School of Electronic Engineering and Computer Science
Queen Mary University of London

Deep Learning Methods for Instrument Separation and Recognition

Carlos Pedro Vianna Lordelo

PhD thesis

Submitted in partial fulfillment of the requirements
of the Degree of Doctor of Philosophy

December 2022

Statement of Originality

I, Carlos Pedro Vianna Lordelo, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

The copyright of this thesis rests with the author and no quotation from it or information derived from it may be published without the prior written consent of the author.

Signature: Carlos Lordelo
Date: 19/12/2022

Details of collaboration and publications: see Section 1.6

Abstract

This thesis explores deep learning methods for timbral information processing in polyphonic music analysis. It encompasses two primary tasks: Music Source Separation (MSS) and Instrument Recognition, with focus on applying domain knowledge and utilising dense arrangements of skip-connections in the frameworks in order to reduce the number of trainable parameters and create more efficient models. Musically-motivated Convolutional Neural Network (CNN) architectures are introduced, emphasizing kernels with vertical, square, and horizontal shapes. This design choice allows for the extraction of essential harmonic and percussive features, which enhances the discrimination of different instruments. Notably, this methodology proves valuable for Harmonic-Percussive Source Separation (HPSS) and instrument recognition tasks.

A significant challenge in MSS is generalising to new instrument types and music styles. To address this, a versatile framework for adversarial unsupervised domain adaptation for source separation is proposed, particularly beneficial when labeled data for specific instruments is unavailable. The curation of the Tap & Fiddle dataset is another contribution of the research, offering mixed and isolated stem recordings of traditional Scandinavian fiddle tunes, along with foot-tapping accompaniments, fostering research in source separation and metrical expression analysis within these musical styles.

Since our perception of timbre is affected in different ways by transient and stationary parts of sound, the research investigates the potential of Transient-Stationary-Noise Decomposition (TSND) as a preprocessing step for frame-level recognition. A method that performs TSND of spectrograms and feeds the decomposed spectrograms to a neural classifier is proposed. Furthermore, this thesis introduces a novel deep learning-based approach for pitch streaming, treating the task as a note-level instrument classification. Such an approach is modular, meaning that it can also successfully stream predicted note-events and not only labelled ground-truth note-event information to corresponding instruments. Therefore, the proposed pitch streaming method enables third-party multi-pitch estimation algorithms to perform multi-instrument AMT.

Keywords— Source Separation, Instrument Recognition, Domain Adaptation, Domain Knowledge, Kernel Shapes, 3W-MDenseNet, Efficient models

Acknowledgements

It is fair to start by giving special acknowledgements to all of my three supervisors. First and foremost, I should thank my primary joint supervisor, Dr. Emmanouil Benetos, who not only allowed me pursuing my research with virtually unlimited freedom, but also for always being expeditiously responsive in any situation. Even during a global pandemic. Thank you very much for all your quick reviews and great feedback of my articles, reports and thesis text. Also, my joint supervisor from QMUL, Prof. Simon Dixon, thanks for all your great support, hand-picked suggestions that made my research more impactful and for making every ISMIR conference more enjoyable participating in our jam sessions and late night celebrations. Lastly, I could not have asked for a better industry supervisor. Thank you, Sven Ahlbäck, for interesting industrial applications to work on, for allowing my research to become part of a final product and for all your supervision regarding psychological and cognitive aspects of music.

This PhD project is the result of a partnership between DoReMIR Music Research AB[1] and Queen Mary University of London. This partnership was made possible due to MIP-frontiers project[2], a European training network for MIR researchers that aims to train a new generation of researchers, bringing together 4 universities, 3 industrial beneficiaries and 15 PhD students. Therefore, it would not be fair to not mention the greatest project manager of all times: Alvaro Bort. Thanks for your support in all administrative issues ranging from planning of virtual training sessions, to arranging travels, conferences and organising workshops. Furthermore, all the other peers from MIP-Frontiers should also be acknowledged. Thanks for all the meetings, brainstorming sessions, criticisms, text reviews and collaborative assignments we did together. Those were all appreciated and important for my research.

I should also be thankful to everyone in C4DM at QMUL and everyone from Doremir Music Research AB for the great time I spent in both places, thanks

[1] `https://doremir.com/`

[2] `https://https://mip-frontiers.eu/`

to all who helped in any way shape or form during my research. In particular, thanks Patrik Ohlsson for all your help, particularly, in the beginning of the PhD regarding Python programming and tensorflow. Moreover, thank you, Sven Emtell, for all your time and efforts to practice Swedish with me and making my adaptation period in Sweden easier.

Last but not least, I should also be thankful to my friends and family, who promptly understood the importance of allowing me to follow my curiosity and dreams wherever it may lead. Thanks for always being supportive, particularly during a global pandemic that directly affected our lives in 2020 and 2021. Last but not least, special thanks should be given to my partner, Anna La Marca, for leaving all their stuff behind and accepting coming to this unexpected journey in Europe with me. Thanks you for not allowing me to go crazy during lock-downs in a dark Swedish winter.

This PhD programme has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 765068. The research was done in collaboration with Doremir Music Research AB as part of the MIP-Frontiers Project.

Contents

List of Figures

List of Tables

List of abbreviations

ADSR	Attack, Decay, Sustain and Release
ADT	Automatic Drum Transcription
AMT	Automatic Music Transcription
ANSI	American National Standards Institute
ASA	Auditory Scene Analysis
CDMA	Code-Division Multiple Access
CMN	Western Common Music Notation System
CNN	Convolutional Neural Network
DFT	Discrete Fourier Transform
DNN	Deep Neural Network
FFT	Fast Fourier Transform
GMM	Gaussian Mixture Models
HMM	Hidden Markov Models
HMRF	Hidden Markov Random Fields
HPSS	Harmonic-Percussive Source Separation
SI	International System of Units
IAD	Instrument Activity Detection
IBM	Ideal Binary Masking
IRM	Ideal Ratio Masking
LReLU	Leaky Rectified Linear Unit
MDX	Music Demixing Challenge
MFCC	Mel-Frequency Cepstrum Coefficients
MIR	Music Information Retrieval
MLP	Multi-Layer Perceptron
MODGDF	Modified Group Delay Feature
MPE	Multi-Pitch Estimation
MSS	Music Source Separation
MWF	Multi-channel Wiener Filtering
NLP	Natural Language Processing
PCA	Principal Component Analysis

ReLU	Rectified Linear Unit
SDA	Supervised Domain Adaptation
SiSEC	Signal Separation Evaluation Campaign
STFT	Short Time Fourier Transform
STN	Sines + Transients + Noise
SVM	Support Vector Machine
TSND	Transient-Stationary-Noise Decomposition
TSS	Transient-Stationary Separation
UDA	Unsupervised Domain Adaptation

Chapter 1

Introduction

This thesis studies deep learning methods for source separation and instrument recognition with the main focus on proposing musically motivated Convolutional Neural Networks (CNN) architectures for each of those tasks. This chapter starts with an overview of the research work in Section 1.1. Afterwards, the context where the research project is inserted in is explained in Section 1.2 while the research objectives are discussed in Section 1.3. The overall structure of the thesis is provided in Section 1.4 and the main contributions of the research are pinpointed in Section 1.5. Finally, in Section 1.6, publications and invited talks associated with the thesis are listed.

1.1 Overview

With the recent increase of computational power, state-of-the-art performance in many applications is dramatically improved with the help of Deep Neural Networks (DNNs). Every day, novel, powerful, and sophisticated deep learning architectures are proposed. Consequently, as researchers have been trying to address progressively more complicated and complex tasks in an end-to-end fashion using DNNs, their architectures have become increasingly larger and deeper.

Even though larger models tend to perform well, they might not necessarily be efficient enough for direct deployment in most applications. Certain deep learning applications need to run in real-time and/or on low-powered devices, such as smartphones, in-car computers or sensors. Furthermore, training and deploying very deep and complex DNNs can be very costly [Menghani, 2021]. For instance, GPT-3 [Brown et al., 2020], a recently developed transformer architecture based on attention layers, comprises around 175 billion parameters, and costs millions of dollars to train [Brown et al., 2020]. Therefore, it is im-

portant to optimise models for more efficiently perform desired tasks.

The strategy of utilising multiple kernel shapes in CNNs is pointed out in recent works as an structured way to exploit the network capacity towards building more efficient musically motivated architectures [Pons et al., 2016; Phan et al., 2016; Pons and Serra, 2017; Pons et al., 2017]. Instead of building increasingly larger architectures hoping to obtain better results, the main focus of this thesis is to propose novel musically motivated CNN architectures that can be smaller and efficient. Initial research addresses the task of *Harmonic-Percussive Source Separation (HPSS)*. By incorporating domain knowledge and promoting parameter sharing in the design of the network architecture, Chapter 3 shows that it is possible to achieve state-of-the art separation performance with a smaller network size (lower number of trainable parameters). Similarly, in Chapter 4 the same methodology is applied to design CNN architectures for performing *frame-level instrument recognition* and *pitch streaming* that also obtain state-of-the-art performance in their respective tasks.

1.1.1 Research on Source Separation

Ensemble music can be seen as a mixture of several audio signals coming from multiple instrumental sources. Even though each of those audio sources overlap in time and in frequency, our auditory system is capable of organising the sounds into perceptually meaningful elements, associating the different notes with their respective instrumental source as soon as the audio signal is heard. For instance, we can automatically recognise multiple vocal lines, string, brass, and wind-based instrumental sounds combined in an audio recording by just listening to it. This process is known as Auditory Scene Analysis [Bregman, 1990] and is the main motivation behind the research topic known as **Music Source Separation (MSS)** [Comon and Jutten, 2010] since the proposal of the 'Cocktail Party' [Cherry, 1953] problem.

In this task, the final goal is to separate a music recording into its constituent source signals. In other words, the primary objective is to group sounds from a single musical instrument into a single source signal while distinguishing them from sounds of other instruments within the same recording. Therefore, processing timbral information — either by mathematically modelling the physics of musical instruments or by automatically learning timbre-related features from data — is of fundamental importance for the MSS task.

MSS is an established research topic within the Music Information Retrieval (MIR) community. Throughout the last two decades, multiple methods for performing this task have been developed and the amount of works submitted to recent public evaluation campaigns and worldwide challenges is enormous

[Stöter et al., 2018; Mitsufuji et al., 2021]. For a brief background on MSS the reader is pointed to Section 2.2.

In this thesis, initial research consists of the proposal of novel data-driven methods for a particular MSS task: the **Harmonic-Percussive Source Separation (HPSS)**, whose objective is to separate a music recording into a non-pitched percussive instrumental source and a pitched harmonic (melodic) instrumental source. It is worth noting that the meaning of "harmonic" in the thesis is not necessarily related to harmony as one might expect as "in close proximity to melody", but referring to the prominence of harmonic series in the spectrum of an individual pitch.

HPSS is a useful preprocessing tool that is beneficial for many downstream tasks. Some examples are the estimation of the beat of a music recording by analysing only the estimated percussive signal [Gkiokas et al., 2012], or the implementation of time-stretching audio effects by manipulating only the harmonic components [Driedger et al., 2014b].

In addition, the thesis also investigates cases to perform HPSS where no labelled data is available for some instrumental sources. The thesis addresses this problem by leveraging unlabelled music data related to those instrument types and a method grounded in unsupervised adversarial domain adaptation is proposed. This contribution is of particular importance given the fact that domain adaptation methods have not been previously explored in MSS-related literature.

1.1.2 Research on Instrument Recognition

The **instrument recognition** task has the objective of automatically classifying the instruments that are active in a music signal. When performed in a frame-level basis, predictions are estimated for each time frame — typically dozens of milliseconds of duration — of a music signal, it is commonly known as **Instrument Activity Detection (IAD)**. However, it can also be performed in a clip-level basis, whose predictions are estimated for longer clips (typically couple of seconds of duration), or in a note-level basis, whose predictions are provided for each note-event, which typically have been previously computed using a Multi-Pitch Estimation (MPE) algorithm. The former is also known as **instrument tagging** while the latter is typically called **pitch streaming** or **instrument assignment** task. For a brief background on instrument recognition the reader is pointed to Section 2.3.

The instrument assignment task is one of the main sub-tasks of the challenging task of **Automatic Music Transcription (AMT)**, which, is also one of the largest topics of research within the MIR field. The AMT task can be

defined as the process of generating a symbolic representation for the contents of a music signal [Benetos et al., 2019]. Sometimes, it is sufficient to fully describe the played notes by generating a representation in the form of a MIDI file or pianoroll of the music signal, where there is an estimation for the localisation of the notes in time, their fundamental frequencies, and their respective durations. This is typically done by using methods known as Multi-Pitch Estimation (MPE) [Christensen et al., 2008] algorithms. However, the primordial AMT goal is to find a more complex and full representation of the music in the form of the actual score of the whole piece. In this case, not only MPE is involved, but many other sub-tasks are included in the process, such as, beat and rhythm tracking, interpretation of expressive timing and dynamics, and score typesetting [Benetos et al., 2013]. When analysing polyphonic audio recordings formed by a combination of sounds of multiple pitched (melodic) instruments, the pitch streaming task is of foremost importance in order to perform multi-instrument AMT. It is the task responsible for associating the transcribed notes to correct instrumental voices in the final staff notation [Benetos and Dixon, 2013; Heittola et al., 2009; Bay and Beauchamp, 2012]. Therefore, proposing novel data-driven methods for pitch streaming is an attractive research topic in order to facilitate multi-instrument AMT.

There are several other applications that can benefit from identifying the sounding sources within a music piece. For example, the obtained frame-level or clip-level instrument information can be used for automatic music tagging, which facilitates indexing and retrieval operations for managing big multimedia archives, for enhancing the quality of music recommendation systems, for performing instrument-specific music equalisation, or even for educational purposes [Fuhrmann, 2012].

Furthermore, even though some works propose systems to transcribe both pitched and non-pitched signals (drum kits) jointly [Benetos et al., 2014], in the vast majority of works in the specialised literature, drum detection and classification is often performed independently because the sound characteristics of drum instruments (unpitched, percussive, transient-like) differs in many aspects from pitched instruments that constitute the melodic and harmonic nature of music. This task is often expressed as the problem of **Automatic Drum Transcription (ADT)** [Wu et al., 2018], which focuses on classifying the input signal into the classes typically found in drum kits of Western music, such as, bass drum, snare drum, hi-hat, cymbals and toms. It is known that it is easier to perform ADT in the presence of isolated percussive notes rather than in a mixture of both percussive and harmonic instruments [Fitzgerald and Paulus, 2006]. Similarly, AMT turns into an even more challenging problem when done in the presence of interference caused by the presence of drum sounds [Herrera-Boyer

et al., 2006]. Under this perspective, it is expected that the methods proposed in this thesis can benefit not only polyphonic multi-instrumental AMT, but also ADT since the proposed HPSS can also be used to isolate drum kits from the mixture, which can then be analysed separately.

In this thesis both the tasks of IAD and pitch streaming are investigated and musically motivated CNNs are proposed to address them. The former is addressed using a Transient-Stationary-Noise Decomposition (TSND) pre-processing step in order to explicitly provide the network with input features related to transient, stationary and residual parts of an audio signal. The latter is addressed using the music spectrogram as input to the framework along with an auxiliary input generated from the onset, offset and pitch information of each note-event.

In the thesis, only real-world music signals are used, meaning no dataset containing artificial music was utilised in any experiment. More specifically, the scope of the analysed music signals is bound to Western-style music and includes music from different genres, instruments and styles. In particular, for the experiments related to source separation, the MUSDB18 dataset [Rafii et al., 2017] is used along with the Tap & Fiddle (T&F) [Lordelo et al., 2020a], which is a dataset that has been curated and published as another contribution of the thesis. It consists of a corpus of Scandinavian fiddle tunes with foot-tapping as accompaniment. It not only contains the final mix with both sources combined, but also includes a track with the fiddle (violin) sounds and another track with foot-tapping sounds in isolation.

While the experiments regarding HPSS include music recordings containing vocals and drums, the experiments related to instrument recognition are performed using the MusicNet dataset [Thickstun et al., 2017], which contains no vocals and no drum. The reason for this is to maintain a controlled environment for instrument recognition only containing mixtures of harmonic-instrument sounds. Moreover, the idea is to propose a method for pitch streaming that is also modular, which means that any MPE algorithm can be used to generate the note-events that will be used to generate inputs for the proposed system. Therefore, the system is limited to not contain singing voice, since vocals have a high variability of timbre and continuous pitch behaviour, which makes much harder to generate note-events with correct onset, constant pitch and offset.

1.2 Context of Project

This PhD project is the result of a partnership between DoReMIR Music Research AB[1] and Queen Mary University of London. This partnership is in the context of the MIP-frontiers project[2], a European training network for MIR researchers that aims to train a new generation of researchers, bringing together 4 universities, 3 industrial beneficiaries and 15 PhD students.

DoReMIR focuses on developing new software and applications to change the way we play and express ourselves in music. During the major part of my programme I have been based in Stockholm, working closely with DoReMIR to identify limitations and conduct research that could lead to a final industry product. One of the company's most revolutionary products is ScoreCloud[3], a software that is able to transcribe music signals directly to a score. However, two current limitations of the system are the fact that it cannot recognise multiple instruments in the signal, which leads to transcription errors, and also the fact that the system does not process drum-based signals, and the presence of this type of sounds in the input signal interfere in the transcription analysis.

In order to overcome those two limitations and improve the polyphonic transcription, the thesis presents novel ideas using deep-learning based source separation and instrument recognition techniques that can also benefit the general Music Information Retrieval (MIR) Community. While most of the contributions have been published in peer-reviewed conferences and journals, some of the proposed algorithms in this thesis have also been adapted to be utilised under industry products related to the aforementioned company.

1.3 Research Objectives

Processing timbral information is of fundamental importance when analysing polyphonic music signals. In cases where sounds of multiple instrumental sources are simultaneously occurring, the task of Music Source Separation (MSS) focuses on separating sounds coming from specific instrumental sources, i.e., sounds with similar timbre, from a music recording. Also, in order to perform multi-instrument Automatic Music Transcription (AMT), not only each note should have its pitch and duration properly estimated, but it is crucial to have a way of recognising the instrument that played each note, so that each sound can be associated to a certain voice in the final staff notation.

While the tasks differ in their primary objectives, both MSS and instrument

[1] `https://doremir.com/`
[2] `https://https://mip-frontiers.eu/`
[3] `https://scorecloud.com/`

recognition tasks are closely related since they share a common foundation in timbral analysis and they can be mutually beneficial in certain applications. For example, an AMT system that utilises an instrument recognition to label notes, could also benefit from MSS by first separating mixed audio into specific sources for more accurate instrument identification.

Therefore, this research endeavour seeks to introduce novel deep learning-based MSS and instrument recognition techniques with focus on not only performance but also efficiency. The methodology and learnings from studying Harmonic-Percussive Source Separation (in Chapter 3) are carried over to instrument recognition proposals (in Chapter 4). The main objective is to show that it is possible to utilise domain knowledge and propose musically-motivated CNNs that use kernels with vertical, square, and horizontal shapes, which enhances the discrimination of more relevant harmonic and percussive features from spectrograms. This design choice allows for building efficient and effective HPSS and instrument recognition models.

The main objectives of the thesis can be summarised below:

- Efficient Musically Motivated Architectures: Develop novel CNN architectures exploiting domain knowledge of music signals to enhance efficiency (smaller network size) while maintaining or even improving performance in source separation and instrument recognition tasks;

- Harmonic-Percussive Source Separation (HPSS): Address the task of HPSS, aiming to separate music recordings into non-pitched percussive and pitched harmonic (melodic) instrumental sources. This research explores techniques to optimize HPSS with a focus on efficiency and effectiveness;

- Domain Adaptation in Music Source Separation: Explore domain adaptation methods to enhance the adaptability of source separation models to diverse musical domains. Investigate the use of unlabelled data and unsupervised adversarial domain adaptation techniques in the context of HPSS;

- Instrument Recognition and Pitch Streaming: Investigate frame-level instrument recognition and pitch streaming tasks. Design musically motivated CNN architectures to achieve state-of-the-art performance in these tasks;

- Interlinking Source Separation and Instrument Recognition: Maintain methodology connections between proposed source separation and instrument recognition methods by leveraging the insights gained from HPSS research;

- Industrial applications: The proposed methods are designed to not only contribute to the Music Information Retrieval (MIR) community, but to also bring innovations and solutions to improve Doremir's music transcription products.

1.4 Thesis structure

In this section, the structure of the thesis along with a brief summary of the contents of each chapter is described.

Chapter 2 starts by defining terms and ideas that are important for the full understanding of the whole thesis. Afterwards, it describes the required background knowledge and previous work in the areas related to the thesis. Specifically, it describes methods related to source separation and instrument recognition, as well as, generative adversarial networks and domain adaptation methods.

Chapter 3 contains contributions of the thesis related to MSS. It starts by describing the proposed method based on musically motivated CNN architectures for performing HPSS. The second part of the chapter explains the proposed unsupervised domain adaptation method for HPSS in order to perform separation in missing data scenarios, where no labels for some instrument sources are explicitly available. In addition, this chapter also provides detailed description of the Tap & Fiddle Dataset, which is also one of the contributions of the PhD research.

Chapter 4 describes contributions of the thesis related to the instrument recognition task. The first part investigates the effects of performing TSND as a pre-processing technique for instrument activity detection. Afterwards, the chapter explains another proposed method that performs pitch streaming using a musically motivated CNN classifier that uses an auxiliary input generated from a note-event information.

Chapter 5 summarises the project and concludes the thesis. The chapter also discusses limitations and talks about potential future directions.

1.5 Contributions

Contributions of this thesis are:

- Chapter 3: Musically motivated CNN architecture for performing HPSS using vertical, horizontal and square filters namely 3W-MDenseNet;

- Chapter 3: Multiple kernel shapes and dense arrangement of skip connections shows 3W-MDenseNet efficiency on performing HPSS. It achieves state of the art performance while keeping the number of parameters low;
- Chapter 3: HPSS is done using a CNN that estimates both sources simultaneously. While more recent deep learning methods for source separation also propose this joint estimation of sources, traditional methodology of spectrogram-based CNN models in 2018-2019, when this research was developed and proposed, was to train DNNs to estimate just a single source;
- Chapter 3: Proposal of an adversarial unsupervised domain adaptation for HPSS that is beneficial for generalising the separation to sounds with different timbre and cases where no labelled data is available for some of the source types;
- Chapter 3: Curation and release of Tap & Fiddle, a dataset with Scandinavian fiddle tunes along with stems for the main melodic instrument (violin) and the accompaniment foot-tapping sounds;
- Chapter 4: Investigation of whether or not explicitly providing transient-like along with stationary-like sounds can help the performance of IAD;
- Chapter 4: Proposal of an adaptation of the 3W-MDenseNet for classification scenarios and verification that the addition of vertical, horizontal and square filters along with the dense arrangement of skip connections is also beneficial for instrument recognition;
- Chapter 4: Proposal of a deep-learning-based pitch streaming method that uses a classifier with a dual channel input: one with the spectrogram around the onset and offset of a target note, and another carrying information regarding the pitch of the target note-event;
- Chapter 4: Modular pitch streaming method. The proposed pitch stream method enables multi-instrument AMT when streaming pitches that have been estimated using third party MPE algorithms;

1.6 Associated publications

Portions of the work detailed in this thesis have been presented in international scholarly publications, as follows:

1.6.1 Journal Publications

– Lordelo, C., Benetos, E., Dixon, S., Ahlbäck, S., and Ohlsson, P. (2021b). Adversarial unsupervised domain adaptation for harmonic-percussive source separation. *IEEE Signal Processing Letters*, 28:81–85

1.6.2 Peer-Reviewed Conference Publications

– Lordelo, C., Benetos, E., Dixon, S., and Ahlbäck, S. (2021a). Pitch-informed instrument assignment using a deep convolutional network with multiple kernel shapes. In *International Society for Music Information Retrieval Conference (ISMIR)*

– Lordelo, C., Benetos, E., Dixon, S., and Ahlbäck, S. (2019). Investigating kernel shapes and skip connections for deep learning-based harmonic-percussive separation. In *Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, USA

1.6.3 Dataset Publications

– Lordelo, C., Ahlbäck, S., Benetos, E., Dixon, S., and Ohlsson, P. (2020a). Tap & Fiddle: A dataset with scandinavian fiddle tunes with accompanying foot-tapping. *Zenodo*. `https://doi.org/10.5281/zenodo.4308731`

1.6.4 Invited Talks

– Lordelo, C., Delgado, A., Ramires, A., "Can transient/non-transient separation improve instrument recognition?". In *MIP Frontiers Workshop vol. 1*, Barcelona, Spain, April 2020

– Lordelo, C., "Deep Learning Methods for Musical Instrument Separation and Recognition". In *Mip-Frontiers Workshop vol. 2*, Online, October 2021

– Lordelo, C., "Pitch-informed Instrument Classification and Unsupervised Domain Adaptation for Source Separation". In *Audio Data Analysis and Signal Processing (ADASP) Group Meeting*, Telecom Paris, October 2021

All work described in this thesis was conducted by the author, including the implementation of experiments, development of code, and writing, with general guidance and feedback given by his supervisors.

Chapter 2

Background

In this chapter, basic terminology and background on state-of-the-art methods for topics of interest for the developed research are provided. Particular importance is given to methods related to source separation and instrument recognition, which are the two main tasks researched in the thesis.

Section 2.1 starts with basic definitions to give and terminology that will be used along the thesis. Basic terms, such as acoustic waves, sound, music, as well as perceptual music features, such as pitch, loudness and timbre will be properly defined. Afterwards, Section 2.2 and Section 2.3 define the tasks of source separation and instrument recognition, respectively, and provide a literature review including state-of-the-art methods for each topic. In Section 2.4 multi-task approaches performing both tasks simultaneously are briefly listed, while in Section 2.5 a discussion related to generative adversarial networks is provided. Lastly, in Section 2.6 the definition of and a literature review on domain adaptation approaches are provided.

2.1 Basic Definitions and Terminology

This section provides basic definitions of important terms and expressions that are important for fully understanding the developed work. Some of them might have multiple meanings in the Music Information Retrieval (MIR) literature, but every time those terms appear in the thesis, they should have the meaning explained in this section.

2.1.1 Acoustic Waves and Sound

An acoustic wave can be defined as a way of transmitting information by the means of compression and decompression of a medium [Rienstra and Hirschberg,

2021]. Even though it is possible to propagate acoustic waves in liquids, such as water, or solids, such as metal, the most familiar transmission medium is the air that exists around us. Every time an object vibrates, an acoustic wave is formed in its vicinity and then propagate through the air. When the small oscillations in the air pressure level reach our eardrums, our auditory system perceives it as a phenomenon we call *sound* [Lapp, 2003].

A good example that is easy to understand this process is the tuning fork. Striking one of the forks causes us to immediately hear a tone. This happens because the strike makes the tuning fork vibrate at its frequency of resonance, which depends of the fork shape, size and material. The fork's vibration creates a small compression of air molecules that travels as an acoustic wave that eventually reaches our ears and makes our eardrums vibrate with the same frequency of the motion of the tuning fork. Our auditory system processes this vibration as a specific tone.

Throughout our lives we learn to associate particular sounds with specific sources. Our auditory system is able to immediately perceive the type of "message" that is being broadcasted because different acoustic waves produce distinct sounds. Based on the sound that we hear we can also identify the source relative location and important characteristics about the environment. The most prominent physical features of acoustic waves that allow us to categorise sounds and uniquely perceive them are their **frequency, amplitude** and **shape**.

- Supposing the acoustic wave is periodic, its **frequency** is the number of oscillations, or cycles, per unit of time. It depends only on the frequency of vibration of the emitting source.

- The **amplitude** of the wave is the maximum deviation in the oscillation relatively to the equilibrium position. The greater the amplitude of an acoustic wave, the louder the sound is perceived if the wave frequency and shape are kept fixed.

- The **shape** of an acoustic wave is directly related to its spectral content, or the frequencies, amplitudes and phases of its components. The spectral content is the primary factor in our perception of timbre [McAdams, 2013]. In Subsection 2.1.6 a detailed discussion about timbre can be found.

2.1.2 Audio and Music Signals

When representing sounds in a machine, it is usual to refer to them as **audio signals**. Since there is a wide variety of sound-related applications, audio signal is a generic term that can be used when processing any type of sound, whether it is speech, music or other.

Music can be broadly understood as an ordered sequence of notes with a rhythmic structure [Lapp, 2003]. For the purpose of the research developed in this thesis, music signals are hereby defined taking into account how musical sound is produced. **Music signals** are treated as particular types of audio signals that are produced by a combination of several concurrent sounds generated by a single or multiple sources, which are typically musical instruments and/or singing voice.

In the thesis the terms music signal and audio signal are used interchangeably.

2.1.3 Fundamental Frequency, Modes and Overtones

A simple sinusoidal audio signal of frequency f_0 and amplitude A can be mathematically modelled as

$$A\sin(2\pi f_0 t + \phi), \tag{2.1}$$

where t represents time and ϕ is the phase of the sinusoidal signal when $t = 0$. In the literature, the term *pure tone* is used to refer to this type of signal [Müller, 2015]. The pure tone has the simplest spectral content and tends to be referred as a sound with no "colour". Even though a pure tone can be digitally synthesised, it is an abstraction that cannot be naturally generated using musical instruments.

In every way we can generate sound, whether it is by striking pieces of wood or metal, by plucking or bowing strings, or by blowing air into tubes, the sound produced by any musical instrument is far more complex than just a single acoustic wave of a desired frequency. In fact, multiple acoustic waves of different frequencies are generated simultaneously.

When musicians play a note using a musical instrument, they do not set a constant frequency of vibration for the medium — a string if it is a stringed instrument, the column of air in the case of woodwind and brass instruments, etc —, in fact, they just provide energy for it to vibrate while fixing the position for nodes of the produced acoustic wave [Fletcher and Rossing, 1998]. The medium always vibrates in multiple **modes** of vibration.

The lowest frequency of vibration attributed to a physical system is called **fundamental frequency** [Lapp, 2003]. It is usual to represent it as f_0 in the literature and this thesis will follow such notation.

The fundamental frequency of a sound is caused by the *first mode* of vibration, which depends on the length of the vibrating medium. Higher modes of vibration are also stimulated simultaneously, generating higher frequency vibrations of the musical instrument medium. The frequencies of these higher modes are known as **overtones** [Loy, 2006]. The *second mode* of vibration produces

the *first overtone*, the *third mode* produces the *second overtone*, and so on. Figure 2.1 shows a simplification of how chordophones (string instruments) and aerophones (woodwind and brass instruments) produce their sound. Observe that not only the fundamental frequency is stimulated, but also integer multiples of this frequency also occur due to higher modes of vibration. In the case of a closed aerophone, only odd-integer multiples of the fundamental frequency are present.

When the relationship between an overtone and the fundamental frequency is an integer, the modes of resonance are named **harmonics** [Müller, 2015], i.e., if we consider f_0 as the fundamental frequency produced by the first mode of vibration, the i-th harmonic f_i can be defined as

$$f_i = i \cdot f_0, \quad \forall i \in \mathbb{N}^+ \tag{2.2}$$

where $\mathbb{N}^+$ is the set of all natural numbers greater than zero[1].

Giving a detailed explanation of how acoustic waves are produced by each type of instrument and how their size, material and shape affect the spectral content of their sound is out of the scope of this thesis. The interested reader is referred to [Rienstra and Hirschberg, 2021; Lapp, 2003; Fletcher and Rossing, 1998; Loy, 2006] to find more information about this topic.

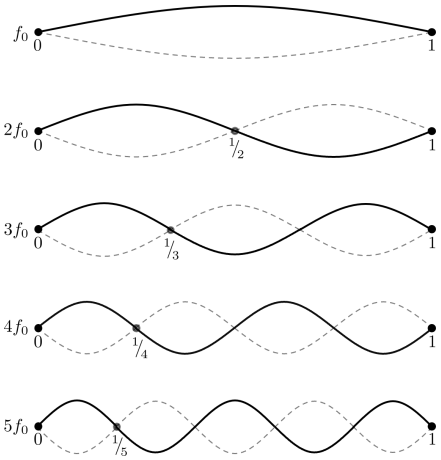
2.1.4 Pitch

According to the American National Standards Institute (ANSI) [ANSI, 1978], **pitch** can be defined as "*that auditory attribute of sound that allows us to order them in a scale from low to high*". In other words, pitch is a subjective attribute reflecting the human auditory perception of the frequencies of sounds [Loy, 2006]. However, while frequency of acoustic waves does not have any constraint, pitch is limited to sounds within the range of human hearing, which is approximately from 20 Hz to 20000 Hz.

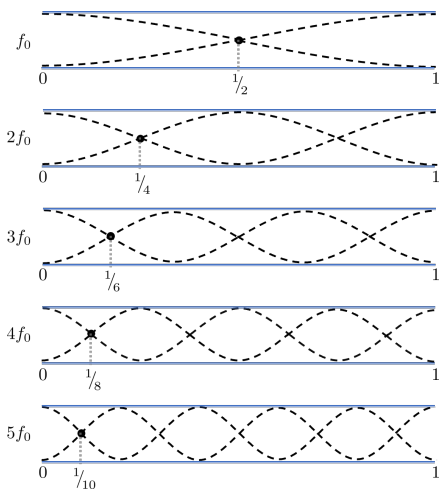
Strictly speaking, it is important to note that our perception of pitch may also be influenced by loudness and the presence of higher or lower non-harmonic frequencies [Yost, 2009; Langner et al., 1998]. However, when analysing harmonic sounds, our sense of pitch is proportional to the fundamental frequency of the produced sound. While sounds produced by two different harmonic instruments may generate a different number of harmonics, if the fundamental frequency of the sounds are the same, they will be perceived by our auditory system as having the same pitch.

As discussed in Subsection 2.1.1, the frequency of a sound wave is a *physical*

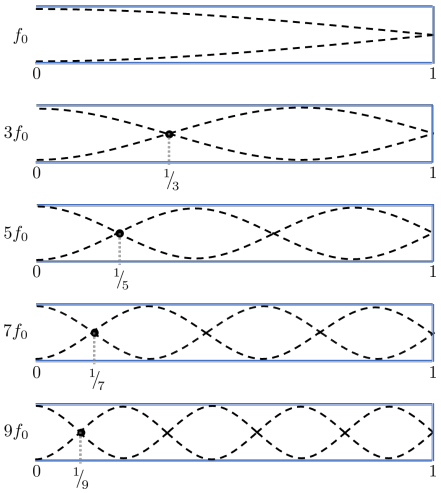
[1] In this thesis it is used the convention that the fundamental frequency is the first harmonic.



(a) First 5 modes of vibration of a fixed string. Since the string is fixed at both ends those positions are nodes for every mode of vibration. Note that the frequencies of higher modes are always a multiple of the fundamental frequency f_0 caused by the first mode of vibration.



(b) First 5 modes of vibration of air inside an open pipe. Since the tube is open at both ends, those positions are antinodes for every mode of vibration. Note that the frequencies of higher modes are always a multiple of the fundamental frequency f_0 caused by the first mode of vibration.



(c) First 5 modes of vibration of air inside an semi-closed pipe. Since one end of the pipe is open while the other is closed, the former position are antinodes while the latter are nodes for every mode of vibration. The higher modes will always be an odd multiple of the fundamental frequency f_0 caused by the first mode of vibration.

Figure 2.1: Modes of vibration in different types of harmonic instruments. The first mode is always shown in the top-most part of each subfigures. The length of the string and tubes are normalised to 1.

measure of vibrations per unit of time. The International System of Units (SI) unit for frequency is Hertz (Hz), which represents cycles per second. On the other hand, in the Western Common Music Notation (CMN) system, an equal tempered scale of 12 notes is utilised to represent the perceptual pitch attribute of sounds. Pitch is identified using a letter (A,B,C,D,E,F,G) and an octave number. As an example, notes with the fundamental frequency of $f_0 = 440$ Hz have their pitch conventionally called as A4, which refers to note A in the 4th octave. The pitch A4 is commonly used by modern Western orchestras as the *reference pitch* to tune all instruments together. Moreover, symbols like ♭ and ♯ can also be used to indicate that a pitch should be "flat" or "sharp", shifting the pitch one semitone lower or higher, respectively.

Due to their direct relationship, in most of the research topics in the area of MIR, both terms are often used as synonyms. For instance, the task of estimating the fundamental frequency of sounds under polyphony is widely known as Multi-Pitch Estimation in the literature [Christensen et al., 2008; Drugman and Alwan, 2011; Kim et al., 2018]. This will follow this practice and the terms *pitch* and *fundamental frequency* can be interchangeably used in the text. Sometimes, the text may also refer to pitch using letters, such as f_0 and units like Hz instead of the regular letters used in the CMN.

2.1.5 Loudness

Loudness is a subjective measure of how humans perceive sound intensity. It is directly related to the amplitude of the acoustic wave but our sense of loudness is also affected by the frequency range and presence or absence of other frequencies Bauer and Torick [1966].

Two acoustic waves of different fundamental frequencies, but with same amplitude are usually perceived as having different loudness levels. For instance, the human auditory system is more sensitive to sounds from a particular pitch range from 1 kHz to 3 kHz. Sounds in this particular range are perceived louder than sounds outside this frequency region (supposing a fixed intensity level) [Fletcher and Munson, 1933; Robinson and Dadson, 1956].

Based upon their construction, the loudness of sounds produced by many musical instruments can be adjusted over what is known as **dynamic range**. In CMN, symbols like *ppp*, *pp*, *p*, *mp*, *mf*, *f*, *ff* and *fff* can be used to indicate the level of loudness of notes, which translates into playing notes as softly as possible to as loudly as possible.

2.1.6 Timbre

Like pitch and loudness, **timbre** is also a perceptual property of sound. However, unlike the other two, timbre is not a well-defined term. Timbre is very hard to completely understand and explain, and, due to its vagueness, definitions of timbre tend to describe the term in an indirect way, indicating what timbre is not rather than defining what it actually is. For example, the definition provided by ANSI [1973] says: "*Timbre is that attribute of auditory sensation in terms of which a listener can judge that two sounds similarly presented and having the same loudness, pitch and duration are dissimilar.*"

In a few words, it is possible to say that timbre is the fundamental aspect of sound that enables us to distinguish among instruments that are playing the same pitch and are equally loud. Sometimes, timbre can be also referred to as 'tone colour' or 'sound quality' and it is common to use terms such as bright, dark and harsh when describing the timbre of musical instruments.

For a long time, researchers have been trying to find a mathematical explanation of what timbre actually is and trying to understand how other more objective and physical properties of sound affect our perception of timbre [Balzano, 1986; Sethares, 2005; McAdams, 2013]. Nevertheless, timbre has proven to be highly complex and abstract, and only some indications of sound characteristics that affect its perception are known; we still do not fully understand how much each characteristic correlates with each other and how much do they actually contribute to timbre perception. For example, when a piano, a violin or a flute plays the same note with the same intensity, the relation between the energy of the harmonics in each sound is different, and this is one of the features that helps us distinguish the instruments. Another important feature is how each part of the Attack, Decay, Sustain and Release (ADSR) modelling of sounds [Müller, 2015] contributes to the envelope of each note. Such modelling represents the 4 stages of envelope modulation and is greatly used in instrument synthesizers to simulate different instrument sounds. An illustration of this time-domain division of a sound generated by a key-press of a keyboard can be seen in Figure 2.2 and is briefly summarised below:

1. **Attack**: time the sound takes to reach its amplitude peak after the key is pressed;

2. **Decay**: how fast the amplitude falls from the sound's attack-peak to the amplitude set by the sustain part;

3. **Sustain**: base amplitude level of the sound during the time the key is pressed and the tone sounds;

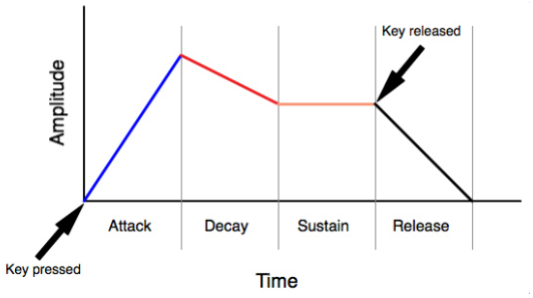


Figure 2.2: ADSR envelope model of a generic piano sound.

4. **Release**: time the amplitude takes to reach zero after the key has been released.

2.1.7 Tone and Note

A **tone** can be defined as a sound that humans perceive with a clearly defined pitch. To fully represent a tone in the CMN, three sonic qualities are needed: *pitch*, *loudness*, and *timbre*. It is important to note that tones have no information regarding time.

When a tone is placed in a temporal context, we say that a **note** is formed. At least two extra pieces of information are necessary to characterise a note: **onset** and **duration**. The onset of a note is the time assigned for the note to start (attack) while its duration is the amount of time the note lasts, i.e., the time until the release stage of the note brings its envelope and energy back to zero.

In reality, a note can encompass many more features, such as, ornamentation — *glissando, trills, appoggiatura* and others — and articulation — *slurs, staccato, legato* and more —. Those features may affect not only a note-event's onset time and duration, but also indicate a variable pitch, or even modify the note's timbre by changing the shape of its attack or decay. The CMN supports those as well as many other features by assigning appropriate symbols in the staff notation in order to represent note events in more detail.

In this thesis I am going to use the definition that a note event has always a constant pitch, a fixed onset, fixed duration and its timbre represents the source (instrument) that produced the note. This follows the definition encountered in the Music Information Retrieval Evaluation eXchange (MIREX)[2] for the tasks of music transcription and note tracking.

[2] `https://www.music-ir.org/mirex/`

2.1.8 Percussive and Harmonic Instruments

Percussive instruments can generally be defined as instruments that produce sound after being struck or scraped by a beater or struck against another similar instrument body. Their typical role in a musical ensemble is to emphasise the rhythmic structure, while harmonic instruments are usually responsible for the harmony and melody. The majority of percussive instruments produce unpitched sounds, i.e. sounds with indefinite pitch. This type of sound is usually the result of a highly *transient* and *impulsive* event, generating a short-time burst of sound and simultaneously stimulating a continuum of frequencies. It is also possible to find percussive instruments, such as cymbals, where the sound emission lasts for a longer time, which makes their sounds non-transient, but with a non-determined pitch and broadband noisy-like spectrum.

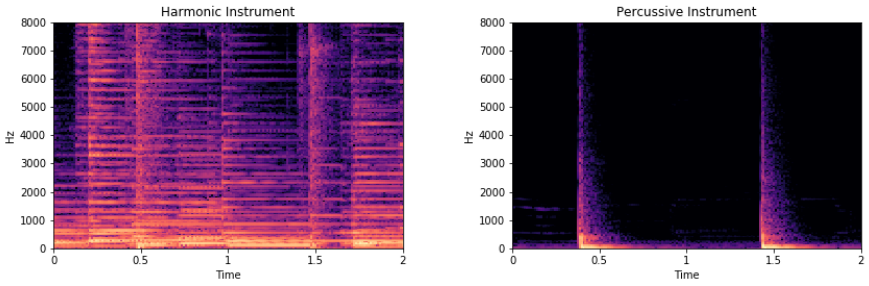
Notwithstanding, some harmonic instruments, such as, piano and guitar, may also be considered pitched percussive instruments in some cases, given the fact that they produce pitched notes with prominent attacks (onsets), which are often perceived with a percussive characteristic. Furthermore, other percussive instruments such as xylophones and marimbas, may also produce sounds with clearly defined pitches, even though their overtones are related to the fundamental frequency in an inharmonic way.

It is important to note that even though unpitched percussive instruments provoke no perception of a clearly defined pitch, they may still produce sensations of lower or higher "global" pitches, a feature that can help in performing Automatic Drum Transcription (ADT) [Wu et al., 2018; Yoshii et al., 2007]. For instance, sounds emitted by a bass drum or a large tom are normally perceived as lower than sounds emitted by a snare drum or a small tom, respectively. Those sensations of pitches, however, relate only to other members of the set of unpitched instruments and not with other pitched percussive or harmonic instruments.

In contrast, harmonic sounds are sounds that are perceived as pitched sounds (or musical notes). This type of sound contains a discrete and finite set of harmonic frequencies that are stimulated simultaneously and usually last for a much longer time if compared to the usual shorter unpitched percussive sounds. In other words, it is possible to say that (unpitched) percussive sounds are well time-localised sounds with a spread out frequency behaviour, while harmonic sounds are harmonically frequency-localised sounds.

Therefore, when analysing a time-frequency representation such as the magnitude spectrogram of an audio signal, where the vertical axis is associated to frequency and horizontal axis to time, unpitched percussive sounds appear as vertical lines and ideal harmonic sounds form horizontal structures. This effect

is known as *anisotropic smoothness* [Driedger et al., 2014a] and is illustrated in Figure 2.3, in which real-world percussive and harmonic sounds are depicted.



(a) Spectrogram of a solo violin clip. Harmonic (pitched) sounds tend to form horizontal lines in the spectrogram, but note that there are also percussive characteristics such as presence of vertical lines, specially around the onsets of notes.

(b) Spectrogram of a bass drum clip. Unpitched percussive sounds form vertical lines in the spectrogram, but note that there are also a few horizontal lines due to their resonance body, but a much more noisy-like characteristic.

Figure 2.3: Examples of real world harmonic and percussive sounds

In an actual music recording, any real-world musical instrument will generate sounds having characteristics of both types of sounds. For instance, a harmonic instrument, such as a guitar or a piano, also have a percussive behaviour during the attack part of their notes due to the plucks of the guitar strings by our fingernails or due to the impact of the piano strings with its internal hammers. Similarly, percussive instruments also have a few horizontal lines in their magnitude spectrogram representation due to their low resonant bodies (this fact can also be seen in Figure 2.3). However, it is still possible to categorise them in one of those two types depending on the predominant patterns they create in their magnitude spectrogram.

In this thesis, when percussive instruments are referred to in the text, they will be representing only the set of *unpitched percussive instruments*. For instance, Chapter 3 addresses the task of Harmonic-Percussive Source Separation (HPSS), which essentially is a synonym for unpitched-pitched source separation, and Section 3.2 describes experiments using harmonic sounds and percussive drum-based sounds, which consists of unpitched percussive sounds from drum kits — bass drum, snare drum, hi-hat, cymbals and toms —. In Section 3.3 foot-tapping sounds are included in the experiments and treated as an unpitched percussive instrument. More details regarding the HPSS task are explained in Subsection 2.2.2 along with descriptions of previous work.

2.2 Source Separation

As mentioned in Subsection 1.1.1, the source separation task is the task of separating signals into its constituent sources. The general problem can be presented considering that R sources emit signals $(s_1, s_2, s_3, \cdots, s_R)$ that are later captured by M sensors (or microphones, when the signals involved are audio or music), which results in M mixtures $(m_1, m_2, m_3, \cdots, m_M)$ being created. Source separation is the task to recover the original values of s_j, for $j \in \{1, 2, 3, \cdots, R\}$ analysing only the mixed signals m_i for $i \in \{1, 2, 3, \cdots, M\}$. It is also possible to estimate, as a by-product of the analysis, how the sources have been mixed. This problem, in its instantaneous mixture-generation formulation, can be mathematically defined as following a linear system of equations:

$$\begin{cases} m_1 = a_{11}s_1 + a_{12}s_2 + a_{13}s_3 + \cdots + a_{1R}s_R \\ m_2 = a_{21}s_1 + a_{22}s_2 + a_{23}s_3 + \cdots + a_{2R}s_R \\ m_3 = a_{31}s_1 + a_{32}s_2 + a_{33}s_3 + \cdots + a_{3R}s_R \\ \vdots \quad\quad \vdots \quad\quad \vdots \quad\quad \vdots \quad\quad \vdots \\ m_M = a_{M1}s_1 + a_{M2}s_2 + a_{M3}s_3 + \cdots + a_{MR}s_R \end{cases}, \tag{2.3}$$

where each term a_{ij} is the weight given to source s_j in the mixture m_i. We can rewrite Equation (2.3) in matrix form as

$$\mathbf{m} = \mathbf{A}\mathbf{s}, \tag{2.4}$$

where $\mathbf{A} \in \mathbb{R}^{M \times R}$ contains the weights a_{ij} of each source and mixture, $\mathbf{s} \in \mathbb{R}^{R \times 1}$ contains the original signals emitted by the sources, both unknowns, and $\mathbf{m} \in \mathbb{R}^{M \times 1}$ is the vector with the different mixtures captured by the sensors, i.e.,

$$\mathbf{m} = \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ \vdots \\ m_M \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1R} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2R} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3R} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{M1} & a_{M2} & a_{M3} & \cdots & a_{MR} \end{bmatrix}, \quad \mathbf{s} = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_R \end{bmatrix} \tag{2.5}$$

Hence, the problem of source separation reduces to the problem of solving Equation (2.4) and inferring the values of vector $\mathbf{s}$. In an initial analysis, one may argue that a straightforward way to solve this linear system would be to use an ordinary technique, such as least squares, and obtain a solution with minimal squared error rate. However, we do not know the coefficients of matrix $\mathbf{A}$, i.e., we do not have information related to the physical system that each

mixture originated from, which makes the problem much more difficult. Due to this fact, some authors even refer to this problem as *blind source separation* [Comon and Jutten, 2010], indicating that the separation algorithm does not utilise any other information about the signals or the mixing process.

In summary, to fully solve the source separation task, it is necessary to estimate the matrix $\mathbf{A}^{-1} \approx \mathbf{W}$ somehow such that the vector of source estimations $\hat{\mathbf{s}} = \mathbf{Wm}$ becomes as close as possible to $\mathbf{s}$. It is trivial to realise that there is no exact solution if matrix $\mathbf{A}$ does not have a full column rank. Notwithstanding, in recent years, a lot of methods have been developed to be used in the most diverse applications of source separation with the objective of obtaining reasonable estimates for the separated signals $\hat{\mathbf{s}}$.

In the case where the number of observed mixtures (or sensors) M is equal to or greater than the number of sources R, a good estimation of the sources can be easily computed by applying Independent Component Analysis (ICA) [Hyvarinen et al., 2001]. This technique makes use of the Central Limit Theorem (CLT) [Peebles, 2000], which essentially establishes that the probability distribution of the sum of stochastically independent variables tends toward a normal distribution, in order to solve the source separation task.

The general idea behind ICA is to assume that the sources originally have a non-Gaussian distribution and are mutually independent, so $\mathbf{W}$ is estimated such that the non-Gaussianity of $\hat{\mathbf{s}} = \mathbf{Wm}$ is maximised. Generally, in order to measure the non-Gaussianity of a signal, the negentropy [Lee et al., 2000] or the absolute value of the kurtosis [Hyvarinen, 1999] is used.

Approaches based on ICA are widely used in many applications, such as text mining, finance analysis, Code-Division Multiple Access (CDMA) communications and others [Girolami, 2000; Oja, 2004; Shiomi, 2021]. In the case of music source separation, ICA cannot be directly applied because either $M = 1$ — in the case of mono music signals — or $M = 2$ — in the case of stereo music signals — and the task becomes a highly underdetermined system of equations with $R >> M$. Moreover, the assumption that the source signals are independent is not reasonable in the music source separation scenario given the fact that, in a music recording, the different instrumental sources are often synchronised and highly correlated. Therefore, approaches based on ICA are not common in this area. More information regarding ICA can be found in [Hyvarinen et al., 2001].

2.2.1 Methods for Music Source Separation

In the MSS task, the final goal is to separate a music recording into its constituting sources. The music signal is, therefore, modelled as being either a single mixture — in the case of mono recordings — or two mixtures — in the

case of stereo recordings — containing multiple sources. The MSS is, therefore, a special type of blind source separation, as explained in Equation (2.4), but with a peculiarity of being a highly underdetermined system of equations, where $M << R$, and having highly correlated concurrent sources.

It is possible to categorise methods for MSS based on the type of sources they estimate. There are MSS algorithms interested in separating the music signal into each constituting single-instrument recording, such as drums, bass, vocals and "rest" [Stöter et al., 2018; Mitsufuji et al., 2021], methods interested in extracting only the singing voice from the recording [Sofianos et al., 2012; Jansson et al., 2017; Luo et al., 2017; Grais et al., 2018; Stoller et al., 2018c,b; Cohen-Hadria et al., 2019], methods for lead-accompaniment separation [Rafii et al., 2018] or even methods to separate harmonic instruments from percussive instruments [Fitzgerald, 2010; Fitzgerald et al., 2014; Lim and Lee, 2017; Drossos et al., 2018].

Depending on the methodology used in the method it is also possible to divide them in knowledge-driven methods and data-driven methods.

2.2.1.1 Knowledge-Driven Methods

Until recently, the majority of methods related to the general field of audio source separation [Makino, 2018] were signal processing algorithms built to explicitly model characteristics of the structure of a music signal or to exploit particular spectro-temporal features of musical instruments. In order to separate the melodic voice from the instrumental accompaniment of a musical piece, for example, a useful assumption is to consider the sounds of accompaniment instrumental sources as repetitive patterns in the mixture's spectrogram while the melody line is the non-repetitive part [Rafii and Pardo, 2013].

However, the problem of separating an audio signal with multiple harmonic and percussive instruments into multiple sources is much more challenging, as the patterns in their spectral and temporal characteristics are difficult to explicitly define. In the literature, Non-negative Matrix Factorization (NMF) [Lee and Seung, 1999] is one of the most traditionally used techniques for this type of separation. NMF can be understood as an iterative algorithm to decompose the magnitude spectrogram of the mixture signal into multiple spectral bases that can be combined to create the sounds of the reference sources. Over the years, various NMF methods have been proposed for source separation [Eggert and Korner, 2004; Virtanen, 2007; Févotte et al., 2009; Weninger et al., 2014], usually including sparsity constraints and other regularisation factors into the cost function. An extension of the classical NMF approach is non-negative tensor factorisation [FitzGerald et al., 2008]. Furthermore, there are more complex

and hybrid methods, where pitch estimation techniques are also included in the separation algorithm. For instance, Rafii et al. [2014] uses a multi-pitch estimation algorithm to identify the pitch contour of the singing voice in order to extract it from the music, while Sofianos et al. [2012] mixes analysis in the frequency domain using ICA and a method denoted by the authors as amplitude discrimination with the posterior time domain analysis and pitch estimation to separate singing voice.

2.2.1.2 Data-Driven Methods

With the increasing utilisation of data-driven approaches and the follow-up improvements of the state-of-the-art performance in closely related music information retrieval tasks such as multi-pitch estimation and instrument tracking, deep learning methods also started to be proposed for MSS [Nugraha et al., 2016b; Chandna et al., 2017; Roma et al., 2018a; Osako et al., 2017]. Moreover, in the Signal Separation Evaluation Campaign (SiSEC) 2018 [Stöter et al., 2018] and the Music Demixing Challenge (MDX) 2021 [Mitsufuji et al., 2021], the vast majority of published methods that obtained higher performance in the task of separating music recordings into their instrumental sources (drums, vocals, bass and other) are based on Deep Neural Networks (DNNs).

The basic idea of the current state-of-the-art methods of source separation is to use a deep neural network capable of learning long-term time-relationships of the music signal. Some examples are the utilisation of Recurrent Neural Networks (RNN) and Long-Short Term Memory (LSTM) networks that have been proven to improve the separation [Huang et al., 2014; Takahashi et al., 2018b]. Another important aspect is that the majority part relies on estimating a time-frequency mask for a pre-defined source, which is then multiplied by the time-frequency representation of the mixture to get the associated sound source signal. Those methods usually ignore phase information and just use the original phase of the mixture in the estimated source, but some works also try to include phase information in the system [Muth et al., 2018; Takahashi et al., 2018a; Drossos et al., 2018]. Other deep-learning methods perform the separation directly in the time-domain and have been proven to also achieve state-of-the-art performance [Stoller et al., 2018c; Pandey and Wang, 2019; Cohen-Hadria et al., 2019].

An architecture of particular interest is the U-Net [Ronneberger et al., 2015], where the authors proposed a deep encoder-decoder network with forward skip-connections between layers of the same resolution with the objective of avoiding vanishing gradients during training and of passing directly to the decoder the high level of details of the previous feature-maps. Despite being originally used

for medical image processing, the U-Net has already been used successfully for singing voice separation when applied to the mixture magnitude spectrogram [Jansson et al., 2017]. Later, it has been adapted to perform end-to-end audio separation directly in the time domain [Stoller et al., 2018c].

An important contribution to the source separation community was the release of Open-Unmix [Stöter et al., 2019], which is a reference state-of-the-art implementation for ASS based on the architecture proposed by Uhlich et al. [2017]. It provides ready-to-use models that allow users to separate music signals into four stems: vocals, drums, bass and the remaining other instruments. Open-Unmix is used as a baseline for the source separation experiments explained in Chapter 3.

However, a limitation of the majority of methods is the fact that they are designed to tackle predefined separation sub-tasks, i.e., they rely on extracting only a specific instrument from the mixture or are focused on performing the separation only in particular music genres. Some examples are methods for melody and vocals extraction [Salamon et al., 2014] and for lead and accompaniment separation [Rafii et al., 2018]. Moreover, data-driven methods do not usually generalise well to different types of music or to other instrumental sources unseen during training [Manilow et al., 2019; Stoller et al., 2018a].

Recently, a different perspective for the source separation problem has been rising in the literature. Instead of creating multiple large models and training each of them to be specialised to extract a single source from the mixture, the idea of those works is to create a general model that can estimate different sources based on an auxiliary input that would condition the network. This approach has been proposed in Meseguer-Brocal and Peeters [2019] and Lee et al. [2019]. The former utilises an auxiliary vector representing the source of interest to guide the separation process of a U-Net-like architecture [Ronneberger et al., 2015], while the latter uses a clip with sounds of the source of interest to guide the separation using a variational auto-encoder [Kingma and Welling, 2014].

After decades of study, while being tackled from various perspectives, the MSS problem has proven to be highly complex. It is still really challenging to create a separation system capable of obtaining high-level results when applied to a mixture of multiple types of musical instruments and to heterogeneous music styles.

2.2.2 Harmonic-Percussive Source Separation

The task of Harmonic-Percussive Source Separation (HPSS) of music signals is already a well-established problem in the Music Information Retrieval (MIR) community. It can be seen as a particular case of MSS, where the objective

is to decompose a given music signal into a sum of two component signals: one containing only the sounds produced by (pitched and melodic) harmonic instruments and the other consisting of all the sounds emitted by (unpitched and non-melodic) percussive instruments. It is known that HPSS is a useful preprocessing tool for many applications, such as the estimation of the beat of a music recording by analysing only the estimated percussive signal [Gkiokas et al., 2012], or the implementation of time-stretching audio effects by manipulating only the harmonic components [Driedger et al., 2014b].

The HPSS task should not be confused with the Transient-Stationary Separation (TSS) task. In TSS, the goal is to decompose the original audio signal into strictly percussive (transient) sounds and strictly harmonic (stationary) sounds. In this case, all the attacks of notes should be considered part of the transient source, while the sustain, decay and release parts generally remain in the stationary source. Conversely, HPSS is, effectively, a pitched-unpitched sound source separation. In this task, the harmonic source should contain the whole set of sounds produced by the harmonic instruments, including the percussive transient part of the sound emission. Moreover, since the set of (unpitched) percussive sources of a music recording usually includes only instruments of a drum kit, the HPSS task is often a synonym of a drum extraction task.

Another important difference between HPSS and TSS is that while it is possible to generate ground-truth data to carry out supervised training of deep learning-based HPSS models, it is impossible to generate a dataset with ground-truth signals for the expected transient and stationary sources. This is due to the fact that it is not possible to find exclusively transient signals and exclusively stationary signals in the real world. So, usually TSS models are either performed via signal processing techniques [Fitzgerald et al., 2014] that try to model transient and non-transient patterns, or as an adaptation of HPSS models with additional sparsity losses [Roma et al., 2018b]. Furthermore, it is common that a deep learning HPSS model ends up doing TSS when applied to music signals with different timbre characteristics from the signals utilised during training.

Traditional methods for performing HPSS usually tend to exploit distinctive patterns of harmonic and percussive signals to perform the separation. A good example is HPSS by median filtering [Fitzgerald, 2010], where the segregation of the vertical and horizontal patterns is done by applying a vertical and a horizontal median filtering technique in the mixture power spectrogram. This method was later extended to an iterative process that uses different filter sizes at each iteration [Driedger et al., 2014a] and improved by combining source-specific proximity kernels [Fitzgerald et al., 2014; Liutkus et al., 2014]. However, methods of this nature have intrinsic performance limitations caused by the

hand-crafted filters and the strict assumptions they are based around. More recently, deep learning based HPSS algorithms began to rise in the literature and have shown a significant improvement over traditional HPSS methods because they can automatically learn features from data [Lim and Lee, 2017; Drossos et al., 2018].

2.2.3 Evaluation Metrics for Source Separation

There are two main types evaluation methodologies for source separation: *subjective* and *objective*. While both types are important and bring complementary benefits for fully evaluating the performance of a separator, each type of evaluation has its own set of drawbacks.

Subjective evaluation procedures necessarily involve human raters, who give scores to predictions based on a reference signal. On the other hand, objective evaluation procedures rate the quality of a prediction by performing a set of automated calculations and generating a numerical measure representing a score. While subjective evaluations are more reliable, they are much time consuming and expensive to obtain as well as subject to variability of human raters. In this thesis, no formal subjective tests and evaluation have been performed, but some subjective comments after listening to the separation estimates are mentioned in the text when discussing the results of the developed source separation experiments

Objective evaluation, however, are much faster and cheaper, but at the same time might struggle since there are many aspects of human perception that are extremely difficult to model using only computational instructions. The evaluation of the methods proposed in this thesis has been done following the usual set of objective measures of research works in the literature related to source separation.

2.2.3.1 Objective Evaluation

There is a set of three objective measures that are, to date, the most widely used metrics for evaluating a MSS framework [Stöter et al., 2018; Mitsufuji et al., 2021]. This set include the Source-to-Distortion Ratio (SDR), Source-to-Interference Ratio (SIR), and Sources-to-Artefact Ratio (SAR) measures [Vincent et al., 2006].

Such evaluation measures can be seen as scores that are computed comparing the output signals of a separation system with the ground-truth isolated sources. In order to mathematically model them, suppose that an estimate $\hat{s}$ of a source

signal s_{source} is composed by four components [Vincent et al., 2006]:

$$\hat{s} = s_{\text{source}} + e_{\text{interf}} + e_{\text{artef}} + e_{\text{noise}}, \tag{2.6}$$

where e_{interf} represents the interference caused by the other sources in the estimated signal and e_{artef} is the part of the signal that contains the artefacts and defects possibly inserted in the source estimation due to the separation procedure. The term e_{noise} is sometimes added in the signal modelling to represent the noise in the mixture. In the cases studied in the thesis, this value is neglected.

Based on Equation (2.6), it is possible to define the following objective measures:

Signal-to-Distortion Ratio (SDR)

The SDR is the main objective measure to evaluate the separation quality. It gives us an insight on how much actual information from the target source is effectively contained in the estimated source signal compared with the signals related to all undesired signal. It is defined as:

$$\text{SDR} = 10\log_{10}\left(\frac{\|s_{\text{source}}\|^2}{\|e_{\text{interf}} + e_{\text{artef}} + e_{\text{noise}}\|^2}\right), \tag{2.7}$$

where $\|\cdot\|^2$ represents the squared norm of the argument sinal.

Signal-to-Interference Ratio (SIR)

The SIR evaluates the quality of the separation measuring how much of the other sources have been inserted in the target source. It ignores the noise and the artefacts inserted by the separation method. It is defined as:

$$\text{SIR} = 10\log_{10}\left(\frac{\|s_{\text{source}}\|^2}{\|e_{\text{interf}}\|^2}\right); \tag{2.8}$$

Sources-to-Artefacts Ratio (SAR)

The SAR carries the information of the unwanted artefacts that have been inserted during the separation process. It is an estimate of how much the other 3 component signals are when compared to the amount of included artefacts, which is represented by e_{artef}. It is defined as:

$$\text{SAR} = 10\log_{10}\left(\frac{\|s_{\text{source}} + e_{\text{interf}} + e_{\text{noise}}\|^2}{\|e_{\text{artef}}\|^2}\right), \tag{2.9}$$

Note that it is necessary to have prior knowledge of not only the ground-truth signal related to the target source that is being evaluated, but also all the other isolated sources in order to compute e_{interf}.

Moreover, while this set of metrics is currently used as the standard practice to evaluate source separation algorithms in the specialised literature, there is a lot of features in the human auditory perception that those metrics fail to capture. Therefore, while objective evaluation gives a general idea of how good an estimate might sound, they are not perfect and, in some cases, having a higher objective metric does not translate in better sound quality. It is always recommended to listen to the resulted estimations in order to carry out final conclusions.

The experiments related to source separation developed as part of this thesis will be explained in Chapter 3, and while being evaluated using the objective measures explained in this subsection, repositories that include music examples are also provided and cited accordingly throughout the chapter.

Algorithms to compute each component signal shown in Equation (2.6) are out of the scope of the thesis and the interested reader is pointed to Vincent et al. [2006]; Le Roux et al. [2019]. In the context of this thesis, all those metrics have been computed using the publicly available Python package `museval`[3].

2.3 Automatic Instrument Recognition

The problem of instrument recognition can be defined as the ability of a system to automatically identify the number and the names of instruments that are emitting sounds throughout the music recording. Such task is usually formulated as a multi-label classification task that can be addressed either on a frame-level [Hung and Yang, 2018; Hung et al., 2019; Wu et al., 2020], where the purpose is to obtain the instrument activations across time, or on a clip-level basis [Han et al., 2017; Gururani et al., 2019; Solanki and Pandey, 2019], where the purpose is to estimate the instruments that are present in an audio clip. The former instrument recognition task is commonly known as *Instrument Activity Detection (IAD)* while the latter as *instrument tagging*.

It is common that sounds from two or more instruments have a high level of perceptual similarity, which makes instrument recognition a challenging task even for humans. In order to address this problem, the system has to be robust and take into account the large variance in timbre and in performance style within a single instrument, as well as, deal with the superposition of multiple instruments in time and frequency.

[3] `https://github.com/sigsep/sigsep-mus-eval`

Notwithstanding, instrument recognition is also deeply related to the task of Automatic Music Transcription (AMT), which is the process of creating any form of notation for a music signal [Benetos et al., 2019]. AMT is currently one of the most challenging and discussed topics in the MIR community [Benetos et al., 2013, 2019]. Most AMT systems are designed to transcribe a single monophonic or a single polyphonic source into a musical score (or piano-roll). In this case, the main sub-task involved in the process is Multi-Pitch Estimation (MPE), where predictions regarding the pitch and time localisation of the musical notes are carried out.

However, when analysing polyphonic multi-instrumental recordings, not only each note should have its pitch and duration properly estimated, but the information regarding the timbre of sounds should also be correctly processed [Duan et al., 2014]. Multi-instrument transcription is still a rarely investigated problem [Wu et al., 2019] since it is mandatory to have a way of recognising the instrument that played each note. This particular task can be formulated as note-level instrument recognition and is also known as *instrument assignment* [Benetos et al., 2013] or *pitch streaming* [Duan et al., 2014].

2.3.1 Methods for Instrument Recognition

Methods for instrument recognition can be divided into knowledge-driven and data-driven methods.

2.3.1.1 Knowledge-Driven Methods

Most of the earlier works of instrument recognition are knowledge-driven methods, focusing on using handcrafted features along with signal processing and probabilistic analysis in order to classify the instruments. Those works usually address solo excerpts of monophonic music signals [Herrera-Boyer et al., 2003; Essid et al., 2004; Diment et al., 2013]. Essid et al. [2004], for example, uses Mel-Frequency Cepstrum Coefficients (MFCCs) along with Principal Component Analysis (PCA) to reduce dimensionality and Gaussian mixture models (GMM) for classifying solo phrases of 5 instruments. On the other hand, Diment et al. [2013] proposes a Modified Group Delay Feature (MODGDF), which is a combination of MFCCs with phase information, and perform the classification based on it to improve the results.

When classifying recordings with multiple monophonic instruments, one way of performing the recognition is to use pattern recognition algorithms, which are usually trained on isolated instrument solos, but applied directly to the polyphonic signal at inference time. Those methods usually use Support Vector Machine (SVM) classifiers [Simmermacher et al., 2006; Essid et al., 2006] or

complex probabilistic models [Kitahara et al., 2007]. Another typical approach is to use the methodology of decompose and recognise, where the polyphonic signal is first decomposed using source separation techniques and later classified. For instance, Heittola et al. [2009] developes a method that uses an NMF algorithm on top of a polyphonic pitch estimation. Each separated source is later analysed by pre-trained GMMs and classified accordingly.

Considering recordings with multiple polyphonic instruments, the task of instrument recognition has shown to be even more complicated [Benetos et al., 2019]. The approaches to the task are similar, but the performance of traditional methods are much worse, and the training is usually still done by analysing isolated notes. For instance, Burred et al. [2009] and Burred et al. [2010] propose to use models for the timbre of each instrument, created from isolated instrument notes by means of sinusoidal modelling, PCA and dynamic spectral-envelope modelling. The classification is done later by grouping and separation of sinusoidal components using timbre template matching.

Regarding specifically instrument assignment, just few works have explored this particular task. For instance, Duan et al. [2014] approached it using a constrained clustering of frame-level pitch estimates obtained from an MPE algorithm via the minimisation of timbre inconsistency within each cluster. The authors tested different timbre features for both music and speech signals. Arora and Behera [2015] proposed a similar method, where Probabilistic Latent Component Analysis (PLCA) is applied in order to decompose the audio signal into multi-pitch estimates and extract source-specific features. Then, clustering is performed under the constraint of cognitive grouping of continuous pitch contours and segregation of simultaneous pitches into different source streams using Hidden Markov Random Fields (HMRF). Both of those works, however, assume that each source is monophonic, i.e., each instrument could only play a single note at a time.

An alternative approach is to model the temporal evolution of musical tones [Benetos and Dixon, 2013]. This method is based around the use of multiple spectral templates per pitch and instrument source that correspond to sound states. The authors use hidden Markov model-based temporal constraints to control the order of the templates and streamed the pitches via shift-invariant PLCA.

2.3.1.2 Data-Driven Methods for Instrument Recognition

Data-driven techniques for instrument recognition have considerably higher performance than the traditional knowledge-driven methods. However, in order to perform a supervised learning approach, labelled data is necessary. Obtaining

frame-level instrument annotations is expensive and time consuming; data containing such types of label is scarce and, consequently, data-driven methods that perform IAD are rare. Labelling clips of audio signals, on the other hand, is much faster and easier to obtain. Due to this fact, the majority of instrument recognition works fall into the category of predominant instrument recognition, or clip-level instrument recognition (instrument tagging). Some works propose the utilisation of spectrogram-based CNNs to perform the recognition [Han et al., 2017; Solanki and Pandey, 2019; Jawaherlalnehru and Jothilakshmi, 2019], while in Li et al. [2015] an end-to-end approach using a time-domain CNN to directly pinpoint the multiple instruments in a small clip of audio is used. Gururani et al. [2018] proposes to utilise a CRNN and compare this approach to a CNN and Multi-Layer Perceptron (MLP).

In an attempt to overcome the lack of strongly (frame-wise) labelled data, some of the more recent approaches use attention mechanisms on models trained on weakly (clip-wise) label data in order to pinpoint parts in the clip where the classified instruments are activated [Gururani et al., 2019; Anhari, 2020; Southall et al., 2017].

Regarding frame-level instrument recognition, or IAD, Hung and Yang [2018] allies a Constant-Q Transform (CQT) with pitch information from a multi-pitch estimation algorithm to perform instrument recognition using the MusicNet dataset [Thickstun et al., 2017]. The authors propose to use a deep neural network based on a combination of convolutional layers with three residual blocks to implement frame-level instrument recognition of multiple instruments. The experiments in this thesis also use MusicNet to train and evaluate the proposed instrument recognition approaches.

Briefly speaking, the MusicNet dataset [Thickstun et al., 2017] contains 330 freely-licensed classical music recordings by 10 composers, written for 11 instruments, along with over 1 million annotated labels indicating the precise time of each note in every recording and the instrument that plays each note.

Regarding pitch streaming, Tanaka et al. [2020] also approaches the task via clustering, but applies to a joint input representation combined of the spectrogram and the pitchgram, which is obtained using an MPE algorithm. In their proposal, each bin of the joint input is encoded onto a spherical latent space taking into account timbral characteristics and the piano-rolls of each instrument is later estimated via masking of the pitchgram based on the results of a deep spherical clustering technique applied on the latent space.

2.4 Multi-task approaches

Instead of using a DNN model to perform a single task, the multi-task learning paradigm [Ruder, 2017] has shown that it is possible to perform multiple tasks simultaneously using a single network and improve its generalisation capabilities. In the same way that additional training examples guide the model parameters towards generalising well to different data, when including a different but similar task to a model, most of its parameters is shared across tasks, and hence become constrained towards good values. In other words, the idea behind multi-task learning is that considering tasks as deeply related, it might be possible to leverage from shared latent representation to perform multiple tasks at once. This makes multi-task models often yield better generalisation [Goodfellow et al., 2014].

Generally, a new task is included in the framework by adding new terms in the loss function that is optimised during training. A couple of recent works propose to use multi-task learning to solve multiple tasks in MIR using a single model. For example, Stoller et al. [2018b] proposes a voice separation system that is also able to perform voice activity detection simultaneously. Since voice activity and voice detection are similar tasks, the multi-task model that uses both loss functions is shown to obtain better results. Hung et al. [2019], is able to improve their previously proposed instrument recognition method [Hung and Yang, 2018] by creating a model that is able to jointly predict the instrument class along with the pitch of the notes. This is done by adding pitch-related loss along with instrument-related loss during the training.

Moreover, other multi-task methods have successfully proposed to perform multi-instrumental AMT that are able to directly estimate the pitches and associate them to their instrumental source jointly [Bittner et al., 2018; Hung et al., 2019; Manilow et al., 2020]. In this case, the instrument assignment task is implicitly performed by the model. In [Bittner et al., 2018], a multitask deep learning network jointly estimated outputs for various tasks including multiple-pitch, melody, vocal and bass line estimation. The Harmonic Constant-Q Transform (HCQT) of the audio signal was used as input and the data used for training was semi-automatically labelled by remixing a diverse set of multitrack audio data from the MedleyDB dataset [Bittner et al., 2017]. Hung et al. [2019] uses a DNN to jointly predict the pitch and instrument for each audio frame. They used the Constant-Q Transform (CQT) as input to their system and trained using a large amount of audio signals synthesised from MIDI piano-rolls. Manilow et al. [2020], on the other hand, is able to jointly transcribe and separate an audio signal into up to 4 instrumental sources — piano, guitar, bass and strings. They propose the Cerberus network architecture, where three outputs (heads)

are jointly estimated: an embedding space (trained via deep clustering loss), the separated source signals (trained via mask inference loss) and the piano-roll transcription of each source (trained via a transcription loss). However, their system is trained with only synthesised signals.

While all methods proposed in this thesis are focused on single tasks, it is possible to understand the proposed methods for HPSS, and explained in detail in Chapter 3, as a multi-task framework, since those methods estimate both sources (harmonic and percussive) simultaneously using a single model. This is reflected in the proposed loss function, which contains two distinct source-specific factors. More information can be seen in Chapter 3.

2.5 Generative Adversarial Networks

Given a set of data instances X and a set of labels Y, while discriminative models capture the conditional probability $p(Y|X)$, generative models capture the joint probability $p(X, Y)$, hence including the distribution of the data itself in the framework.

Generative Adversarial Networks (GANs) are a family of deep generative models that are able to learn a distribution of data via adversarial training [Goodfellow et al., 2016]. The GAN framework can be defined based on a theoretical scenario in which two networks compete against each other. The first network is called *generator* and is responsible for generating samples

$$\mathbf{z} = G(\mathbf{x}, \theta_G), \tag{2.10}$$

where $G(\mathbf{x}, \theta_G)$ represents the transformation that the generator, with its set of parameters θ_G, performs in a generic input $\mathbf{x}$. Its adversary is a classifier that is called *discriminator* and its main purpose is to distinguish samples that are generated by the generator from samples taken from real data. The discriminator can be modelled as outputting the value of $D(\mathbf{z}, \theta_D)$ representing the probability that $\mathbf{z}$ is a real training example rather than a fake sample generated by the generator.

Supposing that the real samples $\mathbf{z}$ follow the distribution $p_{\mathbf{z}}$ while the input $\mathbf{x} \sim p_{\mathbf{x}}$ and the generator output is a mapping from $\mathbf{x}$ into a resulting distribution $q_{\mathbf{z}}$, the idea of the GAN framework is to make $q_{\mathbf{z}}$ approximate $p_{\mathbf{z}}$.

The discriminator can be trained to distinguish real training examples drawn from $p_{\mathbf{z}}$ from fake examples generated by the generator drawn from $q_{\mathbf{z}}$ by max-

imising the following cross-entropy loss:

$$\hat{\theta}_D = \arg\max_{\theta_D} \left(\mathop{\mathbb{E}}_{\mathbf{z}\sim p_{\mathbf{z}}} \big[\log D(\mathbf{z}, \theta_D) \big] + \mathop{\mathbb{E}}_{\mathbf{z}\sim q_{\mathbf{z}}} \big[\log(1 - D(\mathbf{z}, \theta_D)) \big] \right). \tag{2.11}$$

The generator treats the discriminator as its adversary and is optimised to generate samples that are able to fool the discriminator. This can be achieved by optimising the following loss function

$$\hat{\theta}_G = \arg\min_{\theta_G} \left(\mathop{\mathbb{E}}_{\mathbf{x}\sim p_{\mathbf{x}}} \big[\log(1 - D(G(\mathbf{x}, \theta_G), \theta_D)) \big] \right) \tag{2.12}$$

While firstly proposed in [Goodfellow et al., 2014], many variant formulations for GANs have been developed and they have proven to be a powerful approach to modelling complex probability distributions and are incredibly useful for various generation and prediction tasks. Some examples of applications are image generation [Goodfellow et al., 2014; Wang et al., 2018; Noguchi and Harada, 2019; Brock et al., 2019], image super-resolution [Sønderby et al., 2017] and image-to-image translation [Zhou and Zhang, 2017; Isola et al., 2017; Tang et al., 2019, 2020]. For an overview of GANs see Creswell et al. [2018]

GANs have also been used in source separation tasks with the objective of creating more realistic source masks [Stoller et al., 2018a; Fan et al., 2018; Kong et al., 2019]. In those works the separator network is treated as the generator of a traditional GAN framework and a discriminator that learns to distinguish real ground-truth source data from separated (fake) data is also included. The separator is then trained to fool the discriminator while at the same time perform source separation. In Ong et al. [2019] a different framework consisting of multiple separators and multiple discriminators are adversarially trained to perform source separation using a method denoted as cross adversarial source separation by the authors.

In this thesis, the GAN framework is utilised in order to address the domain adaptation problem. In Section 3.3 a method consisting of a separator and a domain discriminator that are adversarially trained with partly unlabelled data is proposed for adapting a source separation system to a new domain.

2.6 Domain Adaptation

Even though it is known that the use of data augmentation techniques, such as random pitch-shifting and random mixing of source signals for the task of source separation can help data-driven methods to generalise to out-of-sample data [Uhlich et al., 2017; Cohen-Hadria et al., 2019], the performance of a data-

driven model will always depend on the type of audio data used during training [Goodfellow et al., 2014].

The paradigm of domain adaptation handles the generalisation problem in cases where two or more distinct domains (datasets) are being utilised. To simplify, suppose that two domains, namely domain A and domain B, are available to train and evaluate a predictor. Each domain is a distinct dataset with different characteristics, and, in the field of MIR, examples of such characteristics can be different types of sources, timbre of instruments, music styles, genres and audio effects. If the predictor is trained using data from A and tested using data from B, its performance will unavoidably become much worse [Abu-Mostafa et al., 2012; Goodfellow et al., 2014]. In other words, when the data distribution of the training set is different from the data distribution of the test set, the performance of any predictor is degraded. This effect is known as *dataset shift* or *domain shift* [Quionero-Candela et al., 2008].

Under this scenario, domain adaptation techniques [Redko et al., 2019] try to adapt a machine learning model to have the same level of performance (or as close as possible) on both domains. Methods of this nature use the following nomenclature: the dataset originally used for training is called **source domain** and the dataset where the model will be evaluated is called **target domain**.

Domain adaptation techniques address the dataset shift problem by adapting predictors from a *source domain*, where usually a large amount of labelled data is available, to a *target domain*, where only few or no labelled data is available. In Section 3.3, a domain adaptation method for performing source separation is proposed and it is shown that the method can improve separation performance on a target domain by leveraging from mixtures from such domain.

2.6.1 Methods for Domain Adaptation

Domain adaptation methods can be either supervised or unsupervised depending on the type of data from the target domain that is used. While Supervised Domain Adaptation (SDA) methods use labelled data from both domains, Unsupervised Domain Adaptation (UDA) exploits only unlabelled data from the target domain.

A typical SDA approach is to first train a model using a large number of labelled samples from the source domain and then re-train some (or all) of its layers using a smaller labelled dataset of interest (target domain). This technique is known as *fine-tuning* since it tunes the weights of the originally trained model according to the new type of data [Oquab et al., 2014; Wang et al., 2018]. Another SDA approach is *joint training*, where the two datasets are merged into a new dataset and only a single training stage is done, using labelled data from

both domains in every batch [Maciejewski et al., 2018; Manilow et al., 2019]. Fine-tuning of models to a different domain provides a boost in performance on the target domain at the cost of a decrease in performance on the source domain, while joint training tends to bias the performance towards the dataset with the larger amount of labelled data.

UDA methods perform unsupervised training using unlabelled data from the target domain to give clues to the model about the new characteristics of the data from this domain. The main idea of the majority of such methods is to assume that the framework is under the *covariate shift paradigm* [Bickel et al., 2009], i.e., even though the marginal distribution of source domain data is different from the marginal distribution of target domain data, the conditional probability of the output remains the same. Therefore, if the marginal distributions can be matched, the same predictor can be applied successfully over samples from either of the two domains [Shimodaira, 2000].

In order to do this, some UDA methods propose to re-weight [Huang et al., 2006] or to select samples from the source domain [Gong et al., 2013] and retrain in order to perform the adaptation, or to automatically detect and label only the most representative data from the target domain and use it in a supervised scenario in a technique called active learning [Su et al., 2020]. Other methods project the data through an embedding function such that not only the marginals become similar on the embedded space, but also the embedded features keep their discrimination potential [Fernando et al., 2013; Xiao et al., 2018]. This is also the assumption behind [Ganin and Lempitsky, 2015; Ganin et al., 2016], but those works find such an embedded space by means of adversarial training.

The task of domain adaptation is already consolidated as an important research topic in computer vision [Daumé III et al., 2010; Long et al., 2013, 2016; Wang et al., 2019], where it is used in complex classification tasks involving data even from multiple source domains [Peng et al., 2019]. Even in fields closer to MIR, such as acoustic scene analysis [Gharib et al., 2018; Wei et al., 2020], speech recognition [Sun et al., 2017], speech enhancement [Meng et al., 2018] and Natural Language Processing (NLP) [Li, 2012], domain adaptation methods have already been proposed. However, as far as I was able to verify, there is no prior work specifically applying the domain adaptation methodology for source separation. See Section 3.3 for a detailed explanation of the contribution of the thesis in this area.

Chapter 3

Musically motivated CNNs for Harmonic-Percussive Source Separation

3.1 Introduction

This chapter provides a detailed description of the research developed with respect to the topic of music source separation along with corresponding experiments and novel contributions. More specifically, rather than of building increasingly larger architectures hoping to obtain better results, the initial focus of this chapter is to propose a novel musically motivated CNN architecture for the task of music source separation, namely 3W-MDenseNet. By incorporating domain knowledge and promoting parameter sharing in the design of the network architecture, 3W-MDenseNet achieves state-of-the art separation performance in HPSS with much more efficiency if compared to other methods in the literature, i.e., with lower number of trainable parameters. In Section 3.2 a comprehensive description of the proposed CNN architecture is given and a systematic investigation of the effects of design choices, such as shape of filters of the convolutional layers, on the model performance is done.

Furthermore,

Afterwards, the chapter addresses the task of domain adaptation (see Section 2.6 for general definition and background) applied to music source separation. Fully supervised approaches have performance limitations based on the characteristics of the dataset used for training. In particular, due to the high variability of the timbre and pitch of sounds that are usually considered part of a single source in MSS approaches, large amounts of labelled data (ground-truth source

signals) are required in order to make models generalise well to out-of-sample music recordings, but publicly available datasets are rather small. While data augmentation approaches, such as, pitch shifting and randomly mixing sources might combat overfitting and reduce the generalisation error, they are not always a suitable technique, specially in cases where no labelled data for particular sounds are available. Therefore, working with two different domains, one of which with a unique set of timbral characteristics for the harmonic and percussive sounds but with no labelled data (ground-truth source signals) availability, Section 3.3 introduces a generic, adversarial unsupervised domain adaptation framework applied to the HPSS problem. By leveraging unlabelled data (mixtures) from this domain, the experiments show that the proposed framework improves separation performance on the new domain without losing any considerable performance on the original domain. Also, in Subsection 3.3.4.1 a description of the Tap & Fiddle dataset, a dataset containing recordings of Scandinavian fiddle tunes along with isolated tracks for foot-tapping and violin, can be found.

The work in Section 3.2 was published in Lordelo et al. [2019] and most of the work discussed in Section 3.3 was published in Lordelo et al. [2021b]. The Tap & Fiddle dataset (Subsection 3.3.4.1) is also another contribution of the thesis and is freely available for research purposes in Lordelo et al. [2020a].

Lastly, Section 3.4 concludes the chapter by discussing our findings and proposals as well providing an outlook of potential future steps.

In summary, the contributions of this chapter include:

- Musically Motivated CNN Architecture for MSS: Proposal of the 3W-MDenseNet, a novel convolutional encoder-decoder for MSS that use musically motivated kernel shapes for the internal convolutions, facilitating learning useful feature-maps related to harmonic and percussive sounds, which consequently improves HPSS performance. It is shown that 3W-MDenseNet is able to achieve state-of-the-art performance in HPSS with a lower number of trainable parameters if compared to other DNN models.

- Unsupervised Domain Adaptation for HPSS: Proposal of an adversarial unsupervised domain adaptation framework for HPSS that can be used with any neural network architecture. The experiments show considerable improvement of HPSS performance over data from other domains, when compared to the performance obtained by the proposed fully supervised HPSS approach and when compared to other benchmark methods.

- Unsupervised Domain Adaptation for MSS: As long as I was able to verify, the proposed unsupervised domain adaptation method is the first work in

the literature discussing and applying domain adaptation methods for the task of MSS. While the prototype experiments were performed for the particular task of HPSS, the proposed framework can easily be adapted to use other types of input representation and to estimate other type of sources;

- Curation of Novel Dataset: Public release of the "Tap & Fiddle Dataset", a dataset containing recordings of traditional Scandinavian fiddle tunes with accompanying foot-tapping along with isolated tracks for "foot-tapping" and "violin". This dataset has different timbral characteristics than MUSDB18 [Rafii et al., 2017], which is the largest publicly available dataset with non-artificial music recordings that can be used for source separation.

3.2 Musically Motivated CNN for HPSS

After recent increases of computational power and free availability of large amounts of data, deep learning methods have become really popular. Due to its impressive representation capacity, DNNs have pushed the boundaries of state-of-the-art performance in many applications and nowadays progressively more complicated and complex tasks are addressed in an end-to-end fashion using methods of this nature. Consequently, their architectures have become increasingly larger and deeper. A similar trend occurs in the topic of music source separation. Analysing the results of recent music source separation evaluation campaigns, such as, the Signal Separation Evaluation Campaign (SiSEC) 2018 [Stöter et al., 2018] and the Music Demixing Challenge (MDX) 2021 [Mitsufuji et al., 2021], it is easy to verify that the vast majority of the top performer methods for MSS are large deep-learning-based models, with dozens of millions of trainable parameters. While large models tend to perform well in most cases, it can be very costly to train, deploy and maintain very large deep neural networks [Menghani, 2021] and they might not necessarily be efficient enough for direct deployment in most applications that require real-time processing and/or deployment on low-powered devices. Moreover, larger models also need larger amounts of data to generalise well and not overfit on training data. In this section the focus is to build more efficient DNNs (high performance and low number of parameters) that can be used for source separation. While the experiments are done for the particular task of HPSS, the model can easily be adapted to separate other type of sources.

A technique that can be used to reduce the number of trainable parameters of a deep neural network is pruning the model's layers [Han et al., 2015; Li et al., 2016]. In this approach, models are first trained with a large number

of channels and afterwards the importance of each channel is evaluated using specific measures. Finally, channels with less importance are discarded from the network and the pruned model is then retrained to recover the degradation of performance caused by the pruning of channels. While pruning can help in building more efficient models, it is out of the scope of this thesis. Here the focus is in adopting a more structural methodology, in which the design of the network architecture will take into account efficiency (low number of parameter and high performance) in its design. The main idea is to promote parameter sharing among the model layers and apply domain knowledge to make more musically motivated choices when designing the architecture.

The proposed model is built upon the MDenseNet [Takahashi and Mitsufuji, 2017], which is a variation of the traditional U-Net architecture [Ronneberger et al., 2015], but with a denser arrangement of skip-connections. The MDenseNet already uses fewer parameters than the U-Net and achieves impressive performance when extracting a single source from a mixture [Takahashi and Mitsufuji, 2017]. Even though the U-Net and MDenseNet are not contributions of this thesis, a full understanding of those architectures is necessary in order to comprehend the proposed musically motivated model. Therefore, Subsection 3.2.4 and Subsection 3.2.5 provide descriptions of each of those architectures, respectively.

In the context of the thesis, the 3W-MDenseNet is proposed, where three different kernel shapes are used in its internal convolutional layers. A branch only with vertical filters, a branch only with square filters and a branch only with horizontal filters. The model is trained in a supervised scenario to perform HPSS and yields estimates of more than just a single source. In other words, both the harmonic and percussive sources are estimated simultaneously. Figure 3.1 shows a block diagram of the proposed framework, where the harmonic and percussive magnitude spectrograms are estimated directly from an input mixture magnitude spectrogram using a single deep convolutional encoder-decoder network. The detailed description of the 3W-MDenseNet can be found in Subsection 3.2.6.

It is important to note that the proposed model has around 0.6 million parameters, which is substantially lower than most of the current DNN-based methods for music source separation. For instance, the traditional U-Net [Jansson et al., 2017] uses 8.7 million parameters, Uhlich et al. [2017] use a BLSTM for source separation that uses more than 30 million parameters, and Mimilakis et al. [2017] propose a complex recurrent encoder-decoder approach for singing voice extraction that uses a total of 24 million parameters.

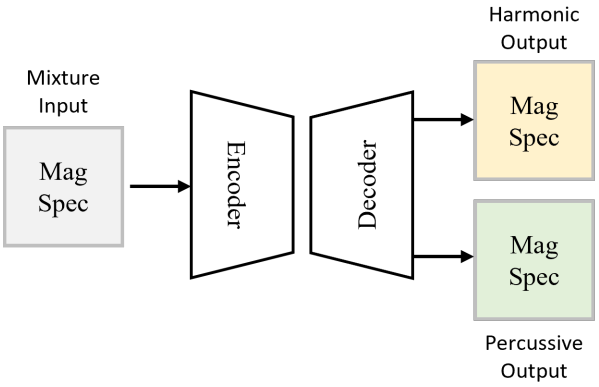


Figure 3.1: Diagram of the proposed HPSS framework. A single network (encoder-decoder) that receives the mixture magnitude spectrogram as input estimates the magnitude spectrograms of both sources: harmonic and percussive.

3.2.1 Parameter Sharing

Promoting parameter sharing is a way of reducing the number of trainable parameters of deep learning models. The most known example of this idea is the development of the convolutional layer, which was first successfully used for image classification [LeCun et al., 1989]. One of the convolutional layer's main advantages, if compared to traditional fully connected layers, is promoting *parameter sharing* by avoiding the necessity to learn separate weights for each input pixel. After their introduction, CNNs quickly became the top performing models for many different tasks and until today they are one of the most commonly used deep learning architectures. However, as the problems researchers try to solve are becoming more and more complex, CNNs have become very deep, with models usually reaching dozens or hundreds of millions of trainable parameters. When designing typical CNNs, it is common practice to double the number of channels (feature-maps) as the depth of network increases. This fact leads to a dramatic increase in the number of parameters required by the model.

In order to avoid this exponential explosion, a different methodology for promoting parameter sharing is adopted by the proposed CNN. The same strategy of the Multi-scale DenseNet (MDenseNet) [Takahashi and Mitsufuji, 2017] is used to build the DNN. The basic idea is that rather than doubling the number of feature-maps after each down-sampling unit of the network, Takahashi and Mitsufuji [2017] proposed to utilise dense blocks (DenseNets) [Huang et al., 2017], a dense arrangement of skip-connections between each of the model's layers, instead of regular stacks of convolutional layers, at every scale (between

each down-sampling unit) so feature-maps of earlier layers can also be "seen" at later layers. This approach makes the model avoid learning redundant feature-maps throughout its layers. Therefore, it is possible to maintain the number of channels at every scale of the network constant. Takahashi and Mitsufuji [2017] proposed the MDenseNet to perform the extraction of a single instrument from a mixture and have shown that this model is able to learn feature-maps more efficiently. Here, the original MDenseNet is adapted for the task of HPSS, estimating both the percussive and harmonic sources simultaneously.

3.2.2 Domain Knowledge

Another contribution of the thesis is to apply domain knowledge when designing the architecture to make CNNs more efficient for the HPSS task. It is important to note that, different than what happens with regular images, when working with audio magnitude spectrograms as inputs for CNNs, the two dimensions of the feature-maps do not have the same meaning of space. The two axes of audio spectrograms correspond to time (horizontal axis) and frequency (vertical axis). Therefore, wider filters are capable of learning longer temporal dependencies in the audio signal while taller filters are capable of learning more frequency-spread timbral features [Pons et al., 2016]. Thus, by tweaking the shapes of the filters of the convolutional layers for the target task it is possible to improve the performance of the model and reduce its computational complexity.

This strategy is grounded in previous works which point out that including multiple kernel shapes in CNNs is an efficient way to exploit the network capacity towards building more efficient musically motivated architectures [Pons et al., 2016; Phan et al., 2016; Pons and Serra, 2017; Pons et al., 2017]. For instance, Pons et al. [2016] study what different filter shapes learn when using CNNs to perform audio classification tasks, while Pons et al. [2017] propose an efficient model for timbre analysis using several filter shapes that minimise the risk of noise-fitting and over-fitting. In this Section, a similar methodology is adopted to build an efficient musically motivated CNN architecture for the task of HPSS. The main idea is to use convolutional layers with multiple kernel shapes — horizontal, square and vertical — in order to facilitate learning of HPSS-relevant time-frequency patterns. A dense arrangement of skip-connections is also utilised with the objective of promoting parameter sharing and reducing the number of trainable parameters of the model.

3.2.3 Dense Block (DenseNet)

Consider mathematically modelling the input-output function of a composite layer. It can be any type of layer, such as, a fully connected, convolutional or

pooling layer, for example. It is possible to write its output y as a series of transformations H performed by the layer on its input x, i.e.:

$$y = H\{x\}. \tag{3.1}$$

In the case of a convolutional layer, the transformation H is a convolution, but it is possible to generalise this notation to any type of layer.

When stacking composite layers, a new layer is applied directly to the output of another composite layer. Thus, in a block of L layers, the output x_i of the i-th layer can be modelled as:

$$x_i = H_i\{x_{i-1}\}, \tag{3.2}$$

where x_0 is the initial input and H_i represents the sequence of transformations performed by the i-th composite layer. Observe that, in this traditional way of stacking composite layers, each layer (other than the last) is connected to just a single other layer. Therefore, a stack of L layers has only L internal connections if the final output is counted as a connection. Figure 3.2 illustrates a regular stack of 4 composite layers. Since the thesis contains work mainly related with convolutional neural networks, the text may refer to those composite layers as convolutional layers, but the mathematical modelling and explanation should hold for any type of composite layer.

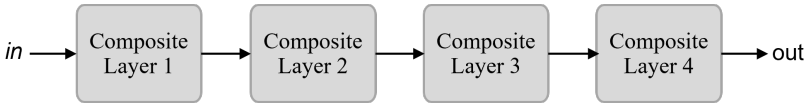


Figure 3.2: Example of a regular stack with 4 composite layers. Observe that each layer is only connected to the subsequent layer, which makes this stack have only 4 total connections.

In a **DenseNet** [Huang et al., 2017], sometimes also called in this thesis as **Dense Block**, the feed-forward nature of the stack is preserved, but a dense pattern of skip-connections is used in the stack. Each layer now obtains additional inputs coming from all preceding layers and passes on its own output (or feature-maps in the case of convolutional layers) to all subsequent layers. In contrast to ResNets [He et al., 2016], in a DenseNet there is never addition of features through summation before passing them into a layer; instead, the combination of features is done by concatenation. Mathematically, the output of the i-th layer becomes:

$$x_i = H_i\{[x_{i-1}, x_{i-2}, x_{i-3}, \ldots, x_0]\}, \tag{3.3}$$

where $[\dots]$ represents the concatenation operation (over the channels dimension). In this densely connected stack of L layers, the i-th layer has now as input a concatenation of i outputs, consisting of the outputs of all preceding layers and the initial input, and its own output x_i is passed on to all $L - i$ subsequent layers. This fact increases the number of internal connections to $\frac{L(L+1)}{2}$. Such dense connectivity pattern was first introduced by Huang et al. [2017] and is illustrated schematically in Figure 3.3 for $L = 4$.

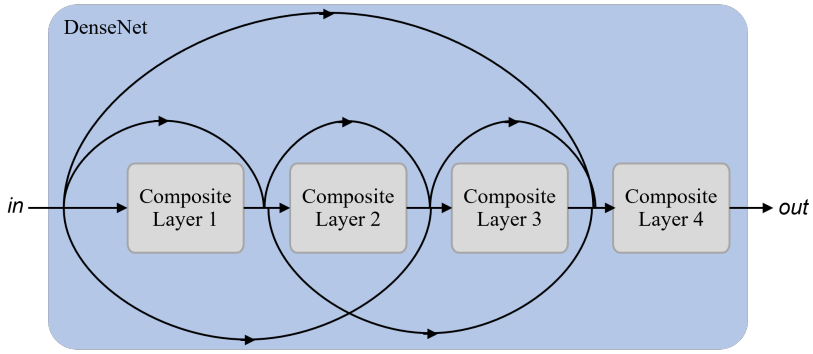


Figure 3.3: Example of a densely connected stack (DenseNet) with 4 layers. Observe that each layer's output is sent to every subsequent layer, which makes the total number of connections become 10 in this 4-layered dense block.

By interconnecting all layers of a convolutional stack in this way, maximum information flow is ensured between the layers of the network, which not only reduces vanishing gradients during training, but also allows the network to have fewer trainable parameters than traditional networks, as there is no need to relearn redundant feature-maps, due to the re-utilisation of feature-maps already computed in the preceding layers. However, it is important to note that as most of the internal connections of a DenseNet consist of concatenations of feature-maps over the channel dimension, it is necessary that all the feature-maps have the same shape. Therefore, it is not possible to utilise layers that change the size of feature-maps, such as pooling layers, as part of a single dense block. Notwithstanding, it is still possible to build densely connected encoder-decoder networks by dividing the network into multiple DenseNets interconnected by down-sampling or up-sampling units. An illustration of this approach is shown in Figure 3.4.

The layers in this densely connected block were proposed with the idea of producing a constant number of new feature-maps (channels) at every layer [Huang et al., 2017]. This number is referred to as the **growth rate** of the network and is denoted by k. Consequently, as each layer produces k feature-maps, it follows that the i-th layer has $k_0 + k(i-1)$ input feature-maps, where k_0 is the number of channels in the initial input layer. In this thesis, the separation

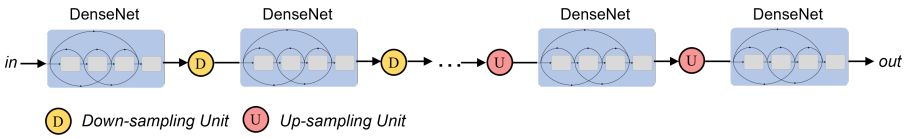


Figure 3.4: A densely connected encoder-decoder network can be built by interconnecting densely connected blocks (DenseNets) of composite layers with down-sampling and up-sampling units. Note that the size of feature-maps does not change inside a DenseNet, but varies between two consecutive DenseNets due to the application of a down-sampling or up-sampling unit.

of monoaural music recordings is performed using an encoder-decoder structure, where stacks of DenseNets are interconnected by down-sampling units (encoder) and up-sampling units (decoder), so the initial DenseNet has have $k_0 = 1$ while the others have as k_0 the growth rate of the previous dense block.

Another way of understanding a DenseNet is to view the feature-maps in the stack as "a global state" of the network. The growth rate k regulates how much new information each layer contributes to this "collective knowledge" of the network. As the information flows forward in the network, each layer adds only k new feature-maps of its own to the global state and, once concatenated, the new global state can be accessed from everywhere within the network. Differently than traditional stacks of convolutional layers, there is no need to replicate the network state from layer to layer, which makes such stack of layers more efficient. The growth rate of DenseNets can be very narrow, such as, only 12 filters per layer, which adds only a small set of feature-maps to the "collective knowledge" of the network and keeps the number of trainable parameters low [Huang et al., 2017].

In the context of this thesis, a convolutional encoder-decoder is built considering that composite layer consists of a convolutional layer with a Rectified Linear Unit (ReLU) activation function [Hahnloser et al., 2000] followed by a batch normalisation layer. The application of padding on the convolutional layers is required to make sure the shape of the feature-maps do not change between the internal composite layers of a single DenseNet.

For the rest of the chapter, let's consider the number of channels (output feature-maps) of each convolutional layer as constant and denoted by the growth rate, k, while the number of layers in a DenseNet denoted by L.

3.2.4 U-Net

The U-Net is a DNN architecture originally proposed for biomedical image segmentation [Ronneberger et al., 2015], but it has also been used for other tasks including source separation [Jansson et al., 2017]. It is an adaptation of the

deconvolutional network [Noh et al., 2015], a convolutional encoder-decoder architecture in which the size of the feature-maps is successively halved on the encoding path due to the application of a series of convolutional and pooling layers, but the number of channels is doubled at every scale. Once a final small encoded representation of the input is generated, it is decoded back to the original size of the input by multiple series of unpooling, deconvolution and rectification operations. The channels at every scale of the decoding path are halved while the dimensions of feature-maps are doubled. Figure 3.5 shows an illustration of the original U-Net architecture.

In a U-Net, the main difference from a regular deconvolutional network is the addition of forward skip-connections between layers at the same resolution level from the encoder directly to the decoder. The skip-connection is done via a concatenation operation and is done to ensure that the decoding (later) layers are exposed to the high level of detail that exists on the earlier encoded feature-maps.

As the layers go deeper in the encoding path, there is a loss in the resolution of the feature-maps due to the application of pooling layers (down-sampling units). This makes the decoder unable to rebuild higher resolution feature-maps with the same level of detail they had before being down-sampled by previous encoder layers. Hence, the U-Net includes forward skip-connections from the encoder path to the corresponding scale in the decoder path, which allows the decoder to directly receive the high level of detail of the earlier feature-maps. This is particularly useful for source separation and image segmentation since every detail of the input spectrogram or of the input image carries important information to solve those tasks and the encoder-decoder should be able to reproduce them on the output. Moreover, the addition of those skip-connections also helps to reduce the problem of vanishing gradients since there is a direct path from later layers to earlier layers during backpropagation.

3.2.5 Multi-scale DenseNet (MDenseNet)

The Multi-scale DenseNet (MDenseNet) [Takahashi and Mitsufuji, 2017] is a deep encoder-decoder network whose architecture is built upon a regular U-Net [Ronneberger et al., 2015]. Instead of using traditional convolutional layers with exponential increase of the number of channels as the layers go deeper, the MDenseNet uses densely connected stacks (DenseNets) with a small growth rate at every scale.

Similar to a U-Net, the MDenseNet also includes forward skip-connections from the encoder path to the corresponding scale in the decoder path, which allows the decoder to directly receive the high-level information of earlier feature-

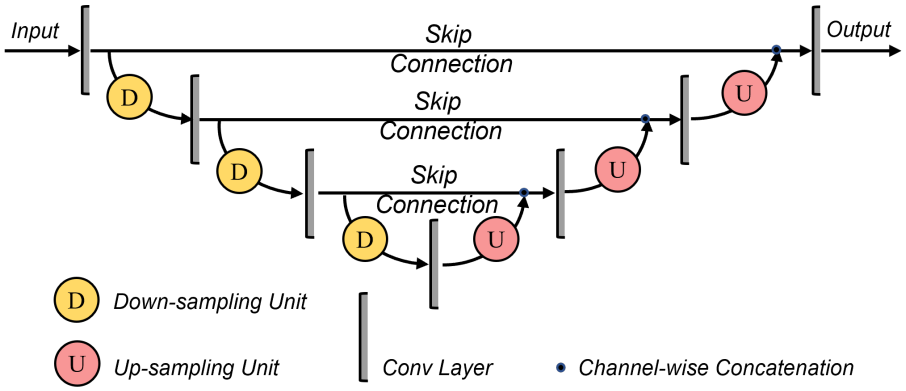


Figure 3.5: Example of a U-Net architecture with depth of 4. Observe the U-shaped structure of the architecture, which inspired its name. The channels of the convolutional layers are omitted, but they double at every scale during the encoder path and are halved at every scale in the decoder.

maps. An example of an MDenseNet architecture with depth 4 is schematically shown in Figure 3.6. It is important to note that the depth of an MDenseNet should not be confused with the depth of its internal DenseNets. The depth of the MDenseNet indicates the number of scales in the network and, consequently, the number of DenseNets in its encoding path, while the depth of a DenseNet represents the number of layers that have been densely connected in a single stack, i.e., at a single scale.

In the context of the proposed HPSS architecture, multiple MDenseNet structures running in parallel are used. All the dense blocks of a single MDenseNet will have the same growth rate k and the same number of layers L independently of the scale at which they are located. Moreover, the depth of each MDenseNet will be notated as d. Max-pooling layers and transposed convolutions with kernel shape of (2×2) are utilised as down-sampling and up-sampling units, respectively.

3.2.6 Proposed Architecture - 3W-MDenseNet

The main idea of the proposed architecture is to adapt the original MDenseNet encoder-decoder structure by including convolutional layers with filters of different shapes to facilitate learning of the harmonic and percussive patterns present in the input spectrogram. In order to do that, the *Three-Way Multi-scale DenseNet* (3W-MDenseNet) is proposed, where instead of using a single encoder-decoder path, 3 separate branches of MDenseNets that are combined later are used. Each branch uses a different filter shape for its internal convolutional layers.

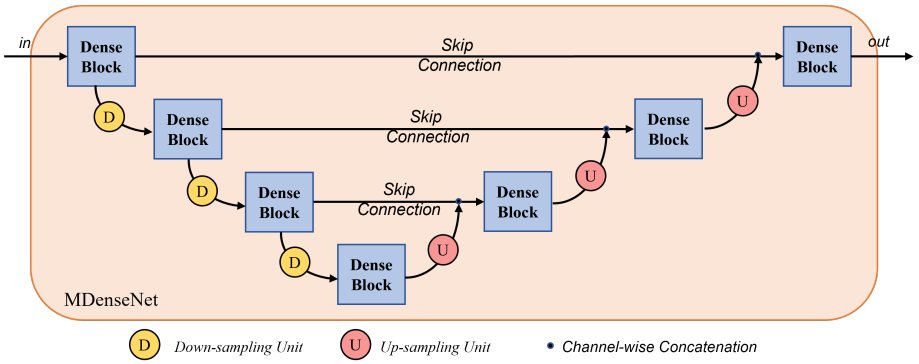


Figure 3.6: MDenseNet architecture with a depth $d = 4$. The major difference from the U-Net is the utilisation of densely connected convolutional stacks (DenseNets) at every scale instead of single or regular stacks of convolutional layers with exponential growth of channels.

A branch with the conventional square kernel shape of (3×3) for all the convolutional layers is used and 2 extra branches, where all the internal convolutional layers have kernel shapes of (13×1) (vertical filters) and (1×13) (horizontal filters), respectively, are also included. Those new shapes were set based on the vertical and horizontal patterns that percussive and harmonic components form in the mixture spectrogram. It is expected that the rectangular filter shapes will help the model to learn those patterns more easily and efficiently. Those shapes were empirically set after the first initial set of experiments varying the rectangular filters length from 7 to 23. The value of 13 seemed to be a good trade-off in having good performance while maintaining the number of trainable parameters of each branch of the 3W-MDenseNet similar, i.e., while each filter of the vertical and horizontal branch uses 13 parameters per kernel, the branch with square filters contains 9 parameters per each (3×3) kernel. A thorough evaluation of multiple kernel shape sizes is left as future work.

Note that a branch with square filters is still used in the architecture because no real-world musical instrument is strictly percussive or harmonic — the percussive-like patterns present in the transient part of harmonic instrumental sounds still need to be learned and correctly associated to the harmonic source, for example.

The detailed architecture of the proposed HPSS model is depicted in Figure 3.7 with the necessary parameters to re-implement the model. The direct outputs of the 3W-MDenseNet are soft masks for the harmonic and percussive sources that are later multiplied by the mixture magnitude spectrogram in order to generate the final estimation of the source magnitude spectrograms.

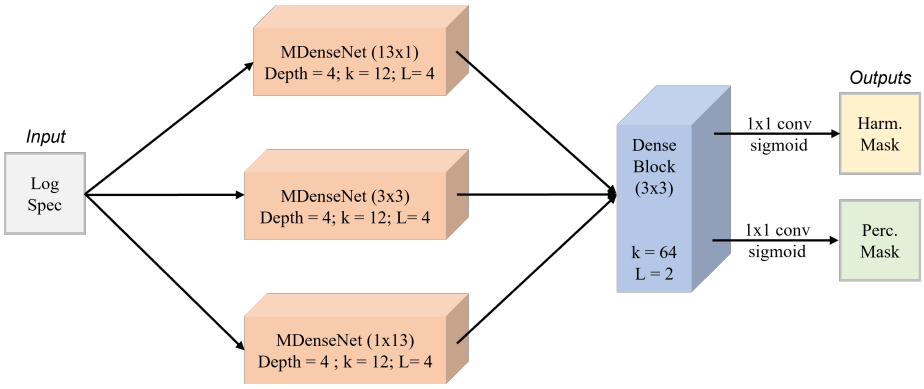


Figure 3.7: Proposed architecture of the 3W-MDenseNet. Three MDenseNets process the same input magnitude spectrogram in parallel and their outputs are concatenated in a final DenseNet with a LeakyReLU activation function. Harmonic and percussive soft masks are generated by two separate (1×1) convolutional layers with sigmoid activation functions. All the convolutions use 'same' zero-padding to ensure the final concatenation of the three branches can be done successfully.

The loss function optimised during the training of the 3W-MDenseNet is a linear combination of the mean squared errors between the magnitude spectrograms of the estimated sources and their respective ground-truths. Supposing the mixture magnitude spectrogram is denoted by X, the ground-truth for the percussive part is P and the ground-truth for the harmonic part is H, it is possible to define the loss function $\mathcal{L}$ as:

$$\mathcal{L} = \lambda_P ||(M_P \odot X) - P||_F^2 + \lambda_H ||(M_H \odot X) - H||_F^2, \tag{3.4}$$

where M_P and M_H are the estimated time-frequency masks for the percussive and harmonic sources respectively, $\odot$ represents the Hadamard (element-wise) product and $||\dots||_F$ the Frobenius norm. Since the interest is to obtain maximum performance on the separation of both sources, the weights λ_P and λ_H are set as 0.5, but these values can be modified according to the application.

The end-to-end framework for performing HPSS of a time-domain audio signal is shown in Figure 3.8. The initial step is to take a short audio clip and, after performing its Short Time Fourier Transform (STFT), use it as a single input to the 3W-MDenseNet model. The DNN directly estimates magnitude spectrograms for the harmonic and percussive sources. The final time-domain harmonic and percussive signals are then generated by computing the inverse STFT (i-STFT) using the mixture phase.

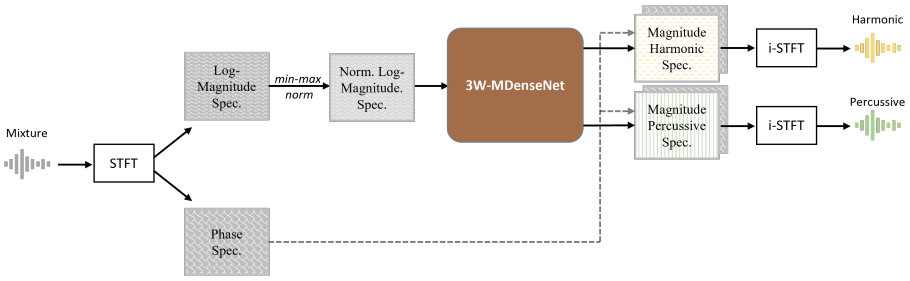


Figure 3.8: End-to-end framework for performing harmonic-percussive source separation of music recordings.

3.2.7 Dataset and Training Details

The dataset utilised for training and testing the model is MUSDB18 [Rafii et al., 2017]. MUSDB18 is the largest public dataset for MSS containing real-world audio recordings. It was the official dataset used for the evaluation of the separation models submitted to SiSEC 2018 [Stöter et al., 2018] and for training models submitted to MDX 2021 [Mitsufuji et al., 2021].

Its corpus has approximately 10h of high quality (sampling rate of 44.1kHz) audio data spread among 150 full-length tracks with different durations. In this dataset, all signals are stereo and mixed using professional digital audio workstations, which makes it a good representative of real application scenario. Furthermore, two other important factors are that it has many musical genres included in its corpora, such as jazz, electro, metal, and others, and for all the recordings not only the full mix is available, but also the original stems for *vocals, bass, drums* and *'other'* source signals. Therefore, MUSDB18 can be used for training source separation systems for those instruments. In the HPSS experiments the *drum* tracks are set as ground-truth sources for the percussive source signals and the sum of other tracks as ground-truth for the harmonic sources.

MUSDB18 is organised into two directories: one containing 100 recordings to be used as a training set and another with 50 recordings to be used as a test set. Supervised training is done using the *drums* stem as ground-truth for the percussive source and the difference between the original mixture and the drums stem as ground-truth for the harmonic source. From the total of 100 recordings assigned as the default training set of MUSDB18, 80 were used for training the model while the other 20 recordings were used as a validation set. The dataset is freely available online[1] along with Python development tools to automatically load, process and evaluate separation performance.

The training was performed using the ADAM optimiser with initial learning

[1]https://sigsep.github.io/datasets/musdb.html#musdb18-compressed-stems

rate of 0.001, which was reduced if the validation loss did not decrease for 3 consecutive epochs. After the validation loss ceases improving for 15 consecutive epochs, the training process is stopped and the model with lower total mean square loss on the validation set is taken.

The initial step is to take a short monoaural audio clip of 1.5 seconds from a music recording and, after performing its Short Time Fourier Transform (STFT) using a window length of 1024 samples with 50% of overlap, a magnitude spectrogram with size (513×128) is computed. Afterwards, a min-max normalisation procedure is performed over the log1p, i.e., $\log(1+x)$, version of the spectrogram and given as input to a 3W-MDenseNet. Such normalisation is done using the global minimum and maximum value over all the spectrograms in the training set. The DNN directly estimates magnitude spectrograms for the harmonic and percussive sources. The final time-domain harmonic and percussive signals are then generated by computing the inverse STFT (i-STFT) using the mixture phase.

3.2.8 Evaluation Metrics

The HPSS performance is measured with the usual set of objective measures for source separation, which include the SDR, SIR and SAR. More information regarding those metrics and what they represent the reader is referred to Subsection 2.2.3. A higher value of each of those metrics means that it is expected that the model provides a better separation quality. The metrics were computed using the `museval`[2] Python package.

3.2.9 Experimental Results

In the experiments, apart from the 3W-MDenseNet, three additional models are implemented, evaluated and compared. The first is the traditional U-Net [Jansson et al., 2017] with square filters of shape (3×3) with exponential channel growth at each scale. This model serves as a baseline approach and uses neither dense blocks in its architecture nor multiple filter shapes. The second is a modified version of the U-Net adopting the same design strategy of multiple branches with different kernel shapes. This model uses the same filter shapes utilised by the 3W-MDenseNet in a similar multi-branch architecture, but does not contain any dense blocks. It is denoted by 3W-U-Net in the rest of the section. The third method is my own implementation of the MDenseNet [Takahashi and Mitsufuji, 2017] using only square (3×3) filters. All the methods use ReLU activation functions in their internal layers with a final sigmoid activation

[2] https://github.com/sigsep/sigsep-mus-eval

function to estimate masks for both harmonic and percussive sources. To properly evaluate the models' capacity, the number of total trainable parameters of each method was kept in the range of 550k–610k.

The performance of the 4 models is evaluated on the default test set of MUSDB18, which has a total of 50 music recordings — a considerably high number of samples if compared to just the 80 music-recordings that were used for training. The results are expressed in dB in Table 3.1 and the highest values for each metric appear in boldface. Overall, it can be seen that the proposed 3W-MDenseNet architecture obtained better average results, having the highest values for all averaged metrics.

Table 3.1: Objective evaluation of HPSS. The values are in dB and represent the mean of the metrics over all 50 songs in the test set of MUSDB18.

Model	Percussive			Harmonic			Average		
	SDR	SIR	SAR	SDR	SIR	SAR	SDR	SIR	SAR
U-Net	3.16	4.32	5.27	9.56	10.84	12.15	6.36	7.58	8.71
3W-U-Net	3.25	4.34	**5.80**	9.80	**10.98**	12.26	6.53	7.66	9.03
MDenseNet	3.48	4.92	5.49	**9.93**	10.84	12.46	**6.71**	7.88	8.98
3W-MDenseNet	**3.70**	**5.84**	5.35	9.71	10.48	**13.32**	**6.71**	**8.16**	**9.34**

3.2.9.1 Effects of Kernel Shapes

The only difference between the U-Net and 3W-U-Net is the presence of multiple kernel shapes in the CNN architecture. When comparing their HPSS performance as shown in Table 3.1 it is easy to note that there was an increase in each of the 3 metrics in both the harmonic and the percussive source estimation. Furthermore, when comparing the regular MDenseNet with the musically motivated 3W-MDenseNet it is also possible to see that there was an increase in average SIR and SAR metrics and no change in the SDR. Therefore, the results show that, by just adding filters of different shapes in the traditional U-Net or in the MDenseNet architectures, the performance of the HPSS is already improved. This fact consolidates the idea that different filter shapes increase network capacity and facilitate learning of high-level features related to harmonic and percussive instrumental sounds.

3.2.9.2 Effects of Skip-Connections

Regarding the effects of adding skip-connections in the network, we can also see from Table 3.1 that there is a gain accross every single metric when going from the *U*-Net to the MDenseNet. When comparing the 3W-U-Net with the

3W-MDenseNet, despite having a drop of 0.45 dB in the SAR metric of for the percussive source and a drop of 0.5 dB in the SIR metric for the harmonic source, the average value of all the metrics across both sources were also improved, which suggests that the addition of skip-connections is also beneficial for the separation.

3.2.9.3 Comparison with other Methods for Drum Extraction

It is important to note that most of the research and experimentation regarding this proposal was done in 2019 and at that time, most state-of-the-art methods for drum separation were evaluated as part of SiSEC 2018 [Stöter et al., 2018] campaign, which also usedthe MUSDB18 dataset for assessment. The results for the percussive separation could be compared with the results published on the campaign for drum extraction. This is depicted in Figure 3.9, where there are box-plots sorted by the highest to the lowest median SDR. Differently than Table 3.1, where the metrics are aggregated by computing the mean value over tracks, in Figure 3.9 the aggregation is done by median value instead in order to follow the same methodology used in the SiSEC 2018 evaluation. Comparing the 3W-MDenseNet with the original MDenseNet, it is possible to see that adding multiple kernel shapes reduced the variance of the overall HPSS performance and also considerably improved the SIR value — this can also be concluded from Table 3.1, where one can notice an increase of 0.9 dB on the SIR value of the percussive separation. Comparing to other methods that only used MUSDB18 training data, one can see that the proposed 3W-MDenseNet model achieves competitive performance.

It's also worth noting that UHL1 and UHL2 [Uhlich et al., 2017], two methods that achieved slightly better aggregated metrics than the 3W-MDenseNet as shown in Figure 3.9, employ a BLSTM model with approximately 30 million parameters for performing source separation. In contrast, the proposed method achieves similar performance with only 0.6 million parameters, highlighting the model's efficiency in source separation. For instance, the number of trainable parameters of the 3W-MDenseNet is considerably lower than most of the current DNN-based methods for audio source separation. The U-Net proposed in Jansson et al. [2017] uses 8.7 millions of parameters, Mimilakis et al. [2017] propose a complex recurrent encoder-encoder approach for singing voice extraction that uses a total of 24 million parameters.

Furthermore, results of pair-wise significance tests following Conover-Inman methodology are displayed in Figure 3.10 to assess which methods have higher statistically significant difference in their results.

More information about the experiments and audio samples can be found in

the experiment repository[3].

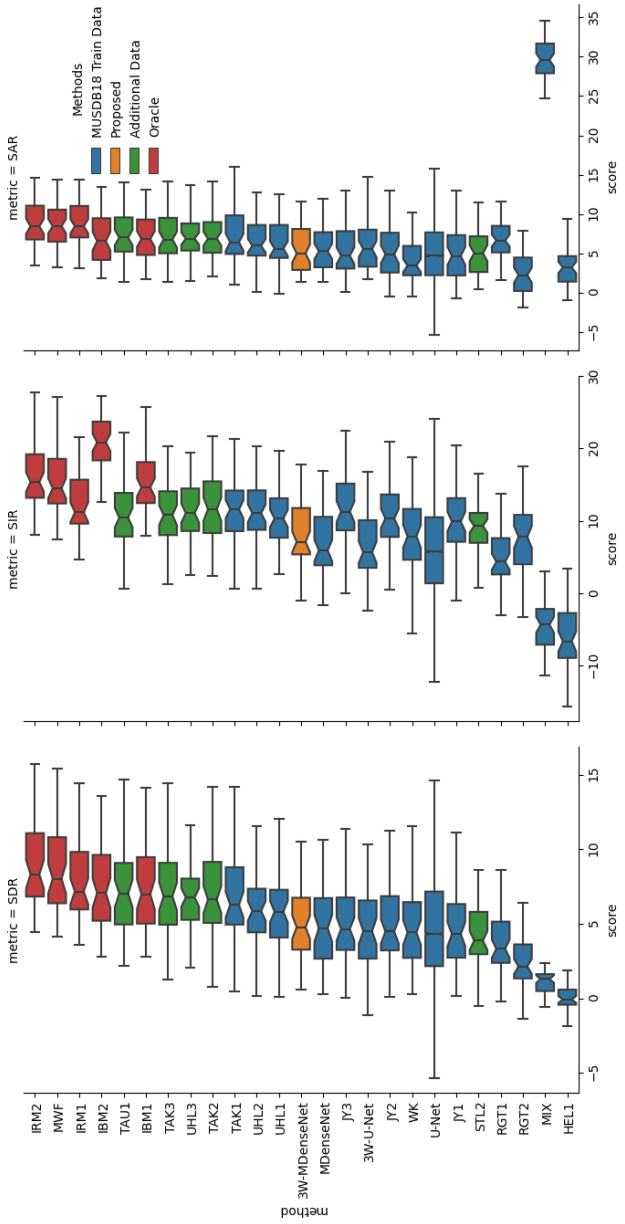


Figure 3.9: Comparison of the SiSEC 2018 results for drums extraction with the results obtained in percussive source estimates in the HPSS experiments. The metrics are aggregated by computing the median value over all 50 songs in the test set of MUSDB18 following the same methodology used in the SiSEC 2018. The oracle methods are Ideal Binary Masks (IBM), Ideal Ratio Masks (IRM) and Multi-channel Wiener Filtering (MWF). For more details regarding each method, the reader is pointed to [Stöter et al., 2018]

[3]http://c4dm.eecs.qmul.ac.uk/WASPAA19-HPSS/

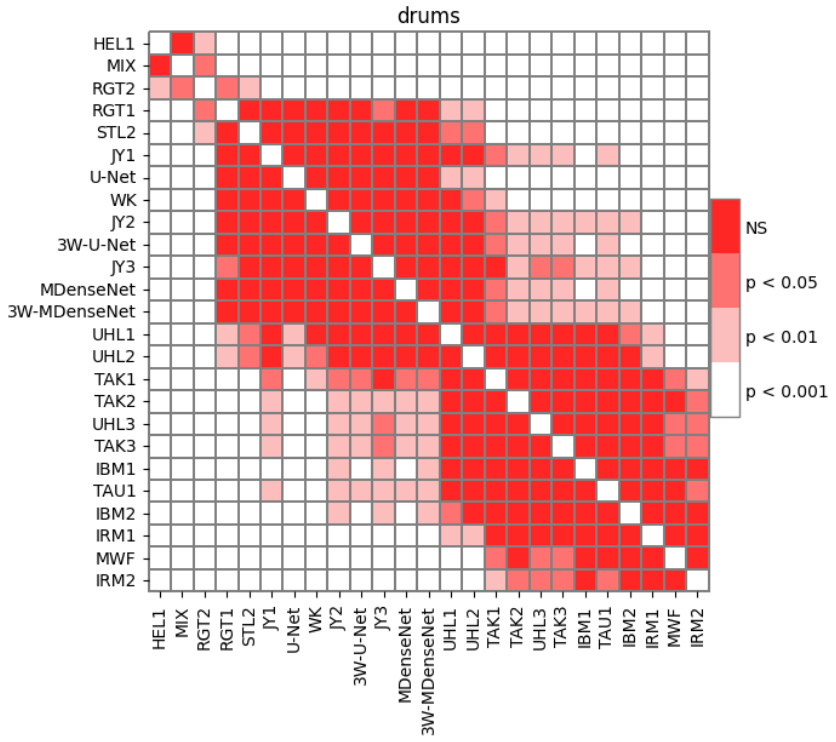


Figure 3.10: Post-hoc pair-wise significance test using Conover-Iman method comparing the SiSEC 2018 results for drums extraction with the results obtained in percussive source estimates in the HPSS experiments. The methods are the same as shown in 3.9. For more details regarding each method, the reader is pointed to [Stöter et al., 2018]

3.3 Adversarial Unsupervised Domain Adaptation for HPSS

In this section a discussion and analysis of the problem of domain adaptation applied to the task of HPSS is provided. In order to increase performance of the separation when performed in music recordings with different characteristics than the original labelled data used when training, an adversarial Unsupervised Domain Adaptation (UDA) approach that exploits unlabelled data from the target domain is proposed. The method is built upon the previously proposed 3W-MDenseNet (see Section 3.2), but here, a domain discriminator is added to the framework and the loss function is modified to support adversarial unsupervised domain adaptation. Furthermore the Tap & Fiddle Dataset, which includes Scandinavian fiddle tunes along with the accompaniment foot-tapping sounds was curated as part of the research. The work explained in this section

was published in Lordelo et al. [2021b] and the dataset is released in Lordelo et al. [2020a].

To my knowledge, this approach is the first attempt on studying and explicitly handling domain adaptation in the MSS topic. By using the mixtures and the available ground-truth signals from MUSDB18 and a set of unlabelled data (mixtures) from the Tap & Fiddle Dataset, a domain with different timbral characteristics, the proposed framework is able to improve separation performance in the new domain while maintaining the original performance on MUSDB18, considerably reducing the degradation effect caused by dataset shift. For an overview of the domain adaptation task including a literature review and a discussion of other related works the reader is referred to Section 2.6.

Although the experiments are carried out for the particular task of Harmonic-Percussive Source Separation (HPSS), the proposed framework can be easily adapted to other MSS tasks with different types of sources and domains.

3.3.1 Motivation

The task of HPSS and similar music source separation tasks pose unique challenges when dealing with music recordings from different domains. In practice, the characteristics of audio signals can vary significantly between these domains, leading to a degradation in separation performance when models trained on one domain are applied to another. To address changes in data distributions between audio data that are used for training (source domain) and for testing (target domain), the goal here is to project the audio data into an embedding space where it is possible to align the distributions regardless if data belongs to source or target domain.

Such distribution alignment is achieved by the utilisation of a domain discriminator. This component distinguishes between source and target domain embeddings during training, encouraging the model to create domain-agnostic representations. Another benefit of the proposed method is that it performs domain adapted HPSS using labelled data (mix and ground-truth source signals) from the source domain, but only unlabelled data (only mixed signals) from the target domain. The proposed approach, inspired by related adversarial unsupervised domain adaptation works in computer vision [Ganin and Lempitsky, 2015; Ganin et al., 2016], employs Convolutional Neural Networks (CNNs) for both encoding and domain discrimination, and is tailored specifically to audio-related tasks like HPSS. These adaptations make the proposed method uniquely suited to addressing the challenges of music source separation and dataset shift.

As explained in Section 2.6, aligning these distributions offers several benefits. But specifically for the music source separation task, the main motivation

is to enhance model generalisation by enabling adaptation to diverse domains without the need of supervised source separation training using labelled data from the target domain. This can ensure robustness across various timbral characteristics and musical styles.

3.3.2 Differences from Previous Work

The closest approaches to the proposed framework are Ganin and Lempitsky [2015] and Ganin et al. [2016]. The proposed method is also looking for a *domain-invariant* and *separation-discriminative* embedding space that can be learned directly from data via adversarial training. However, differently from those two works, the focus of the proposal is to deal with the task of music source separation (regression) instead of image recognition (classification). In addition, CNNs are used for the encoder-decoder and the domain discriminator, while in Ganin and Lempitsky [2015] and Ganin and Lempitsky [2015]; Ganin et al. [2016] simple feed-forward networks are used. Moreover, both of those works perform adversarial training using the gradient reversal layer method, the hereby approach conducts conditional GAN iterative optimisation as in Goodfellow et al. [2014].

Furthermore, since the proposed approach is also grounded in GANs [Goodfellow et al., 2014], before describing it in more detail, it is important to point out some key aspects in which the framework proposed in this Section is different from other GAN-based source separation methods [Stoller et al., 2018a; Fan et al., 2018; Ong et al., 2019].

Discriminator

Works on GAN-based MSS use a *source discriminator*, which is trained to differentiate **real** source signals from **fake** source signals. This is different from the proposed approach, where only a *domain discriminator* is trained to differentiate mixtures **across two different domains**.

Unlabelled data

In order to train a source discriminator, the other methods require large number of **single-source** signals, even though those signals do not necessarily have to be paired with a music mixture. For the method proposed in the thesis, only **mixtures** from each of the two domains are needed to successfully train the domain discriminator.

Input to discriminator

The input to a source discriminator of GAN-based MSS works is the **output** of the separator network. The proposed approach applies the domain discriminator on the **encoded feature-maps**, in the middle of the separator network, and not directly on its output.

3.3.3 Proposed Framework

Assume that both the input data and the outputs are $(F \times T)$ magnitude spectrograms, where F is the number of frequency bins and T the number of frames. To simplify the notation, it is possible to treat them as vectors in $\mathbb{R}^K$, where $K = FT$. Hence, the input (mixture signal) is notated as $\mathbf{x}$ and its labels (ground-truth isolated source signals) as the $(K \times 2)$ matrix $\mathbf{Y} = \begin{bmatrix}\mathbf{h}\ \mathbf{p}\end{bmatrix}$, where the first column is the original harmonic vector $\mathbf{h} \in \mathbb{R}^K$ and the second column is the original percussive vector $\mathbf{p} \in \mathbb{R}^K$. Furthermore, consider that the mixture-label pairs follow the joint distribution $p_{\mathcal{A}}(\mathbf{x}, \mathbf{Y})$, or, in other words, it is possible to say that the data "come from domain $\mathcal{A}$". For the general supervised HPSS case, the goal is to train a model based on this data that can be a good predictor of $p(\mathbf{Y}|\mathbf{x} \sim p_{\mathcal{A}}(\mathbf{x}))$.

In this section, I build upon the regular HPSS task performed through any encoder-decoder architecture. For instance, in Section 3.2 the 3W-MDenseNet, a convolutional encoder-decoder for HPSS was proposed. The output of the separator can be represented as an estimate $\hat{\mathbf{Y}} = \begin{bmatrix}\hat{\mathbf{h}}\ \hat{\mathbf{p}}\end{bmatrix}$ of $\mathbf{Y}$ while the encoder-decoder-based separation process can be mathematically modelled as a sequence of two mappings. First, the encoder $\mathcal{E}$ with parameters $\theta_{\mathcal{E}}$ maps the input to an embedded feature space $\mathbf{z} = \mathcal{E}(\mathbf{x}; \theta_{\mathcal{E}})$ and then the decoder $\mathcal{D}$, with parameters $\theta_{\mathcal{D}}$, maps $\mathbf{z}$ to the output $\hat{\mathbf{Y}}$ such that:

$$\hat{\mathbf{Y}} = \mathcal{D}(\mathbf{z}; \theta_{\mathcal{D}}) = \mathcal{D}(\mathcal{E}(\mathbf{x}; \theta_{\mathcal{E}}); \theta_{\mathcal{D}}). \tag{3.5}$$

This separator can be optimised for the general supervised HPSS case using the mean squared error as the loss function. Equation (3.4) describes such type of loss. However, in order to take into account the new notation introduced in this section, now the mean square loss can be denoted as $\mathcal{L}_{\text{S}}$ using the index S to indicate that it is the **Supervised** part of the total loss function of the domain adaptation framework and write:

$$\mathcal{L}_{\text{S}}(\theta_{\mathcal{E}}, \theta_{\mathcal{D}}) = \mathop{\mathbb{E}}_{\mathbf{x} \sim p_{\mathcal{A}}(\mathbf{x})} \left[\lambda_h ||\hat{\mathbf{h}} - \mathbf{h}||^2 + \lambda_p ||\hat{\mathbf{p}} - \mathbf{p}||^2\right] = \mathop{\mathbb{E}}_{\mathbf{x} \sim p_{\mathcal{A}}(\mathbf{x})} \left[||(\hat{\mathbf{Y}} - \mathbf{Y})\mathbf{\Lambda}||_F^2\right] \tag{3.6}$$

$$\mathcal{L}_{\text{S}}(\theta_{\mathcal{E}}, \theta_{\mathcal{D}}) = \mathop{\mathbb{E}}_{\mathbf{x} \sim p_{\mathcal{A}}(\mathbf{x})} \left[\, ||(\mathcal{D}(\mathcal{E}(\mathbf{x}; \theta_{\mathcal{E}}); \theta_{\mathcal{D}}) - \mathbf{Y})\mathbf{\Lambda}||_F^2 \,\right], \tag{3.7}$$

where λ_h and λ_p are weights for the harmonic and percussive outputs respectively — The value of 0.5 is used in the experiments for each, since it is desired the assignment of equal importance to each source — , $||\dots||$ represents the Euclidean norm, $||\dots||_F$ the Frobenius norm and $\mathbf{\Lambda}$ is the vector $\left[\sqrt{\lambda_h}\ \sqrt{\lambda_p}\right]^\top$. Note that Equation (3.7) is the same as Equation (3.4), but with different matrix notation.

Furthermore, now it is assumed that there also exists a new domain $\mathcal{B}$, where mixtures follow the marginal distribution $p_{\mathcal{B}}(\mathbf{x})$, which is considered different from $p_{\mathcal{A}}(\mathbf{x})$. Then the main goal is to be able to robustly predict labels $\hat{\mathbf{Y}}$ given that the input can be from either domain $\mathcal{A}$ or $\mathcal{B}$. Apart from the labelled samples from domain $\mathcal{A}$, we have access to a set of unlabelled data (mixtures) from $\mathcal{B}$ that can be used for performing unsupervised domain adaptation.

Observe that in here it is considered that the system is under the *covariate shift paradigm* [Bickel et al., 2009]. Thus, while $p_{\mathcal{A}}(\mathbf{x}) \neq p_{\mathcal{B}}(\mathbf{x})$, the task is the same across both domains and, therefore, the underlying conditional probability of the output given data comes from either of the domains $p(\mathbf{Y}|\mathbf{x} \sim p_{\mathcal{A},\mathcal{B}}(\mathbf{x}))$ remains fixed.

The proposed framework is depicted in Figure 3.11. The main idea is to learn encoded features $\mathbf{z}$ that are able to not only guarantee a good separation performance, but that are also invariant to domain changes. This means that $\mathbf{z}$ must not contain any discriminative information about the origin of the input ($\mathcal{A}$ or $\mathcal{B}$). By doing so, it is possible to make the distributions $p(\mathbf{z}|\mathbf{x} \sim p_{\mathcal{A}}(\mathbf{x})) = \{\mathcal{E}(\mathbf{x};\theta_{\mathcal{E}})|\mathbf{x} \sim p_{\mathcal{A}}(\mathbf{x})\}$ and $p(\mathbf{z}|\mathbf{x} \sim p_{\mathcal{B}}(\mathbf{x})) = \{\mathcal{E}(\mathbf{x};\theta_{\mathcal{E}})|\mathbf{x} \sim p_{\mathcal{B}}(\mathbf{x})\}$ to become as similar as possible. In order to measure their similarity, a domain discriminator $\mathcal{C}(\mathbf{z},\theta_{\mathcal{C}})$ is used to discriminate the encoded feature-maps between the two domains, i.e., to classify whether the input signal is from domain $\mathcal{A}$ or domain $\mathcal{B}$. Such domain discriminator is a binary classifier that can be trained using only mixture signals by minimising the binary cross-entropy $\mathcal{L}_{\text{U}}$ (index U is used to indicate that it is the **Unsupervised** loss of the proposed framework):

$$\mathcal{L}_{\text{U}}(\theta_{\mathcal{C}},\theta_{\mathcal{E}}) = -\mathop{\mathbb{E}}_{\mathbf{z}\sim p_{\mathcal{B}}(\mathbf{z})}\left[\log \mathcal{C}(\mathbf{z},\theta_{\mathcal{C}})\right] - \mathop{\mathbb{E}}_{\mathbf{z}\sim p_{\mathcal{A}}(\mathbf{z})}\left[\log(1-\mathcal{C}(\mathbf{z},\theta_{\mathcal{C}}))\right]. \tag{3.8}$$

It is possible to ensure that $\mathbf{z}$ will become domain-invariant by forcing the encoder sub-network to generate feature-maps that can fool the domain discriminator. This is achieved by maximising $\mathcal{L}_{\text{U}}$ when training the encoder weights. Such a min-max game is played by the encoder sub-network and the domain discriminator during training just like in GAN training [Goodfellow et al., 2014]. At the same time, $\mathbf{z}$ can keep its separation-discriminative properties if the minimisation of $\mathcal{L}_{\text{S}}$ is included in the loss function. The final encoder loss is, therefore, a combination of the (unsupervised) *adversarial* loss $\mathcal{L}_{\text{U}}$, which can

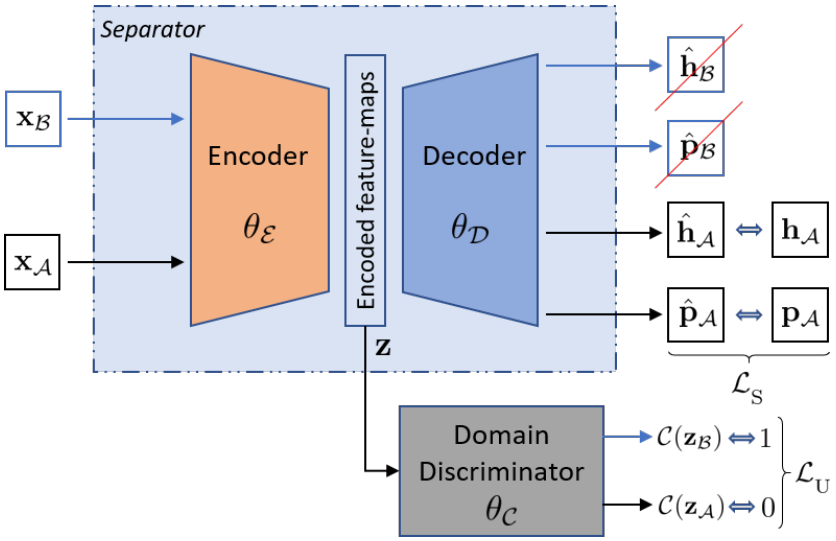


Figure 3.11: Schematic of proposed adversarial Unsupervised Domain Adaptation (UDA) for HPSS. A regular encoder-decoder-like separator is utilised to perform HPSS while a domain discriminator is utilised to ensure the encoded feature-maps are domain-invariant.

be optimised using only mixture signals from each of the two domains, and the (supervised) loss $\mathcal{L}_{\mathrm{S}}$, which can be optimised based only on samples from $\mathcal{A}$ since it requires labelled data. In summary, the loss function of each sub-network is:

$$\hat{\theta}_{\mathcal{C}} = \arg\min_{\theta_{\mathcal{C}}} \mathcal{L}_{\mathrm{U}}(\theta_{\mathcal{E}}, \theta_{\mathcal{C}}) \tag{3.9}$$

$$\hat{\theta}_{\mathcal{E}} = \arg\min_{\theta_{\mathcal{E}}} \big[-\gamma_{\mathrm{U}} \mathcal{L}_{\mathrm{U}}(\theta_{\mathcal{E}}, \hat{\theta}_{\mathcal{C}}) + \gamma_{\mathrm{S}} \mathcal{L}_{\mathrm{S}}(\theta_{\mathcal{E}}, \hat{\theta}_{\mathcal{D}}) \big] \tag{3.10}$$

$$\hat{\theta}_{\mathcal{D}} = \arg\min_{\theta_{\mathcal{D}}} \mathcal{L}_{\mathrm{S}}(\theta_{\mathcal{E}}, \theta_{\mathcal{D}}) \tag{3.11}$$

where γ_{U} and γ_{S} are weights given to the unsupervised part and to the supervised part of the loss.

It should be noted that $\mathcal{C}$, $\mathcal{E}$ and $\mathcal{D}$ must be trained together in an iterative way as in GAN training [Goodfellow et al., 2014]. If $\mathcal{C}$ is optimised to completion, the encoder sub-network will not be able to increase the domain-discriminator confusion, causing the separator performance to overfit over domain $\mathcal{A}$ [Goodfellow et al., 2014].

3.3.3.1 Separator Architecture

The 3W-MDenseNet (see Subsection 3.2.6) is used as the separator architecture. However, the hyperparameters are set differently this time as the experiments have been done in 16kHz. The growth rate of the DenseNets in each branch

has now been set to 20, 24 and 20 at every scale while the kernel shapes for the convolutions to (1×7), (3×3), and (7×1) respectively. Note that those kernel shapes are a bit smaller if compared to the original shapes in the 3W-MDenseNet of Subsection 3.2.6. The reason is due to the fact that now the model process data at 16 kHz and in the previous experiments the original 44.1 kHz was used. The final DenseNet, after the concatenation of branches, uses (3×3) convolutions, has a depth of 3 and growth rate of 64. ReLU activation functions are used after every convolution operation in the separator and a batch normalisation layer is applied between the concatenation layer and the final DenseNet.

In order to construct the encoded feature-maps $\mathbf{z}$, the deepest encoder layer of each of the three branches of the 3W-MDenseNet are concatenated and a single vector $\mathbf{z}$ is generated.

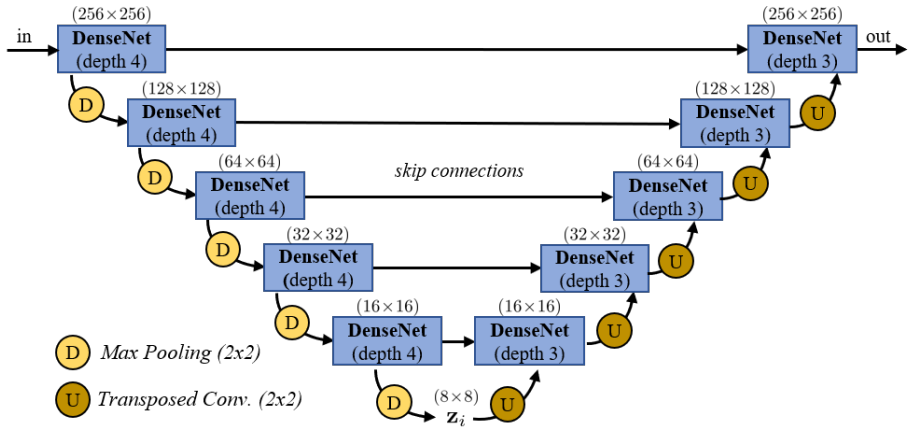


Figure 3.12: Architecture of a single branch of the 3W-MDenseNet. Note that $\mathbf{z}_i$, with $i \in \{1, 2, 3\}$ are each branch's encoded feature-maps, whose concatenation form the final encoded features $\mathbf{z}$ that is used as input for the domain discriminator.

3.3.3.2 Discriminator Architecture

The architecture of the domain-discriminator network is depicted in Fig. 3.13. It consists of a regular stack of 3 convolutional stages, each of which is a (3×3) convolutional layer followed by (2×2) max pooling, with 80, 160 and 320 channels respectively, and three fully connected layers with 128, 32 and 1 channel respectively.

This discriminator is trained as a binary classifier, so a sigmoid activation function is applied after the last fully convolutional layer and Equation (3.8) is used as loss function.

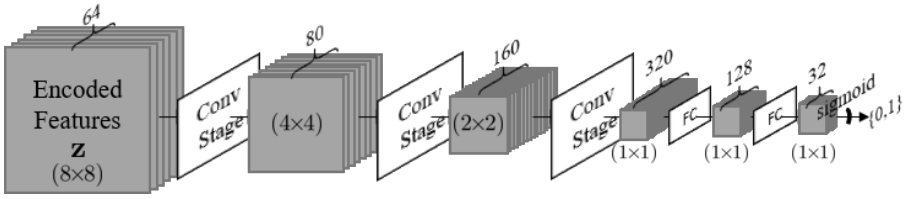


Figure 3.13: Architecture of the domain discriminator. Each "Conv Stage" is a (3×3) convolutional layer followed by (2×2) max pooling. "FC" is a fully connected layer. ReLU activation functions are used after every convolutional layer and fully connected layer apart from the last one, for which a sigmoid activation layer is applied.

3.3.4 Datasets

To perform the experiments related to domain adaptation, two different datasets are used. As *source domain*, or domain $\mathcal{A}$ — as denoted in Subsection 3.3.3 —, the MUSDB18 [Rafii et al., 2017] was used. MUSDB18 is the same dataset utilised for the regular HPSS experiments in Section 3.2, therefore, for details regarding its corpora the reader is referred to Subsection 3.2.7.

To serve as *target domain*, or domain $\mathcal{B}$ — as denoted in Subsection 3.3.3 — a new dataset has been curated containing recordings of Scandinavian fiddle tunes with accompanying foot-tapping, namely "Tap & Fiddle". This dataset contains a completely different timbral characteristic if compared to MUSDB18, while the former has recordings containing only violin and foot-tapping sounds, the latter has much more variability in its instruments, MUSDB18 includes vocals and recordings from many different musical genres such as jazz, pop, electro, metal, and others. The Tap & Fiddle is freely available online for research purposes and was released to the community in Lordelo et al. [2020a] as another contribution of the research developed during the PhD programme.

3.3.4.1 Tap & Fiddle Dataset

The Tap & Fiddle dataset consists of 28 stereo recordings of traditional Scandinavian fiddle tunes with accompanying foot-tapping, which is standard performance practice within these musical styles. Its corpora contains not only the mixed signals, but also the two isolated instrumental tracks that can be used as ground-truths for music source separation algorithms:

- The fiddle track (harmonic track);
- The foot-tapping track (percussive track).

Foot-tapping is very often an integral part of the musical expression in Scan-

dinavian fiddle music as a percussive accompaniment. For instance, some studies have shown that the dance beat of the music can even be unintelligible without the foot-tapping part [Hopkins, 1978]. Notwithstanding, foot-tapping in performance of fiddle music has not been systematically studied yet. Hence, apart from contributing to the music source separation community, the release of Tap & Fiddle to the community can also bring contributions to researchers and enthusiasts working with analysis of fiddle music as well as studies of metrical expression in music in general.

The audio files are uncompressed and saved as stereo ".wav" files with sampling frequency of 44100 Hz and 32 bits per sample. The average duration for a recording in Tap & Fiddle is 65 seconds, totalling around $65 \times 28 = 30$ m 20 s of full play time.

The dataset is divided into a training set with 23 recordings and a test set with 5. It is also important to note that some of the recordings in the dataset are variations of the same Scandinavian fiddle tune. Those recordings are versions containing different acoustic conditions and audio characteristics for the foot-tapping and/or for the violin sound within the same tune.

Audio and Repertoire Characteristics: Tap & Fiddle contains recordings of different dance types, including Norwegian Halling music, with straight single and double tapping as well as Swedish polska tunes, where tapping is considerably scarcer with tapping on beat 1 and 3 in 3-beat time being the most common way to tap. The sound of the foot-tapping ranges from more soft foot-tapping produced by a sock-covered foot, to sharp, distinct and loud foot-tapping produced by shoes with hard heels on parquet. In addition, the loudness of the foot-tapping regardless of sound source in relation to the fiddle is varied between and within recordings.

Recording Methodology: Each isolated signal was recorded by one fiddle player in a natural 30 m^2 room with separate miking for the foot and the fiddle (violin), using close-up Shure SM-58 microphones and a Focusrite sound card recorded in Audacity on a Macbook PRO. The mixture signals were created by adding the two isolated signals together. All the recordings have been made using the same instrument, which was played by the same performer.

3.3.5 Experimental Setup

Preprocessing: Before performing the domain adaptation experiments, it has been noted that the regular HPSS results using the MUSDB18 dataset have similar performance when using either the full-band $44,1$kHz music signals as

input or resampled versions of the input to 16kHz. The music signals of both datasets are then resampled to 16kHz in order to make training faster. The inputs are normalised magnitude spectrograms of size (257×256) generated by the application of an STFT of size 512 with 75% overlap. A validation split of 20% of all labelled data available for training is used.

Once more, the drum stem is used as ground-truth for the percussive source and the sum of all the other stems as the harmonic source when concerning data from MUSD18, while the foot-tapping and violin sounds are used as ground-truth for the percussive and harmonic sources respectively, when evaluating models on Tap & Fiddle.

Postprocessing: After experiments of Section 3.2, the separation results were further improved when Wiener filtering [Nugraha et al., 2016a] was applied to the source estimates, therefore a post-processing stage of Wiener filtering is now applied in the experiments of this section. This goes inline with most other spectrogram-based music source separation models in the literature [Takahashi et al., 2018b; Takahashi and Mitsufuji, 2021; Hennequin et al., 2020; Stöter et al., 2019], which also use Wiener filtering as an efficient post-processing step to improve separation performance.

Training Details: After initial experimentation, the values of 1 for γ_{S} and 0.001 for γ_{U} were chosen. Training is performed using the ADAM optimiser with an initial learning rate of 0.001, which is reduced by a factor of 0.25 if the supervised validation loss $\mathcal{L}_{\mathrm{S}}$ stops improving for 50 consecutive epochs, and if no improvement happens in 200 epochs the training is stopped. In the experiments, at every training iteration, I perform 5 updates on $\theta_{\mathcal{C}}$ before updating $\theta_{\mathcal{E}}$ and $\theta_{\mathcal{D}}$. The full training procedure can be found in Algorithm 1.

3.3.6 Evaluation Metrics

The separation quality is evaluated using the set of objective metrics that are largely used by the MSS community: SDR, SIR and SAR. Those objective measures were also used to evaluate the performance of the HPSS experiments performed in Section 3.2 and are explained in detail in Subsection 2.2.3. Once again, a higher value of each of those metrics means that it is expected that the model provides a better separation quality. Those metrics were computed using the `museval`[4] Python package.

[4]https://github.com/sigsep/sigsep-mus-eval

Algorithm 1: Adversarial Unsupervised Domain Adaptation for HPSS

Result: Adaptation of HPSS to domain $\mathcal{B}$

Inputs : Current weights at iteration i: $\theta_{\mathcal{C}}^{i}, \theta_{\mathcal{E}}^{i}$ and $\theta_{\mathcal{D}}^{i}$,
labelled data from domain $\mathcal{A}$,
unlabelled data from domain $\mathcal{B}$

Outputs: Updated weights $\theta_{\mathcal{C}}^{i+1}, \theta_{\mathcal{E}}^{i+1}$ and $\theta_{\mathcal{D}}^{i+1}$

d_steps $\leftarrow 5$;

$\lambda_h, \lambda_p \leftarrow 0.5$;

$\gamma_{\mathrm{U}} \leftarrow 0.001$; $\gamma_{\mathrm{S}} \leftarrow 1$

for *(**i** = 0 ; **i** <* ***total_batches****; **i**++)* **do**

  for *(**d** = 0 ; **d** <* ***d_steps****; **d**++)* **do**

    sample N samples $\{\mathbf{x}_1^{\mathcal{A}}, \mathbf{x}_2^{\mathcal{A}}, \cdots, \mathbf{x}_N^{\mathcal{A}}\}$ from $\mathcal{A}$

    sample N samples $\{\mathbf{x}_1^{\mathcal{B}}, \mathbf{x}_2^{\mathcal{B}}, \cdots, \mathbf{x}_N^{\mathcal{B}}\}$ from $\mathcal{B}$

    update discriminator $\mathcal{C}$ by gradient ascent on:

$$\nabla_{\theta_{\mathcal{C}}} \frac{1}{N} \sum_{i=1}^{N} \Big[log\left(\mathcal{C}(\mathcal{E}(\mathbf{x}_i^{\mathcal{A}}))\right) + log\left(1 - \mathcal{C}(\mathcal{E}(\mathbf{x}_i^{\mathcal{B}}))\right) \Big]$$

  end

  sample N samples $\{\mathbf{x}_1^{\mathcal{A}}, \mathbf{x}_2^{\mathcal{A}}, \cdots, \mathbf{x}_N^{\mathcal{A}}\}$ from $\mathcal{A}$

  take their respective labels $\{\mathbf{Y}_1^{\mathcal{A}}, \mathbf{Y}_2^{\mathcal{A}}, \cdots, \mathbf{Y}_N^{\mathcal{A}}\}$

  update separator ($\mathcal{E}$ and $\mathcal{D}$) by gradient descent on

$$\nabla_{\theta_{\mathcal{E}}, \theta_{\mathcal{D}}} \frac{\gamma_{\mathrm{S}}}{N} \sum_{i=1}^{N} \left|\left|(\mathcal{D}(\mathcal{E}(\mathbf{x}_i^{\mathcal{A}}; \theta_{\mathcal{E}})) - \mathbf{Y}_i^{\mathcal{A}})\mathbf{\Lambda}\right|\right|_F^2$$

  sample N samples $\{\mathbf{x}_1^{\mathcal{B}}, \mathbf{x}_2^{\mathcal{B}}, \cdots, \mathbf{x}_N^{\mathcal{B}}\}$ from $\mathcal{B}$

  update $\mathcal{E}$ by gradient descent on

$$\nabla_{\theta_{\mathcal{E}}} \frac{\gamma_{\mathrm{U}}}{N} \sum_{i=1}^{N} \Big[log\left(\mathcal{C}(\mathcal{E}(\mathbf{x}_i^{\mathcal{A}}))\right) + log\left(1 - \mathcal{C}(\mathcal{E}(\mathbf{x}_i^{\mathcal{B}}))\right) \Big]$$

end

3.3.7 Results

In the experiments of this section, recordings from MUSDB18 represent the source domain $\mathcal{A}$ while recordings from Tap & Fiddle represent the target domain $\mathcal{B}$ for the domain adapted methods. The aim is to investigate how different training frameworks perform across the two domains and investigate if the domain adapted methods are capable of successfully address the domain shift

Table 3.2: Objective evaluation of HPSS on MUSDB18 Dataset. The values are in dB and represent the median of metrics over tracks in the test set. HPSS_MUSDB represents the method trained only using data from MUSDB18 and HPSS_T&F the method trained only with data from Tap & Fiddle. SDA_joint and SDA_tuned are supervised domain adaptation models. The former is trained with joint data from both domains while the latter is first trained with data from MUSDB18 and later finetuned on Tap & Fiddle dataset. IBM is the Ideal Binary Masking and IRM represents the Ideal Ratio Masking oracle methods. Open_Unmix (UMX) is the baseline DNN proposed in Stöter et al. [2019] while Median_Filtering is the method in Fitzgerald [2010].

Method (Training Set)	Test Set: MUSDB18 (Domain $\mathcal{A}$)						Type of Data	
	Percussive			Harmonic				
	SDR	SIR	SAR	SDR	SIR	SAR	$\mathcal{A}$	$\mathcal{B}$
HPSS_MUSDB ($\mathcal{A}$)	4.5	13.0	5.0	10.0	13.4	12.3	labelled	—
HPSS_T&F ($\mathcal{B}$)	−0.2	0.3	10.5	3.1	16.5	5.2	—	labelled
SDA_joint ($\mathcal{A}+\mathcal{B}$)	4.8	13.3	5.1	10.2	13.9	12.1	labelled	labelled
SDA_tuned ($\mathcal{A}\rightarrow\mathcal{B}$)	2.9	8.6	3.3	7.1	9.3	10.5	labelled	labelled
HPSS_UDA_small	4.8	12.2	5.1	10.0	13.4	11.8	labelled	unlabelled
HPSS_UDA_large	4.6	12.9	4.9	10.1	14.1	12.0	labelled	unlabelled
Open_Unmix (UMX)	5.2	11.2	6.0	10.1	17.7	10.7	labelled	—
Median_Filtering	1.0	0.1	4.1	5.3	5.5	12.5	—	—
IBM	7.8	16.4	7.9	11.9	17.9	13.2	—	—
IRM	8.0	12.4	9.7	12.2	15.8	15.0	—	—

problem.

The UDA proposal is compared not only to traditional supervised HPSS approaches that use only labelled data from one of the domains, but also to SDA frameworks, which include joint training using labelled data from both datasets and fine-tuning over samples from the training set of Tap & Fiddle after pre-training on the training set of MUSDB18, and to other two baseline methods from the literature. The first is a state-of-the-art DNN for MSS named OpenUnmix (UMX) Stöter et al. [2019]. This method is fully supervised trained on an augmented version of MUSDB18. The second baseline is a traditional HPSS method [Fitzgerald, 2010] denoted here as "Median_Filtering", in which a vertical and a horizontal median filtering are applied to the mixture spectrogram with the objective of separating percussive from harmonic sounds. The chosen median filter lengths are 35 for the experiments in this section.

In addition to the mixtures in the Tap & Fiddle dataset (domain $\mathcal{B}$), a larger collection of 50 extra recordings of Scandinavian fiddle tunes with accompanying foot-tapping was also accessible for training. Despite not being release as part of Tap & Fiddle, it can be assumed that such collection is also representative of

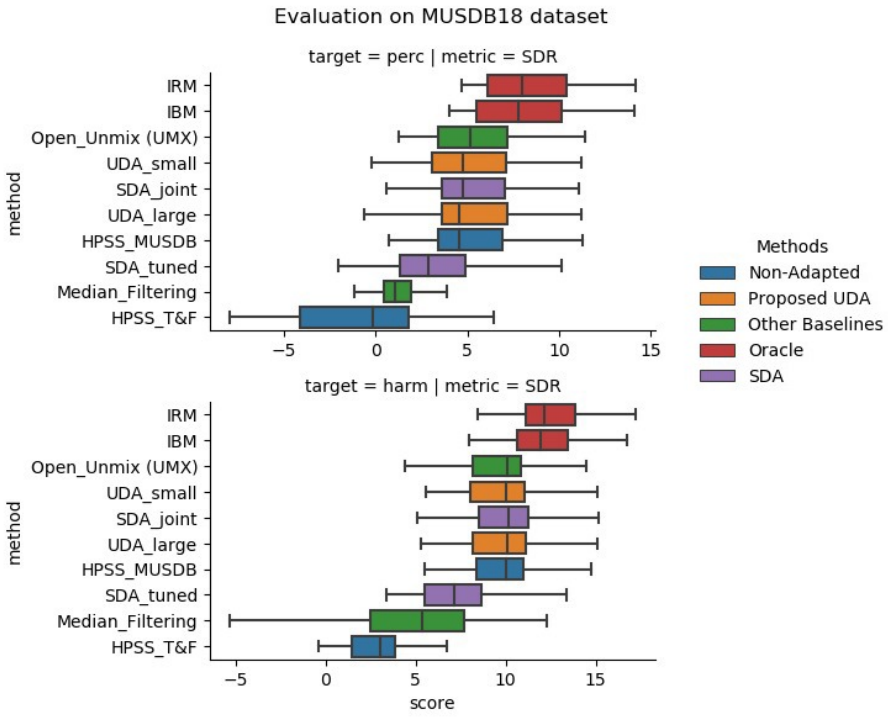


Figure 3.14: Boxplots showing SDR distribution of different methods on the MUSDB18 Dataset. Methods are ordered by median SDR value. For detailed information of each method see Table 3.2.

data from domain $\mathcal{B}$ and although no labels (ground-truth source signals) are available, this data can also be used by the proposed UDA method to adapt the model to perform the separation on data from the Tap & Fiddle. Therefore, two versions of the proposed approach are tested:

- HPSS_UDA_small: trained using ground-truth harmonic and percussive source signals from MUSDB18 along with the mixtures only from the train set of Tap & Fiddle for performing the adaptation to domain $\mathcal{B}$;
- HPSS_UDA_large: trained using the larger set of mixtures from the internal collection along with the ground-truth harmonic and percussive source signals from MUSDB18.

Results of the experiments evaluated on MUSDB18 (Source Domain $\mathcal{A}$) are shown in Table 3.2 and corresponding detailed boxplots for the SDR metric appear in Figure 3.14, while the results of the experiments on Tap & Fiddle (Target Domain $\mathcal{B}$) appear on Table 3.3 along with corresponding detailed boxplots for the SDR metric in Figure 3.15.

By inspecting the results, the first observation one can easily notice is that

Table 3.3: Objective evaluation of HPSS on Tap & Fiddle. The values are in dB and represent the median of metrics over tracks in the test set. HPSS_MUSDB represents the method trained in MUSDB18 and HPSS_T&F the method trained in Tap & Fiddle. SDA_joint and SDA_tuned are supervised domain adaptation models. The former is trained with joint data from both domains while the latter is first trained with data from MUSDB18 and later finetuned on Tap & Fiddle dataset. IBM is the Ideal Binary Masking and IRM represents the Ideal Ratio Masking oracle methods. Open_Unmix (UMX) is the baseline DNN proposed in Stöter et al. [2019] while Median_Filtering is the method in Fitzgerald [2010].

Method (Training Set)	Test Set: Tap & Fiddle (Domain $\mathcal{B}$)						Type of Data	
	Percussive			Harmonic				
	SDR	SIR	SAR	SDR	SIR	SAR	$\mathcal{A}$	$\mathcal{B}$
HPSS_MUSDB ($\mathcal{A}$)	1.3	15.8	0.3	22.0	23.0	29.7	labelled	—
HPSS_T&F ($\mathcal{B}$)	10.2	16.9	12.7	35.0	36.4	34.3	—	labelled
SDA_joint ($\mathcal{A}+\mathcal{B}$)	4.6	18.1	6.4	27.5	28.9	30.2	labelled	labelled
SDA_tuned ($\mathcal{A}\rightarrow\mathcal{B}$)	12.1	18.8	12.6	35.3	37.1	35.6	labelled	labelled
HPSS_UDA_small	3.4	13.0	2.9	25.0	25.9	30.8	labelled	unlabelled
HPSS_UDA_large	7.4	18.0	8.4	29.2	30.6	33.1	labelled	unlabelled
Open_Unmix (UMX)	6.7	7.0	5.1	28.6	36.8	25.9	labelled	—
Median_Filtering	1.5	4.8	1.7	21.8	30.1	22.2	labelled	—
IBM	13.5	20.8	13.7	37.8	41.3	37.7	—	—
IRM	13.4	19.5	13.8	37.2	42.0	37.2	—	—

the overall metrics for the harmonic source evaluation on Tap & Fiddle are much higher than the corresponding metric from the same methods when evaluated on MUSDB18 data. This is due to the fact that harmonic sources from Tap & Fiddle are only violin sounds, which should be easier to separate and obtain higher metrics across the board if compared to the harmonic source signals from MUSDB18, which contains a much higher genre, timbrel and instrument variability.

Furthermore, it is straightforward from analysing the results that models trained only with samples from one dataset had poor performance on the other, meaning that MUSDB18 and Tap & Fiddle have very different priors over the data. This fact is also reflected in the performance of Open_Unmix, which was trained in an augmented version of MUSDB18. Its performance is much lower on Tap & Fiddle if compared with the performance provided by methods that used labelled data strictly from Tap & Fiddle.

Moreover, as expected, the jointly trained model, SDA_joint, achieved relatively good performance overall because it uses supervised data from both domains. The SDA_tuned model, which started training with the same model weights as HPSS_ MUSDB model but was fine-tuned using Tap & Fiddle data,

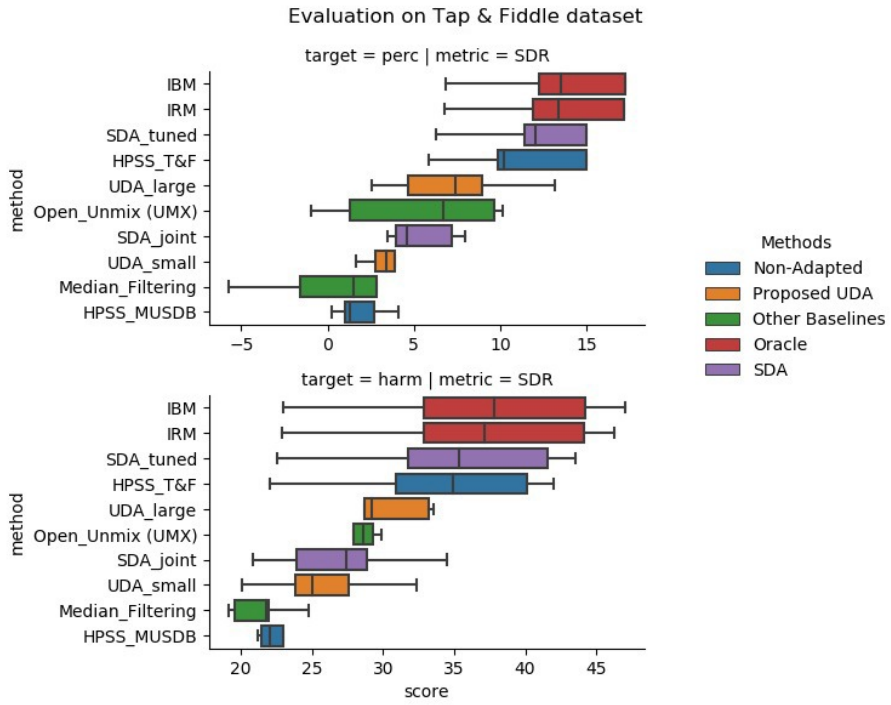


Figure 3.15: Boxplots showing SDR distribution of different methods on the Tap & Fiddle Dataset. Methods are ordered by median SDR value. For detailed information of each method see Table 3.3.

was indeed greatly improved when evaluated over this domain, but, as a trade-off, it lost a lot of its original performance on the original MUSDB18 dataset.

On the other hand, both versions of the proposed UDA approach got a boost in performance on all 3 of the metrics on Tap & Fiddle without losing any considerable performance on MUSDB18. This means that the proposed UDA approach can perform HPSS on both domains successfully, even though the labelled data used for training came only from domain $\mathcal{A}$. This shows that the proposed framework is a useful way of exploiting unlabelled data from the target domain to improve performance of HPSS.

The quantity of unlabelled data from domain $\mathcal{B}$ seems also to have impacted the performance of the proposed method. Even though the results of UDA_large are similar to UDA_small over the original source domain $\mathcal{A}$, the former performs much better over samples from the target domain $\mathcal{B}$ than the latter due to the fact that it uses more than double the amount of mixtures from this particular domain during training to perform domain adaptation.

Another interesting result is that UDA_large, which is a semi-supervised framework, had similar performance over MUSDB18, but much better over Tap

& Fiddle if compared to SDA_joint, which is a fully supervised method. This means that UDA using large amounts of unlabelled data can be much more promising than joint training using a smaller amount of labelled data.

More information about this work including audio samples can be found in the supplementary webpage[5].

3.4 Conclusions

In Section 3.2 a novel CNN architecture for music source separation named 3W-MDenseNet was proposed. It is an extension of the original MDenseNet architecture and its performance was investigated for the HPSS task. This novel musically motivated architecture network combines vertical, square and horizontal filters in its internal convolutional layers with the objective of facilitating the learning of time-frequency patterns associated with percussive and harmonic sounds. Differently than the MDenseNet, the proposed model estimates more than a single source simultaneously. Other than this, the 3W-MDenseNet also maintains the dense arrangement of skip-connections originally used in the MDenseNet in order to increase information flow in the network, avoid learning redundant feature-maps and, consequently, reduce the number of trainable parameters. As indicated by the experiments, it outperforms both the U-Net and the MDenseNet architectures when trained under comparable settings. The 3W-MDenseNet is able to maintain a high level of performance with a low number of parameters, compared to the majority of state-of-the-art DNN-based methods. For instance, Figure 3.16 shows a bar chart comparing how the proposed 3W-MDenseNet model and recently introduced methods for source separation perform when evaluated on drum extraction in MUSDB18. The number of parameters of each method is shown on the x axis.

Furthermore, no form of data augmentation has been used in the experiments of this section. By adding techniques of this nature and using larger datasets, it is expected that the model could achieve even higher performance. It is important to remember that the model has been trained only on 80 songs (20 songs were used for validation) and tested on 50. This experimentation is left as future work.

In Section 3.3 a framework for the domain adaptation problem under the source separation scenario was proposed. The experiments extended the previously proposed 3-MDenseNet to be able to not only perform the separation but to also generate domain-invariant encoded features. A domain-discriminator is included in the framework and the separator and discriminator are itera-

[5]http://c4dm.eecs.qmul.ac.uk/auda-hpss

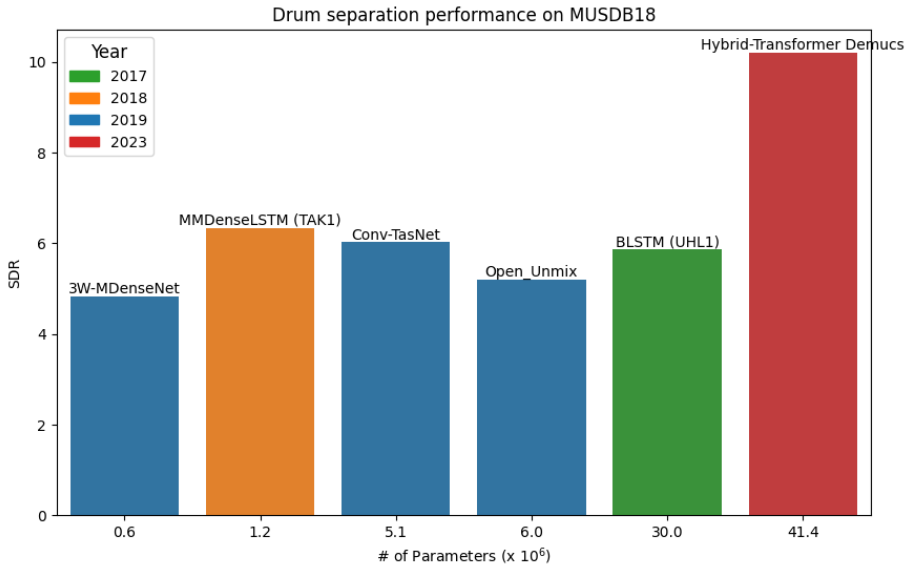


Figure 3.16: Median SDR on MUSDB18 test set for multiple methods that perform drum separation. The number of parameters of each method is shown in the x-axis. Apart from the 3W-MDenseNet, the other methods are: 'MM-DenseLSTM (TAK1)' [Takahashi et al., 2018b], 'Conv-TasNet' [Luo and Mesgarani, 2019], 'Open_Unmix' [Stöter et al., 2019], 'BLSTM (UHL1)' [Uhlich et al., 2017], 'Hybrid-Transformer Demucs' [Rouard et al., 2023].

tively and adversarially trained. The proposed framework is trained under semi-supervision exploiting unlabelled mixtures from a target domain in order to improve HPSS generalisation to new signals from this particular domain. The results showed that the this unsupervised domain adaptation approach is able to improve separation performance on a new domain without losing any considerable performance on the source domain.

Last but not least, the Tap & Fiddle Dataset, a dataset containing Scandinavian fiddle tunes with foot-tapping accompaniment was detailed (Subsection 3.3.4). Its corpus consists of not only the final mixes containing the two sources but also the original fiddle (violin) stem and foot-tapping stem. It is expected that this new dataset can not only be used for source separation but can also bring contributions to analysis of fiddle music and metrical expression in music. It is currently available to the community for research purposes in Lordelo et al. [2020a]

Chapter 4

Musically Motivated CNNs for Instrument Recognition

4.1 Introduction

This chapter provides a detailed description of the research, along with experiments and novel contributions, with respect to the topic of instrument recognition. Following the same strategy used when building the DNNs for music source separation in Chapter 3, all the models proposed in this chapter are also focused on efficiency (low number of trainable parameters) via the application of domain knowledge to build musically motivated CNN architectures. Hence, similar architectures are herein proposed and adapted for this new task of classification rather than separation.

The research can be divided into two related but distinct tasks. In Section 4.2, the task of Instrument Activity Detection (IAD), where instrument activations are estimated on a frame-level basis, is studied. Special interest is given in the fact that transient and stationary parts of sounds affect our perception of timbre in different ways and a method that first decomposes a music spectrogram into its constituent transient, stationary and noisy-like sounds is proposed. In Section 4.3 the pitch streaming (also known as instrument assignment) task is addressed as note-level instrument classification and a method utilising an auxiliary input carrying the pitch, onset and offset information of note-events alongside the time-frequency representation of the audio signal is proposed. The initial findings of the experiments in Section 4.2 were presented in Lordelo et al. [2020b] while most of the research done in Section 4.3 was published in Lordelo et al. [2021a]. Lastly, Section 4.4 concludes the chapter by discussing our findings and proposals as well as suggesting future potential

work.

The contributions of this chapter can be summarised as follows:

- Instrument Activity Detection: Proposal of a novel DNN for frame-level instrument recognition, i.e., each frame of a music spectrogram is associated to one or multiple instrument classes.
- Transient, Steady-State and Noisy-like sound features: Utilisation of a Transient-Stationary-Noise Decomposition (TSND) method [Driedger et al., 2014a] as pre-processing step in order to extract features related to transient, steady-state and residual sounds.
- Pitch Streaming: Proposal of a novel DNN that associates each note from a music signal to its instrumental source.
- Modular Framework for Pitch Streaming: Pitch streaming approach works with any MPE method. The streaming performance is evaluated when using ground-truth note labels as well as 2 state-of-the-art MPE algorithms [Thomé and Ahlbäck, 2017; Wu et al., 2019] to estimate note-events (pitch, onset and offset).
- Input Representation: Comparison of different representations for the input audio and auxiliary information for the task of pitch streaming.
- Musically Motivated CNN Architectures: Proposal of CNN architectures for frame-level instrument activity detection and for pitch streaming tasks that use musically motivated kernel shapes for the convolutions, facilitating learning representations for different instruments and note sound states. It is shown that their use improves the performance in both tasks.

4.2 Instrument Activity Detection

In this section, the task of instrument activity detection, where multi-label instrument classes are predicted for every frame of an input audio spectrogram, is addressed. Such problem is addressed as a frame-level instrument classification and a novel classifier architecture is proposed following the same methodology utilised in Chapter 3: the utilisation of domain knowledge to build more efficient musically motivated CNN architectures.

Studies on music cognition [McAdams et al., 1995; Clark et al., 1964] show that information regarding the production mode of the sound, i.e., if the sound is generated by bowing, plucking or hammering of a string, or by impelling air into a wind instrument, is essentially located at the beginning and at the end of the notes (transient parts). Some features taken from the attack transients

of sound, such as onset duration and crest factor, have also proved to be helpful for instrument family identification on isolated notes. On the other hand, the relation between the energy of the harmonics (stationary part) also has a fundamental impact on our perception of timbre.

For instance, Essid et al. [2005] performed experiments evaluating the usefulness of a combination of transient-based features and steady-state features for instrument identification. The authors showed that an SVM classifier achieves better performance in solo instrument recognition when using only attack-transient features compared to only steady-state features. However, this is valid only when the decision window for classification is short (48 ms and 80 ms). When larger decision windows (496 ms and 1936 ms) were used, the authors verified that methods not considering the distinction between transients and steady-state features performed better, indicating that musical context also has a central role in automatic recognition of instruments. They concluded their study proposing that a technique mixing both transient and steady-state segments and specialised classifiers would be the ideal approach to achieve a better overall performance.

Motivated by those works and the fact that one of the fundamental ideas of the research performed in the thesis is the exploitation of harmonic and percussive filters and features (see also Chapter 3 for experiments related to harmonic-percussive source separation), the experiments in this section also investigate whether the task of instrument classification can be improved by explicitly providing both types of sound features to the framework. The final proposal is to split a music spectrogram into a *transient-enhanced spectrogram*, a *steady-state-enhanced spectrogram* and a *noisy-like residual spectrogram*, which are estimated by a Transient-Stationary-Noise Decomposition (TSND) method [Driedger et al., 2014a]. Those 3 spectrograms are jointly used as inputs to the DNN classifier.

A preliminary version of the work presented in this section was presented in Lordelo et al. [2020b].

4.2.1 Proposed Method

The proposed approach is similar to other data-driven instrument recognition methods in the sense that instrument activity detection is addressed as a multi-label classification task, in which the objective is to pinpoint all the instruments that are active in each frame across time, and a DNN is used as the classifier [Hung and Yang, 2018; Gururani et al., 2018; Hung et al., 2019]. However, the proposed method is primarily different from previous work in the following ways:

1. Instead of utilising raw audio spectrograms as inputs to the neural network, transient and steady-state sound information is explicitly provided as different inputs to the framework;

2. The classifier architecture is musically motivated, including vertical, square and horizontal kernel shapes to help learning harmonic and percussive features directly from the time-frequency representation, which helps in recognising instruments more efficiently;

3. The classifier also contains a dense arrangement of skip-connections in order to avoid learning redundant feature-maps, reducing the number of trainable parameters;

4. Instead of performing the classification for every frame of the input at once, only the central frame of the input spectrogram is classified, and musical contexts varying from 50 ms to 800 ms around the central frame are evaluated.

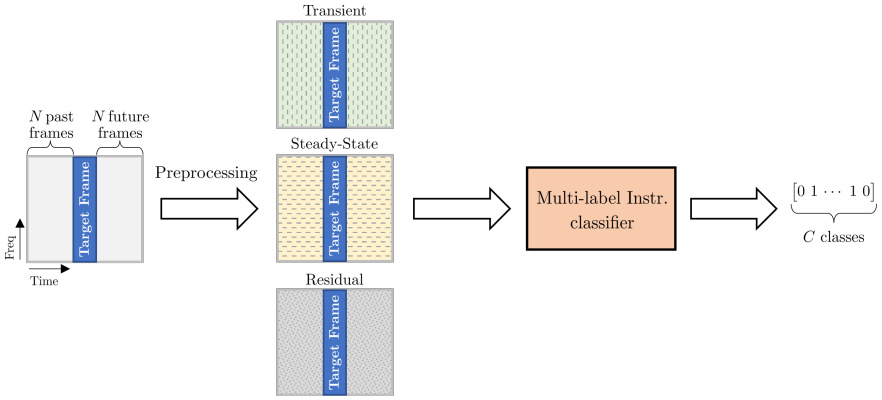


Figure 4.1: Overview of the proposed framework for IAD. During the preprocessing stage, transient-enhanced, steady-state-enhanced and residual spectrograms are estimated and are later used as inputs to a classifier. The different colours represent a different type of sound-enhanced spectrogram

Figure 4.1 depicts an overview of the proposed frame-level instrument recognition approach. The main motivation of the method is to facilitate the learning of transient-related and steady-state-related features by explicitly providing separated sources to the classifier as different input channels. More specifically, transient-enhanced, steady-state-enhanced and residual magnitude spectrograms are estimated from the original spectrogram of the music signal during a preprocessing step and later used as joint inputs to a deep-learning-based multi-label instrument classifier.

Furthermore, since musical context is also a fundamental aspect for discriminating the timbre of sounds, instead of classifying the information contained in just a single frame of the spectrogram, a context of N future frames and N past frames is used as a **decision window** for the classifier. In other words, time-frequency representations of shape $(F \times (2N+1))$ are used as inputs to the classifier, where F is the number of frequency bins and $2N+1$ is the number of frames per input (decision window). The classifier is trained to recognise the active instruments in the $(N+1)$-th frame, i.e., the instrument activation labels associated with the middle frame of the decision window is used as ground-truth targets for the classification task. In the experiments, values of N varying from 2 to 40 frames, which are equivalent of using decision windows from 50 ms to 810 ms respectively, were tested and the value of $N = 35$ obtained the best overall performance. See Subsection 4.2.5.1 for details.

4.2.1.1 Preprocessing Step

The main preprocessing step of the proposed framework is the application of a method to estimate transient-sound-enhanced and stationary-sound-enhanced spectrograms.

Even though in Chapter 3 data-driven methods for performing HPSS were proposed, it is important to note that they are not considered the best approach to be used in this case. The HPSS methods estimate a source with all unpitched-percussive instrument sounds and another with all pitched-harmonic instrumental (or vocal) sounds, but both of those sources yield sounds containing transient and stationary parts. A good example of this is that the attack part of sounds generated by harmonic instruments is essentially transient, but it is still associated with the estimated harmonic source. The Transient-Stationary Separation (TSS) is not performed by HPSS. The reader is referred to Subsection 2.2.2 where differences between harmonic-percussive source separation and transient-stationary separation tasks are discussed.

Consequently, the primary choice for the preprocessing step is a different method taken directly from the literature that essentially performs Transient-Stationary-Noise Decomposition (TSND) [Driedger et al., 2014a]. This method is an extension of the TSS method of Fitzgerald [2010], where a third residual component is added to the separation, which captures sounds that lie in between the clearly harmonic and percussive sounds of the music signal. This type of decomposition of music signals is inspired by the Sines + Transients + Noise (STN) audio model [Levine and Smith III, 1998; Petrovski et al., 2011], whose traditional application lies in low bitrate audio coding. In those works, the goal is to represent a given input audio signal in terms of a parameterised set of sine

waves, transient sounds, and noise that can be processed and later resynthesised back.

Note that instead of denoting it a "*Separation*" method, the term "*Decomposition*" is chosen here instead. The reason for this is the fact that the preprocessing stage is performing a **decomposition** of a single magnitude spectrogram into 3 disjoint magnitude spectrograms. The resulting signals are still time-frequency representations, i.e., there is no waveform estimation via an application of inverse-STFT and the phase is ignored. Therefore, the method is not performing source separation, strictly speaking.

In summary, the main preprocessing step is the application of a TSND method [Driedger et al., 2014a] for estimating 3 time-frequency representations for an input music signal:

- Transient-Enhanced Representation: the representation of stationary parts of the sounds should be suppressed while the representation of the transient parts of the sounds should be salient;
- Stationary-Enhanced Representation: the representation of transient parts of the sounds should be suppressed while the representation of the stationary parts of the sounds should be salient;
- Residual (Noise) Representation: the representation of parts of the sounds that are neither transient nor stationary.

The main idea of this method [Driedger et al., 2014a] is to perform the decomposition by exploiting the anisotropic smoothness of music spectrograms, which is based on the fact that the representation of transient-percussive sounds tend to form straight vertical lines while stationary-harmonic (steady-state) sounds are represented as horizontal structures, considering the vertical axis associated to frequency and horizontal axis to time. For more details, the reader is referred to Subsection 2.1.8, where a brief discussion about this phenomenon is exposed.

Hence transient-percussive parts of sounds can be removed from the spectrogram by a process which emphasises horizontal lines and suppresses vertical ones. On the other hand, a process that emphasises the vertical lines while suppressing the horizontal ones should result in a spectrogram which has most of the pitched-stationary sounds removed.

Such processes can be achieved using a combination of two median filters. Median filters operate by replacing the central sample of a windowed signal by the median value of the window. Mathematically, given a median filter of an odd length $L_{\mathcal{M}}$, the output $y[n]$ of the application of a median filter to a digital

input signal $x[n]$ can be defined as

$$y[n] = \mathcal{M}\left(\{x[n-k], x[n-k+1], \cdots, x[n+k-1], x[n+k]\}\right), \tag{4.1}$$

where $\mathcal{M}$ represents the median operation and $k = \frac{L_{\mathcal{M}}-1}{2}$. In effect, by performing a horizontal median operation across a fixed frequency bin of the spectrogram, fast variations with respect to time, i.e., transient-percussive events will be smoothed out, resulting in a harmonic or stationary-enhanced spectrogram. Similarly, by applying a vertical median filtering operation across a fixed frame of the spectrogram, pitched-stationary sounds will be considered outliers and will be removed from the spectrogram, generating a transient-enhanced spectrogram.

The whole method to perform the decomposition can be defined as follows. Consider an input magnitude spectrogram $X[f, n]$. Initial transient-enhanced $X'_{\mathrm{T}}[f, n]$ and stationary-enhanced $X'_{\mathrm{S}}[f, n]$ decompositions can be computed using a median filter of odd length L_{T} applied vertically (frequency-wise) and a median filter of odd length L_{S} applied horizontally (time-wise) as

$$X'_{\mathrm{T}}[f, n] = \mathcal{M}\left(\{X[f - k_{\mathrm{T}},\, n], \cdots, X[f + k_{\mathrm{T}},\, n]\}\right), \quad k_{\mathrm{T}} = \frac{L_{\mathrm{T}} - 1}{2}, \tag{4.2}$$

$$X'_{\mathrm{S}}[f, n] = \mathcal{M}\left(\{X[f,\, n - k_{\mathrm{S}}], \cdots, X[f,\, n + k_{\mathrm{S}}]\}\right), \quad k_{\mathrm{S}} = \frac{L_{\mathrm{S}} - 1}{2}. \tag{4.3}$$

As pointed out in previous work [Driedger et al., 2014a; Fitzgerald, 2010], the resulting decomposition is not deeply affected by the values of L_{T} and L_{S} as long as they are not extreme. In the experiments the values suggested by Driedger et al. [2014a] were used, which are $L_{\mathrm{T}} = 21$ frequency bins (equivalent to approximately 475 Hz) and $L_{\mathrm{S}} = 21$ frames (equivalent to 210 ms).

Since we are interested in decomposing the original spectrogram, two binary masks are then computed by comparing the values of the initial estimations $X'_{\mathrm{T}}[f, n]$ and $X'_{\mathrm{S}}[f, n]$. A transient-sound mask $M_{\mathrm{T}}[f, n]$ and a stationary-sound mask $M_{\mathrm{S}}[f, n]$ according to

$$M_{\mathrm{T}}[f, n] = \begin{cases} 1, & \text{if} \quad \dfrac{X'_{\mathrm{T}}[f, n]}{X'_{\mathrm{S}}[f, n]} > \beta, \\ 0, & \text{if} \quad \dfrac{X'_{\mathrm{T}}[f, n]}{X'_{\mathrm{S}}[f, n]} \leq \beta; \end{cases} \tag{4.4}$$

$$M_{\mathrm{S}}[f, n] = \begin{cases} 1, & \text{if} \quad \dfrac{X'_{\mathrm{S}}[f, n]}{X'_{\mathrm{T}}[f, n]} > \beta, \\ 0, & \text{if} \quad \dfrac{X'_{\mathrm{S}}[f, n]}{X'_{\mathrm{T}}[f, n]} \leq \beta, \end{cases} \tag{4.5}$$

where $\beta \in \mathbb{R}, \beta \geq 1$ is a separation factor. Intuitively, we can think that for a time-frequency bin to be included in the transient component, it is required that its value in $X'_{\mathrm{T}}[f, n]$ stands out from the stationary portion $X'_{\mathrm{S}}[f, n]$ by at least a factor of β and vice versa. In the experiments, we tested multiple values of β varying from 1.0 to 3.5. See Subsection 4.2.5.3 for details.

When $\beta = 1$ the method reduces to the transient-stationary separation method proposed by Fitzgerald [2010], but a higher value of β allows us to control how strict the decomposition should be when deciding if sounds should be considered transient or stationary. Note that both masks are disjoint, i.e., it is not possible to have both $M_{\mathrm{T}}[f, n] = M_{\mathrm{S}}[f, n] = 1$ given that $\beta \geq 1$. Therefore, in the cases where both masks are zero, the sounds can be considered as part of a residual component (noise) of the decomposition, whose corresponding binary mask $M_{\mathrm{N}}[f, n]$ can be computed as

$$M_{\mathrm{N}}[f, n] = 1 - M_{\mathrm{T}}[f, n] - M_{\mathrm{S}}[f, n] \tag{4.6}$$

In the end, the three masks are applied to the original magnitude spectrogram in order to obtain the final decomposition

$$X_{\mathrm{T}}[f, n] = M_{\mathrm{T}}[f, n] \odot X[f, n]; \tag{4.7}$$

$$X_{\mathrm{S}}[f, n] = M_{\mathrm{S}}[f, n] \odot X[f, n]; \tag{4.8}$$

$$X_{\mathrm{N}}[f, n] = M_{\mathrm{N}}[f, n] \odot X[f, n]; \tag{4.9}$$

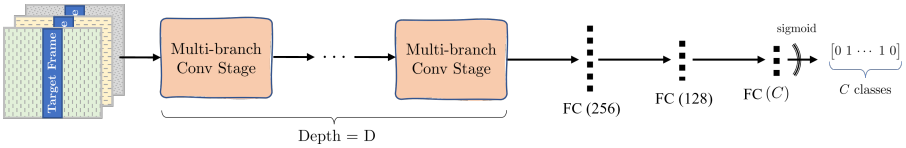
where $\odot$ represents point-wise multiplication.

The estimated spectrograms $X_{\mathrm{T}}[f, n]$, $X_{\mathrm{S}}[f, n]$ and $X_{\mathrm{N}}[f, n]$ are then concatenated in the channel dimension and used as a multi-channel input for the classifier.

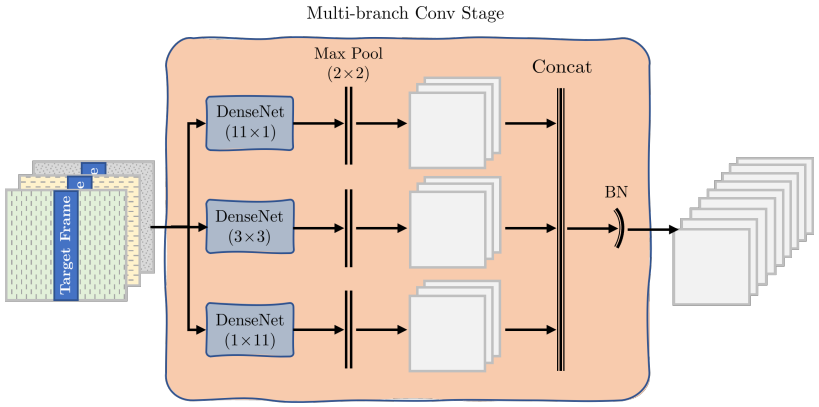
4.2.1.2 Proposed Musically Motivated Classifier Architecture

Briefly speaking, the classifier architecture is a variation of the 3W-MDenseNet, which is another contribution of this thesis (see Subsection 3.2.6 for details), but adapted to perform classification rather than source separation. The main idea behind the methodology is kept: proposal of a musically motivated CNN, whose convolution kernels are a combination of vertical, square and horizontal filters, and utilisation of DenseNets [Huang et al., 2017] instead of regular stacks of convolutions in order to avoid learning redundant feature-maps and have more representative capacity with fewer number of trainable parameters. The proposed architecture is depicted in Figure 4.2. It consists of a sequence of D multi-branch convolutional stages and three fully connected layers. When using $D = 5$ and $N = 35$, the number of trainable parameters of the proposed network

is approximately 2 million.



(a) Overview of the proposed instrument classifier. The transient-enhanced, stationary-enhanced and noise spectrograms are concatenated as a single multi-channel input. The model consists of D multi-branch convolutional stages and 2 fully-connected hidden layers before a final fully connected layer classifies them into one or more of C instrument classes.



(b) Internal structure of a multi-branch convolutional stage. Observe that 3 DenseNets with unique filter shapes run in parallel but their final feature-maps are concatenated at every stage.

Figure 4.2: Proposed architecture for instrument activity detection using TSND as preprocessing stage. The different colours of the input spectrograms represent different types of sounds: transient-enhanced, stationary-enhanced and residual spectrograms.

In the original 3W-MDenseNet, three encoder-decoders run in parallel in separate branches, each with a unique kernel shape: vertical, square or horizontal (see Figure 3.7 for details). The feature-maps generated by each encoder-decoder are later concatenated at a final layer. In the instrument classification task, a similar methodology is adopted by taking only the encoder layers from the 3W-MDenseNet and complementing with fully connected layers at the end in order to perform classification rather than source separation. In addition, a few modifications to the original encoder layers, which here are called a *multi-branch convolutional stage* are also proposed.

The main difference from the regular 3W-MDenseNet encoder layers is that instead of only concatenating the feature-maps computed using different choices of kernel shapes (vertical, square and horizontal) at a later layer in the network,

the encoder (multi-branch convolutional stage) now concatenates the feature-maps of each branch at every stage, right after each down-sampling stage, which is still done using a (2×2) max-pooling layer. By doing so, the feature-maps of every convolutional layer are given access to feature-maps computed using all different choices of kernel shapes from a previous stage.

In Figure 4.2b the internal structure of a multi-branch convolutional stage is shown. Internally, each multi-branch convolutional stage contains 3 separate branches whose convolutions have unique kernel shapes. A branch with horizontal (1×11), a branch with square (3×3), and a branch with vertical (11×1) convolutions are used. In each path, a Densely connected convolutional Network (DenseNet) [Huang et al., 2017] with growth rate $k = 24$ and number of layers $L = 4$ is used. In short, a DenseNet is a stack of L k-channel convolutional layers — each with its own activation function — with a dense pattern of skip-connections, where each layer receives the concatenation of all previous layers' outputs as input. The reader is referred to Subsection 3.2.3 for the detailed internal structure of a DenseNet. After the DenseNet, a (2×2) max-pooling layer is applied in order to down-sample the feature-maps by reducing their dimensions and increase the receptive field at each branch. Afterwards, the three branches are concatenated and the batch is normalised. The final feature-maps are used as input for the next multi-branch convolutional stage. Since we need to concatenate feature-maps that are originated by multiple kernel shapes, padding is necessary to be applied on the convolutions and on the max-pooling layers to ensure the feature-maps maintain the same dimensions across branches.

The ReLU function is used as activation for all layers apart from the last, for which a sigmoid function activation is used (final activation in the interval of $[0, 1]$) in order to perform multi-label classification among C classes. The Binary Cross-Entropy (BCE) loss is used to train the framework.

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{B}\frac{1}{C}\sum_{i=1}^{B}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij}) + (1 - y_{ij})\log(1 - \hat{y}_{ij}), \tag{4.10}$$

where B represents the number of samples in a training batch, the value of $j \in \{1, \cdots, C\}$ represents one of the C different instrument classes, y_{ij} is the ground-truth label with respect to sample i and instrument j, i.e., y_{ij} is 1 if input i has instrument j active or 0 otherwise. Similarly, $\hat{y}_{ij}$ is the prediction of the model with respect to sample i and instrument j (after the application of the final sigmoid activation function).

Furthermore, threshold values V_j, with $j \in \{1, \cdots, C\}$ are used in order to consider the j-th instrument class as positive (instrument is active) or nega-

tive (instrument is not active). In other words, if the final activation of the j-th instrument class is greater than or equal to V_j, that instrument is considered active, otherwise, it is inactive on the target frame. The values used for V_j can vary across the instrument classes and are learned after the model is trained using a validation set. See Subsection 4.2.4 for more details on how the experiments are set up.

4.2.2 Dataset

Despite existing large-scale publicly available datasets, such as IRMAS [Bosch et al., 2012] or OpenMIC-2018 [Humphrey et al., 2018], that could potentially be used for the general instrument recognition task, they are often weakly labelled, meaning that they contain only annotations for the instrument on a clip-level basis. There is no information regarding the time stamps when each instrument is active. In order to be able to train the proposed method in a supervised manner, frame-level instrument annotations are necessary.

Datasets that contain such type of annotations are called strongly labelled datasets and they include Bach10 [Duan et al., 2010], TRIOS [Fritsch and Plumbley, 2013], Slakh2100 [Manilow et al., 2019] and MusicNet [Thickstun et al., 2017]. Bach10 and TRIOS are small scale datasets with only 10 and 5 recordings respectively. On the other hand, Slakh2100 is a large-scale dataset containing 2100 recordings, but it consists of only synthesised data and not real-world recordings. Since the goal of the research is mainly work with real world data, avoiding the analysis of artificial sounds, the MusicNet dataset [Thickstun et al., 2017] was chosen as the ideal dataset for the experiments of this chapter.

The MusicNet dataset [Thickstun et al., 2017] is the largest publicly available dataset with non-synthesised data that is strongly labelled for the task of instrument recognition. This means that we know the exact frames where the instruments are active in the signal, which permits the training of supervised models to perform instrument recognition at the frame-level, note-level, and clip-level. The dataset contains 330 freely-licensed classical music recordings by 10 composers, written for 11 instruments, along with over 1 million annotated labels indicating the onset, offset and pitch of each note in the recordings and the instruments that play them.

The instrument taxonomy included in MusicNet is: *piano, violin, viola, cello, french horn, bassoon, clarinet, harpsichord, bass, oboe* and *flute*. However, the last 4 instruments (harpsichord, bass, oboe and flute) do not appear in the original test set provided by the authors. Therefore, in all the experiments using this dataset the labels related to those instruments were ignored and a 7-class instrument classification using the following classes: **piano, violin,**

viola, cello, french horn, bassoon and **clarinet** was performed. It is worth noting that even though the sounds of those 4 non-classified instruments are not recognised, they are not removed from the training data. The reason for this is not only to have as many recordings as possible for training, but also to allow the model to be more robust and recognise the target instruments in the potential presence of extra unwanted sounds.

4.2.3 Evaluation Metrics

The classification performance is evaluated by computing the frame-level F-score (F_s), which is directly related to the precision (P) and recall (R) according to:

$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}},\ R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}},\ F_\mathrm{s} = \frac{2PR}{P + R}, \tag{4.11}$$

where TP is the number of true positives, FP the false positives and FN the false negatives.

Given a music track of the test set of MusicNet, an STFT magnitude spectogram with a time resolution of 10 ms is computed and each frame is classified. The numbers of true positives, true negatives, false positives and false negatives are computed on a per-class (per-instrument) basis and then those values are aggregated across all the tracks in the test set.

In some tables in this section, the per-class F-scores are directly provided, however, in order to obtain a single metric value that represents the overall performance of an evaluated method, the per-instrument F-score values can be averaged. In the literature, there are 3 most common ways of computing an average value of F-score and all of them are used here.

Macro-Averaged F-score

The *macro-averaged F-score* (or just *macro F-score*) is computed using the arithmetic mean of all the per-class F-scores. This method treats all classes equally regardless of the number of occurrences of each class in the dataset.

This is probably the most straightforward way of averaging the methods and is also the most common metric used in the literature.

Weighted-Averaged F-score

Instead of taking the arithmetic mean of all the per-class F-scores, the *weighted-averaged F-score* (or just *weighted F-score*) is calculated by taking a weighted average. The weight of a particular class is set to the number of examples of that class.

Differently than the macro F-score, the weighted F-score takes into account the proportion of data in each class and is a "fairer" metric when working with imbalanced datasets.

Micro-Averaged F-score

The *micro-averaged F-score* (or just *micro F-score*) is a global computation of an overall F-score by counting the true positives, false negatives, and false positives across all the different classes and then using Equation (4.11).

Essentially, it is computing the proportion of correctly classified observations out of all observations. This definition matches the overall accuracy definition in cases when we are working with a single-label method, i.e., when it is guaranteed that each test example should be assigned to exactly one class. In the general multi-label classification case the two metrics are not equivalent.

4.2.4 Experiment Setup

In all experiments the original train/test split provided in the original MusicNet dataset is used. The sampling frequency of the audio input is set to the original 44100 Hz and an STFT using Blackman-Harris windows of 1024 samples (23 ms) with a hop size of 441 samples (10 ms) is computed. As mentioned in Subsection 4.2.1, I set a decision window of N past frames and N future frames of the target frame to be classified. This means that each input is a magnitude spectrogram with shape $(F \times (2N + 1))$, where $F = 513$ and $2N + 1 = 71$.

Since MusicNet provides ground-truth instrument-activation labels for every audio sample in the resolution of 44100 Hz, the target associated with any input can be computed by taking the instrument labels associated to the middle audio sample of the middle $(N + 1) = 36$-th frame of the input. Mathematically, this means that if the first frame of the i-th input has $x[i]$ as its middle sample, the target instrument for that particular input is the label associated with the sample $x[441 \cdot N + i] = x[441 \cdot 35 + i]$. The labels associated to the N first (and last) frames of every track in the train set were ignored because the associated input to the model would require padding. However, during evaluation on test set those frames were not ignored.

It is important to mention that a lot of different inputs can be generated by treating each frame of the STFT of a music recording as a potential central frame to be classified. Therefore, if the STFT of a single music recording has a total of T frames, $T - 2N$ different inputs can be generated by moving a $2N + 1$ decision window accross the original spectrogram. For instance, despite having only 10 recordings in the MusicNet test set, the experiments are evaluated in a total of effectively different 73259 input samples.

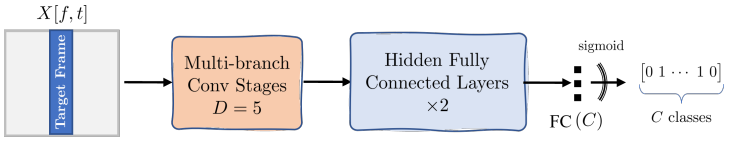
From the training set 15% of inputs and corresponding labels of each class are picked and put into a validation set. The models were trained using the Adam optimiser with an initial learning rate of 0.001, which was reduced by a factor of 0.2 if the binary cross-entropy loss stopped improving for 2 consecutive epochs on the validation set. If no improvement happens after 5 consecutive epochs, the training was stopped early. The experiments were performed using the Tensorflow/Keras Python package.

As shown in Figure 4.3, three model variations are implemented and compared:

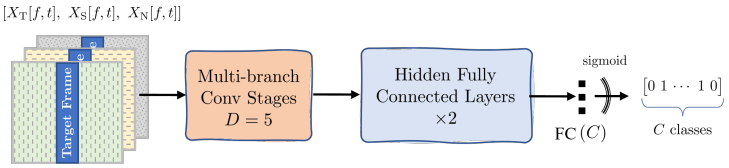
- Baseline Model: There is no TSND preprocessing stage and the original spectrogram of the music clip is used as input to the model. Block diagram is depicted in Figure 4.3a.
- Multi-Channel Model: TSND is performed and the three resulting spectrograms are concatenated, yielding a multi-channel spectrogram that is used as input to the model. Block diagram is depicted in in Figure 4.3b.
- Multi-Input Model: TSND is performed and three single-channel input heads, each with a type of estimated spectrogram, are used in the model. Block diagram is depicted in in Figure 4.3c.

In order to keep the number of trainable parameters of each type of model close to 2 million, the growth rate of every the DenseNets in each multi-branch convolutional stage is set to 24 for the baseline and the TSND-multi-channel models, and to 12 for the TSND-multi-input model. In addition, the experiments involving models with only the set of square (3×3) filters, had their DenseNet's growth rate increased to 62 at each multi-branch convolutional stage.

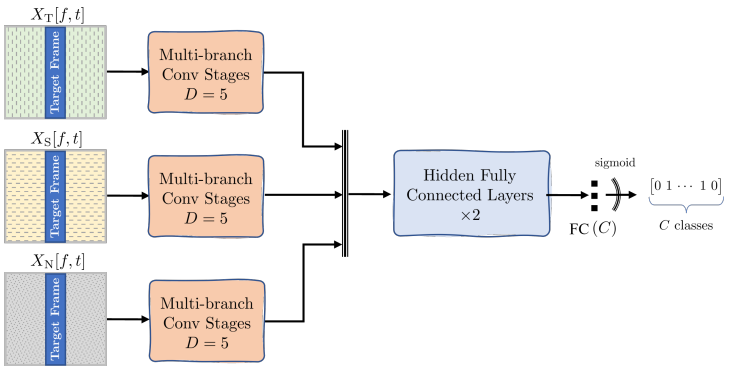
Regarding the threshold values V_j, with $j \in \{1 \cdots C\}$, that are used to verify if j-th instrument class should be considered active or inactive, for every experiment in this section, an exhaustive search is done using data from the validation set and the threshold values that obtain the best performance are later used for evaluating the models on the test set. More specifically, the per-instrument F-score is computed on the validation set using values of V_j ranging from 0.10 to 0.95 with a step size of 0.05 and the values that obtain the best per-class metric are then picked for performing evaluation on the test set.



(a) Baseline: This model has no TSND preprocessing step. Its input is generated directly taking a decision window of N future frames and N past frames from the original magnitude spectrogram of the recording.



(b) Multi-channel: This model uses TSND as preprocessing step and the transient-enhanced, stationary-enhanced and residual spectrograms are concatenated and used as a single 3-channel input for the model.



(c) Multi-Input: This model uses TSND as preprocessing step but the transient-enhanced, stationary-enhanced and residual spectrograms are used as unique input heads for the model.

Figure 4.3: Different models evaluated in the experiments. Different input colours represent different types of estimated sources: transient, stationary and residual.

4.2.5 Results

The results of experiments related to frame-level instrument classification using the previously explained methods are provided in this Subsection. It is started by showing preliminary studies performed using the baseline model only with the objective of finding the best choice of length for the decision window. Then experiments to verify whether the utilisation of multiple kernel shapes in the architecture is beneficial for the task is shown, comparing the proposed architecture with other DNNs that utilise higher number of trainable parameters.

Afterwards, experiments using TSND as preprocessing step are analysed and compared with baselines and an exhaustive test to verify how the separation factor β impacts the performance is discussed.

4.2.5.1 Length of Decision Window

This initial set of experiments was implemented in order to determine the best decision window to be used under this classification scenario. Different decision windows varying from 50 ms to 810 ms were evaluated and the results are shown on Figure 4.4. Since the STFT has a step size of exactly 441 samples, the frame-wise resolution is 10 ms and consequently, the values shown in the legend can be mapped directly to $2N + 1$ by dividing by 10.

Furthermore, note that for models with lower dimension, slightly shallower versions of the model using $D = 4$ (when $N = 15$ or $N = 10$) and $D = 3$ (when $N = 5$) had to be used instead of the regular $D = 5$ of other methods. This is due to the fact that the length of the input $(2N + 1)$ has to be at least 2^D to take into account the D stages of down-sampling by 2 via max-pooling.

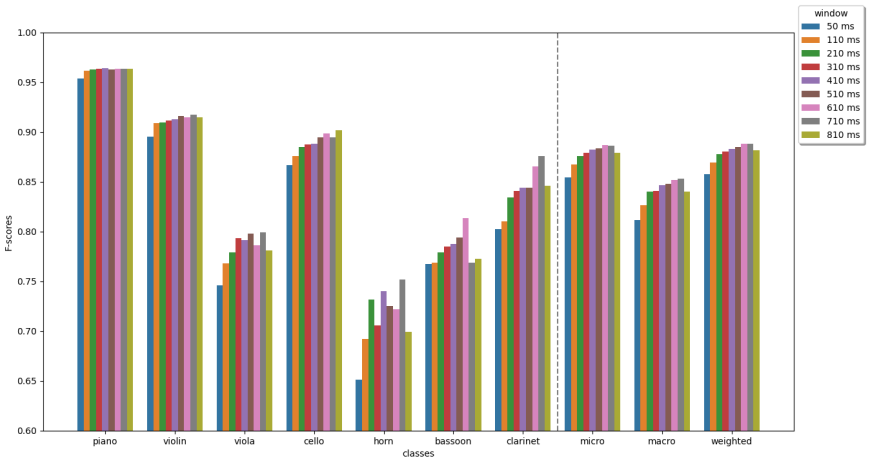


Figure 4.4: Comparison of the performance of the baseline approach (without TSND) using different duration of the input's musical context (length of decision window).

Analysing Figure 4.4, it is possible to verify that the length of decision window has more impact on the detection of woodwind instruments and horn than string instruments. It is believed that this is due to the fact that, in general, not only the average duration of the notes played by string instruments is shorter than the average duration of the notes played by woodwind and brass instruments in musical pieces, but also, their attacks are more prominent and, consequently, most of the information required to recognise string-based instru-

ment sounds is in the frequency envelope of the note attacks [McAdams, 2013; Essid et al., 2004]. Analysing shorter or longer musical contexts does not bring much benefit.

However, analysing longer windows increases the bassoon F-score from 0.7674, at 50 ms, to 0.8134 at 610 ms and can bring improvements of 7.5% in the clarinet F-score and 10.5% for horn F-score. The overall performance of the model using a context window of 710 ms (equivalent to an input length of $2N + 1 = 71$ frames) obtained the best macro F-score of 0.8529. Therefore, the value of $N = 35$ was kept for the subsequent experiments.

4.2.5.2 Effects of Kernel Shapes and Skip-Connections

This experiment has the purpose of validating the hypothesis that the addition of vertical and horizontal kernel shapes to the convolutional layers of the network is beneficial for the instrument recognition task. There is also interest in verifying whether or not using the dense arrangement of skip-connections in DenseNets can also avoid learning of redundant feature-maps and obtain high performance with a reduced number of trainable parameters.

In this case the baseline model (without TSND) is still used because it is faster to train and evaluate due to the absence of the preprocessing stage and because the effects of the application of TSND are not important to the current evaluation.

Two variations of model architecture were evaluated: an architecture using regular stacks of convolutional layers + max-pooling instead of DenseNets (no internal skip-connection) and an architecture using regular DenseNets of depth 4 at each convolutional stage (with dense arrangement of skip-connections). For each of the two types of architectures, a model using only square (3×3) kernel shapes is compared with a model that uses not only square, but also horizontal and vertical kernel shapes. Table 4.1 provides the choice of parameters for each model and the total number of trainable parameters in each case.

The results can be seen in Table 4.2, which shows the F-score metric for every instrument class and the micro, macro and weighted average metric. It is possible to note that the models that do not use DenseNets have double the number of trainable parameters, but perform similarly or, in some cases, worse than their corresponding model using DenseNets. This allow us to conclude that, indeed, the dense arrangement of skip-connections of the DenseNets facilitates the learning of more discriminative feature-maps and effectively helps in reducing the number of trainable parameters in the model.

Moreover, when comparing models using only square filters to models using a combination of square, horizontal and vertical, we see that the macro-averaged

Table 4.1: Choice of the parameters for the convolutional stages for the the different tested models. The models using multiple kernel shapes have 3 branches using (1×11), (3×3), (11×1) kernel shapes respectively. The values that appear with a $(3\times)$ prefix indicate that each of the 3 branches use that value for the respective parameter. The fully connected layers of all models are the same and follow Figure 4.2a

	Parameter	Without DenseNets		With DenseNets	
		Square	Multiple	Square	Multiple
Conv. Stage 1	Growth Rate k	—	—	60	$(3\times)$ 24
	# of Channels	64	$(3\times)$ 8	—	—
	# of Layers L	1	$(3\times)$ 1	4	$(3\times)$ 4
Conv. Stage 2	Growth Rate k	—	—	60	$(3\times)$ 24
	# of Channels	128	$(3\times)$ 16	—	—
	# of Layers L	1	$(3\times)$ 1	4	$(3\times)$ 4
Conv. Stage 3	Growth Rate k	—	—	60	$(3\times)$ 24
	# of Channels	256	$(3\times)$ 32	—	—
	# of Layers L	1	$(3\times)$ 1	4	$(3\times)$ 4
Conv. Stage 4	Growth Rate k	—	—	60	$(3\times)$ 24
	# of Channels	512	$(3\times)$ 64	—	—
	# of Layers L	1	$(3\times)$ 1	4	$(3\times)$ 4
Conv. Stage 5	Growth Rate k	—	—	60	$(3\times)$ 24
	# of Channels	512	$(3\times)$ 128	—	—
	# of Layers L	1	$(3\times)$ 1	4	$(3\times)$ 4
# of trainable parameters		5.8 M	4.2 M	2.0 M	2.0 M

F-scores are between 2 to 3 percentage points lower for the former models. This confirms the initial claim that using domain knowledge to build musically motivated CNN architectures is also beneficial for the task of instrument recognition. The findings in this experiment conform to the findings of the thesis regarding HPSS discussed in Chapter 3.

4.2.5.3 Varying Separation Factor β

In this set of experiments the models that perform classification using TSND as preprocessing stage are evaluated and compared. While the lengths of the vertical and horizontal median filtering are fixed as mentioned in Subsection 4.2.1.1, different values of the separation factor β were tested and the results are summarised in Table 4.3. In the left half of the table, metrics for multi-channel models, whose architectures follow Figure 4.3b, are shown, while in the right half, the results of the evaluation of multi-input models following Figure 4.3c are shown.

Table 4.2: Evaluation of models with different kernel shapes in the architecture as well as models whose internal convolutional stages are composed of DenseNets (dense arrangement of skip-connections) or regular convolutional layers (no skip-connections). The baseline model was used in both cases, i.e., no TSND is performed as preprocessing step.

F-score	Without DenseNets		With DenseNets	
	Square	Multiple	Square	Multiple
Piano	0.9605	0.9647	0.9615	0.9637
Violin	0.9154	0.9184	0.9063	0.9171
Viola	0.7915	0.7918	0.7686	0.7990
Cello	0.8868	0.8909	0.8814	0.8946
Horn	0.7379	0.7411	0.7038	0.7520
Bassoon	0.7518	0.8188	0.7596	0.7684
Clarinet	0.8169	0.8615	0.8191	0.8758
Micro	0.8737	0.8885	0.8669	0.8863
Macro	0.8372	0.8553	0.8287	0.8529
Weighted	0.8765	0.8894	0.8706	0.8881
# params	5.8 M	4.2 M	2.0 M	2.0 M

It is easy to realise that the multi-channel models have better performance regardless of the value of β. This should be due to the fact that using a multi-channel input allows the model to learn feature-maps directly using the information of every decomposed sound feature. When a multi-input framework is used, the convolutional stages for each type of input are isolated and therefore it is harder for the network to learn relations between feature-maps of distinct input heads during backpropagation training.

The separation factor that obtained the best overall performance was $\beta = 2.0$, but just for a marginal difference when compared to other multi-channel models. This was the value chosen for the rest of the experiments involving TSND, but using a different value should not considerably affect the results.

4.2.5.4 Smoothing Instrument Activations

A common problem of most frame-level classification models is that they are susceptible to estimating erroneous classes for a short sequence of frames. This means that it is common to find short duration segments where particular instrument activations are missing or that the instrument activations change from positive to negative during a few consecutive frames (high variance).

In order to smooth the time series composed of the binary instrument activations across time, a postprocessing step that consists of the application of a fixed-length median filtering was tested. A preliminary study testing odd-length

Table 4.3: Comparison of overall performance of the models using TSND with different values for the separation factor β. Multi-Channel are models following Figure 4.3b while Multi-Input are models following Figure 4.3c. The vertical median filtering was set to 11, which is approximately 475 Hz and the horizontal median filtering length was set to 21, which is 210 ms. The decision window is set to 710 ms. Only the averaged F-score values are shown.

Models	Multi-Channel			Multi-Input		
	micro	macro	weighted	micro	macro	weighted
$\beta = 1.0$	0.8724	0.8323	0.8755	0.8596	0.8115	0.8615
$\beta = 1.5$	0.8710	0.8325	0.8748	0.8496	0.8017	0.8559
$\beta = 2.0$	0.8803	0.8448	0.8825	0.8589	0.8163	0.8613
$\beta = 2.5$	0.8729	0.8327	0.8760	0.8481	0.8067	0.8516
$\beta = 3.0$	0.8726	0.8319	0.8749	0.8471	0.8009	0.8538
$\beta = 3.5$	0.8761	0.8362	0.8780	0.8117	0.7521	0.8022

filter lengths ranging from 3 to 49 was done for every evaluated model and the value of a fixed length of 19 samples was verified to consistently provide the most gain in performance on average across different models. Therefore, this value was then used when performing the smoothing technique.

Table 4.4 includes the F-scores of the top two proposed methods: one using the raw music spectrogram as input according to Figure 4.3a and the other using TSND and following the framework in Figure 4.3b. The table provides the F-scores when using the raw predictions (after thresholding) and when using smoothed predictions via a median filtering of length 19 across time for each instrument.

We can easily see that the smoothing technique improves the metrics of both models. As mentioned before, this is due to the fact that the median filtering postprocessing stage reduces quick variations in the instrument predictions across time. As a way of visualising the effects, Figure 4.5 shows the instrument activations for each of the 10 tracks contained in the MusicNet test set.

4.2.5.5 Usefulness of TSND for Instrument Classification

After performing the experiment to determine best choices of TSND and model parameters, it is possible to compare the performance of the instrument classification task performed by DNN classifiers when applied directly to the music spectrogram or to the decomposed sound estimates.

In addition to using the proposed musically motivated CNN classifier, which was inspired by the 3W-MdenseNet, two other traditional CNN architectures that are commonly used as classifiers due to their high performance in computer

Table 4.4: Comparison between using the raw instrument activations (direct output of the models after thresholding) and using smoothed activations, which is done through the application of a median filtering of a fixed length of 19 frames (the equivalent of 190 ms)

F-score	Best Overall Methods			
	Raw Predictions		Smoothed Predictions	
	Baseline	With TSND	Baseline	With TSND
Piano	0.9637	0.9635	0.9636	0.9631
Violin	0.9171	0.9138	0.9174	0.9147
Viola	0.7990	0.8011	0.8075	0.8044
Cello	0.8946	0.8869	0.8968	0.8914
Horn	0.7520	0.7174	0.7534	0.7270
Bassoon	0.7684	0.7897	0.7618	0.7884
Clarinet	0.8758	0.8413	0.8776	0.8442
Micro	0.8863	0.8803	0.8872	0.8821
Macro	0.8529	0.8448	0.8540	0.8476
Weighted	0.8881	0.8825	0.8891	0.8844

vision tasks. were also included in the comparison

One of the models is the VGG16 [Simonyan and Zisserman, 2015], which consists of stacks of multiple convolutional layers and max-pooling layers (down-sampling by a factor of two a total of 5 times) and two fully connected layers, with 4096 units and another with 10 (number of instrument classes). In summary, the VGG16 has 16 layers and a 14.7M trainable parameters after being adapted to perform the instrument classification task of this section. This architecture uses only square (3×3) filters and no skip-connections.

The other architecture is the ResNet50 [He et al., 2016], which is inspired by the VGG16, but uses more stacks of convolutional layers reaching a total of 50 layers internally. Also, the ResNet50 architecture has a path called the *residual path*, which consists of a path of forward skip-connections that skips the following 2 convolutional layers in the original path. Those residual connections (skip-connections) prevent vanishing of gradients during training. The ResNet50 has a total of 23.5 M trainable parameters after being adapted to perform the instrument classification task of this section. Explaining those architectures in detail is out of the scope of this thesis and the reader is referred to previous work [Simonyan and Zisserman, 2015; He et al., 2016] in order to know more about them.

Table 4.5 shows the F-scores of each evaluated model. First of all it is important to note that the proposed architecture obtains much higher performance for each instrument when compared to the other deeper classifiers. While using

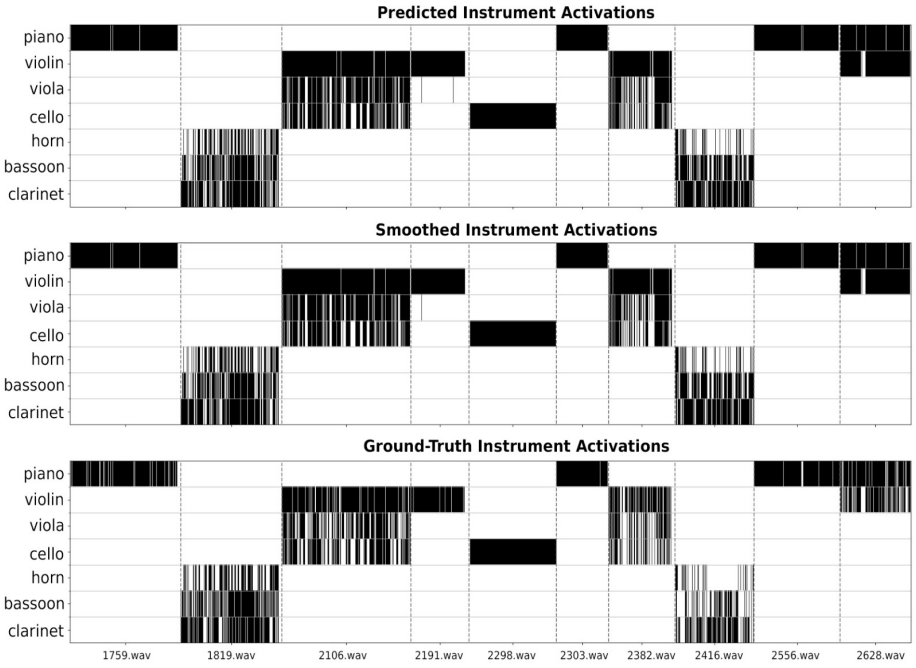


Figure 4.5: Visualisation of instrument activations across time. The activations are smoothed by applying a median filter to the binary instrument activation (after thresholding). Observe that the postprocessed models have short-time spurious activations eliminated; a good example is verifying that for track `2191.wav` the wrong short-duration viola activations are reduced after the smoothing approach.

a tenth of number of trainable parameters of a ResNet50, it surpasses the performance of the latter. This corroborates the methodology that it is essential to focus on building more efficient networks, whose architectures are adapted to a desired task using domain knowledge. However, it is important to note that performing the instrument classification using only data from MusicNet might be a simple problem to be directly addressed using deeper models, such as ResNet50 or $VGG16$. Those models need more data to not overfit to the training data. This might have happened in this case; more experiments using larger datasets and other types of instruments are necessary in order to draw a better conclusion.

Regarding TSND, it seems that it was not effective in helping the network to learn better timbre-discriminative features. While our perception of timbre is affected differently by transient and stationary parts of the sounds, having isolated transients and steady-state sounds as distinct and isolated inputs does not provide any benefit for training a neural network framework. When think-

ing more critically about this proposed methodology, it is possible to conclude that the chosen TSND procedure does not provide any new information to the framework, all the information used by the model was already in the original music spectrogram, and what the method essentially does is assigning each bin to one of the three inputs. While this could provide benefit to downstream tasks if we process each type of source separately, this type of decomposition is a transformation that essentially is transparent to the model since all the estimated sources are given back to the model as a multi-channel input. Providing a single music spectrogram as input is equivalent to providing disjoint parts of it as a multi-channel inputs. Therefore, there is neither significant improvement nor reductions of the performance.

Table 4.5: Evaluation of different CNN classifiers performing instrument activity detection on MusicNet dataset. The classifier architecture proposed in Subsection 4.2.1.2 is compared with VGG16 and ResNet50 architectures.

F-score	Classifier Architecture					
	Without TSND			With TSND		
	Proposed	VGG16	ResNet50	Proposed	VGG16	ResNet50
Piano	0.9637	0.9623	0.9431	0.9635	0.9600	0.9594
Violin	0.9171	0.9156	0.9065	0.9138	0.9086	0.9003
Viola	0.7990	0.7773	0.7377	0.8011	0.7699	0.7387
Cello	0.8946	0.8908	0.8674	0.8869	0.8825	0.8792
Horn	0.7520	0.7472	0.6885	0.7174	0.7349	0.7521
Bassoon	0.7684	0.7983	0.7456	0.7897	0.7722	0.7649
Clarinet	0.8758	0.8347	0.7640	0.8413	0.8062	0.8038
Micro	0.8863	0.8804	0.8491	0.8803	0.8691	0.8638
Macro	0.8529	0.8466	0.8076	0.8448	0.8335	0.8284
Weighted	0.8881	0.8825	0.8526	0.8825	0.8723	0.8665

4.3 Pitch-informed Instrument Assignment

While in Section 4.2 the task of instrument recognition was addressed with the goal of detecting instrument activations across time, in this section, the work in this section is focused towards processing polyphonic multi-instrumental recordings with the objective of assigning note-events to their corresponding instrumental source.

This task is known as **instrument assignment** (or **pitch streaming**) and when used along with an MPE method, allows multi-instrument transcription. Each note can not only have its pitch, onset and offset properly estimated, but the information regarding the instrumental sources that produce them can also

be recognised. This task was discussed in Section 2.3, where more information regarding this task along with a literature review is provided.

In summary, the proposal here is to address this task as a note-level instrument classification, where the main objective is to associate each note-event of a music signal to an instrument class. This method contrasts to other state-of-the-art instrument recognition approaches since it analyses each note-event individually while typical instrument recognition approaches usually address the instrument classification task on a frame-level basis, as the method proposed in Section 4.2, or on a clip-level basis [Han et al., 2017; Gururani et al., 2019].

Previous work has shown that the use of pitch information can help frame-level instrument recognition [Hung and Yang, 2018]. Inspired by this method, a framework that uses a main input carrying information about the music signal along with an auxiliary input encoding information of single note-events is proposed here. It is also shown that this approach can obtain good performance when the note-event information is predicted using state-of-the-art MPE algorithms [Thomé and Ahlbäck, 2017; Wu et al., 2019] rather than using ground-truth note labels. Therefore, the proposed method is designed as a modular framework, which can be combined with any MPE algorithm in order to obtain multi-instrumental pitch predictions.

The proposed method here was published in Lordelo et al. [2021a] but a more detailed explanation and discussion can be found in the next subsections.

Furthermore, throughout the thesis the strategy of building musically motivated CNNs by including kernels with multiple shapes in the network has proven to be an efficient way of applying domain knowledge for the tasks of source separation and instrument classification. In this section, the methodology is maintained by adapting from the proposed architecture for a frame-level classifier explained in Section 4.2, which, in turn, was a variation of the 3W-MDenseNet (explained in Subsection 3.2.6) for performing classification instead of source separation. The experiments in this section verify that this strategy can also improve instrument assignment performance.

4.3.1 Differences from Previous Work

One of the closest work to the proposed approach is Hung and Yang [2018], where a frame-level instrument recogniser is proposed using the CQT spectrogram of the music signal allied with the pitch information of the note-events as inputs. Here, the pitch annotations are also used to guide the instrument classifier, but the proposal in this thesis differs from Hung and Yang [2018] in the fact that classification is performed for each note-event individually, while Hung and Yang [2018] use the whole piano-roll at once to guide frame-level

instrument recognition. While they are able to obtain instrument activations leveraging from the pitch information, they do not stream the note events into their corresponding instruments. Similarly, the proposed approach explained in Section 4.2 also cannot associate specific notes-events to instrument classes. In fact, it predicts all instruments that are active in a target frame.

Moreover, in the task of score-informed source separation the main idea is to use the score information, usually in the form of a MIDI piano-roll, of the target instrument as a guide for the separation of that particular source. Also inspired by this task, the proposed method uses note-event information to address the problem of note-level instrument recognition as a "pitch-informed instrument assignment" task. Here, each note is considered as having constant pitch, onset and offset values and this information is used to guide the network for performing instrument recognition.

In order to do this, the whole pitchgram of a music clip is broken down into its constituent note-events, which are used separately as an auxiliary signal to condition the network to predict the correct instrument. Differently than other instrument recognition approaches, instead of just using the spectrogram of a clip to try to recognise the instruments that are active, the spectrogram along with the information of a specific note-event is used to recognise the unique instrument that is related to that note.

4.3.2 Proposed Method

In the proposed method, the same definition of note-events as in the MIREX MPE task[1] is used. Each note N is considered an event with a constant pitch f_0, an onset time T_{on} and an offset time T_{off}. Therefore, if a music signal has a total of M notes, any note N_i, with $i \in \{1, \cdots, M\}$, can be uniquely defined by the tuple $(f_0^i, T_{\text{on}}^i, T_{\text{off}}^i)$. In the experiments, two ways of obtaining this note information were used. The first is utilising ground-truth pitch labels provided by the employed dataset (MusicNet) [Thickstun et al., 2017] and the second using pitch estimates predicted by state-of-the-art MPE algorithms [Thomé and Ahlbäck, 2017; Wu et al., 2019]. The f_0 granularity is considered to follow the semitone scale, ranging from $A0$ to $G\sharp 7$ (MIDI $\#21 - 104$).

In this framework polyphony is allowed, so, most of the time more than a single note will be active, but the objective is to analyse each note of the audio signal separately in order to be able to assign an instrument class to it. This is done by using two inputs to the model: the main input $X(f, t)$, with f representing frequency and t representing time, is a time-frequency representation of a segment of the audio signal around the value of T_{on}, and an auxiliary input

[1] `https://www.music-ir.org/mirex/`

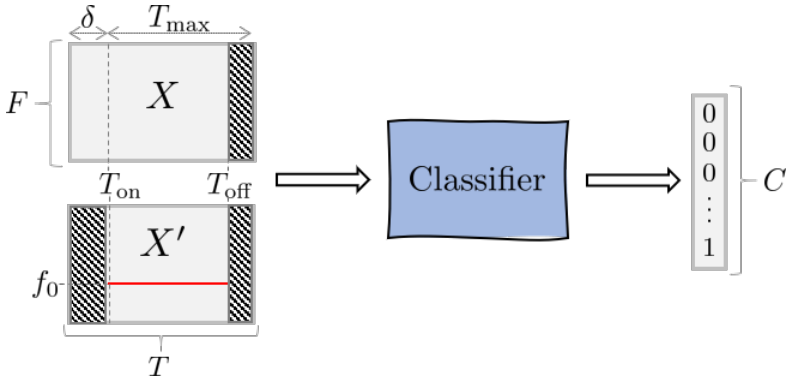


Figure 4.6: Overview of the proposed framework for instrument assignment. See Section 4.3.2 for the detailed explanation of the variables in the figure.

$X'(f,t)$, which carries information regarding a note-event (f_0, T_{on} and T_{off}). The two inputs are concatenated into a two-channel input $\mathbf{X}(f,t,c)$, where $c \in \{1,2\}$ represents the channel dimension, that is fed to the model. In Figure 4.6 an overview of the proposed framework is shown.

4.3.2.1 Main Audio Input

The main input is a time-frequency representation $X(f,t) \in \mathbb{R}^{F \times T}$ of a small clip of the music signal, where F is the number of frequency bins and T is the number of time frames. The clip is generated by first setting a maximum duration $T_{\max}$ for the note. Values of $T_{\max}$ ranging from 400 ms to 1 s were tested (see Subsection 4.3.6 for details) and 400 ms obtained the best results, so this value was kept for all of the other experiments. If any note N_i has a duration D_i greater than $T_{\max}$, i.e., $D_i = T_{\text{off}} - T_{\text{on}} > T_{\max}$, only its initial time span of $T_{\max}$ seconds is considered.

Next, for every note N_i, $X_i(f,t)$ is constructed by picking a segment of duration $T = T_{\max} + \delta$ from the original music signal starting from $T^i_{\text{on}} - \delta$, where δ is a small interval to take into account deviations between the true onset value and the annotated (or estimated) value. The inclusion of the extra window of δ from the music signal also helps the convolutional layers since it brings some context of the signal before the note onset time. The value of δ was fixed to 30 ms after initial tests. Lastly, if the note duration D_i is less than $T_{\max}$, we set the values of $X_i(f, t > D_i + \delta)$ to zero, where $D_i = T^i_{\text{off}} - T^i_{\text{on}}$.

4.3.2.2 Auxiliary Note-Related Input

In order to provide the note-event information to guide the model to recognise its corresponding instrument, an auxiliary input is necessary. It serves as a

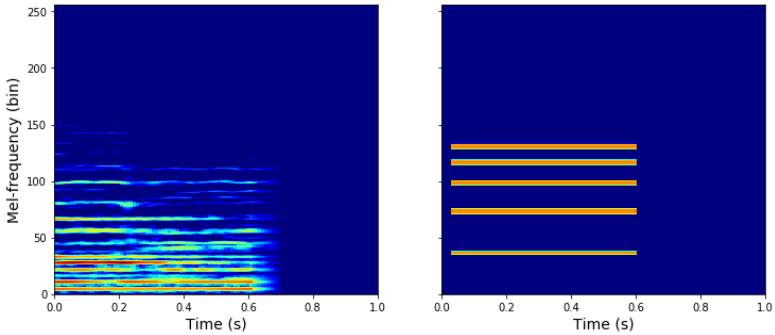


Figure 4.7: Pair of inputs using 256 Mel-frequency spectrogram. On the left is depicted $X(f,t)$, where three pitches are simultaneously active (MIDI # 58, 62 and 74) and on the right $X'(f,t)$, where the note-event with pitch # 74 is represented using a harmonic comb of $H = 5$. In this example, $D_i = 600\,\text{ms}$ and $T_{\text{max}} = 1\,\text{s}$.

"salience function" to the model, where only the frequency bins related to the fundamental frequency of a note and its harmonics will be 'highlighted'.

Effectively, the auxiliary input $X'(f,t) \in \mathbb{R}^{F \times T}$ is defined as a harmonic comb representation using the pitch value f_0 as the first harmonic[2], such that,

$$X'(f,t) = \begin{cases} 1, & \text{if } 2^{-1/24} h f_0 < f \leq 2^{1/24} h f_0 \quad \text{and} \quad T_{\text{on}} \leq t \leq T_{\text{off}} \\ 0, & \text{otherwise} \end{cases}, \quad (4.12)$$

where $h = \{1, 2, 3, \cdots, H\}$ with H being the total number of harmonics in the representation. Multiple values for H were tested (see Section 4.3.5). Note that, a tolerance of half a semitone for each harmonic value is used when constructing $X'(f,t)$ from the magnitude STFT. Furthermore, even though this representation starts as binary, the linear frequency is converted to Mel-frequency, which can result in having non-binary values due to the linear-to-Mel transformation. Also, it is important to note that the values of X' before T_{on} and after T_{off} are set to zero, indicating that the note-event that should be classified starts at T_{on} and ends at T_{off}. In Figure 4.7 we show an example of a pair of inputs for the framework.

4.3.2.3 Output

The note-level instrument assignment task is addressed as a multi-class single-label classification task. Given a pair of inputs $\mathbf{X}$, the objective is to classify them as belonging to one of C instrument classes. Therefore, a deep neural network receives $\mathbf{X}$ as input and outputs a C-dimensional vector $\hat{y}$ with a final

[2]Considering the definition that f_0 corresponds to the first harmonic.

softmax activation function, ensuring the values of $\hat{y}$ represent probabilities that sum up to 1. The model is trained using the cross-entropy loss. At inference time, the class corresponding to the dimension with the highest value in $\hat{y}$ is predicted. See Section 4.3.3 for details regarding the network architecture.

It is worth noting that in the cases where two or more instruments are playing the same pitch simultaneously, the small differences between the notes' onset and offset values can generate different inputs $\mathbf{X}$. Thus, it would still allow the instrument assignment task to be properly executed as a single-label classification scenario. However, when the pitch, onset and offset values of notes from different instruments exactly match, this system will consider them as a single note and only a single instrument will be estimated. This case rarely happens in real-world scenarios for many musical styles. For instance, from a total of 1089540 labelled note-events in the MusicNet dataset, only 10140 had the same pitch, onset and offset values, which is around 0.9%. For the experiments, we have considered note-events that were performed by a single instrument, and discarded the events that were concurrently produced (in terms of the same pitch, onset, and offset times) by multiple instruments. As a proof of concept, this is not considered a severe limitation for the framework and multi-labelled approaches are left as future work.

Also, two choices for time-frequency representations were compared: the Mel-frequency Short-Time Fourier Transforms (Mel-STFT) and the CQT. The mel-STFT is constructed by first forming X' according to Equation (4.12) and then applying the mel-frequency transformation while the CQT is also straightforward to compute by using corresponding CQT-bins based on the corresponding MIDI value of f_0.

We consider two ways of getting note-event information in the experiments. One is using human-labelled ground-truth note labels provided by MusicNet, but since in real-word applications, it is not possible to have direct access to this type of label, we also evaluated the performance of instrument assignment task by using note-events estimated by two state-of-the-art MPE algorithms: Thomé and Ahlbäck [2017] and Wu et al. [2019].

4.3.3 Musically Motivated Classifier Architecture

The architecture used in the experiments for instrument assignment is the same architecture used for the frame-level instrument recognition with few hyperparameter modifications. The architecture is based on the utilisation of vertical, horizontal and square kernel shapes and dense arrangements of skip-connections via the usage of DenseNets rather than regular stacks of convolutional layers. For a detailed explanation of the CNN architecture the reader is referred to

4.2.1.2.

Regarding the small changes performed in the architecture, in Section 4.2 raw linear-frequency scaled magnitude spectrograms were used as input to the model. Here, experiments using either Mel-STFT or CQT are implemented. This fact decreases the frequency resolution of the input representation and reduces the corresponding input dimension. In comparison, in Section 4.2 513 frequency bins are used and here either 256 Mel-bins or 115 CQT bins are used. Due to this fact initial tests suggested that smaller vertical kernels were getting slightly better results. Therefore, the vertical kernel shapes were changed from (1×11) to (1×9) and a similar follow-up change was performed on the horizontal kernels, i.e., changed from (11×1) to (9×1). The reason for the latter was because there was no considerable effect on performance and by doing so, each branch of the multi-branch convolutional stages would have the same number of trainable parameters. The square kernels kept the shape of (3×3)

Other hyperparameter choices that are different is the fact that this model uses a softmax activation layer as the final layer and uses $D = 4$ multi-branch convolutional stages and a single hidden fully connected layer of 64 neurons. A summary of the architecture adopted for the experiments in this section appear in Figure 4.8.

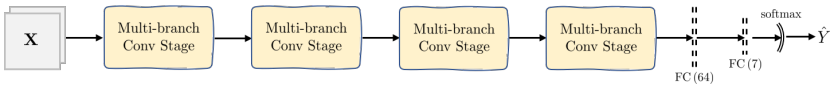


Figure 4.8: Classifier architecture. Each of the 3 DenseNets in a multi-branch convolutional stage has growth rate $k = 25$ and $L = 4$ layers. Relu is used as activation function after each convolutional layer. The number of trainable parameters is 1.2 million his time.

4.3.4 Dataset

The dataset utilised in the experiments is the MusicNet dataset [Thickstun et al., 2017], which is the largest publicly available dataset with non-synthesised data that is strongly labelled (with frame-level instrument annotations and note onset, offsets and pitch annotations). This dataset was also utilised in the experiments related to the frame-level recognition in Section 4.2, so the reader is pointed to Subsection 4.2.2 for more details about this dataset.

In the experiments of this section the same procedure of performing a 7-class instrument classification using the following classes: *piano, violin, viola, cello, french horn, bassoon* and *clarinet* was kept since the other 4 instrument classes are not in the official test set provided by the authors.

In Table 4.6 it is possible to see the statistics of the note-event information in

MusicNet. Observe that the dataset is heavily biased towards piano and violin given their frequent presence in Western classical music recordings.

Set	Piano	Violin	Viola	Cello	Horn	Bassoon	Clarinet	Harps.	Bass	Oboe	Flute	Total
Train	628549	197229	88446	89356	10770	13874	22873	4914	3006	8624	8310	1075951
	58.4%	18.3%	8.2%	8.3%	1.0%	1.2%	2.1%	0.5%	0.3%	0.8%	0.8%	100%
Test	5049	3238	842	1753	557	873	1277	0	0	0	0	13589
	37.2%	23.8%	6.2%	12.9%	4.1%	6.4%	9.4%	0	0	0	0	100%

Table 4.6: Statistics of the note-events information in MusicNet across train and test sets.

4.3.5 Experimental Setup

In all experiments the original train/test split provided by MusicNet was used with the original sampling frequency of 44100 Hz. For experiments that involved the computation of Short Time Fourier Transform (STFT) we used Blackman-Harris windows of 4096 samples to compute the Discrete Fourier Transform (DFT). The hop size was always set to 10 ms in every experiment. The hop size was kept the same as in the experiments in Section 4.2, but the STFT window was now changed to a higher number since it is beneficial to have a higher frequency resolution before converting to the mel-frequency scale.

From the training set, 5% of the notes of each class were picked and a validation set was created. The models were trained using the Adam optimiser with an initial learning rate of 0.001, which was reduced by a factor of 0.2 if the cross-entropy loss stopped improving for 2 consecutive epochs on the validation set. If no improvement happened after 10 epochs, the training was stopped early. The experiments were performed using the Tensorflow/Keras Python package.

The classification performance was evaluated by computing the note-level F-score (F_s), which was already explained in Section 4.2.3 with Equation (4.11).

For the cases when the instrument assignment is done on top of MPE algorithms, 2 groups of metrics that are generated following the MIREX evaluation protocol for the music transcription task are provided. In the first group, an estimated note is assumed correct if its onset time is within 50 ms of a reference note and its pitch is within a quarter tone of the corresponding reference note. The offset values are ignored. In the second group, on top of those requirements, the offsets are also taken into consideration. An estimated note is only considered correct if it also has an offset value within 50 ms or within 20% of the reference note's duration around the original note's offset, whichever is largest. After all notes are verified, the F-score is computed note-wise across time and the average value (micro F-score) is provided here. This evaluation method was

computed using the `mir_eval.transcription`[3] toolbox.

4.3.6 Results

4.3.6.1 Effects of the Kernel Shapes

First, the effects of the inclusion of multiple kernel shapes in the architecture of the CNN are analysed. Table 4.7 compares 3 versions of the model: one that uses only square filters in a single branch; a version using the branched structure, but with (3×3) kernels in each; and another model with the proposed multi-branch structure with horizontal, square, and vertical kernels. For the single-branched case the growth rate of the DenseNets was increased to 57 channels in order to keep the number of trainable parameters of the network close to 1.2 million.

Analysing the results it is possible to realise that the addition of new kernel shapes improved the average F-score across all classes. Regarding each instrument class, one can say that for string instruments (piano, violin, viola and cello) there is a gain in performance, while for non-string instruments (horn, bassoon, clarinet) the performance either drops or remains with a negligible gain if compared to the models that used only square filters. This suggests that the inclusion of vertical the kernel helped the model in learning the percussive characteristics of the timbre of string musical instruments. Similarly, horizontal filters, helped the model in learning harmonic-related timbre-discriminative features.

Table 4.7: Instrument assignment performance based on the kernel shapes used in network. The metrics shown are the F-scores achieved by each instrument class and the average value (macro F-score) across all instruments.

Kernel	Piano	Violin	Viola	Cello	Horn	Bassoon	Clarinet	**Macro**
(3×3)	0.994	0.936	0.757	0.954	**0.826**	0.864	0.954	0.898
$3 \times (3 \times 3)$	0.995	0.939	0.764	0.945	0.819	**0.896**	0.965	0.903
Multiple	**0.997**	**0.944**	**0.775**	**0.958**	0.810	0.879	**0.967**	**0.904**

4.3.6.2 Evaluation of Different Input Sizes

Different values for the input size (decision window) were also evaluated. More specifically, multiple values for $T_{\max}$, which is the maximum valid window of analysis for a note event, were compared. The results are shown in Table 4.8. It is possible to see that the shortest input size of 400 ms obtained the best results. It is believed that this is due to the fact that the average duration of a note-event in the test set of MusicNet is 260 ms and the 90th percentile is 0.464 ms.

[3] `https://craffel.github.io/mir_eval/`

So, the value of 400 ms is already good enough to represent the vast majority of the notes. Moreover, when the analysed note event is longer than 400 ms, the 400 ms initial window contains most of the important timbre-discriminative features for the model. This seems to go in line with human perception of timbre, which according to music psychology studies [McAdams et al., 1995], the most important information for humans to recognise instruments is around the onset of the notes.

Table 4.8: Comparison between different values for the input size. The values shown in first column represent the maximum valid note duration $T_{\max}$. The metrics are the F-scores achieved by each class and the average value (macro F-score) across all instruments .

$T_{\max}$	Piano	Violin	Viola	Cello	Horn	Bassoon	Clarinet	**Macro**
400 ms	**0.997**	**0.944**	**0.775**	**0.958**	0.810	0.879	**0.967**	**0.9043**
600 ms	0.996	0.942	0.771	0.954	**0.826**	**0.881**	0.959	0.9043
800 ms	0.996	0.944	0.772	0.957	0.814	0.868	0.965	0.9022
1 s	0.997	0.931	0.740	0.954	0.742	0.871	0.948	0.8832

4.3.6.3 Auxiliary Input and Types of Representations

To test the importance of the auxiliary input and how its modification would affect the performance of the model, a version of the model using only the main mel spectrogram input and versions using different numbers of harmonics H in the auxiliary input (from $H = 1$ to $H = 5$) were compared. Also two types of input representation for the model were tested: the Constant-Q Transform (CQT) and the mel-frequency spectrogram. The CQT was computed using 12 bins per octave and a total of 115 bins starting from $G\sharp 0$ (MIDI #20). The mel-frequency spectrogram was computed by a linear transformation of an STFT onto a mel-scaled frequency axis, using 256 mel-bins. The results are provided in Table 4.9.

Analysing the results, it is possible to say that the auxiliary input is extremely necessary for the framework. Without it, the average F-score only reaches 60.9%, while with it the performance improves up to 90.4%. Apart from piano, all other classes have a large decrease in performance when the auxiliary input is excluded from $\mathbf{X}$. It is believed that the results for the piano class continue to be high not only because of the MusicNet bias towards piano, but also because some recordings of the test set are solo piano recordings, which facilitates the classification of piano notes when analysing only the main input signal due to the absence of other classes. Regarding the number of harmonics used in the auxiliary input, in general, the CQT seems to work best with few

Table 4.9: Evaluation of instrument assignment task when using CQT or Mel spectrograms as input representation for the network as well as a comparison between models trained with no auxiliary input and models trained with different number of harmonics in the auxiliary input. This experiment was performed using $T_{max} = 400\,\text{ms}$

Main Input	Aux Input	Piano	Violin	Viola	Cello	Horn	Bassoon	Clarinet	**Macro**
CQT	—	0.960	0.732	0.116	0.725	0.512	0.296	0.681	0.575
	$H = 1$	**0.994**	0.934	0.763	0.955	0.783	**0.888**	0.950	**0.895**
	$H = 2$	0.993	0.939	0.771	**0.960**	**0.785**	0.858	0.929	0.891
	$H = 3$	0.992	**0.946**	**0.772**	0.957	0.754	0.884	**0.952**	0.894
	$H = 4$	0.993	0.938	0.766	0.958	0.784	0.869	0.950	0.894
	$H = 5$	0.993	0.939	0.767	0.952	0.769	0.874	0.949	0.892
Mel STFT	—	0.967	0.742	0.222	0.730	0.607	0.306	0.690	0.609
	$H = 1$	0.996	0.939	0.759	0.958	0.780	0.867	0.958	0.895
	$H = 2$	0.994	0.945	0.779	0.956	0.809	0.864	0.946	0.899
	$H = 3$	**0.997**	0.944	0.775	**0.958**	0.8104	0.879	**0.967**	**0.904**
	$H = 4$	0.996	0.935	0.747	0.945	**0.839**	**0.891**	0.960	0.902
	$H = 5$	0.996	**0.947**	**0.783**	0.954	0.801	0.876	0.954	0.902

harmonics, while the Mel-STFT prefers higher values. A possible explanation for this is the fact that it is harder to represent higher order odd harmonics on the CQT using a log-frequency resolution of 12 bins per octave. However, more experiments are needed in order to better investigate this assumption.

4.3.6.4 Streaming of Multi-Pitch Estimations

Once it was verified that the model obtains impressive performance when original ground-truth note-event information is used, the classifier was evaluated in a more realistic environment, where no note-event labels were readily available. Frame-level pitch values were estimated using two third-party MPE algorithms [Thomé and Ahlbäck, 2017; Wu et al., 2019]. For the algorithm in Thomé and Ahlbäck [2017], an implementation from the original authors was obtained, while an implementation of Wu et al. [2019] is available via the project Omnizart[4]. Both MPE algorithms were performed on the multi-instrumental music recordings to obtain note-event information such as pitch, onset and offset. The pair of inputs of the classifier were generated based on the estimated data.

It is important to observe that errors in the MPE estimation will be carried over to the instrument assignment task. If a note is wrongly estimated, no ground-truth class for the instrument assignment task exists, so it is hard to evaluate the results in the same way done for the other experiments. Therefore, in this experiment, transcription metrics explained in the last paragraph of

[4] `https://github.com/Music-and-Culture-Technology-Lab/omnizart`

Table 4.10: Transcription results when considering only the estimated pitches, onset times and instrument classified by the proposed model using Ground-Truth (GT) labels for note-event information and when using two different MPE methods instead. MPE-1 is Thomé and Ahlbäck [2017] and MPE-2 is Wu et al. [2019]. In the row "MPE-only", no instrument assignment is done, the multi-pitch estimates are evaluated using the reference ground-truth notes ignoring the instrument annotations, i.e., only taking into account pitches and onsets.

Instr.	Onset Only		
	GT	MPE-1	MPE-2
MPE-only	1	0.633	0.480
piano	0.997	0.745	0.451
violin	0.942	0.529	0.499
viola	0.775	0.366	0.308
cello	0.954	0.596	0.570
horn	0.804	0.460	0.429
bassoon	0.874	0.473	0.373
clarinet	0.967	0.616	0.456

Section 4.3.5 are used. The results appear in Table 4.10 ignoring note offset values and in Table 4.11 considering them.

In the first row of each table (MPE_only) the metrics are computed for each MPE algorithm individually, i.e., f-scores of all the estimated note-event information (pitch, onset and offset (Table 4.11)) with respect to the ground-truth note-event values provided by MusicNet without considering any instrumental source since MPE algorithms do not handle instrument information. Then, the evaluation on each instrumental source after using the proposed instrument assignment method on top of the raw estimated pitches and onsets is provided. Given the limitations of each MPE method used, it is possible to see that the approach can generate good multi-instrument transcriptions.

4.4 Conclusions

Research related to instrument recognition performed during the course of the PhD project was presented in this chapter.

In Section 4.2 a deep learning-based frame-level instrument recognition approach was proposed while in Section 4.3 the model was adapted to perform note-wise instrument assignment instead. In both cases the proposed methods were based on a CNN classifier, whose architecture was built to foster efficiency (high performance and low number of trainable parameters) by applying do-

Table 4.11: Transcription results considering only the estimated pitches, onset, offset times, and instrument classified by the proposed model when using Ground-Truth (GT) labels for note-event information and when using two different MPE methods instead. MPE-1 is Thomé and Ahlbäck [2017] and MPE-2 is Wu et al. [2019]. In the row "MPE-only", no instrument assignment is done, the multi-pitch estimates are evaluated using the reference ground-truth notes ignoring the instrument annotations, i.e., only taking into account pitches, onsets and offset times.

Instr.	Onset + Offset		
	GT	MPE-1	MPE-2
MPE-only	1	0.423	0.200
piano	0.997	0.497	0.196
violin	0.942	0.381	0.225
viola	0.775	0.227	0.116
cello	0.954	0.507	0.258
horn	0.804	0.232	0.166
bassoon	0.874	0.193	0.130
clarinet	0.967	0.344	0.165

main knowledge to choose better kernel shapes for the convolutions and a dense arrangement of skip-connections to avoid learning redundant feature-maps. It was shown that the proposed musically motivated CNN architecture is beneficial for both tasks.

The investigation of whether providing separated transient, stationary and residual sounds to a deep-learning model is beneficial for performing instrument recognition, has shown to bring no overall improvement to the framework. The chosen approach to estimate such type of sources based in a TSND preprocessing step estimates disjoint spectrogram masks that are used to generate 3 disjoint spectrograms, which are then used as joint inputs to the model. The disjoint separation along the channel dimension of the input seems to be ignored by the model and have no effect in performance. Using other type of transient-stationary separation seems to be necessary ideal in order to draw better conclusions.

Regarding the instrument assignment task, a proposal of addressing it as by a deep learning-based note-informed instrument recogniser was detailed in Section 4.3. The proposed framework uses the pitch, onset and offset information of the note-events as a way of guiding the classification. It was verified that the proposed approach is able to assign note-events into 7 different classes of instruments with a macro-averaged F-score of 90.4%. Furthermore, the proposed

approach could also successfully classify notes provided by an MPE algorithm, which permits generating multi-instrument transcriptions.

As future work a suggestion is to investigate more deeply the interaction of the instrument assignment method with other MPE algorithms, as well as, how to leverage the two proposed models of this section in order to build better frame and clip-level instrument recognisers.

Chapter 5

Conclusions and Future Work

The main goal of this thesis is to propose data-driven approaches for music source separation and instrument recognition taking into account music domain knowledge while keeping numbers of parameters low. To combat prohibitive memory usage that is usually exacerbated by large-scale model training, architectural improvements to deep-learning-based models are developed. In an attempt to reduce the number of trainable parameters and still achieve state-of-the-art performance, the methodology applied throughout the whole thesis consists of the exploitation of features related to harmonic and percussive sounds in order to build musically motivated CNN architectures.

As discussed in Section 1.6, most of the work presented in this thesis was presented in international peer-reviewed conferences and journals. In this chapter, the contributions of the thesis are summarised and directions for future work are presented.

5.1 Summary of Contributions

5.1.1 Musically Motivated CNN Architectures

All the proposed methods for music source separation studied in Chapter 3, as well as the methods for instrument recognition studied in Chapter 4 make use of novel CNN architectures. Those methods involve the use of vertical, square and horizontal kernel shapes in the internal convolutions of the CNNs, which improves performance when compared to traditional CNN architectures.

One of the key advantages of this methodology is that it makes use of do-

main knowledge about music to create more efficient CNN architectures that are able to achieve state-of-the-art performance. By incorporating vertical and horizontal kernel shapes, these CNNs are able to learn feature-maps that are better able to discriminate between different timbral characteristics of musical signals, such as the harmonic and percussive elements of a recording.

Additionally, the use of these kernels allows for the creation of musically motivated CNN architectures with a considerably lower number of trainable parameters compared to other methods. This makes them more computationally efficient, which is important for many practical applications of music source separation and instrument recognition.

In conclusion, the addition of vertical and horizontal kernel shapes in the internal convolutions of a CNN is an effective way of improving the performance of music source separation and instrument recognition tasks. By leveraging domain knowledge, it is possible to create more efficient CNN architectures that achieve state-of-the-art performance while using fewer trainable parameters. This has important implications for the field of music processing, as it allows for the development of more effective and efficient methods for separating and identifying individual instruments in complex musical recordings.

5.1.2 Music Source Separation

In Chapter 3 the 3W-MDenseNet is proposed, a musically motivated convolutional encoder-decoder where vertical, horizontal and square kernels are used in order to facilitate learning of features related to percussive and harmonic sounds. In Section 3.2 the task of HPSS is addressed and it is shown that this CNN architecture is beneficial for source separation, being able to achieve state-of-the-art performance with a lower number of trainable parameters if compared to other deep-learning-based source separation methods. While at an initial analysis the proposed model seems to be a single-task model since it only performs the task of source separation, in reality it is able to estimate both sources simultaneously via the utilisation of a loss function with 2 factors: one related to the reconstruction of the harmonic-source signal and another related to the reconstruction of the percussive-source signal. If we consider the estimation of each source as a single task, the proposed framework is, essentially, a multi-task learning method. This is also another contribution for the topic of music source separation as the original MDenseNet [Takahashi and Mitsufuji, 2017] and most spectrogram-based state-of-the-art music source separation models estimate only a single source.

In Section 3.3 another contribution is the development of a novel framework based on adversarial learning that allows incorporating additional types of

datasets into training. The proposed framework is able to adapt source separation models to perform the separation on music data from other domains, i.e., on music signals with different timbre of sounds, different genres and instruments. The proposed unsupervised adversarial domain adaptation technique is applied in the context of HPSS, where a labelled dataset from a source domain and an additional dataset comprising unlabelled data (no access to the original ground-truth signals of each source) from a target domain are used to train the model. It is important to note that often, unlabelled data from other domains (datasets) are available but traditional fully supervised learning frameworks cannot be used.

The experiments extended the 3W-MDenseNet to also generate domain-invariant encoded features. In order to do this, a domain-discriminator is included in the framework and the separator and discriminator are adversarially trained similar to a GAN [Goodfellow et al., 2014] scenario. It is shown that this semi-supervised approach leads to considerable increase in performance in the target domain over baseline models that are not able to make use of the additional data. Furthermore, the model maintains the original level of performance when evaluated over data from the source domain. This work was published in Lordelo et al. [2021b], which as far as the authors were able to verify, was the first work where adversarial unsupervised domain adaptation techniques are investigated for the task of music source separation.

Finally, a novel dataset is publicly released as part of the research developed in this thesis (described in detail in Subsection 3.3.4.1). The "Tap & Fiddle Dataset" is a dataset containing recordings of traditional Scandinavian fiddle tunes with accompanying foot-tapping along with isolated tracks for"foot-tapping" and "violin". The main repertoire characteristics consists of Scandinavian tunes of different dance types, including Norwegian Halling music, as well as Swedish polska tunes. This dataset can not only contribute for training novel source separation methods, but also be used for analysis of metrical expression in music and fiddle music studies in general.

5.1.3 Instrument Recognition

Contributions related to the topic of instrument recognition can be found in Chapter 4. The 3W-MDenseNet is adapted to perform classification rather than source separation, hence once more, musically motivated CNNs that use vertical, square and horizontal kernel shapes are proposed and applied in the problem of frame-level instrument classification (Section 4.2) and pitch streaming (Section 4.3). It is shown that the proposed architecture obtains considerably higher performance in both tasks when compared to other traditional CNN architectures

while using a much lower number of trainable parameters.

The experiments in Section 4.2 are inspired by works in cognitive musicology, which indicate that transient parts of sounds such as onset attacks and stationary parts such as note sustain affect our perception of timbre differently. An investigation of whether or not explicitly providing transient enhanced and stationary enhanced spectrograms to the CNN is beneficial for performing instrument classification is performed. In the proposed framework, a transient-stationary-noise decomposition of the original music spectrogram is performed using the method of Driedger et al. [2014a] and the resulting spectrograms are used as input to an instrument classifier. The results suggest that the preprocessing step brings no improvement to the instrument classification, indicating that the CNN can focus on different parts of a music spectrogram during training and the model weights can be adapted without the necessity of explicitly providing separated transients and stationary sounds.

In Section 4.3 the main contribution consists of a novel framework for assigning note-events in a music recording to their corresponding instruments. The task is addressed as a pitch-informed instrument classification, where an auxiliary input carrying information about the pitch, onset and offset of a single note-event is used along the music spectrogram to guide the separation. The experiments show that the proposed method is able to achieve 0.904 macro F-score when evaluated over 7 instrument classes on the MusicNet dataset. Finally, in Subsection 4.3.6.4 it is also shown that the proposed method is modular, meaning that note-events estimated by third party MPE algorithms can also be streamed into their corresponding instruments, which enables MPE methods to perform multi-instrument transcription.

5.2 Limitations and Future Work

It is important to think about efficiency when building neural networks. In this thesis, it was shown that it is possible to apply domain knowledge and foster parameter sharing in the model layers in order to build more efficient neural networks. The proposed musically motivated CNN architectures are beneficial for harmonic-percussive source separation, frame-level and note-level instrument recognition, but the thesis did not apply the proposed methodology to other source separation or MIR tasks. A direct follow-up of the experiments in Chapter 3 would be to generalise the two proposed frameworks to other types of source separation tasks, such as vocal-accompaniment separation and specific instrumental source extraction. Since the $3W$-MDenseNet is able to output more than a single source, it would be compelling to investigate the possibility of adapting this model to work with other types of instrumental sources. Also, exploring

the performance of the $3W$-MDenseNet for other MIR tasks, such as melody or multi-pitch estimation, music auto-tagging, drum detection/transcription and song segmentation, is also an interesting research direction towards demonstrating the general network effectiveness for MIR.

Also, the model's performance could be further investigated when adding dilated convolutions and/or other rectangular kernel shapes to the framework. Due to the harmonically spaced frequency patterns of harmonic instrument sounds, dilated convolutions can potentially increase the network performance even more. The fully vertical and fully horizontal kernel shapes were proposed with the idea of capturing percussive and harmonic sound features that appear in the spectrograms. However, a study regarding the utilisation of other non-traditional rectangular kernel shapes in the architecture could bring more insights to creating better CNN architectures for machine listening tasks.

Regarding the unsupervised adversarial domain adaptation method for HPSS proposed in Section 3.3, it was focused on integrating labelled and unlabelled datasets as a means of improving model generalisation. Using unlabelled data from another domain is an effective way of imbuing models with additional knowledge about a different distribution of data. However, investigating ways of finding the most informative data from the target domain that could potentially be manually labelled and have a high positive impact on performance if used under supervision could be a promising research direction. Inspiration comes from the topic of active adversarial domain adaptation from computer vision [Su et al., 2020], which not only proposes a domain-discriminative model to align domains, but also uses a model to weight samples to account for distribution shifts. The idea is to apply *active learning* [Kao et al., 2018], which is the process of quantifying the informativeness of unlabelled data so that they are maximally useful when annotated. It is expected that this "smart few-shot" adaptation can be useful in improving source separation performance in the absence of many labelled data samples. Moreover, the proposed adversarial learning framework is based on the standard GAN framework [Goodfellow et al., 2014]. The iterative adversarial training involved in GANs is known to be unstable, so it would be desirable to extend the framework to other more stable GAN formulations such as the Wasserstein GAN [Arjovsky et al., 2017].

It also would be interesting to extend the experiments to other types of labelled and unlabelled music data. For instance, the method proposed for adapting HPSS to other domains was evaluated on two domains. As a proof-of-concept, this suggests that the method successfully achieves the expected behaviour of improving performance on the target domain without losing performance on the source domain, but evaluating it using more data from other domain distributions is a desirable topic for further work. The same can be

concluded with the instrument recognition experiment, which used data only from MusicNet, which is not ideal. MusicNet provides frame-level instrument and pitch annotations, which allows us to train supervised models, but it is heavily unbalanced towards piano and most tracks in the test set are recordings with a solo instrument, which makes the instrument recognition task easier. A suggestion would be to extend all the proposed methods to utilise the Slakh2100 [Manilow et al., 2019] dataset that has a larger instrument taxonomy and also allows training and evaluation of source separation methods using much more data. The reason why it was not originally used in the thesis is due to the fact that it consists of artificial music signals while MUSDB18 [Rafii et al., 2017], Tap & Fiddle (Section 3.3.4.1) and MusicNet have real-world music recordings.

Finally, ideas for further work involving the proposed frame-level instrument classification and the proposed pitch streaming approach involve exploring other types of transient-stationary separation methods as a preprocessing step and extending the models to perform both tasks simultaneously, as they have shown to be deeply related tasks, using multi-task-learning frameworks.

Bibliography

Abu-Mostafa, Y. S., Magdon-Ismail, M., and Lin, H.-T. (2012). *Learning From Data*. AMLBook, 1st edition.

Anhari, A. K. (2020). Learning multi-instrument classification with partial labels. *arXiv e-prints*.

ANSI, A. N. S. I. (1973). S3.20-psychoacoustical terminology. Technical report, Acoustical Standards Association.

ANSI, A. N. S. I. (1978). S3.20-r1978-american national standard on bioacoustical terminology. Technical report, American Standards Association.

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein GAN. *arXiv e-prints*, pages 1–32.

Arora, V. and Behera, L. (2015). Multiple f0 estimation and source clustering of polyphonic music audio using PLCA and HMRFs. *IEEE/ACM Transactions on Audio Speech and Language Processing (TASLP)*, 23:278–287.

Balzano, G. J. (1986). What Are Musical Pitch and Timbre? *Music Perception*, 3(3):297–314.

Bauer, B. B. and Torick, E. L. (1966). Researches in loudness measurement. *IEEE Transactions on Audio and Electroacoustics*, 14(3):141–151.

Bay, M. and Beauchamp, J. W. (2012). Multiple-timbre fundamental frequency tracking using an instrument spectrum library. *The Journal of the Acoustical Society of America*, 132(3):1886.

Benetos, E. and Dixon, S. (2013). Multiple-instrument polyphonic music transcription using a temporally constrained shift-invariant model. *The Journal of the Acoustical Society of America*, 133(3):1727–1741.

Benetos, E., Dixon, S., Duan, Z., and Ewert, S. (2019). Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, 36(1):20–30.

Benetos, E., Dixon, S., Giannoulis, D., Kirchhoff, H., and Klapuri, A. (2013). Automatic music transcription: Challenges and future directions. *Journal of Intelligent Information Systems*, 41(3):407–434.

Benetos, E., Ewert, S., and Weyde, T. (2014). Automatic transcription of pitched and unpitched sounds from polyphonic music. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3107–3111, Florence, Italy. IEEE.

Bickel, S., Brückner, M., and Scheffer, T. (2009). Discriminative learning under covariate shift. *Journal of Machine Learning Research*, 10(9).

Bittner, R., McFee, B., Salamon, J., Li, P., and Bello, J. P. (2017). Deep salience representations for f0 estimation in polyphonic music. In *International Society for Music Information Retrieval Conference (ISMIR)*, volume 18, pages 63–70, Suzhou, China.

Bittner, R. M., Mcfee, B., and Bello, J. P. (2018). Multitask learning for fundamental frequency estimation in music. *arXiv e-prints*, pages 1–13.

Bosch, J., Janer, J., Fuhrmann, F., and Herrera, P. (2012). A comparison of sound segregation techniques for predominant instrument recognition in musical audio signals. In *International Society for Music Information Retrieval Conference (ISMIR)*, volume 13, pages 559–564, Porto, Portugal.

Bregman, A. S. (1990). *Auditory Scene Analysis: The Perceptual Organization Of Sound*. The MIT Press, Cambridge, UK.

Brock, A., Donahue, J., and Simonyan, K. (2019). Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations (ICLR)*, New Orleans, USA.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., Mccandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1877–1901.

Burred, J., Robel, A., and Sikora, T. (2010). Dynamic Spectral Envelope Modelling for Timbre Analysis of Musical Instrument Sounds. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):663–674.

Burred, J. J., Robel, A., and Sikora, T. (2009). Polyphonic musical instrument recognition based on a dynamic model of the spectral envelope. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 173 – 176, Taipei, Taiwan.

Chandna, P., Miron, M., Janer, J., and Gómez, E. (2017). Monoaural audio source separation using deep convolutional neural networks. In *International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, volume 13, pages 258–266, Grenoble, France. Springer.

Cherry, C. (1953). Some experiments on the recognition of speech, with one and with two ears. *Journal of the Acoustical Society of America*, 25(5):975–979.

Christensen, M. G., Stoica, P., Jakobsson, A., and Holdt Jensen, S. (2008). Multi-pitch estimation. *Signal Processing*, 88(4):972–983.

Clark, J. M., Robertson, P., and Luce, D. (1964). A preliminary experiment on the perceptual basis for musical instrument families. *Journal of Audio Engineering Society (AES)*, 12(3):199–203.

Cohen-Hadria, A., Roebel, A., and Peeters, G. (2019). Improving singing voice separation using deep U-Net and Wave-U-Net with data augmentation. In *European Signal Processing Conference (EUSIPCO)*, A Coruna, Spain. IEEE.

Comon, P. and Jutten, C. (2010). *Handbook of Blind Source Separation: Independent Component Analysis and Applications*. Academic Press, Inc., Orlando, USA, 1st edition.

Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., and Bharath, A. A. (2018). Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35:53–65.

Daumé III, H., Kumar, A., and Saha, A. (2010). Frustratingly easy semi-supervised domain adaptation. In *Workshop on Domain Adaptation for Natural Language Processing*, pages 53–59, Uppsala, Sweden. Association for Computational Linguistics (ACL).

Diment, A., Rajan, P., Heittola, T., and Virtanen, T. (2013). Modified group delay feature for musical instrument recognition. In *International Symposium on Computer Music Multidisciplinary Research*, volume 10, pages 431–438, Marseille, France.

Driedger, J., Müller, M., and Disch, S. (2014a). Extending harmonic-percussive separation of audio signals. In *International Society for Music Information Retrieval Conference (ISMIR)*, volume 15, pages 611–616, Taipei, Taiwan.

Driedger, J., Müller, M., and Ewert, S. (2014b). Improving time-scale modification of music signals using harmonic-percussive separation. *IEEE Signal Processing Letters*, 21(1):105–109.

Drossos, K., Magron, P., Mimilakis, S., and Virtanen, T. (2018). Harmonic-percussive source separation with deep neural networks and phase recovery. In *International Workshop on Acoustic Signal Enhancement (IWAENC)*, volume 16, pages 421–425, Tokyo, Japan. IEEE.

Drugman, T. and Alwan, A. (2011). Joint robust voicing detection and pitch estimation based on residual harmonics. In *Conference of the International Speech Communication Association (INTERSPEECH)*, Florence, Italy.

Duan, Z., Han, J., and Pardo, B. (2014). Multi-pitch streaming of harmonic sound mixtures. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 22(1):138–150.

Duan, Z., Pardo, B., and Zhang, C. (2010). Multiple fundamental frequency estimation by modeling spectral peaks and non-peak regions. *IEEE Transactions on Audio, Speech and Language Processing*, 18(8):2121–2133.

Eggert, J. and Korner, E. (2004). Sparse coding and NMF. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, volume 4, pages 2529–2533.

Essid, S., Leveau, P., Richard, G., Daudet, L., and David, B. (2005). On the usefulness of differentiated transient steady-state processing in machine recognition of musical instruments. In *Audio Engineering Society (AES) Convention*, volume 118, Barcelona, Spain.

Essid, S., Richard, G., and David, B. (2004). Musical instrument recognition on solo performances. In *European Signal Processing Conference*, volume 12, pages 1289–1292.

Essid, S., Richard, G., and David, B. (2006). Instrument recognition in polyphonic music based on automatic taxonomies. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(1):68–80.

Fan, Z. C., Lai, Y. L., and Jang, J.-S. R. (2018). SVSGAN: Singing voice separation via generative adversarial network. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 726–730, Calgary, Canada. IEEE.

Fernando, B., Habrard, A., Sebban, M., and Tuytelaars, T. (2013). Unsupervised visual domain adaptation using subspace alignment. In *IEEE International Conference on Computer Vision*, pages 2960–2967, Sydney, Australia. IEEE.

Févotte, C., Bertin, N., and Durrieu, J. L. (2009). Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis. *Neural Computation*, 21(3):793–830.

Fitzgerald, D. (2010). Harmonic/percussive separation using median filtering. In *International Conference on Digital Audio Effects (DAFx)*, volume 13, Graz, Austria.

FitzGerald, D., Cranitch, M., and Coyle, E. (2008). Extended nonnegative tensor factorisation models for musical sound source separation. *Computational Intelligence and Neuroscience*.

Fitzgerald, D., Liukus, A., Rafii, Z., Pardo, B., and Daudet, L. (2014). Harmonic/percussive separation using kernel additive modelling. In *IET Irish Signals Systems Conference and China-Ireland International Conference on Information and Communications Technologies (ISSC/CIICT)*, volume 25, pages 35–40, Limerick, Ireland.

Fitzgerald, D. and Paulus, J. (2006). Unpitched percussion transcription. In Klapuri, A. and Davy, M., editors, *Signal Processing Methods for Music Transcription*, chapter 5, pages 131–162. Springer, New York, USA, 1 edition.

Fletcher, H. and Munson, W. A. (1933). Loudness, its definition, measurement and calculation. *Journal of the Acoustical Society of America*, 5:82–108.

Fletcher, N. H. and Rossing, T. D. (1998). *The Physics of Musical Instruments*. Springer New York, 2 edition.

Fritsch, J. and Plumbley, M. D. (2013). Score informed audio source separation using constrained nonnegative matrix factorization and score synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 888–891.

Fuhrmann, F. (2012). *Automatic Musical Instrument Recognition From Polyphonic Music Audio Signals*. Phd thesis, Universitat Pompeu Fabra, Barcelona, Spain.

Ganin, Y. and Lempitsky, V. (2015). Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning (ICML)*, pages 1180–1189, Lille, France. JMLR.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35.

Gharib, S., Drossos, K., E. Çakir, C., Serdyuk, D., and Virtanen, T. (2018). Unsupervised adversarial domain adaptation for acoustic scene classification. In *Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, pages 138–142, Surrey, UK.

Girolami, M., editor (2000). *Advances in Independent Component Analysis*. Springer, London, UK.

Gkiokas, A., Katsouros, V., Carayannis, G., and Stajylakis, T. (2012). Music tempo estimation and beat tracking by applying source separation and metrical relations. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 37, pages 421–424, Kyoto, Japan.

Gong, B., Grauman, K., and Sha, F. (2013). Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *International Conference on Machine Learning (ICML)*, pages 222–230, Atlanta, USA. JMLR.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press, 1st edition.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Conference on Neural Information Processing Systems (NeurIPS)*, volume 28, pages 2672–2680, Montreal, Canada.

Grais, E., Wierstorf, H., Ward, D., and Plumbley, M. (2018). Multi-resolution fully convolutional neural networks for monaural audio source separation. In *International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, volume 10891 LNCS, pages 340–350, Guildford, UK. Springer.

Gururani, S., Sharma, M., and Lerch, A. (2019). An attention mechanism for musical instrument recognition. In *International Society for Music Information Retrieval Conference (ISMIR)*, Delft, Netherlands.

Gururani, S., Summers, C., and Lerch, A. (2018). Instrument activity detection in polyphonic music using deep neural networks. In *International Society for Music Information Retrieval Conference (ISMIR)*, Paris, France.

Hahnloser, R. H. R., Sarpeshkar, R., Mahowald, M. A., Douglas, R. J., and Seung, H. S. (2000). Digital Selection and Analogue Amplification Coexist in a Cortex-Inspired Silicon Circuit. *Nature*, 405(6789):947–951.

Han, S., Pool, J., Tran, J., and Dally, W. (2015). Learning both weights and connections for efficient neural network. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems NIPS*, volume 28, pages 1135–1143. Curran Associates, Inc.

Han, Y., Kim, J., and Lee, K. (2017). Deep convolutional neural networks for predominant instrument recognition in polyphonic music. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 25(1):208–221.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA. IEEE.

Heittola, T., Klapuri, A., and Virtanen, T. (2009). Musical instrument recognition in polyphonic audio using source-filter model for sound separation. In *International Society for Music Information Retrieval Conference (ISMIR)*, volume 10, pages 327–332, Kobe, Japan.

Hennequin, R., Khlif, A., Voituret, F., and Moussallam, M. (2020). Spleeter: a fast and efficient music source separation tool with pre-trained models. *Journal of Open Source Software*, 5(50):2154. Deezer Research.

Herrera-Boyer, P., Klapuri, A., and Davy, M. (2006). Automatic classification of pitched musical instrument sounds. In Klapuri, A. and Davy, M., editors, *Signal Processing Methods for Music Transcription*, chapter 6, pages 163 – 200. Springer, New York, USA, 1 edition.

Herrera-Boyer, P., Peeters, G., and Dubnov, S. (2003). Automatic classification of musical instrument sounds. *Journal of New Music Research*, 32(1):3–21.

Hopkins, P. (1978). *Aural Thinking in Norway: The Lore of the Harding Fiddle*. Phd thesis, University of Pennsylvania, Pennsylvania, USA.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4700–4708, Honolulu, Hawaii.

Huang, J., Smola, A., Gretton, A., Borgwardt, K. M., and Schölkopf, B. (2006). Correcting sample selection bias by unlabeled data. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 601–608, Vancouver, CA.

Huang, P., Kim, M., Hasegawa-Johnson, M., and Smaragdis, P. (2014). Deep learning for monaural speech separation. In *IEEE International Conference*

on Acoustics, Speech and Signal Processing (ICASSP), pages 1562–1566, Florence, Italy.

Humphrey, E. J., Durand, S., and Mcfee, B. (2018). OpenMIC-2018: An open dataset for multiple instrument recognition. In *International Society for Music Information Retrieval Conference (ISMIR)*, Paris, France.

Hung, Y. N., Chen, Y. A., and Yang, Y. H. (2019). Multitask learning for frame-level instrument recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 381–385, Brighton. UK. IEEE.

Hung, Y.-N. and Yang, Y.-H. (2018). Frame-level instrument recognition by timbre and pitch. In *International Society for Music Information Retrieval Conference (ISMIR)*, Paris, France.

Hyvarinen, A. (1999). Gaussian moments for noisy independent component analysis. *IEEE Signal Processing Letters*, 6:145–147.

Hyvarinen, A., Karhunen, J., and Oja, E. (2001). *Independent Component Analysis*. Wiley, New York, USA.

Isola, P., Zhu, J. Y., Zhou, T., and Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, Honolulu, USA.

Jansson, A., Humphrey, E., Montecchio, N., Bittner, R., Kumar, A., and Weyde, T. (2017). Singing voice separation with deep U-Net convolutional networks. In *International Society for Music Information Retrieval Conference (ISMIR)*, volume 18, Suzhou, China.

Jawaherlalnehru, G. and Jothilakshmi, S. (2019). Music instrument recognition from spectrogram images using convolution neural network. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, 8(9):1076–1079.

Kao, C.-C., Lee, T.-Y., Sen, P., and Liu, M.-Y. (2018). Localization-aware active learning for object detection. *arXiv e-prints*, pages 1–21.

Kim, J. W., Salamon, J., Li, P., and Bello, J. P. (2018). Crepe: A convolutional representation for pitch estimation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 161–165, Calgary, Canada. IEEE.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations*, volume 2, Banff, Canada.

Kitahara, T., Goto, M., Komatani, K., Ogata, T., and Okuno, H. G. (2007). Instrogram: Probabilistic representation of instrument existence for polyphonic music. *IPSJ Journal*, 48(1):214–226.

Kong, Q., Xu, Y., Wang, W., Jackson, P., and Plumbley, M. D. (2019). Single-channel signal separation and deconvolution with generative adversarial networks. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2747–2753, Macao, China.

Langner, G., Schreiner, C., and Biebel, U. (1998). *Functional Implications of Frequency and Periodicity Coding in the Auditory Midbrain*, pages 277–285. Whurr Publishers Ltd., London.

Lapp, D. R. (2003). *The Physics of Music and Musical Instruments*. Wright Center for Science Education (Tufts University), Medford, USA, 1st edition.

Le Roux, J., Wisdom, S., Erdogan, H., and Hershey, J. R. (2019). SDR - half-baked or well done? In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 626–630, Brighton, UK.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551.

Lee, D. D. and Seung, H. S. (1999). Learning the parts of objects by nonnegative matrix factorization. *Nature*, 401:788–791.

Lee, J. H., Choi, H.-S., and Lee, K. (2019). Audio query-based music source separation. In *International Society for Music Information Retrieval Conference (ISMIR)*, Delft, Netherlands.

Lee, T., Girolami, M., Bell, A., and Sejnowski, T. (2000). A unifying information-theoretic framework for independent component analysis. *Computers & Mathematics with Applications*, 39:1–21.

Levine, S. N. and Smith III, J. O. (1998). A sines+transients+noise audio representation for data compression and time/pitch scale modiications. In *Audio Engineering Society Convention*.

Li, H., Kadav, A., Durdanovic, I., Samet, H., and Graf, P. (2016). Pruning filters for efficient convnets. *ArXiv e-prints*, page arXiv:1608.08710.

Li, P., Qian, J., and Wang, T. (2015). Automatic instrument recognition in polyphonic music using convolutional neural networks. *arXiv e-prints*, page arXiv:1511.05520.

Li, Q. (2012). Literature survey: Domain adaptation algorithms for natural language processing. Technical report, Department of Computer Science, City University of New York, New York.

Lim, W. and Lee, T. (2017). Harmonic and percussive source separation using a convolutional auto encoder. In *European Signal Processing Conference (EUSIPCO)*, volume 25, pages 1804–1808, Kos Island, Greece. IEEE.

Liutkus, A., Fitzgerald, D., Rafii, Z., Pardo, B., and Daudet, L. (2014). Kernel additive models for source separation. *IEEE Transactions on Signal Processing*, 62(16):4298–4310.

Long, M., Wang, J., Ding, G., Sun, J., and Yu, P. S. (2013). Transfer feature learning with joint distribution adaptation. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2200–2207, Sydney, Australia. IEEE.

Long, M., Zhu, H., Wang, J., and Jordan, M. I. (2016). Unsupervised domain adaptation with residual transfer networks. In *Conference on Neural Information Processing Systems (NeurIPS)*, pages 136–144, Barcelona, Spain.

Lordelo, C., Ahlbäck, S., Benetos, E., Dixon, S., and Ohlsson, P. (2020a). Tap & Fiddle: A dataset with scandinavian fiddle tunes with accompanying foot-tapping. *Zenodo*.

Lordelo, C., Benetos, E., Dixon, S., and Ahlbäck, S. (2019). Investigating kernel shapes and skip connections for deep learning-based harmonic-percussive separation. In *Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, USA.

Lordelo, C., Benetos, E., Dixon, S., and Ahlbäck, S. (2021a). Pitch-informed instrument assignment using a deep convolutional network with multiple kernel shapes. In *International Society for Music Information Retrieval Conference (ISMIR)*.

Lordelo, C., Benetos, E., Dixon, S., Ahlbäck, S., and Ohlsson, P. (2021b). Adversarial unsupervised domain adaptation for harmonic-percussive source separation. *IEEE Signal Processing Letters*, 28:81–85.

Lordelo, C., Delgado, A., and Ramires, A. (2020b). Can transient/non-transient separation improve instrument recognition? In *MIP-Frontiers Workshop vol. 1*. Barcelona, Spain, April.

Loy, G. (2006). *Musimathics: The Mathematical Foundations of Music*. The MIT Press, 1st edition.

Luo, Y., Chen, Z., Hershey, J. R., Le Roux, J., and Mesgarani, N. (2017). Deep clustering and conventional networks for music separation: Stronger together. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 61–65, New Orleans, USA. IEEE.

Luo, Y. and Mesgarani, N. (2019). Conv-TasNet: surpassing ideal time-frequency magnitude masking for speech separation. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 27(8):1256–1266.

Maciejewski, M., Sell, G., Garcia-Perera, L. P., Watanabe, S., and Khudanpur, S. (2018). Building corpora for single-channel speech separation across multiple domains. *arXiv e-print:1811.02641*.

Makino, S. (2018). *Audio Source Separation*. Signals and Communication Technology. Springer International Publishing, 1 edition.

Manilow, E., Seetharaman, P., and Pardo, B. (2020). Simultaneous separation and transcription of mixtures with multiple polyphonic and percussive instruments. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 771–775, Barcelona, Spain.

Manilow, E., Wichern, G., Seetharaman, P., and Le Roux, J. (2019). Cutting music source separation some Slakh: A dataset to study the impact of training data quality and quantity. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 45–49, New Paltz, USA.

McAdams, S. (2013). Musical timbre perception. In Deutsch, D., editor, *The Psychology of Music*, chapter 2, pages 35–67. Elsevier Academic Press, San Diego, CA, 3rd edition.

McAdams, S., Winsberg, S., Donnadieu, S., De Soete, G., and Krimphoff, J. (1995). Perceptual scaling of synthesized musical timbres: Common dimensions, specificities, and latent subject classes. *Psychological Research*, 58(3):177–192.

Meng, Z., Li, J., Gong, Y., and Juang, B.-H. (2018). Adversarial feature-mapping for speech enhancement. In *Interspeech*, pages 3259–3263, Hyderabad, India. ISCA.

Menghani, G. (2021). Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *ArXiv e-prints*.

Meseguer-Brocal, G. and Peeters, G. (2019). Conditioned-U-Net: Introducing a control mechanism in the u-net for multiple source separations. In *International Society for Music Information Retrieval Conference (ISMIR)*, Delft, Netherlands.

Mimilakis, S. I., Drossos, K., Virtanen, T., and Schuller, G. (2017). A recurrent encoder-decoder approach with skip-filtering connections for monaural singing voice separation. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, volume 27, pages 1–7, Tokyo, Japan.

Mitsufuji, Y., Fabbro, G., Uhlich, S., and Stöter, F. R. (2021). Music demixing challenge 2021.

Müller, M. (2015). *Fundamentals of Music Processing - Audio, Analysis, Algorithms, Applications*. Springer International Publishing, Switzerland, 1 edition.

Muth, J., Uhlich, S., Perraudin, N., Kemp, T., Cardinaux, F., and Mitsufuji, Y. (2018). Improving DNN-based music source separation using phase features. In *Joint Workshop on Machine Learning for Music at ICML, IJCAI/ECAI and AAMAS*, Stockholm, Sweden.

Noguchi, A. and Harada, T. (2019). Image generation from small datasets via batch statistics adaptation. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2750–2758, Seoul, Korea.

Noh, H., Hong, S., and Han, B. (2015). Learning deconvolution network for semantic segmentation. In *International Conference on Computer Vision (ICCV)*, pages 1520–1528. IEEE.

Nugraha, A., Liutkus, A., and Vincent, E. (2016a). Multichannel music separation with deep neural networks. In *The European Signal Processing Conference (EUSIPCO)*, pages 1748–1752, Budapest, Hungary.

Nugraha, A. A., Liutkus, A., and Vincent, E. (2016b). Multichannel audio source separation with deep neural networks. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 24(9):1652–1664.

Oja, E. (2004). Applications of independent component analysis. In Pal, N. R., Kasabov, N., Mudi, R., Pal, S., and Parui, S. K., editors, *Neural Information Processing*, pages 1044–1051. Springer, Berlin, Germany.

Ong, Y. Z., Chui, C. K., and Yang, H. (2019). CASS: Cross adversarial source separation via autoencoder. *arXiv e-prints*.

Oquab, M., Bottou, L., Laptev, I., and Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1717–1724, Columbus, USA. IEEE Computer Society.

Osako, K., Mitsufuji, Y., Singh, R., and Raj, B. (2017). Supervised monaural source separation based on autoencoders. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11–15, New Orleans, USA. IEEE.

Pandey, A. and Wang, D. (2019). TCNN: Temporal convolutional neural network for real-time speech enhancement in the time domain. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6875–6879, Brighton. UK. IEEE.

Peebles, P. (2000). *Probability, Random Variables, and Random Signal Principles*. McGraw-Hill, New York, USA.

Peng, X., Bai, Q., Huang, Z., Saenko, K., and Wang, B. (2019). Moment matching for multi-source domain adaptation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea.

Petrovski, A., Azarov, E., and Petrovsky, A. (2011). Hybrid signal decomposition based on instantaneous harmonic parameters and perceptually motivated wavelet packets for scalable audio coding. *Signal Processing*, 91:1489–1504.

Phan, H., Hertel, L., Maass, M., and Mertins, A. (2016). Robust audio event recognition with 1-max pooling convolutional neural networks. In *Conference of the International Speech Communication Association (INTERSPEECH)*, pages 3653–3657, San Francisco, USA.

Pons, J., Lidy, T., and Serra, X. (2016). Experimenting with musically motivated convolutional neural networks. In *International Workshop on Content-Based Multimedia Indexing (CBMI)*, Bucharest, Romania. IEEE.

Pons, J. and Serra, X. (2017). Designing efficient architectures for modeling temporal features with convolutional neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 42, pages 2472–2476, New Orleans, USA. IEEE.

Pons, J., Slizovskaia, O., Gong, R., Gómez, E., and Serra, X. (2017). Timbre analysis of music audio signals with convolutional neural networks. In *European Signal Processing Conference (EUSIPCO)*, volume 25, pages 2744–2748, Kos Island, Greece.

Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N., editors (2008). *Dataset Shift in Machine Learning*. Neural Information Series. The MIT Press.

Rafii, Z., Duan, A., and Pardo, B. (2014). Combining rhythm-based and pitch-based methods for background and melody separation. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 22(12):1884–1893.

Rafii, Z., Liutkus, A., Stöter, F. R., Mimilakis, S. I., and Bittner, R. (2017). The MUSDB18 corpus for music separation.

Rafii, Z., Liutkus, A., Stöter, F. R., Mimilakis, S. I., FitzGerald, D., and Pardo, B. (2018). An overview of lead and accompaniment separation in music. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 26(8):1307–1335.

Rafii, Z. and Pardo, B. (2013). REpeating Pattern Extraction Technique (REPET): A simple method for music/voice separation. *IEEE Transactions on Audio, Speech and Language Processing*, 21(1):73–84.

Redko, I., Habrard, A., Morvant, E., Sebban, M., and Bennani, Y. (2019). *Advances in Domain Adaptation Theory*. Elsevier.

Rienstra, S. W. and Hirschberg, A. (2021). *An Introduction to Acoustics*. Eindhoven University of Technology, Eindhoven, Netherlands.

Robinson, D. W. and Dadson, R. S. (1956). A re-determination of the equal-loudness relations for pure tones. *British Journal of Applied Physics*, 7(5):181.

Roma, G., Green, O., and Tremblay, P. A. (2018a). Improving single-network single-channel separation of musical audio with convolutional layers. In *International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, volume 14, pages 306–315, Guildford, UK. Springer.

Roma, G., Green, O., and Tremblay, P. A. (2018b). Stationary/transient audio separation using convolutional autoencoders. In *International Conference on Digital Audio Effects (DAFx)*, pages 65–71, Aveiro, Portugal.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 18, pages 234–241, Munich, Germany.

Rouard, S., Massa, F., and Défossez, A. (2023). Hybrid transformers for music source separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, Rhodes Island, Greece.

Ruder, S. (2017). An overview of multi-task learning in deep neural networks. *arXiv e-prints*.

Salamon, J., Gómez, E., Ellis, D., and Richard, G. (2014). Melody extraction from polyphonic music signals: Approaches, applications and challenges. *IEEE Signal Processing Magazine*, 31(2):118–134.

Sethares, W. A. (2005). *Tuning, Timbre, Spectrum, Scale*. Springer, London, UK, 2nd edition.

Shimodaira, H. (2000). Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244.

Shiomi, H. (2021). Anti-collision of RFID tags with blind DS-CDMA using ICA. In *International Symposium on Antennas and Propagation (ISAP)*, pages 47–48.

Simmermacher, C., Deng, D., and Cranefield, S. (2006). Feature analysis and classification of classical musical instruments: An empirical study. In Perner, P., editor, *Advances in Data Mining. Applications in Medicine, Web Mining, Marketing, Image and Signal Mining*, pages 444–458, Berlin, Heidelberg. Springer Berlin Heidelberg.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations(ICLR)*, San Diego, USA.

Sofianos, S., Ariyaeeinia, A., Polfreman, R., and Sotudeh, R. (2012). H-semantics: A hybrid approach to singing voice separation. *Journal of the Audio Engineering Society*, 60(10):831–841.

Solanki, A. and Pandey, S. (2019). Music instrument recognition using deep convolutional neural networks. *International Journal of Information Technology*, pages 1–10.

Southall, C., Stables, R., and Hockman, J. (2017). Automatic drum transcription for polyphonic recordings using soft attention mechanisms and convolutional neural networks. In *International Society for Music Information Retrieval Conference (ISMIR)*, pages 606–612, Suzhou, China.

Stoller, D., Ewert, S., and Dixon, S. (2018a). Adversarial semi-supervised audio source separation applied to singing voice extraction. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2391–2395, Calgary, Canada. IEEE.

Stoller, D., Ewert, S., and Dixon, S. (2018b). Jointly detecting and separating singing voice: A multi-task approach. In *International Conference on Latent*

Variable Analysis and Signal Separation (LVA/ICA), volume 14, pages 329–339, Guildford, UK. Springer.

Stoller, D., Ewert, S., and Dixon, S. (2018c). Wave-U-Net: A multi-scale neural network for end-to-end audio source separation. In *International Society for Music Information Retrieval Conference (ISMIR)*, volume 19, Paris, France.

Stöter, F. R., Liutkus, A., and Ito, N. (2018). The 2018 signal separation evaluation campaign. In *International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, volume 14, pages 293–305, Guildford, UK. Springer.

Stöter, F.-R., Uhlich, S., Liutkus, A., and Mitsufuji, Y. (2019). Open-Unmix - a reference implementation for music source separation. *Journal of Open Source Software*, 4(41).

Su, J.-C., Tsai, Y.-H., Sohn, K., Liu, B., Maji, S., and Chandraker, M. (2020). Active adversarial domain adaptation. In *Winter Conference on Applications of Computer Vision (WACV)*, Colorado, USA. IEEE.

Sun, S., Zhang, B., Xie, L., and Zhang, Y. (2017). An unsupervised deep domain adaptation approach for robust speech recognition. *Neurocomputing*, 257:79–87.

Sønderby, C. K., Caballero, J., Theis, L., Shi, W., and Huszár, F. (2017). Amortised map inference for image super-resolution. In *International Conference on Learning Representations (ICLR)*, Toulon, France.

Takahashi, N., Agrawal, P., Goswami, N., and Mitsufuji, Y. (2018a). PhaseNet: Discretized phase modeling with deep neural networks for audio source separation. In *Conference of the International Speech Communication Association (INTERSPEECH)*, pages 2713–2717, Hyderabad, India.

Takahashi, N., Goswami, N., and Mitsufuji, Y. (2018b). MMDenseLSTM: An efficient combination of convolutional and recurrent neural networks for audio source separation. In *International Workshop on Acoustic Signal Enhancement (IWAENC)*, volume 16, pages 106–110, Tokyo, Japan. IEEE.

Takahashi, N. and Mitsufuji, Y. (2017). Multi-scale multi-band DenseNets for audio source separation. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 21–25, New Paltz, USA.

Takahashi, N. and Mitsufuji, Y. (2021). D3Net: Densely connected multidilated densenet for music source separation. *ArXiv e-prints*.

Tanaka, K., Nakatsuka, T., Nishikimi, R., Yoshii, K., and Morishima, S. (2020). Multi-instrument music transcription based on deep spherical clustering of spectrograms and pitchgrams. In *International Society for Music Information Retrieval Conference (ISMIR)*, Montreal, Canada.

Tang, H., Liu, H., Xu, D., Torr, P. H. S., and Sebe, N. (2020). Attention-gan: Unpaired image-to-image translation using attention-guided generative adversarial networks. *arXiv e-prints*.

Tang, H., Xu, D., Sebe, N., and Yan, Y. (2019). Attention-guided generative adversarial networks for unsupervised image-to-image translation. In *International Joint Conference on Neural Networks (IJCNN)*, Budapest, Hungary.

Thickstun, J., Harchaoui, Z., and Kakade, S. M. (2017). Learning features of music from scratch. In *International Conference on Learning Representations (ICLR)*.

Thomé, C. and Ahlbäck, S. (2017). Polyphonic pitch detection with convolutional recurrent neural networks. In *Music Information Retrieval Evaluation eXchange (MIREX)*.

Uhlich, S., Porcu, M., Giron, F., Enenkl, M., Kemp, T., Takahashi, N., and Mitsufuji, Y. (2017). Improving music source separation based on deep neural networks through data augmentation and network blending. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 261–265, New Orleans, USA. IEEE.

Vincent, E., Gribonval, R., and Févotte, C. (2006). Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech and Language Processing*, 14(4):1462–1469.

Virtanen, T. (2007). Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *IEEE Transactions on Audio, Speech and Language Processing*, 15(3):1066–1074.

Wang, Y., Gonzalez-Garcia, A., Berga, D., Herranz, L., Khan, F. S., and van de Weijer, J. (2019). MineGAN: Effective knowledge transfer from GANs to target domains with few images. *arXiv e-prints*.

Wang, Y., Wu, C., Herranz, L., van de Weijer, J., Gonzalez-Garcia, A., and Raducanu, B. (2018). Transferring GANs: Generating images from limited data. In *European Conference on Computer Vision (ECCV)*, volume 11210 LNCS, pages 220–236, Munich, Germany. Springer International Publishing.

Wei, W., Zhu, H., Benetos, E., and Wang, Y. (2020). A-CRNN: A domain adaptation model for sound event detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 276–280, Barcelona, Spain.

Weninger, F., Le Roux, J., Hershey, J. R., and Watanabe, S. (2014). Discriminative nmf and its application to single-channel source separation. In *Conference of the International Speech Communication Association (INTERSPEECH)*, pages 865–869.

Wu, C., Dittmar, C., Southall, C., Vogl, R., Widmer, G., Hockman, J., Müller, M., and Lerch, A. (2018). A review of automatic drum transcription. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 26(9):1457–1483.

Wu, Y.-T., Chen, B., and Su, L. (2019). Polyphonic music transcription with semantic segmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, UK.

Wu, Y.-T., Chen, B., and Su, L. (2020). Multi-instrument automatic music transcription with self-attention-based instance segmentation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 28:2796–2809.

Xiao, P., Du, B., Wu, J., Zhang, L., Hu, R., and Li, X. (2018). TLR: Transfer latent representation for unsupervised domain adaptation. In *IEEE International Conference on Multimedia and Expo (ICME)*, pages 0–5, San Diego, USA. IEEE.

Yoshii, K., Goto, M., and Okuno, H. G. (2007). Drum sound recognition for polyphonic audio signals by adaptation and matching of spectrogram templates with harmonic structure suppression. *IEEE Transactions on Audio, Speech and Language Processing*, 15(1):333–345.

Yost, W. (2009). Pitch perception. *Attention, Perception, & Psychophysics*, 71:1701–1715.

Zhou, W. and Zhang, Y. (2017). Speech separation in multi-channel by ICA and in monaural by deep recurrent neutral networks. Technical report, University of Rochester - ECE Department - ECE 477 Course, Rochester, USA.