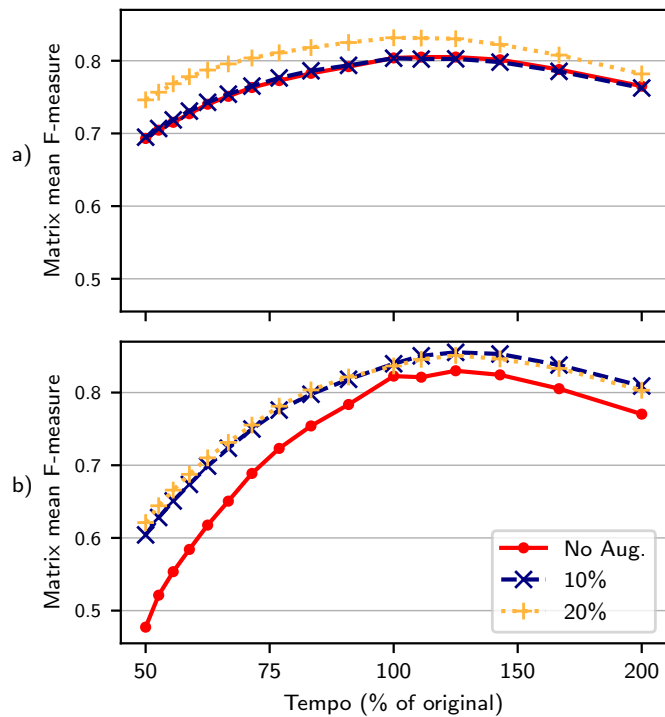
In the following we examine these qualitative findings in a quantitative fashion. To do so, we apply the modifications to all acts of all performances in the test set, detect leitmotif activity in these modified inputs using our networks, and then evaluate with our usual procedure.

### 9.4.1 Tempo Changes

First, we simulate global tempo changes in our test recordings by stretching or compressing our CQT representation along the time axis using bilinear filtering (see also Figure 9.4b).<sup>43</sup> Figure 9.5a shows the matrix mean F-measure obtained by the RNN on the test set for different tempo changes. For example, at 50% tempo, the input is stretched to twice its original length (i. e., slower), whereas for 200% tempo, the input is compressed to half its original length (i. e., faster). The solid red curve in Figure 9.5 demonstrates the effect of this transformation on our model. The resulting F-measure steadily decreases for slower inputs (from  $F=0.80$  at 100% to  $F=0.69$  at 50%). For faster inputs, the F-measure remains higher compared to slower inputs (e. g.,  $F=0.76$  at 200%). Nevertheless, most results are above  $F=0.70$ , meaning that our model can deal even with considerable tempo changes. It should be noted that all test performances are longer (i. e., slower) than an average performance in the training set. This may be the reason why our activity detection procedure is more robust to speeding up test performances while being more sensitive towards slowing them down.

<sup>43</sup> The experiments in this and the following section yield similar trends and conclusions when performed using a phase vocoding technique for time-scale modification.

**Figure 9.5:** Results for our (a) RNN-based and (b) CNN-based leitmotif activity detection systems on the test set under tempo changes. The CQT input is stretched in time (using bilinear resampling) by the given percentage.

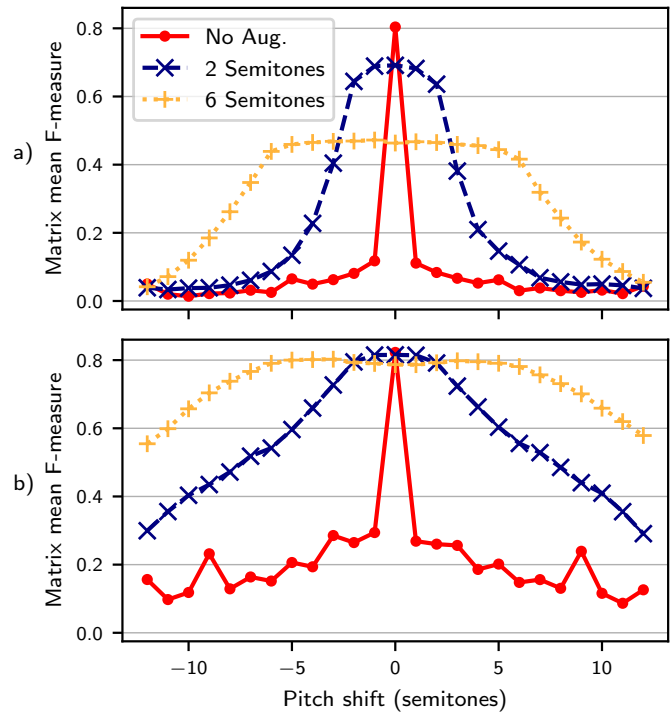


A similar trend can be observed for the CNN-based model in Figure 9.5b. Here, we observe a stronger drop in results for slower inputs (from  $F=0.82$  at 100% to  $F=0.48$  at 50%). We hypothesize that this is due to the fixed size of the CNN's receptive field, which means that its predictions are based on less musical content for inputs at slower tempos and on more musical content for inputs at faster tempos.

We now conduct the same experiment with an additional data augmentation strategy, as is common practice in deep learning [62], by also simulating global tempo changes during training. The dashed blue curve in Figure 9.5a shows the RNN's results in this experiment. This way, training examples are randomly stretched or compressed to be at most 10% slower or faster. The solid red and dashed blue curves are almost identical, meaning that this augmentation does not affect results much. We repeat this experiment with training augmentations of up to 20% change in tempo, indicated by the dotted orange curve. Here, test F-measure increases for all amounts of tempo changes (including  $F=0.83$  at 100%). For the CNN, we observe a similar behavior in Figure 9.5b. Here, both augmentation experiments yield improved results, although there is still a drop for very slow inputs ( $F=0.62$  at 50% for augmentations up to 20%). From these experiments, we conclude that training on ten different performances of the *Ring* already introduces some robustness to minor tempo changes in our model, which may further be enhanced through augmentations.



**Figure 9.6:** Results for our (a) RNN-based and (b) CNN-based leitmotif activity detection systems on the test set under pitch shifts. The CQT input has been shifted (using nearest-neighbor padding) on the pitch axis by the given number of semitones (corresponding to CQT bins).



### 9.4.2 Pitch Shifts

Second, we simulate transpositions in our test recordings by shifting our CQT representations along the pitch axis (using nearest-neighbor padding at the boundaries, i. e., the value for the lowest/highest CQT bin is replicated), see also Figure 9.4c. Figure 9.6a (solid red curve) shows matrix mean F-measures obtained with the RNN after modifying the test recordings in this way. This curve demonstrates that pitch shifts have a dramatic effect. For example, the test results drop to  $F=0.11$  for a shift of one semitone upwards. Shifting by more semitones, the F-measure drops further. We conclude that our model crucially relies on absolute pitch information. Even though leitmotif instances of the same motif appear in different registers and keys, the model has not learned their properties in a transposition-invariant way. As such, the model can only detect transposed motifs seen during training and would fail to generalize to new, unseen transpositions.

Convolutional architectures such as our CNN-based model are usually ascribed a certain degree of translation-invariance due to the weight-sharing and pooling operations [62]. Performing the pitch shift experiment for our CNN (Figure 9.6b), we can indeed observe better results than for the RNN when applying pitch shifting to the model input. For example, a shift of one semitone upwards now yields  $F=0.26$  and F-measures never drop below 0.1 for any considered shift. However, all shifts yield F-measures below 0.3, meaning that absolute pitch information is still highly important for our CNN-based model.

We repeat this experiment with an augmentation strategy, using pitch shifting also for the training set. Here, training examples are randomly shifted at most two semitones in either direction along the pitch axis. The dashed blue curve in Figure 9.6a shows the corresponding results for the RNN. We observe that applying this augmentation decreases results for the unmodified test inputs (i. e.,  $F=0.69$  for a shift of 0), but increases results for transformations considered during training (shifts of  $-2$  to  $+2$  semitones). Larger shifts still cause the model to fail. The same effect is seen in the dotted orange curve, where shifts of up to  $\pm 6$  semitones were applied as augmentation during training. Here, the result for unmodified model input drops to  $F=0.46$ , but the model can now cope with pitch shifts within the same range as used for augmentation (e. g.,  $F=0.42$  for a shift of  $+6$  semitones). In addition, the slopes of the F-measure curve are less steep, implying better generalization (e. g.,  $F=0.26$  for a shift of minus eight semitones, even though only shifts up to  $\pm 6$  semitones were included during training).

Figure 9.6b shows the corresponding curves for the CNN. Here, results for unmodified model input (shift of 0 semitones) drop only slightly when adding augmentations (e. g.,  $F=0.79$  for up to  $\pm 6$  semitones pitch shift augmentation compared to  $F=0.82$  without augmentation). Additionally, the slopes of the F-measure curves are even less steep (e. g.,  $F=0.74$  for a shift of minus eight semitones and up to  $\pm 6$  semitones as augmentation).

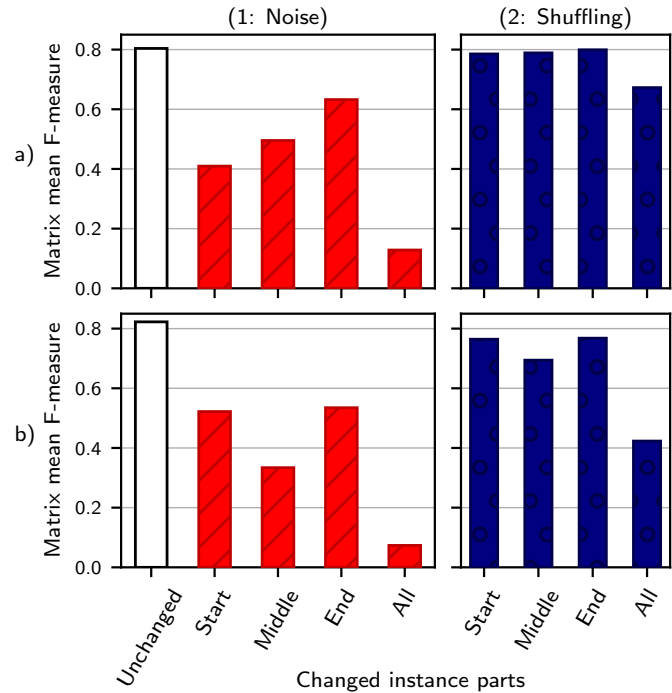
### 9.4.3 Noise

Third, we study the effect of completely removing all information in leitmotif regions from our test set. To do so, we replace all frames within a leitmotif instance by uniform noise (see Figure 9.4d). The impact of this modification on the RNN's results is shown in Figure 9.7a (1). When replacing all leitmotif frames (denoted as "All"), we obtain a much lower F-measure ( $F=0.13$ ) compared to the original model input ("Unchanged,"  $F=0.80$ ). In order to see whether our model responds to certain parts of leitmotif instances, we further modify only the first ("Start"), the middle ("Middle"), or the last third of frames ("End") for each leitmotif instance. The drop in F-measure is most pronounced for the beginning of motif instances (leading to  $F=0.42$  when replacing the first third but preserving the rest, compared to  $F=0.63$  for the last third). Yet, the overall F-measure does not drop entirely even when replacing all frames by noise. This implies that context around the leitmotif instances can help in identifying motifs even when the actual motif frames are absent. Again, we observe similar results for the CNN in Figure 9.7b (1). Here, frames in the middle of each leitmotif affect results more strongly and results drop even further when replacing all leitmotif frames ( $F=0.07$ ). Overall, we can conclude that our CNN-based model exploits context around leitmotif regions in a similar fashion as the RNN does.

### 9.4.4 Shuffling

Fourth, we study the effect of removing the temporal order from the leitmotif activity regions. To do so, we shuffle the frames within a leitmotif instance along the time axis, see also Figure 9.4e. The impact

**Figure 9.7:** Results for our (a) RNN-based and (b) CNN-based leitmotif activity detection systems on the test set when (1) replacing leitmotif frames by noise or (2) shuffling them along the time axis. The modifications have been applied to either the first, middle, or last third of each leitmotif instance (Start, Middle, End), for none (Unchanged), or for all leitmotif frames (All).



of this modification on the RNN is shown in Figure 9.7a (2), again for different parts of a leitmotif instance. We can observe that shuffling has only a minor impact on results (giving  $F=0.79$  when shuffling only the first third or  $F=0.67$  for all frames). Since shuffling along the time axis destroys any rhythmic information as well as the temporal aspects of melody (the order of notes), we conclude that such rhythmic or melodic cues are largely ignored by our model. We hypothesize that our model instead captures the pitch distributions in leitmotif instances, which are related to harmony. These distributions are mostly preserved when shuffling leitmotif frames, explaining the high results even for shuffling all frames of a leitmotif instance. Our experiments on pitch shifting (see Section 9.4.2) further suggest that the model depends on absolute pitch distributions rather than relative harmonic relationships (since pitch shifting preserves relative pitch relationships but changes absolute pitch distributions, leading to worse results).

The CNN reacts more strongly to this input modification, see Figure 9.7b (2). When shuffling all frames, for example, the F-measure drops to 0.42. F-measures remain high when only individual parts of the instances are shuffled (e. g.,  $F=0.77$  when shuffling only the end). Therefore, we hypothesize that our CNN only weakly reacts to temporal relationships.

Summarizing the insights obtained from the input modifications, we find that our models are to some degree robust to global tempo changes, which is a desirable property. However, we also found that they rely on pitch distributions within leitmotif instances (which is undesirable since these distributions can be affected by other musical parts) instead of capturing many musical cues that human listeners would associate with specific leitmotifs (such as temporal aspects of melody and rhythm). We further found that

our recurrent and convolutional architectures behave similarly under input modifications, with some slight differences. While the CNN is affected more strongly by slowed down input, it is more robust to pitch shifts, especially when using additional augmentation. In addition, the CNN is affected slightly more strongly by shuffling of leitmotif frames than the RNN.

## 9.5 Towards Less Informed Scenarios

This chapter approached the task of detecting leitmotif activity in a continuous (frame-wise) fashion over the course of entire opera recordings. As a more informed scenario, Chapter 8 considered classification of pre-segmented audio excerpts according to the leitmotif played. Additionally, we ruled out excerpts where multiple leitmotifs were played simultaneously. Compared to this constrained scenario, the leitmotif activity detection task is more challenging since no pre-segmented instances are given and inputs may contain no motif or simultaneously active motifs. In Chapter 8, we report F-measures of about 0.9 for a leitmotif classification setting with the first ten motifs of Table 9.1. While our results cannot be compared directly (especially since we evaluate on a frame level instead of an excerpt level as in Krause et al. [104]), we can see that the detection F-measures obtained with our deep learning systems (Table 9.4) are lower, at roughly 0.8 on average.

To approach scenarios with an even lower degree of side information, our systems must be able to handle previously unseen leitmotif occurrences. The classification experiments reported in Chapter 8 demonstrate that generalizing to unseen leitmotif occurrences is more challenging than generalizing to unseen performances of known occurrences. To this end, different splits of the dataset were considered in Chapter 8. In a similar way, we performed a preliminary experiment where we split the dataset across operas instead of performances. Here, we trained on all operas except for *Das Rheingold* in all 16 performances (Figure 8.2). We then evaluated on a test set containing only *Das Rheingold*, again in all 16 performances. From this experiment, we obtained low evaluation measures with  $P=0.17$ ,  $R=0.07$  and  $F=0.10$  (matrix mean) for the RNN-based system, as well as  $P=0.18$ ,  $R=0.13$  and  $F=0.15$  for the CNN-based system.

The discrepancy between the performance and the opera split's results may be explained with the models relying on confounding factors such as pitch distributions in leitmotif instances, while ignoring musically relevant aspects of leitmotifs such as rhythmic or melodic progressions. In other words, our models can be said to be overfitted towards the specific motif instances in the training set. In order to approach less informed scenarios such as the opera split (i. e., generalizing to unseen pattern occurrences) or the discovery of unknown leitmotifs (i. e., discovering unknown patterns in an unsupervised fashion), it becomes important to limit the impact of confounding factors. For this purpose, using more diverse data is recommended in the machine learning literature [62]. This could be realized, e. g., by adding more performances, considering data augmentation strategies, or utilizing artificial training data to expose the models to a larger variety of tempo, key, or timbre. As a different approach, one might annotate additional

musical works and utilize transfer-learning techniques [32]. Another improvement strategy could be the use of more elaborate neural network architectures by increasing the number of network parameters or by using convolutional-recurrent architectures [25] and other recent models proposed for SED tasks [121]. Additionally, dedicated architectures introducing invariance to tempo [60], key [46] or other properties [112] may be useful.

## 9.6 Conclusion

In this chapter, we approached the task of detecting leitmotif activity in opera recordings as a case study for the detection of complex musical patterns in audio. For our experiments, we considered a scenario comprising 3569 annotated occurrences of 20 characteristic leitmotifs in Wagner’s *Ring* cycle, realized in 16 different performances and, thus, summing up to 57 104 activity regions within more than 200 hours of audio. As our main contributions, we tested two deep learning models for leitmotif activity detection and analyzed their behavior under different input modifications. Our models provided good numerical results on a held-out test set but captured confounding factors such as absolute pitch distributions, rather than relying on characteristic musical properties of leitmotifs such as rhythmic or melodic patterns.

Thus, our study demonstrates the challenges faced by neural networks for detecting musical patterns. Future work may employ more elaborate model architectures and dedicated training strategies in order to handle this task in a more robust way and to proceed towards approaching other, less-informed scenarios. In the final Chapter 10 of this thesis, we further discuss these challenges and possible solutions.



## 10 Summary and Future Work

This thesis studied activity detection for musical sound events in audio recordings. We focused on orchestral and opera music as a complex and often neglected application scenario. In this context, we covered the detection of four different kinds of musical sound events using deep learning (DL), exploring different techniques in the process. We began by looking at singing voice as the event class to be detected. In Chapter 3, we compared a classical and a DNN-based approach for singing voice detection and showed that both perform roughly on par. Our study yielded insights on the advantages and drawbacks of DNNs for activity detection, which are more computationally expensive but also able to better utilize large training datasets than classical methods. Afterwards, in Chapter 4, we extended this scenario towards simultaneously classifying singer gender and voice type. In this novel setting, we compared different approaches for utilizing the hierarchical relationships between the event classes. As a main contribution, we proposed new consistency loss terms and found that a joint classification approach incorporating our losses is particularly effective. Thus, our work demonstrates the benefits of adding musically motivated constraints (such as hierarchies) to DL systems in a soft way. In Chapter 5, we explored these hierarchical techniques in more depth, using instrument sounds as our target events. As one contribution, we showed that utilizing hierarchy information reduces the amount of instrument annotations required, which has so far hindered research progress on this task. Additionally, we analyzed our models with regard to confounding effects and contributed to a better understanding of DL models for instrument activity detection. Afterwards, in Chapter 6, we presented an approach for representation learning that requires no instrument labels, relying instead on correspondences between different recorded versions of a piece. Our approach therefore constitutes another example of exploiting musical structure in DL systems. As the third kind of events in this thesis we considered pitches. In Chapter 7, we presented an approach for learning multi-pitch estimation from weakly-aligned training examples using soft dynamic time warping (SoftDTW) and showed its effectiveness compared to the state of the art for this task. Moreover, our work demonstrates how differentiable alignments may be used for arbitrary alignment problems in MIR. Fourth, in Chapters 8 and 9, we discussed leitmotifs as musical sound events that are especially challenging to detect. To the best of our knowledge, leitmotif classification and detection has not previously been studied within MIR research. In Chapter 8, we considered a scenario with pre-segmented motif excerpts and showed that an RNN can effectively classify these excerpts. In Chapter 9, we extended this towards a continuous detection scenario while covering additional motifs and classifier types. We also discussed the generalization capabilities of our models in-depth, yielding general insights on the automated detection of musical patterns in audio recordings.

We conclude this thesis with a discussion of promising directions for future work, which we organize into four broad categories.

**Applying and Extending our Detection Systems.** First, one may apply our detection systems for downstream applications such as music tagging or audio-based retrieval in large music catalogs. For example, the user of a digital audio workstation may find it helpful to segment an audio track according to singing or instrument activity. Our systems could also be used to aid music visualization or to display helpful additional information alongside live orchestral and opera performances [135]. Some of the methodologies presented in this thesis could also be applied to other tasks or music from other genres. For example, it could be insightful to explore the confounding factors affecting instrument activity detection systems for popular music. SoftDTW, as another example, could be applied to lyrics alignment and transcription [65, 193]. Furthermore, one may utilize soft alignments in scenarios involving feature sequences from arbitrary domains, such as multi-modal music synchronization between score, audio, and video [204]. There are also several natural extensions to our detection systems that could be explored. For instance, one might attempt to jointly detect multiple event classes simultaneously, e. g., instruments and pitches [84, 232]). One may also further improve the results of our detection systems by employing standard DL tricks, including more augmentations, larger model architectures, rigorous hyper-parameter tuning and increased amount of data [62]. However, as discussed throughout this thesis, the cost of obtaining annotated orchestral data may be prohibitively high. Future work may consider using music synchronization to generate even more annotated versions of the same pieces, leveraging alternative sources of annotations (such as lyric subtitles in video recordings of opera), or exploring synthetically generated versions.

**Utilizing Weakly- and Unsupervised Training Targets.** To alleviate the need for large amounts of annotations, a second promising direction is the use of weakly- and unsupervised learning paradigms. In this thesis, we made some contributions in this area, including the use of hierarchies to reduce the need for fine-grained labels, our proposed cross-version approach for annotation-free representation learning, and the use of SoftDTW for learning from weakly-aligned labels. As discussed in Chapter 6, there exists a large body of research on audio representation learning using self-supervised training strategies. In recent years, methods following a mask-and-predict paradigm have shown great promise. There, a portion of an audio input is masked and a DNN is trained to predict the masked content given some context (where “masking” may mean removing information directly from the waveform or within some latent space as in, e. g., the successful HuBERT [81] model). This idea originated from natural language processing [37] and has also been applied to computer vision [76]. We refer to [126] for a comprehensive survey on self-supervised learning for audio. Another interesting direction is multi-modal representation learning, where large databases of text–audio or image–audio pairs are used for learning [68, 169]. This approach has also recently been applied to the music domain [130]. In future work, the representations learned by these systems may be utilized for musical sound event detection. Often, however, such models are quite large and resource hungry. On the opposite end of that spectrum, incorporating domain knowledge can help to build models that require less data, may be easier to interpret, and have fewer parameters.



For example, the authors in [10] propose a system for unsupervised piano transcription that incorporates knowledge about the mechanisms of sound production in pianos. Choi et al. [31] build a differentiable drum synthesizer that learns—without annotations—to transcribe drum recordings through re-synthesis. This research direction, also termed analysis-by-synthesis [33, 177], has become more prominent with the availability of differentiable digital signal processing toolboxes [47], which have been used, e. g., to learn fundamental frequency estimation in a semi-supervised setting [48].

**Understanding and Addressing Confounding Factors.** Another prominent topic in this thesis was the presence of confounding factors and related overfitting effects. We repeatedly found that our models respond to patterns in the input which do not correspond to the musical attributes we wish to detect (e. g., spectral statistics for leitmotif detection in Chapter 9 or simultaneous instrument activity in Chapter 5). This behavior limits the generalization properties of our systems. As a third future research direction, we suggest to further investigate such confounding effects. While many MIR systems are known to exhibit such problems [196], DL systems are particularly prone to a phenomenon called *shortcut learning* in the wider machine learning literature. Shortcut learning refers to a DNN choosing simple decision-making strategies (shortcuts) over complex, more plausible ones. For example, an image classification system may respond to accidental pixel patterns in the training data [86, 199]. Similar effects have been observed in the music domain [168]. Likewise, an object detector may identify coffee cups based on patches of brown liquid rather than geometry [113]. We refer to [59] for a comprehensive survey on this topic. To address these problems in our setting, one may consider more controlled scenarios than real orchestral recordings, e. g., performing instrument classification on simple, monophonic sounds as opposed to polyphonic orchestral mixtures. Alternatively, using larger and more diverse datasets may also reduce the extent of overfitting, though eliminating all confounders in this way is challenging.

**Utilizing Generative Models for Activity Detection.** Finally, as the fourth area for future work, one may review recent advances in generative modeling for various modalities (including music) and investigate how they might contribute to improving musical sound event detection. Modern generative models in natural language processing such as BERT [37] or GPT [22] have shown impressive performance, not only in generating convincing natural language output, but also in representing higher-level semantic concepts [173]. Similarly, the success of generative vision models like Stable Diffusion [174] or multi-modal models like CLIP [169] suggests that such semantic understanding can extend to visual concepts. We hypothesize the following explanation for this: In order to produce convincing and high-resolution outputs, these generative models need a good understanding of semantics. In contrast, for many classification problems, shortcut solutions can suffice (see above). The hypothesis is supported by recent works showing how generative models may be adapted to reduce the impact of shortcuts and improve the robustness of DL models [180, 235]. An analogy from the MIR domain is the relationship between music source separation and activity detection. State-of-the-art systems for music source separation achieve good results when evaluated for activity detection, even when they have not been trained for that task [111]. Again, we hypothesize that separating a music mixture into individual tracks requires a DNN to learn more complex and robust features than merely learning activity detection. Another desirable property of generative

models is that they may be trained on large, unannotated datasets (see also the second research direction discussed above). In future work, such generative models may be adapted and fine-tuned for analysis tasks, such as sound event detection. In the context of music processing, some works have attempted to adapt a large generative model for music (called Jukebox [38]) by finetuning it for music classification tasks [27] or source separation [132]. Their success has so far been hindered by the limited capability of Jukebox to model long-range dependencies in music recordings. Recently released generation systems like MusicLM [3] claim to overcome this. However, like most models in MIR research, none of these models are trained on large amounts of orchestral or opera data and are thus not directly applicable in our scenario. Finally, it is important to emphasize that large generative models also suffer from biases and confounding effects [77]. In other words, building MIR models that are transferable to new music styles and genres remains a challenge.

To conclude, we believe that music in general and musical sound event detection in particular offer unique opportunities to advance the state of the art and gain a better understanding of machine learning algorithms. In contrast to, e. g., object detection in images, the musical scenarios approached in this thesis remain far from solved. The event types we discussed may appear in a highly correlated fashion (like instruments), may merge together into a single sound (e. g., in an orchestral tutti), and their definition may be highly ambiguous (for leitmotifs or note offsets). As we showed in this thesis, by building musical sound event detection systems and analyzing their behavior, we gain an understanding not just of the neural networks we use but also of the musical concepts themselves.

## Abbreviations

<b>CPU</b>	central processing unit	<b>MCTC</b>	multi-label connectionist temporal classification
<b>CTC</b>	connectionist temporal classification	<b>MPE</b>	multi-pitch estimation
<b>CQT</b>	constant-Q transform	<b>MIREX</b>	Music Information Retrieval Evaluation eXchange
<b>CNN</b>	convolutional neural network	<b>MIR</b>	music information retrieval
<b>DL</b>	deep learning	<b>MIDI</b>	musical instrument digital interface
<b>DNN</b>	deep neural network	<b>RAM</b>	random access memory
<b>DCASE</b>	Detection and Classification of Acoustic Scenes and Events	<b>RFC</b>	random forest classifier
<b>DP</b>	dynamic programming	<b>ReLU</b>	rectified linear unit
<b>DTW</b>	dynamic time warping	<b>RNN</b>	recurrent neural network
<b>GPU</b>	graphics processing unit	<b>SWD</b>	Schubert Winterreise Dataset
<b>HCQT</b>	harmonic constant-Q transform	<b>STFT</b>	short-time Fourier transform
<b>IAD</b>	instrument activity detection	<b>SVD</b>	singing voice detection
<b>ISMIR</b>	International Society for Music Information Retrieval	<b>SoftDTW</b>	soft dynamic time warping
<b>LSTM</b>	long short-term memory	<b>SED</b>	sound event detection
<b>MFCC</b>	mel-frequency cepstral coefficient	<b>TPU</b>	tensor processing unit
		<b>VRAM</b>	video random access memory



# Bibliography

- [1] Jakob Abeßer. A review of deep learning based methods for acoustic scene classification. *Applied Sciences*, 10(6):2020, 2020. doi: 10.3390/app10062020.
- [2] Jakob Abeßer and Meinard Müller. Jazz bass transcription using a U-net architecture. *Electronics*, 10(6):670, 2021. doi: 10.3390/electronics10060670.
- [3] Andrea Agostinelli, Timo I. Denk, Zalán Borsos, Jesse H. Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, Matthew Sharifi, Neil Zeghidour, and Christian Havnø Frank. MusicLM: Generating music from text. *arXiv*, abs/2301.11325, 2023. doi: 10.48550/arXiv.2301.11325.
- [4] Ruchit Agrawal, Daniel Wolff, and Simon Dixon. A convolutional-attentional neural framework for structure-aware performance-score synchronization. *IEEE Signal Processing Letters*, 29:344–348, 2021. doi: 10.1109/LSP.2021.3135192.
- [5] Henning Albrecht and Klaus Frieler. The perception and recognition of Wagnerian leitmotifs in multimodal conditions. In *Proceedings of the International Conference of Students of Systematic Musicology (SysMus)*, London, UK, 2014. URL <https://journals.gold.ac.uk/index.php/sysmus14/article/view/220>.
- [6] Pablo Alonso-Jiménez, Xavier Serra, and Dmitry Bogdanov. Music representation learning based on editorial metadata from discogs. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 825–833, Bengaluru, India, 2022. doi: 10.5281/zenodo.7316790.
- [7] David J. Baker and Daniel Müllensiefen. Perception of leitmotives in Richard Wagner’s *Der Ring des Nibelungen*. *Frontiers in Psychology*, 8:662, 2017. doi: 10.3389/fpsyg.2017.00662.
- [8] Emmanouil Benetos, Simon Dixon, Zhiyao Duan, and Sebastian Ewert. Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, 36(1):20–30, 2019. doi: 10.1109/MSP.2018.2869928.
- [9] Adam L. Berenzweig and Daniel P. W. Ellis. Locating singing voice segments within music signals. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 119–122, New Paltz, New York, USA, 2001. doi: 10.1109/ASPAA.2001.969557.
- [10] Taylor Berg-Kirkpatrick, Jacob Andreas, and Dan Klein. Unsupervised transcription of piano music. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 1538–1546, Montréal, Canada, 2014. URL <https://proceedings.neurips.cc/paper/2014/hash/3b5dca501ee1e6d8cd7b905f4e1bf723-Abstract.html>.

## Bibliography

- [11] Luca Bertinetto, Romain Müller, Konstantinos Tertikas, Sina Samangooei, and Nicholas A. Lord. Making better mistakes: Leveraging class hierarchies with deep networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12503–12512, Seattle, WA, USA, 2020. doi: 10.1109/CVPR42600.2020.01252.
- [12] Rachel M. Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Pablo Bello. MedleyDB: A multitrack dataset for annotation-intensive MIR research. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 155–160, Taipei, Taiwan, 2014. doi: 10.5281/zenodo.1417889.
- [13] Rachel M. Bittner, Brian McFee, Justin Salamon, Peter Li, and Juan P. Bello. Deep salience representations for F0 tracking in polyphonic music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 63–70, Suzhou, China, 2017. doi: 10.5281/zenodo.1417937.
- [14] Sebastian Böck, Florian Krebs, and Gerhard Widmer. Joint beat and downbeat tracking with recurrent neural networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 255–261, New York City, New York, USA, 2016. doi: 10.5281/zenodo.1415835.
- [15] Dmitry Bogdanov, Minz Won, Philip Tovstogan, Alastair Porter, and Xavier Serra. The MTG-Jamendo dataset for automatic music tagging. In *Proceedings of the Workshop on Machine Learning for Music Discovery, International Conference on Machine Learning (ICML)*, Long Beach, CA, USA, 2019. URL <https://drive.google.com/open?id=1tRRLI0Wz80TrUpcQnW0okeI0XePkbbGy>.
- [16] Juan J. Bosch, Rachel M. Bittner, Justin Salamon, and Emilia Gómez. A comparison of melody extraction methods based on source-filter modelling. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 571–577, New York City, New York, USA, 2016. doi: 10.5281/zenodo.1418166.
- [17] Karlheinz Brandenburg, Christian Dittmar, Matthias Grühne, Jakob Abeßer, Hanna Lukashevich, Peter Dunker, Daniel Gärtner, Kay Wolter, Stefanie Nowak, and Holger Großmann. Music search and recommendation. In Borko Furht, editor, *Handbook of multimedia for digital entertainment and arts*, volume 3, pages 349–383. Springer, New York, USA, 2009. ISBN 0-387-89023-8. doi: 10.1007/978-0-387-89024-1\_16.
- [18] Matthew Bribitzer-Stull. *Understanding the Leitmotif*. Cambridge University Press, 2015. ISBN 978-1316161678. doi: 10.1017/CBO9781316161678.
- [19] Howard Mayer Brown, Ellen Rosand, Reinhard Strohm, Roger Parker, Arnold Whittall, Roger Savage, and Barry Millington. Opera. In Stanley Sadie, editor, *The New Grove Dictionary of Music and Musicians*, pages 416–471. Macmillan Publishers, London, 2 edition, 2001. ISBN 978-0-333-60800-5. URL <https://www.oxfordmusiconline.com/grovemusic>.
- [20] Judith C. Brown. Calculation of a constant Q spectral transform. *Journal of the Acoustical Society of America*, 89(1):425–434, 1991. doi: 10.1121/1.400476.
- [21] Judith C. Brown and Miller S. Puckette. An efficient algorithm for the calculation of a constant Q transform. *Journal of the Acoustic Society of America (JASA)*, 92:2698–2701, 1992. doi: 10.1121/1.404385.

- [22] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1877–1901, Virtual, 2020. URL <https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfc4967418bfb8ac142f64a-Abstract.html>.
- [23] Morgan Buisson, Brian McFee, Slim Essid, and H el ene C. Crayencour. Learning multi-level representations for hierarchical music structure analysis. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 591–597, Bengaluru, India, 2022. doi: 10.5281/zenodo.7343060.
- [24] Christoph B ohm, David Ackermann, and Stefan Weinzierl. A multi-channel anechoic orchestra recording of Beethoven’s Symphony no. 8 op. 93. *Journal of the Audio Engineering Society*, 68(12):977–984, 2021. doi: 10.17743/jaes.2020.0056.
- [25] Emre  akir, Giambattista Parascandolo, Toni Heittola, Heikki Huttunen, and Tuomas Virtanen. Convolutional recurrent neural networks for polyphonic sound event detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(6):1291–1303, 2017. doi: 10.1109/TASLP.2017.2690575.
- [26] Estefan a Cano, Derry FitzGerald, Antoine Liutkus, Mark D. Plumbley, and Fabian-Robert St oter. Musical source separation: An introduction. *IEEE Signal Processing Magazine*, 36(1):31–40, 2019. doi: 10.1109/MSP.2018.2874719.
- [27] Rodrigo Castellon, Chris Donahue, and Percy Liang. Codified audio language modeling learns useful representations for music information retrieval. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 88–96, Online, 2021. doi: 10.5281/zenodo.5624604.
- [28] Ricardo Cerri, Rodrigo C. Barros, and Andr e Carlos Ponce de Leon Ferreira de Carvalho. Hierarchical multi-label classification using local neural networks. *Journal of Computer and System Sciences*, 80(1): 39–56, 2014. doi: 10.1016/j.jcss.2013.03.007.
- [29] Chien-Yi Chang, De-An Huang, Yanan Sui, Li Fei-Fei, and Juan Carlos Nieves. D3TW: Discriminative differentiable dynamic time warping for weakly supervised action alignment and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3546–3555, Long Beach, CA, USA, 2019. doi: 10.1109/CVPR.2019.00366.
- [30] Kin Wai Cheuk, Yin-Jyun Luo, Emmanouil Benetos, and Dorien Herremans. Revisiting the onsets and frames model with additive attention. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, Shenzhen, China, 2021. doi: 10.1109/IJCNN52387.2021.9533407.
- [31] Keunwoo Choi and Kyunghyun Cho. Deep unsupervised drum transcription. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 183–191, Delft, The Netherlands, 2019. doi: 10.5281/zenodo.3527773.
- [32] Keunwoo Choi, Gy orgy Fazekas, Mark B. Sandler, and Kyunghyun Cho. Transfer learning for music classification and regression tasks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 141–149, Suzhou, China, 2017. doi: 10.5281/zenodo.1418014.

## Bibliography

- [33] Nicolae Cleju, Maria G. Jafari, and Mark D. Plumbley. Analysis-based sparse reconstruction with synthesis-based solvers. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 5401–5404, Kyoto, Japan, 2012. doi: 10.1109/ICASSP.2012.6289142.
- [34] Jason Cramer, Vincent Lostanlen, Andrew Farnsworth, Justin Salamon, and Juan Pablo Bello. Chirping up the right tree: Incorporating biological taxonomies into deep bioacoustic classifiers. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 901–905, Barcelona, Spain, 2020. doi: 10.1109/ICASSP40776.2020.9052908.
- [35] Marco Cuturi and Mathieu Blondel. Soft-DTW: a differentiable loss function for time-series. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 894–903, Sydney, NSW, Australia, 2017. URL <http://proceedings.mlr.press/v70/cuturi17a.html>.
- [36] Steven B. Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, 1980. doi: 10.1109/TASSP.1980.1163420.
- [37] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, (NAACL-HLT)*, pages 4171–4186, Minneapolis, MN, USA, 2019. doi: 10.18653/v1/n19-1423.
- [38] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv*, abs/2005.00341, 2020. doi: 10.48550/arXiv.2005.00341.
- [39] Christian Dittmar, Bernhard Lehner, Thomas Prätzlich, Meinard Müller, and Gerhard Widmer. Cross-version singing voice detection in classical opera recordings. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 618–624, Málaga, Spain, October 2015. doi: 10.5281/zenodo.1416958.
- [40] Simon Dixon and Gerhard Widmer. MATCH: A music alignment tool chest. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 492–497, London, UK, 2005. doi: 10.5281/zenodo.1416952.
- [41] Laurence Dreyfus and Carolin Rindfleisch. Using digital libraries in the research of the reception and interpretation of Richard Wagner’s leitmotifs. In *Proceedings of the International Workshop on Digital Libraries for Musicology (DLfM)*, pages 1–3, London, UK, 2014. doi: 10.1145/2660168.2660181.
- [42] Zhiyao Duan, Bryan Pardo, and Changshui Zhang. Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(8): 2121–2133, 2010. doi: 10.1109/TASL.2010.2042119.
- [43] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv*, abs/1603.07285, 2016. doi: 10.48550/arXiv.1603.07285.
- [44] Thibaut Durand, Nazanin Mehrasa, and Greg Mori. Learning a deep convnet for multi-label classification with partial labels. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 647–657, Long Beach, CA, USA, 2019. doi: 10.1109/CVPR.2019.00074.



- [45] Jana Eggink and Guy J. Brown. A missing feature approach to instrument identification in polyphonic music. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 553–556, Hong Kong, China, 2003. doi: 10.1109/ICASSP.2003.1200029.
- [46] Anders Elowsson and Anders Friberg. Modeling music modality with a key-class invariant pitch chroma CNN. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 541–548, Delft, The Netherlands, 2019. doi: 10.5281/zenodo.3527864.
- [47] Jesse Engel, Lamtharn Hantrakul, Chenjie Gu, and Adam Roberts. DDSP: Differentiable digital signal processing. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Virtual, 2020. URL <https://openreview.net/forum?id=B1x1ma4tDr>.
- [48] Jesse Engel, Rigel Swavely, Lamtharn Hanoi Hantrakul, Adam Roberts, and Curtis Hawthorne. Self-supervised pitch detection by inverse audio synthesis. In *International Conference on Machine Learning (ICML), Workshop on Self-Supervision in Audio and Speech*, Vienna, Austria, 2020. URL <https://openreview.net/forum?id=RlVTYWhsky7>.
- [49] Antti J. Eronen and Anssi P. Klapuri. Musical instrument recognition using cepstral coefficients and temporal features. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, pages II753–II756, Istanbul, Turkey, 2000. doi: 10.1109/ICASSP.2000.859069.
- [50] Slim Essid, Gaël Richard, and Bertrand David. Hierarchical classification of musical instruments on solo recordings. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 817–820, Toulouse, France, 2006. doi: 10.1109/ICASSP.2006.1661401.
- [51] Sebastian Ewert, Meinard Müller, and Peter Grosche. High resolution audio synchronization using chroma onset features. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, 2009. doi: 10.1109/ICASSP.2009.4959972.
- [52] Sebastian Ewert, Meinard Müller, Verena Konz, Daniel Müllensiefen, and Gerraint A. Wiggins. Towards cross-version harmonic analysis of music. *IEEE Transactions on Multimedia*, 14(3-2):770–782, 2012. doi: 10.1109/TMM.2012.2190047.
- [53] Arthur Flexer. A closer look on artist filters for musical genre classification. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 341–344, Vienna, Austria, 2007. doi: 10.5281/zenodo.1415668.
- [54] Jonathan Foote. Automatic audio segmentation using a measure of audio novelty. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 452–455, New York, NY, USA, 2000. doi: 10.1109/ICME.2000.869637.
- [55] Peter Foster, Siddharth Sigtia, Sacha Krstulovic, Jon Barker, and Mark D. Plumbley. Chime-home: A dataset for sound source recognition in a domestic environment. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, USA, 2015. doi: 10.1109/WASPAA.2015.7336899.
- [56] Joachim Fritsch and Mark D. Plumbley. Score informed audio source separation using constrained nonnegative matrix factorization and score synthesis. In *Proceedings of the IEEE International Conference*

## Bibliography

- on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 888–891, Vancouver, Canada, May 2013. doi: 10.1109/ICASSP.2013.6637776.
- [57] Ferdinand Fuhrmann and Perfecto Herrera. Polyphonic instrument recognition for exploring semantic similarities in music. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, Graz, Austria, 2010. URL <https://www.dafx.de/paper-archive/details.php?id=cSZcp2xDEkQDPQGDJKDA3w>.
- [58] Hugo Flores Garcia, Aldo Aguilar, Ethan Manilow, and Bryan Pardo. Leveraging hierarchical structures for few-shot musical instrument recognition. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 220–228, Online, 2021. doi: 10.5281/zenodo.5624615.
- [59] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2: 665–673, 2020. doi: 10.1038/s42256-020-00257-z.
- [60] Bruno Di Giorgi, Matthias Mauch, and Mark Levy. Downbeat tracking with tempo-invariant convolutional neural networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 216–222, Montréal, Canada, 2020. doi: 10.5281/zenodo.4245408.
- [61] Eleonora Giunchiglia and Thomas Lukasiewicz. Coherent hierarchical multi-label classification networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9662–9673, Virtual, 2020. URL <https://proceedings.neurips.cc/paper/2020/hash/6dd4e10e3296fa63738371ec0d5df818-Abstract.html>.
- [62] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge and London, 2016. ISBN 978-0-262-03561-3. URL <http://www.deeplearningbook.org>.
- [63] Mikus Grasis, Jakob Abeßer, Christian Dittmar, and Hanna M. Lukashevich. A multiple-expert framework for instrument recognition. In *Proceedings of the International Symposium on Sound, Music, and Motion (CMMR)*, volume 8905 of *Lecture Notes in Computer Science*, pages 619–634, Marseille, France, 2014. Springer. doi: 10.1007/978-3-319-12976-1\_38.
- [64] Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 369–376, Pittsburgh, Pennsylvania, USA, 2006. doi: 10.1145/1143844.1143891.
- [65] Chitralkha Gupta, Emre Yılmaz, and Haizhou Li. Automatic lyrics alignment and transcription in polyphonic music: Does background music help? In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 496–500, Barcelona, Spain, 2020. doi: 10.1109/ICASSP40776.2020.9054567.
- [66] Siddharth Gururani and Alexander Lerch. Semi-supervised audio classification with partially labeled data. In *IEEE International Symposium on Multimedia (ISM)*, pages 111–114, Naples, Italy, 2021. doi: 10.1109/ISM52913.2021.00027.
- [67] Siddharth Gururani, Cameron Summers, and Alexander Lerch. Instrument activity detection in polyphonic music using deep neural networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 569–576, Paris, France, 2018. doi: 10.5281/zenodo.1492479.

- [68] Andrey Guzhov, Federico Raue, Jörn Hees, and Andreas Dengel. AudioCLIP: Extending clip to image, text and audio. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 976–980, Singapore, 2022. doi: 10.1109/ICASSP43922.2022.9747631.
- [69] Isma Hadji, Konstantinos G. Derpanis, and Allan D. Jepson. Representation learning via global temporal alignment and cycle-consistency. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11068–11077, Virtual, 2021. doi: 10.1109/CVPR46437.2021.01092.
- [70] Philippe Hamel, Sean Wood, and Douglas Eck. Automatic identification of instrument classes in polyphonic and poly-instrument audio. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 399–404, Kobe, Japan, 2009. doi: 10.5281/zenodo.1415091.
- [71] Yoonchang Han, Jae-Hun Kim, and Kyogu Lee. Deep convolutional neural networks for predominant instrument recognition in polyphonic music. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(1):208–221, 2017. doi: 10.1109/TASLP.2016.2632307.
- [72] Yushen Han and Christopher Raphael. Informed source separation of orchestra and soloist. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 315–320, Utrecht, The Netherlands, 2010. doi: 10.5281/zenodo.1416750.
- [73] Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse H. Engel, Sageev Oore, and Douglas Eck. Onsets and frames: Dual-objective piano transcription. In *Proceedings of the International Society for Music Information Retrieval Conference, (ISMIR)*, pages 50–57, Paris, France, 2018. doi: 10.5281/zenodo.1492341.
- [74] Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse H. Engel, and Douglas Eck. Enabling factorized piano music modeling and generation with the MAESTRO dataset. In *Proceedings of the International Conference on Learning Representations (ICLR)*, New Orleans, Louisiana, USA, 2019. URL <https://openreview.net/forum?id=r11YRjC9F7>.
- [75] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, Nevada, USA, 2016. doi: 10.1109/CVPR.2016.90.
- [76] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15979–15988, New Orleans, LA, USA, 2022. doi: 10.1109/CVPR52688.2022.01553.
- [77] Melissa Heikkilä. The viral AI avatar app Lensa undressed me—without my consent. MIT Technology Review, 2022. URL <https://www.technologyreview.com/2022/12/12/1064751/the-viral-ai-avatar-app-lensa-undressed-me-without-my-consent/>.
- [78] Toni Heittola, Anssi P. Klapuri, and Tuomas Virtanen. Musical instrument recognition in polyphonic audio using source-filter model for sound separation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 327–332, Kobe, Japan, 2009. doi: 10.5281/zenodo.1417377.
- [79] Toni Heittola, Annamaria Mesaros, and Tuomas Virtanen. Acoustic scene classification in DCASE 2020 challenge: Generalization across devices and low complexity solutions. In *Proceedings*

## Bibliography

- of the Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE), pages 56–60, Virtual, 2020. URL [http://dcase.community/documents/workshop2020/proceedings/DCASE2020Workshop\\_Heittola\\_56.pdf](http://dcase.community/documents/workshop2020/proceedings/DCASE2020Workshop_Heittola_56.pdf).
- [80] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, November 1997. doi: 10.1162/neco.1997.9.8.1735.
- [81] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhota, Ruslan Salakhutdinov, and Abdelrahman Mohamed. HuBERT: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 29:3451–3460, 2021. doi: 10.1109/TASLP.2021.3122291.
- [82] Eric J. Humphrey, Simon Durand, and Brian McFee. OpenMIC-2018: An open data-set for multiple instrument recognition. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 438–444, Paris, France, 2018. doi: 10.5281/zenodo.1492445.
- [83] Eric J. Humphrey, Sravana Reddy, Prem Seetharaman, Aparna Kumar, Rachel M. Bittner, Andrew Demetriou, Sankalp Gulati, Andreas Jansson, Tristan Jehan, Bernhard Lehner, Anna Krupse, and Luwei Yang. An introduction to signal processing for singing-voice analysis: High notes in the effort to automate the understanding of vocals in music. *IEEE Signal Processing Magazine*, 36(1):82–94, 2019. doi: 10.1109/MSP.2018.2875133.
- [84] Yun-Ning Hung and Yi-Hsuan Yang. Frame-level instrument recognition by timbre and pitch. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 135–142, Paris, France, 2018. doi: 10.5281/zenodo.1492363.
- [85] Yun-Ning Hung, Yi-An Chen, and Yi-Hsuan Yang. Multitask learning for frame-level instrument recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 381–385, Brighton, UK, 2019. doi: 10.1109/ICASSP.2019.8683426.
- [86] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 125–136, Vancouver, BC, Canada, 2019. URL <https://papers.nips.cc/paper/2019/hash/e2c420d928d4bf8ce0ff2ec19b371514-Abstract.html>.
- [87] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 448–456, Lille, France, 2015. URL <https://proceedings.mlr.press/v37/ioffe15.html>.
- [88] Arindam Jati, Naveen Kumar, Ruxin Chen, and Panayiotis G. Georgiou. Hierarchy-aware loss function on a tree structured label space for audio event detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP*, pages 6–10, Brighton, United Kingdom, 2019. doi: 10.1109/ICASSP.2019.8682341.
- [89] Dasaem Jeong, Taegyun Kwon, Yoojin Kim, Kyogu Lee, and Juhan Nam. VirtuosoNet: A hierarchical RNN-based system for modeling expressive piano performance. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 908–915, Delft, The Netherlands, 2019. doi: 10.5281/zenodo.3527962.

- [90] Il-Young Jeong and Kyogu Lee. Learning temporal features using a deep neural network and its application to music genre classification. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 434–440, New York City, New York, USA, 2016. doi: 10.5281/zenodo.1416416.
- [91] Cyril Joder, Slim Essid, and Gaël Richard. Temporal integration for audio classification with application to musical instrument classification. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(1): 174–186, 2009. doi: 10.1109/TASL.2008.2007613.
- [92] Rainer Kelz and Gerhard Widmer. An experimental analysis of the entanglement problem in neural-network-based music transcription systems. In *Proceedings of the AES International Conference on Semantic Audio*, pages 194–201, Erlangen, Germany, 2017. URL <https://www.aes.org/e-lib/browse.cfm?elib=18761>.
- [93] Rainer Kelz, Matthias Dorfer, Filip Korzeniowski, Sebastian Böck, Andreas Arzt, and Gerhard Widmer. On the potential of simple framewise approaches to piano transcription. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 475–481, New York City, New York, USA, 2016. doi: 10.5281/zenodo.1416488.
- [94] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference for Learning Representations (ICLR)*, San Diego, California, USA, 2015. URL <http://arxiv.org/abs/1412.6980>.
- [95] Tetsuro Kitahara. *Computational Musical Instrument Recognition and its Application to Content-based Music Information Retrieval*. PhD thesis, Kyoto University, Japan, 2007. URL <http://hdl.handle.net/2433/135955>.
- [96] Sefki Kolozali, Mathieu Barthet, György Fazekas, and Mark B. Sandler. Knowledge representation issues in musical instrument ontology design. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 465–470, Miami, Florida, USA, 2011. doi: <https://doi.org/10.5281/zenodo.1416141>.
- [97] Verena Konz, Meinard Müller, and Rainer Kleinertz. A cross-version chord labelling approach for exploring harmonic structures—a case study on Beethoven’s *Appassionata*. *Journal of New Music Research*, 42(1): 61–77, 2013. doi: 10.1080/09298215.2012.750369.
- [98] Andreas Kornstädt. The JRing system for computer-assisted musicological analysis. In *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, pages 93–98, Bloomington, Indiana, USA, 2001. doi: 10.5281/zenodo.1416100.
- [99] Filip Korzeniowski and Gerhard Widmer. Genre-agnostic key classification with convolutional neural networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 264–270, Paris, France, 2018. doi: 10.5281/zenodo.1492399.
- [100] Aris Kosmopoulos, Ioannis Partalas, Éric Gaussier, Georgios Paliouras, and Ion Androutsopoulos. Evaluation measures for hierarchical classification: a unified view and novel approaches. *Data Mining and Knowledge Discovery*, 29(3):820–865, 2015. doi: 10.1007/s10618-014-0382-x.

## Bibliography

- [101] Agelos Kratimenos, Kleanthis Avramidis, Christos Garoufis, Athanasia Zlatintsi, and Petros Maragos. Augmentation methods on monophonic audio for instrument classification in polyphonic music. In *Proceedings of the European Signal Processing Conference (EUSIPCO)*, pages 156–160, Amsterdam, Netherlands, 2020. doi: 10.23919/Eusipco47968.2020.9287745.
- [102] Michael Krause and Meinard Müller. Hierarchical classification for singing activity, gender, and type in complex music recordings. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 406–410, Virtual and Singapore, 2022. doi: 10.1109/ICASSP43922.2022.9747690.
- [103] Michael Krause and Meinard Müller. Hierarchical classification for instrument activity detection in orchestral music recordings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 31: 2567–2578, 2023. doi: 10.1109/TASLP.2023.3291506.
- [104] Michael Krause, Frank Zalkow, Julia Zalkow, Christof Weiß, and Meinard Müller. Classifying leitmotifs in recordings of operas by Richard Wagner. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 473–480, Montréal, Canada, 2020. doi: 10.5281/zenodo.4245472.
- [105] Michael Krause, Meinard Müller, and Christof Weiß. Towards leitmotif activity detection in opera recordings. *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 4(1):127–140, 2021. doi: 10.5334/tismir.116.
- [106] Michael Krause, Meinard Müller, and Christof Weiß. Singing voice detection in opera recordings: A case study on robustness and generalization. *Electronics*, 10(10):1214:1–14, 2021. doi: 10.3390/electronics10101214.
- [107] Michael Krause, Sebastian Strahl, and Meinard Müller. Weakly supervised multi-pitch estimation using cross-version alignment. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Milano, Italy, 2023.
- [108] Michael Krause, Christof Weiß, and Meinard Müller. A cross-version approach to audio representation learning for orchestral music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Milano, Italy, 2023.
- [109] Michael Krause, Christof Weiß, and Meinard Müller. Soft dynamic time warping for multi-pitch estimation and beyond. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Rhodes Island, Greece, 2023. doi: 10.1109/ICASSP49357.2023.10095907.
- [110] Anna M. Kruspe. *Application of Automatic Speech Recognition Technologies to Singing*. PhD thesis, Technische Universität Ilmenau, Germany, 2018. URL [https://www.db-thueringen.de/receive/dbt\\_mods\\_00035065](https://www.db-thueringen.de/receive/dbt_mods_00035065).
- [111] Rajath Kumar, Yi Luo, and Nima Mesgarani. Music source activity detection and separation using deep attractor network. In *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech)*, pages 347–351, Hyderabad, India, 2018. doi: 10.21437/Interspeech.2018-2326.
- [112] Stefan Lattner, Monika Dörfler, and Andreas Arzt. Learning complex basis functions for invariant representations of audio. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 700–707, Delft, The Netherlands, 2019. doi: 10.5281/zenodo.3527906.

- [113] Guillaume Leclerc, Hadi Salman, Andrew Ilyas, Sai Vemprala, Logan Engstrom, Vibhav Vineet, Kai Yuanqing Xiao, Pengchuan Zhang, Shibani Santurkar, Greg Yang, Ashish Kapoor, and Aleksander Madry. 3DB: A framework for debugging computer vision models. *arXiv*, abs/2106.03805, 2021. doi: 10.48550/arXiv.2106.03805.
- [114] Kyungyun Lee, Keunwoo Choi, and Juhan Nam. Revisiting singing voice detection: A quantitative review and the future outlook. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 506–513, Paris, France, 2018. doi: 10.5281/zenodo.1492463.
- [115] Simon Leglaive, Romain Hennequin, and Roland Badeau. Singing voice detection with deep recurrent neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 121–125, Brisbane, Australia, 2015. doi: 10.1109/ICASSP.2015.7177944.
- [116] Bernhard Lehner, Gerhard Widmer, and Reinhard Sonnleitner. On the reduction of false positives in singing voice detection. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 7480–7484, Florence, Italy, 2014. doi: 10.1109/ICASSP.2014.6855054.
- [117] Bernhard Lehner, Gerhard Widmer, and Sebastian Böck. A low-latency, real-time-capable singing voice detection method with LSTM recurrent neural networks. In *Proceedings of the European Signal Processing Conference (EUSIPCO)*, pages 21–25, Nice, France, 2015. doi: 10.1109/EUSIPCO.2015.7362337.
- [118] Bernhard Lehner, Jan Schlüter, and Gerhard Widmer. Online, loudness-invariant vocal detection in mixed music signals. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(8):1369–1380, 2018. doi: 10.1109/TASLP.2018.2825108.
- [119] Alexander Lerch, Claire Arthur, Ashis Pati, and Siddharth Gururani. Music performance analysis: A survey. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 33–43, Delft, The Netherlands, 2019. doi: 10.5281/zenodo.3527735.
- [120] Bochen Li and Aparna Kumar. Query by video: Cross-modal music retrieval. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 604–611, Delft, The Netherlands, 2019. doi: 10.5281/zenodo.3527882.
- [121] Yanxiong Li, Mingle Liu, Konstantinos Drossos, and Tuomas Virtanen. Sound event detection via dilated convolutional recurrent neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 286–290, 2020. doi: 10.1109/ICASSP40776.2020.9054433.
- [122] Che-Yuan Liang, Li Su, Yi-Hsuan Yang, and Hsin-Ming Lin. Musical offset detection of pitched instruments: The case of violin. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 281–287, Málaga, Spain, 2015. doi: 10.5281/zenodo.1416834.
- [123] Cynthia C. S. Liem and Chris Mostert. Can’t trust the feeling? How open data reveals unexpected behavior of high-level music descriptors. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 240–247, Montreal, Canada, 2020. doi: 10.5281/zenodo.4245413.
- [124] Cynthia C. S. Liem, Emilia Gómez, and George Tzanetakis. Multimedia technologies for enriched music performance, production, and consumption. *IEEE MultiMedia*, 24(1):20–23, 2017. doi: 10.1109/MMUL.2017.20.

## Bibliography

- [125] David Little and Bryan Pardo. Learning musical instruments from mixtures of audio with weak labels. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 127–132, Philadelphia, Pennsylvania, USA, 2008. doi: 10.5281/zenodo.1417815.
- [126] Shuo Liu, Adria Mallof-Ragolta, Emilia Parada-Cabaleiro, Kun Qian, Xin Jing, Alexander Kathan, Bin Hu, and Björn W. Schuller. Audio self-supervised learning: A survey. *Patterns*, 3(12):100616, 2022. doi: 10.1016/j.patter.2022.100616.
- [127] Gilles Louppe. *Understanding Random Forests: From Theory to Practice*. PhD thesis, University of Liege, Belgium, 2014. URL <https://hdl.handle.net/2268/170309>.
- [128] Mehran Maghoubi, Eugene Matthew Taranta, and Joseph LaViola. DeepNAG: Deep non-adversarial gesture generation. In *Proceedings of the International Conference on Intelligent User Interfaces (IUI)*, pages 213–223, College Station, Texas, USA, 2021. doi: 10.1145/3397481.3450675.
- [129] Thor Magnusson. Musical organics: A heterarchical approach to digital organology. *Journal of New Music Research*, 46(3):286–303, 2017. doi: 10.1080/09298215.2017.1353636.
- [130] Ilaria Manco, Emmanouil Benetos, Elio Quinton, and György Fazekas. Learning music audio representations via weak language supervision. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 456–460, Singapore, 2022. doi: 10.1109/ICASSP43922.2022.9746996.
- [131] Ethan Manilow, Gordon Wichern, and Jonathan Le Roux. Hierarchical musical instrument separation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 376–383, Montreal, Canada, 2020. doi: 10.5281/zenodo.4245448.
- [132] Ethan Manilow, Patrick O’Reilly, Prem Seetharaman, and Bryan Pardo. Source separation by steering pretrained music models. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 126–130, Singapore, 2022. doi: 10.1109/ICASSP43922.2022.9747909.
- [133] Matthew C. McCallum. Unsupervised learning of deep features for music segmentation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 346–350, Brighton, UK, 2019. doi: 10.1109/ICASSP.2019.8683407.
- [134] Matthew C. McCallum, Filip Korzeniowski, Sergio Oramas, Fabien Gouyon, and Andreas Ehmann. Supervised and unsupervised learning of audio representations for music understanding. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 256–263, Bengaluru, India, 2022. doi: 10.5281/zenodo.7316644.
- [135] Mark S Melenhorst, Ron van der Sterren, Andreas Arzt, Agustín Martorell, and Cynthia C.S. Liem. A tablet app to enrich the live and post-live experience of classical concerts. In *Proceedings of the International Workshop on Interactive Content Consumption (WSICC)*, Brussels, Belgium, 2015. URL <https://ceur-ws.org/Vol-1516/p4.pdf>.
- [136] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. Metrics for polyphonic sound event detection. *Applied Sciences*, 6(6):162, 2016. doi: 10.3390/app6060162.
- [137] Annamaria Mesaros, Toni Heittola, Tuomas Virtanen, and Mark D. Plumbley. Sound event detection: A tutorial. *IEEE Signal Processing Magazine*, 38(5):67–83, 2021. doi: 10.1109/MSP.2021.3090678.



- [138] Alessandro Ilic Mezza, Emanuël A. P. Habets, Meinard Müller, and Augusto Sarti. Unsupervised domain adaptation for acoustic scene classification using band-wise statistics matching. In *Proceedings of the European Signal Processing Conference (EUSIPCO)*, pages 11–15, Amsterdam, The Netherlands, 2020. doi: 10.23919/Eusipco47968.2020.9287533.
- [139] Stylianos I. Mimilakis, Christof Weiß, Vlora Arifi-Müller, Jakob Abeßer, and Meinard Müller. Cross-version singing voice detection in opera recordings: Challenges for supervised learning. In *Machine Learning and Knowledge Discovery in Databases – Proceedings of the International Workshops of ECML PKDD 2019, Part II*, volume 1168 of *Communications in Computer and Information Science*, pages 429–436, Würzburg, Germany, 2019. doi: 10.1007/978-3-030-43887-6\_35.
- [140] Marius Miron, Julio J. Carabias-Orti, Juan J. Bosch, Emilia Gómez, and Jordi Janer. Score-informed source separation for multichannel orchestral recordings. *Journal of Electrical and Computer Engineering*, 2016 (8363507), 2016. doi: 10.1155/2016/8363507.
- [141] Veronica Morfi, Yves Bas, Hanna Pamula, Hervé Glotin, and Dan Stowell. NIPS4Bplus: A richly annotated birdsong audio dataset. *PeerJ Computer Science*, 5:e223, 2019. doi: 10.7717/peerj-cs.223.
- [142] Yuko Morimoto, Toru Kamekawa, and Atsushi Marui. Verbal effect on memorisation and recognition of Wagner’s leitmotifs. In *Triennial Conference of the European Society for the Cognitive Sciences of Music (ESCOM)*, pages 357–361, Jyväskylä, Finland, 2009. URL <https://jyx.jyu.fi/handle/123456789/20903>.
- [143] Daniel Müllensiefen, David Baker, Christophe Rhodes, Tim Crawford, and Laurence Dreyfus. Recognition of leitmotives in Richard Wagner’s music: An item response theory approach. In *Analysis of Large and Complex Data*, pages 473–483. Springer, Cham, Switzerland, 2016. doi: 10.1007/978-3-319-25226-1.
- [144] Matthias Müller, Thilo Schulz, Tatiana Ermakova, and Philipp P. Caffier. Lyric or dramatic - vibrato analysis for voice type classification in professional opera singers. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:943–955, 2021. doi: 10.1109/TASLP.2021.3054299.
- [145] Meinard Müller. *Fundamentals of Music Processing – Using Python and Jupyter Notebooks*. Springer Verlag, 2nd edition, 2021. ISBN 978-3-030-69807-2. doi: 10.1007/978-3-030-69808-9.
- [146] Meinard Müller, Thomas Prätzlich, and Jonathan Driedger. A cross-version approach for stabilizing tempo-based novelty detection. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 427–432, Porto, Portugal, 2012. doi: 10.5281/zenodo.1417753.
- [147] Meinard Müller, Yigitcan Özer, Michael Krause, Thomas Prätzlich, and Jonathan Driedger. Sync Toolbox: A Python package for efficient, robust, and accurate music synchronization. *Journal of Open Source Software (JOSS)*, 6(64):3434:1–4, 2021. doi: 10.21105/joss.03434.
- [148] Juhan Nam, Keunwoo Choi, Jongpil Lee, Szu-Yu Chou, and Yi-Hsuan Yang. Deep learning for audio-based music classification and tagging: Teaching computers to distinguish rock from bach. *IEEE Signal Processing Magazine*, 36(1):41–51, 2019. doi: 10.1109/MSP.2018.2874383.
- [149] Inês Nolasco and Dan Stowell. Rank-based loss for learning hierarchical representations. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 3623–3627, Singapore, 2022. doi: 10.1109/ICASSP43922.2022.9746907.

## Bibliography

- [150] Tin Lay Nwe and Ye Wang. Automatic detection of vocal segments in popular songs. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 138–144, Barcelona, Spain, 2004. doi: 10.5281/zenodo.1417846.
- [151] Hiroshi G. Okuno, Tetsuya Ogata, and Kazunori Komatani. Computational auditory scene analysis and its application to robot audition: Five years experience. In *International Conference on Informatics Research for Development of Knowledge Society Infrastructure (ICKS)*, pages 69–76, Kyoto, Japan, 2007. doi: 10.1109/ICKS.2007.7.
- [152] Yigitcan Özer, Michael Krause, and Meinard Müller. Using the sync toolbox for an experiment on high-resolution music alignment. In *Demos and Late Breaking News of the International Society for Music Information Retrieval Conference (ISMIR)*, Online, 2021. URL <https://archives.ismir.net/ismir2021/latebreaking/000025.pdf>.
- [153] Kevin R. Page, Terhi Nurmikko-Fuller, Carolin Rindfleisch, David M. Weigl, Richard Lewis, Laurence Dreyfus, and David De Roure. A toolkit for live annotation of opera performance: Experiences capturing Wagner’s Ring cycle. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 211–217, Málaga, Spain, 2015. doi: 10.5281/zenodo.1415582.
- [154] Elias Pampalk, Arthur Flexer, and Gerhard Widmer. Improvements of audio-based music similarity and genre classification. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 628–633, London, UK, 2005. doi: 10.5281/zenodo.1418083.
- [155] Emilia Parada-Cabaleiro, Maximilian Schmitt, Anton Batliner, Simone Hantke, Giovanni Costantini, Klaus R. Scherer, and Björn W. Schuller. Identifying emotions in opera singing: Implications of adverse acoustic conditions. In *Proceedings of the International Conference on Digital Libraries for Musicology (DLfM)*, pages 376–382, Paris, France, 2018. doi: 10.5281/zenodo.1492429.
- [156] Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. SpecAugment: A simple data augmentation method for automatic speech recognition. In *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech)*, pages 2613–2617, Graz, Austria, 2019. doi: 10.21437/Interspeech.2019-2680.
- [157] Kailash Patil and Mounya Elhilali. Biomimetic spectro-temporal features for music instrument recognition in isolated notes and solo phrases. *EURASIP Journal on Audio, Speech, and Music Processing*, 2015(27), 2015. doi: 10.1186/s13636-015-0070-9.
- [158] Johan Pauwels, Ken O’Hanlon, Emilia Gómez, and Mark B. Sandler. 20 years of automatic chord recognition from audio. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 54–63, Delft, The Netherlands, 2019. doi: 10.5281/zenodo.3527739.
- [159] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011. URL <http://jmlr.org/papers/v12/pedregosa11a.html>.

- [160] Graham E. Poliner and Daniel P.W. Ellis. A discriminative model for polyphonic piano transcription. *EURASIP Journal on Advances in Signal Processing*, 2007(1), 2007. doi: 10.1155/2007/48317.
- [161] Archontis Politis, Annamaria Mesaros, Sharath Adavanne, Toni Heittola, and Tuomas Virtanen. Overview and evaluation of sound event localization and detection in DCASE 2019. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:684–698, 2021. doi: 10.1109/TASLP.2020.3047233.
- [162] Archontis Politis, Kazuki Shimada, Parthasaarathy Sudarsanam, Sharath Adavanne, Daniel Krause, Yuichiro Koyama, Naoya Takahashi, Shusuke Takahashi, Yuki Mitsufuji, and Tuomas Virtanen. STARSS22: A dataset of spatial recordings of real scenes with spatiotemporal annotations of sound events. In *Proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, Nancy, France, 2022. Tampere University. URL [https://dcase.community/documents/workshop2022/proceedings/DCASE2022Workshop\\_Politis\\_51.pdf](https://dcase.community/documents/workshop2022/proceedings/DCASE2022Workshop_Politis_51.pdf).
- [163] Edward Polrolniczak and Michal Kramarczyk. Estimation of singing voice types based on voice parameters analysis. In *Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*, pages 63–68, Poznan, Poland, 2017. doi: 10.23919/SPA.2017.8166839.
- [164] Jordi Pons, Thomas Lidy, and Xavier Serra. Experimenting with musically motivated convolutional neural networks. In *Proceedings of International Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 1–6, Bucharest, Romania, 2016. doi: 10.1109/CBMI.2016.7500246.
- [165] Jordi Pons, Oriol Nieto, Matthew Prockup, Erik Schmidt, Andreas Ehmann, and Xavier Serra. End-to-end learning for music audio tagging at scale. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 637–644, Paris, France, 2018. doi: 10.5281/zenodo.1492497.
- [166] Thomas Prätzlich, Meinard Müller, Benjamin W. Bohl, and Joachim Veit. Freischütz Digital: Demos of audio-related contributions. In *Demos and Late Breaking News of the International Society for Music Information Retrieval Conference (ISMIR)*, Málaga, Spain, 2015. URL <https://ismir2015.ismir.net/LBD/LBD18.pdf>.
- [167] Thomas Prätzlich, Jonathan Driedger, and Meinard Müller. Memory-restricted multiscale dynamic time warping. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 569–573, Shanghai, China, March 2016. doi: 10.1109/ICASSP.2016.7471739.
- [168] Katharina Prinz, Arthur Flexer, and Gerhard Widmer. On end-to-end white-box adversarial attacks in music information retrieval. *Transactions of the International Society for Music Information Retrieval (TISMIR)*, 4(1):93, 2021. doi: 10.5334/tismir.85.
- [169] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 8748–8763, Virtual, 2021. URL <https://proceedings.mlr.press/v139/radford21a>.
- [170] Mathieu Ramona, Gaël Richard, and Bertrand David. Vocal detection in music with support vector machines. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1885–1888, Las Vegas, Nevada, USA, 2008. doi: 10.1109/ICASSP.2008.4518002.

## Bibliography

- [171] Lise Regnier and Geoffroy Peeters. Singing voice detection in music tracks using direct voice vibrato detection. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1685–1688, Taipei, Taiwan, 2009. doi: 10.1109/ICASSP.2009.4959926.
- [172] Francisco Rodríguez-Algarra, Bob L. Sturm, and Hugo Maruri-Aguilar. Analysing scattering-based music content analysis systems: Where’s the music? In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 344–350, New York City, New York, USA, 2016. doi: 10.5281/zenodo.1414723.
- [173] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020. doi: 10.1162/tacl\_a\_00349.
- [174] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685, New Orleans, LA, USA, 2022. doi: 10.1109/CVPR52688.2022.01042.
- [175] Aaqib Saeed, David Grangier, and Neil Zeghidour. Contrastive learning of general-purpose audio representations. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 3875–3879, Toronto, Canada, 2021. doi: 10.1109/ICASSP39728.2021.9413528.
- [176] Justin Salamon and Juan Pablo Bello. Unsupervised feature learning for urban sound classification. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 171–175, South Brisbane, Australia, 2015. doi: 10.1109/ICASSP.2015.7177954.
- [177] Justin Salamon, Rachel M. Bittner, Jordi Bonada, Juan José Bosch Vicente, Emilia Gómez, and Juan P. Bello. An analysis/synthesis framework for automatic f0 annotation of multitrack datasets. In *Proceedings of International Society for Music Information Retrieval Conference (ISMIR)*, pages 71–78, Suzhou, China, 2017. doi: 10.5281/zenodo.1415588.
- [178] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2488–2498, Montréal, Canada, 2018. URL <https://proceedings.neurips.cc/paper/2018/hash/905056c1ac1dad141560467e0a99e1cf-Abstract.html>.
- [179] Saurjya Sarkar, Emmanouil Benetos, and Mark B. Sandler. EnsembleSet: A new high quality dataset for chamber ensemble separation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, Bengaluru, India, 2022. doi: 10.5281/zenodo.7316740.
- [180] Axel Sauer and Andreas Geiger. Counterfactual generative networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Virtual, 2021. URL <https://openreview.net/forum?id=BXewfAYMmJw>.
- [181] Jan Schlüter and Thomas Grill. Exploring data augmentation for improved singing voice detection with neural networks. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 121–126, Málaga, Spain, 2015. doi: 10.5281/zenodo.1417745.

- [182] Jan Schlüter and Bernhard Lehner. Zero-mean convolutions for level-invariant singing voice detection. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 321–326, Paris, France, 2018. doi: 10.5281/zenodo.1492413.
- [183] Florian Scholz, Igor Vatolkin, and Günter Rudolph. Singing voice detection across different music genres. In *Proceedings of the AES International Conference on Semantic Audio*, pages 140–147, Erlangen, Germany, 2017. URL <https://www.aes.org/e-lib/browse.cfm?elib=18771>.
- [184] Christian Schörkhuber and Anssi P. Klapuri. Constant-Q transform toolbox for music processing. In *Proceedings of the Sound and Music Computing Conference (SMC)*, Barcelona, Spain, 2010. doi: 10.5281/zenodo.849741.
- [185] Hendrik Schreiber, Christof Weiß, and Meinard Müller. Local key estimation in classical music audio recordings: A cross-version study on Schubert’s Winterreise. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 501–505, Barcelona, Spain, 2020. doi: 10.1109/ICASSP40776.2020.9054642.
- [186] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, Boston, Massachusetts, USA, 2015. doi: 10.1109/CVPR.2015.7298682.
- [187] Kilian Schulze-Forster, Clement S. J. Doire, Gaël Richard, and Roland Badeau. Phoneme level lyrics alignment and text-informed singing voice separation. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 29:2382–2395, 2021. doi: 10.1109/TASLP.2021.3091817.
- [188] Simon Schwär, Michael Krause, Michael Fast, Sebastian Rosenzweig, Frank Scherbaum, and Meinard Müller. Singing voice reconstruction from larynx microphone signals. *Submitted for publication*, 2023.
- [189] Carlos Nascimento Silla, Jr. and Alex Alves Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2):31–72, 2011. doi: 10.1007/s10618-010-0175-9.
- [190] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, California, USA, 2015. URL <http://arxiv.org/abs/1409.1556>.
- [191] Janne Spijkervet and John Ashley Burgoyne. Contrastive learning of musical representations. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 673–681, Online, 2021. doi: 10.5281/zenodo.5624573.
- [192] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, June 2014. URL <https://jmlr.org/papers/v15/srivastava14a.html>.
- [193] Daniel Stoller, Simon Durand, and Sebastian Ewert. End-to-end lyrics alignment for polyphonic music using an audio-to-character recognition model. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 181–185, Brighton, UK, 2019. doi: 10.1109/ICASSP.2019.8683470.

## Bibliography

- [194] Dan Stowell, Dimitrios Giannoulis, Emmanouil Benetos, Mathieu Lagrange, and Mark D. Plumbley. Detection and classification of acoustic scenes and events. *IEEE Transactions on Multimedia*, 17(10):1733–1746, 2015. doi: 10.1109/TMM.2015.2428998.
- [195] Dan Stowell, Mike Wood, Yannis Stylianou, and Hervé Glotin. Bird detection in audio: A survey and a challenge. In *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, Vietri sul Mare, Salerno, Italy, 2016. doi: 10.1109/MLSP.2016.7738875.
- [196] Bob L. Sturm. A simple method to determine if a music information retrieval system is a “horse”. *IEEE Transactions on Multimedia*, 16(6):1636–1644, 2014. doi: 10.1109/TMM.2014.2330697.
- [197] Bob L. Sturm. The "horse" inside: Seeking causes behind the behaviors of music content analysis systems. *Computers in Entertainment*, 14(2):3:1–3:32, 2016. doi: 10.1145/2967507.
- [198] Bob L. Sturm, Corey Kereliuk, and Jan Larsen. ¿El caballo viejo? Latin genre recognition with deep learning and spectral periodicity. In *International Conference on Mathematics and Computation in Music (MCM)*, pages 335–346, London, UK, 2015. doi: 10.1007/978-3-319-20603-5\_34.
- [199] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Banff, AB, Canada, 2014. URL [https://openreview.net/forum?id=kk1r\\_MTHMRQjG](https://openreview.net/forum?id=kk1r_MTHMRQjG).
- [200] Michael Taenzer, Jakob Abeßer, Stylianos I. Mimilakis, Christof Weiß, Hanna Lukashevich, and Meinard Müller. Investigating CNN-based instrument family recognition for Western classical music recordings. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 612–619, Delft, The Netherlands, 2019. doi: 10.5281/zenodo.3527884.
- [201] Zheng Tang and Dawn A. A. Black. Melody extraction from polyphonic audio of Western opera: A method based on detection of the singer’s formant. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 161–166, Taipei, Taiwan, October 2014. doi: 10.5281/zenodo.1416714.
- [202] Hiroko Terasawa, Malcolm Slaney, and Jonathan Berger. The thirteen colors of timbre. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 323–326, New Paltz, NY, USA, 2005. doi: 10.1109/ASPAA.2005.1540234.
- [203] John Thickstun, Zaïd Harchaoui, and Sham M. Kakade. Learning features of music from scratch. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Toulon, France, 2017. URL <https://openreview.net/forum?id=rkFBJv9gg>.
- [204] Verena Thomas, Christian Fremerey, David Damm, and Michael Clausen. SLAVE: a Score-Lyrics-Audio-Video-Explorer. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 717–722, Kobe, Japan, 2009. doi: 10.5281/zenodo.1418029.
- [205] Carl Thomé, Sebastian Piwell, and Oscar Utterbäck. Musical audio similarity with self-supervised convolutional neural networks. In *Demos and Late Breaking News of the International Society for Music Information Retrieval Conference (ISMIR)*, Online, 2021. URL <https://archives.ismir.net/ismir2021/latebreaking/000012.pdf>.

- [206] Steven K. Tjoa and K. J. Ray Liu. Musical instrument recognition using biologically inspired filtering of temporal dictionary atoms. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 435–440, 2010. doi: 10.5281/zenodo.1416166.
- [207] Shuhei Tsuchida, Satoru Fukayama, Masahiro Hamasaki, and Masataka Goto. AIST dance video database: Multi-genre, multi-dancer, and multi-camera database for dance information processing. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 501–510, Delft, The Netherlands, 2019. doi: 10.5281/zenodo.3527854.
- [208] Nicolas Turpault, Romain Serizel, Justin Salamon, and Ankit Parag Shah. Sound event detection in domestic environments with weakly labeled data and soundscape synthesis. In *Proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, pages 253–257, New York, NY, USA, 2019. URL [http://dcase.community/documents/workshop2019/proceedings/DCASE2019Workshop\\_Turpault\\_44.pdf](http://dcase.community/documents/workshop2019/proceedings/DCASE2019Workshop_Turpault_44.pdf).
- [209] Shankar Vembu and Stephan Baumann. Separation of vocals from polyphonic audio recordings. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 337–344, London, UK, 2005. doi: 10.5281/zenodo.1414852.
- [210] Tuomas Virtanen, Mark D. Plumbley, and Dan Ellis. *Computational Analysis of Sound Scenes and Events*. Springer, 2018. ISBN 978-3-319-63449-4. doi: 10.1007/978-3-319-63450-0.
- [211] Marcel A. Vélez Vásquez and John Ashley Burgoyne. Tailed U-Net: Multi-scale music representation learning. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 67–75, Bengaluru, India, 2022. doi: 10.5281/zenodo.7316596.
- [212] Richard Wagner. On the application of music to the drama. In *Prose Works*, pages 175–191. Broude Brothers, New York, 1966. Translation of the original edition from 1879.
- [213] Richard Wagner. *Opera and Drama*. University of Nebraska Press, 1995. ISBN 978-0-803-29765-4. Translation of the original edition from 1851.
- [214] Richard Wagner. *Der Ring des Nibelungen. Vollständiger Text mit Notentafeln der Leitmotive*. Schott Music, Mainz, 2013. ISBN 978-3-254-08229-9. Reprint of the original edition from 1913 (Ed. Julius Burghold).
- [215] Keigo Wakayama and Shoichiro Saito. CNN-Transformer with self-attention network for sound event detection. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 806–810, Singapore, 2022. doi: 10.1109/ICASSP43922.2022.9747762.
- [216] Ju-Chiang Wang, Jordan B. L. Smith, Wei Tsung Lu, and Xuchen Song. Supervised metric learning for music structure features. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 730–737, Online, 2021. doi: 10.5281/zenodo.5624427.
- [217] Ye Wang, Min-Yen Kan, Tin Lay Nwe, Arun Shenoy, and Jun Yin. LyricAlly: Automatic synchronization of acoustic musical signals and textual lyrics. In *Proceedings of the ACM International Conference on Multimedia*, pages 212–219, New York, NY, USA, 2004. doi: 10.1145/1027527.1027576.

## Bibliography

- [218] Yu Wang, Justin Salamon, Nicholas J. Bryan, and Juan Pablo Bello. Few-shot sound event detection. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 81–85, Barcelona, Spain, 2020. doi: 10.1109/ICASSP40776.2020.9054708.
- [219] Yuxuan Wang, Pascal Getreuer, Thad Hughes, Richard F. Lyon, and Rif A. Saurous. Trainable frontend for robust and far-field keyword spotting. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 5670–5674, New Orleans, USA, 2017. doi: 10.1109/ICASSP.2017.7953242.
- [220] Jonatas Wehrmann, Ricardo Cerri, and Rodrigo C. Barros. Hierarchical multi-label classification networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 5225–5234, Stockholm, Sweden, 2018. URL <https://proceedings.mlr.press/v80/wehrmann18a.html>.
- [221] Christof Weiß and Geoffroy Peeters. Learning multi-pitch estimation from weakly aligned score-audio pairs using a multi-label CTC loss. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 121–125, New Paltz, USA, 2021. doi: 10.1109/WASPAA52581.2021.9632740.
- [222] Christof Weiß, Vlora Arifi-Müller, Thomas Prätzlich, Rainer Kleinertz, and Meinard Müller. Analyzing measure annotations for Western classical music recordings. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 517–523, New York, USA, 2016. doi: 10.5281/zenodo.1417449.
- [223] Christof Weiß, Frank Zalkow, Meinard Müller, Stephanie Klauk, and Rainer Kleinertz. Versionsübergreifende Visualisierung harmonischer Verläufe: Eine Fallstudie zu Wagners Ring-Zyklus. In *Proceedings of the Jahrestagung der Gesellschaft für Informatik (GI)*, pages 205–217, Chemnitz, Germany, 2017. doi: 10.18420/in2017\_14.
- [224] Christof Weiß, Hendrik Schreiber, and Meinard Müller. Local key estimation in music recordings: A case study across songs, versions, and annotators. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2919–2932, 2020. doi: 10.1109/TASLP.2020.3030485.
- [225] Christof Weiß, Frank Zalkow, Vlora Arifi-Müller, Meinard Müller, Hendrik Vincent Koops, Anja Volk, and Harald Grohgan. Schubert Winterreise dataset: A multimodal scenario for music analysis. *ACM Journal on Computing and Cultural Heritage (JOCCH)*, 14(2):25:1–18, 2021. doi: 10.1145/3429743.
- [226] Christof Weiß, Johannes Zeitler, Tim Zunner, Florian Schuberth, and Meinard Müller. Learning pitch-class representations from score–audio pairs of classical music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 746–753, Online, 2021. doi: 10.5281/zenodo.5624549.
- [227] Christof Weiß, Vlora Arifi-Müller, Michael Krause, Frank Zalkow, Stephanie Klauk, Rainer Kleinertz, and Meinard Müller. Wagner Ring Dataset: A complex opera scenario for music processing and computational musicology. *Transactions of the International Society for Music Information (TISMIR)*, 2023.
- [228] Felix Weninger, Jean-Louis Durrieu, Florian Eyben, Gaël Richard, and Björn W. Schuller. Combining monaural source separation with long short-term memory for increased robustness in vocalist gender recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2196–2199, Prague, Czech Republic, 2011. doi: 10.1109/ICASSP.2011.5946764.



- [229] Curtis Wigington, Brian L. Price, and Scott Cohen. Multi-label connectionist temporal classification. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*, pages 979–986, Sydney, Australia, 2019. doi: 10.1109/ICDAR.2019.00161.
- [230] Chih-Wei Wu, Christian Dittmar, Carl Southall, Richard Vogl, Gerhard Widmer, Jason Hockman, Meinard Müller, and Alexander Lerch. A review of automatic drum transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(9):1457–1483, 2018. doi: 10.1109/TASLP.2018.2830113.
- [231] Ho-Hsiang Wu, Chieh-Chi Kao, Qingming Tang, Ming Sun, Brian McFee, Juan Pablo Bello, and Chao Wang. Multi-task self-supervised pre-training for music classification. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 556–560, Toronto, Canada, 2021. doi: 10.1109/ICASSP39728.2021.9414405.
- [232] Yu-Te Wu, Berlin Chen, and Li Su. Multi-instrument automatic music transcription with self-attention-based instance segmentation. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 28:2796–2809, 2020. doi: 10.1109/TASLP.2020.3030482.
- [233] Xianjun Xia, Roberto Togneri, Ferdous Sohel, Yuanjun Zhao, and Defeng Huang. A survey: Neural network-based deep learning for acoustic event detection. *Circuits, Systems, and Signal Processing*, 38(8): 3433–3453, 2019. doi: 10.1007/s00034-019-01094-1.
- [234] Yong Xu, Qiang Huang, Wenwu Wang, and Mark D. Plumbley. Hierarchical learning for DNN-based acoustic scene classification. In *Proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, pages 110–114, Budapest, Hungary, 2016. URL <https://dcase.community/documents/workshop2016/proceedings/Xu-a-DCASE2016workshop.pdf>.
- [235] Wanqian Yang, Polina Kirichenko, Micah Goldblum, and Andrew Gordon Wilson. Chroma-VAE: Mitigating shortcut learning with generative classifiers. In *Advances in Neural Information Processing Systems (NeurIPS)*, New Orleans, LA, USA, 2022. URL <https://openreview.net/forum?id=WWVcsfI0jGH>.
- [236] Frank Zalkow and Meinard Müller. Learning low-dimensional embeddings of audio shingles for cross-version retrieval of classical music. *Applied Sciences*, 10(1), 2020. doi: 10.3390/app10010019.
- [237] Frank Zalkow, Christof Weiß, and Meinard Müller. Exploring tonal-dramatic relationships in Richard Wagner’s Ring cycle. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 642–648, Suzhou, China, 2017. doi: 10.5281/zenodo.1415760.
- [238] Frank Zalkow, Christof Weiß, Thomas Prätzlich, Vlora Arifi-Müller, and Meinard Müller. A multi-version approach for transferring measure annotations between music recordings. In *Proceedings of the AES International Conference on Semantic Audio*, pages 148–155, Erlangen, Germany, 2017. URL <https://www.aes.org/e-lib/browse.cfm?elib=18772>.
- [239] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Toulon, France, 2017. URL <https://openreview.net/forum?id=Sy8gdB9xx>.
- [240] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, Vancouver, BC, Canada, 2018. URL <https://openreview.net/forum?id=r1Ddp1-Rb>.

## Bibliography

- [241] Huan Zhang, Jingjing Tang, Syed Rifat Mahmud Rafee, Simon Dixon, and György Fazekas. ATEPP: A dataset of automatically transcribed expressive piano performance. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 446–453, Bengaluru, India, 2022. doi: 10.5281/zenodo.7342764.
- [242] Xulong Zhang, Yi Yu, Yongwei Gao, Xi Chen, and Wei Li. Research on singing voice detection based on a long-term recurrent convolutional network with vocal separation and temporal smoothing. *Electronics*, 9(9): 1458, 2020. doi: 10.3390/electronics9091458.
- [243] Arman Zharmagambetov, Qingming Tang, Chieh-Chi Kao, Qin Zhang, Ming Sun, Viktor Rozgic, Jasha Droppo, and Chao Wang. Improved representation learning for acoustic event classification using tree-structured ontology. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 321–325, Singapore, 2022. doi: 10.1109/ICASSP43922.2022.9746266.
- [244] Zhi Zhong, Masato Hirano, Kazuki Shimada, Kazuya Tateishi, Shusuke Takahashi, and Yuki Mitsufuji. An attention-based approach to hierarchical multi-label music instrument classification. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Rhodes Island, Greece, 2023. Uploaded to arXiv (2302.08136) on 16.02.2023. doi: 10.48550/arXiv.2302.08136.
- [245] Fu Zih-Sing and Li Su. Hierarchical classification networks for singing voice segmentation and transcription. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 900–907, Delft, The Netherlands, 2019. doi: 10.5281/zenodo.3527960.