Musical Source Separation with Deep Learning and Large-Scale Datasets

Andreas Jansson

November, 2019

Acknowledgments

I would like to thank my past and present colleagues and friends in the "MIQ" team at Spotify: Tristan Jehan, Nicola Montecchio, Rachel Bittner, Sebastian Ewert, Eric Humphrey, Simon Durand, Keunwoo Choi, Thor Kell, David Rubinstein, Brian Brost, and Andrew Demetriou — I am privileged to have worked with such brilliant people for the past several years. Thanks to Spotify for letting me pursue a part-time PhD that allowed me to work on projects that overlapped with my academic interests.

I would also like to thank City University, London, and specifically my PhD supervisor Tillman Weyde for having confidence in me and pushing me to continue working on this thesis to completion. Thanks to my colleagues in the City University Music Informatics Research Group: Reinier de Valk, Daniel Wolff, Srikanth Cherla, Emmanouil Benetos, and Andy Elmsley.

Finally, thanks to my family and friends who have supported me with encouragement and advice during these past seven years.

Contents

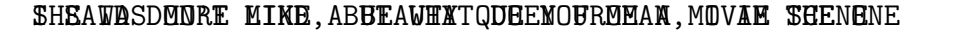
List of Figures

Chapter 1

Introduction

As humans, we have an astonishing ability to focus our auditory attention to specific objects or events. For example, when we attend parties where many guests are talking simultaneously, we are able to selectively listen to certain individuals, while ignoring the surrounding noise from the other guests. And when we listen to music, we hear distinct instruments, even though they are mixed together to a single track. All that our ears receive is a continuous stream of sound waves, the sum of all currently active sounds. It is up to our brains to disentangle this chaos into a multitude of perceived concurrent sources. And not only that, we also have to piece together sounds temporally by consistently attributing sounds at different instants of time to the same sources.

To illustrate the difficulty of this task, consider the following two lines from a famous pop song, overlaid on a single line:



This is analogous to the mixture sound wave that our ears receive when two party guests are talking. In order to untangle this jumble of characters, we have to both figure out which two characters are overlaid at each vertical character position, and which characters go together horizontally to form words and sentences. In the same way, when we hear a musical recording of two instruments playing together, we have to both determine which sonic

events belong to which instrument at each point in time, as well as follow the two instruments over time to form coherent melodic lines.

The answer to the riddle above is this:

```
SHE WAS MORE LIKE A BEAUTY QUEEN FROM A MOVIE SCENE
I SAID DON'T MIND, BUT WHAT DO YOU MEAN, I AM THE ONE
```

While our eyes have not developed the ability to easily disentangle overlapping characters, if we were presented with a recording of these two lines of text read aloud simultaneously, we would likely be able to separate the two sources in our mind. This process, when emulated mathematically and in software, is known as *Blind Source Separation* (BSS). It is "blind" since we do not have any information about the sources other than the signals themselves. In the real world, non-blind source separation could correspond to visually seeing the two speakers while they read the words, seeing their lips move, etc. In blind source separation we are deprived of any non-sonic cues. While non-blind source separation using multi-modal approaches is an active field of research, it is outside the scope of this thesis. Therefore we will use the term "source separation" to refer to blind source separation explicitly for the remainder of this document, unless explicitly state otherwise.

Furthermore, we will investigate a subset of source separation concerned with music: *Musical Source Separation* (MSS). Musical source separation differs from other types of source separation in several ways. In speech separation, speakers tend to talk at their own pace, independent of each other. However, when several people play music together in a group, they specifically attempt to play at the same pace, or tempo. Speakers tend to have different vocal pitch ranges, but musicians often play the same notes in unison or in close harmony. This makes musical source separation a particularly challenging task.

Why then would we invest time and effort to try to solve source separation in general, and specifically musical source separation? General source separation has many proven real-world applications, from separating radar sources, to mother vs. fetus heartbeat separation, hearing aids, etc.(Deville, Jutten, & Vigario, 2010). Products and tools based on musical source separation have yet to be developed and deployed at scale, but there are many promising av-

enues. For example, music education could greatly benefit from the ability to mute and solo individual instruments in arbitrary recordings. Source separation could enable re-mixing of existing musical recordings to better balance instrument levels, or to replace instruments or re-record instruments. One could imagine a future where the listener can control the levels of individual instruments in real-time, or where music software intelligently mixes music to the personal tastes of the listener.

Vocal source separation has the potential to revolutionize the karaoke industry, through the ability to attenuate or remove the vocals from any recording. It could also enable "gamification" in karaoke applications, e.g. by awarding points to performances that are more similar to the original.

Source separation can also simplify the time-consuming task of transcription. When a human transcriber manually transcribes polyphonic music to score, they first have to intently and carefully listen for the source to be transcribed. Transcribing from individual sources is both faster and less error-prone.

The same is true for automatic transcription systems. In fact, many common tasks in Music Information Retrieval (MIR) could benefit from using source separation as a preprocessing step. Automatic drum transcription and vocal pitch transcription systems are likely to yield more accurate models when presented with solo drums and solo vocals; chord detection systems may improve in the absence of drums; automatic lyric transcription is very difficult task when the input is musical mixtures, but could be achievable on solo vocals, etc. Conversely, we believe that semi-blind source separation, where attributes such as beat positions or pitch salience are available, could show improvements over purely blind source separation. This "chicken-and-egg" problem could be overcome by advances in multi-task learning, combining source separation and other MIR tasks in jointly optimized machine learning models.

It is also conceivable that source separation could play a major role in future generations of creative tools. Sampling, re-mixing, "mash-ups", etc., have become commonplace in modern music, thanks to innovations in hardware and software such as the Akai MPC60 and Ableton Live. Yet these tools are limited to the multi-source content that is present in existing recordings. Automatic source separation could enable creative uses of extracted instrument and vocal tracks from existing music, launching entirely new forms of musical expression.

1.1 Thesis outline

Throughout this thesis we will explore automatic music source separation by utilizing modern (at the time of writing) techniques and tools from machine learning and big data processing. The bulk of this work was carried out between 2016 and 2019.

In Chapter 2 we conduct a review of source separation literature. We start by outlining a subset of applications of source separation in some depth. We describe some of the early, pioneering work in automatic source separation: Auditory Scene Analysis, and its digital counterpart, Computational Auditory Scene Analysis.

We then introduce matrix decomposition-based methods such as Independent Component Analysis and Non-Negative Matrix factorization, and pitch-informed methods where the separation algorithm is guided by pitch information that is known *a priori*. We briefly discuss user-guided methods, before conducting a thorough review of Deep Learning-based source separation, including recurrent, convolutional, deep clustering-based, and Generative Adversarial Networks.

We then proceed to describe common evaluation metrics and training datasets. Finally, we list a number of current challenges and drawbacks of current systems.

Chapter 3 focuses on datasets for musical source separation. First we show the growth of dataset sizes for both machine learning in general and music information retrieval specifically. We give several examples of the complexities and idiosyncrasies that are intrinsic to music datasets. We then proceed to present a method for extracting ground truth data for source separation from large unstructured musical catalogs.

In Chapter 4 we design a novel deep learning-based source separation algorithm. Motivation is provided by means of a musicological study[1] that showed the high importance of vocals relative to other musical factors, in the minds of listeners. At the core of the vocal separation algorithm is the *U-Net*, a deep learning architecture that uses skip connections to preserve fine-grained detail. It was originally developed in the biomedical imaging

[1]This work was carried out in collaboration with Andrew Demetriou, PhD candidate at Delft University.

domain, and later adapted to image-to-image translation. We adapt it to the source separation domain by treating spectrograms as images, and we use the dataset mining methods from Chapter 3 to generate sufficiently large training data. We evaluate our model objectively using standard evaluation metrics, subjectively using "crowdsourced" human subjects. To the best of our knowledge, this is the first use of U-Nets for source separation.

In the introduction above we proposed joint learning to optimize source separation and other objectives. In Chapter 5 we investigate one such instance: multi-task learning of vocal removal and vocal pitch tracking. We combine the vocal separation model from Chapter 4 with a state of the art pitch salience estimation model[2], exploring several ways of combining the two models. We find that vocal pitch estimation benefits from joint learning when the two tasks are trained in sequence, with the source separation model preceding the pitch estimation model. We also report benefits from fine-tuning by iteratively applying the model.

Chapter 6 extends the U-Net model to multiple instruments. In order to minimize the phase artifacts that were a common issue in Chapter 4, we modify the model to operate in the complex domain. We run experiments with several loss functions: Time-domain loss, magnitude-only frequency-domain loss, and joint time and frequency-domain loss. Our experiments are evaluated both objectively and subjectively, and we carry out extensive qualitative analysis to investigate the effects of complex masking.

Finally, we conclude the thesis in Chapter 7 by summarizing this work and highlighting several future directions of research.

1.2 Notation

1.2.1 Notational conventions

Throughout this document we will use the following conventions:

Example	Name	Description
x	Lowercase italic	Scalar values
$\mathbf{x}$	Lowercase bold	Vector
$\mathbf{X}$	Uppercase bold	Matrix

[2]The pitch salience model was created by Dr. Rachel Bittner.

$X_{i,j}$	Uppercase italic with subscript	Single matrix element

1.2.2 Symbols

We will use these symbols consistently:

Notation	Description
$\mathbf{x}$	Mixture audio signal
$\mathbf{y}$	Single source audio signal
$\mathbf{X}$	Mixture spectrogram
$\mathbf{Y}$	Single source spectrogram
$\mathbf{S}$	f_0 salience produced by Deep Salience
$\hat{\mathbf{y}}$	Model estimate of single source audio signal
$\hat{\mathbf{Y}}$	Model estimate of single source spectrogram
$\hat{\mathbf{S}}$	Model estimate of f_0 salience, S_v
$\mathbf{M}$	Soft ratio mask
$\hat{\mathbf{O}}$	Final neural network layer output

1.2.3 Abbreviations

These abbreviations are used in this thesis:

Abbreviation	Meaning
AI	Artificial Intelligence
API	Application Programming Interface
ASA	Auditory Scene Analysis
BLSTM	Bidirectional Long Short-Term Memory
BSS	Blind Source Separation
CASA	Computational Auditory Scene Analysis
cIRM	Complex Ideal Ratio Mask
CNN	Convolutional Neural Network
DL	Deep Learning
DNN	Deep Neural Network
f_0	Fundamental Frequency
FFT	Fast Fourier Transform
FMA	Free Music Archive
GAN	Generative Adversarial Network
GFS	Google File System

GMM	Gaussian Mixture Model
GRU	Gated Recurrent Unit
HCQT	Harmonic Constant-Q Transform
HPSS	Harmonic / Percussive Source Separation
IBM	Ideal Binary Mask
ICA	Independent Component Analysis
IRM	Ideal Ratio Mask
ISA	Independent Subspace Analysis
ISMIR	International Society for Music Information Retrieval
ISTFT	Inverse Short-Time Fourier Transform
LSTM	Long Short-Term Memory
MIR	Music Information Retrieval
MIREX	Music Information Retrieval Evaluation eXchange
MPI	Message Passing Interface
MSS	Musical Source Separation
NMF	Non-negative Matrix Factorization
NSDR	Normalized Signal-to-Distortion Ratio
OA	Overall Accuracy
PCA	Principal Component Analysis
RAID	Redundant Array of Inexpensive Disks
RAM	Random Access Memory
RBM	Restricted Boltzmann Machine
RNN	Recurrent Neural Network
RPA	Raw Pitch Accuracy
RPCA	Robust Principal Component Analysis
ReLU	Rectified Linear Unit
SAR	Signal-to-Artifacts Ratio
SDR	Signal-to-Distortion Ratio
SIR	Signal-to-Interference Ratio
SiSEC	Signal Separation Evaluation Campaign
SQL	Structured Query Language
SSD	Solid State Drive
STFT	Short-time Fourier Transform
STOI	Short-Time Objective Intelligibility
T-F	Time-Frequency
VAD	Vocal Activity Detection

1.2.4 *Pluralis modestiae*

Throughout this thesis I use the first-person plural pronoun "we" to describe my own work, unless explicitly stated otherwise.

Chapter 2

Literature Review

2.1 Mathematical preliminaries

2.1.1 Time-domain Audio Signals

Sound is made from fluctuations in air pressure. Digitally, this is represented as a stream of numbers, usually ranging between -1 and 1, corresponding to the displacement of air at discrete instances in time. We notate this *audio signal* x_t, where t is the time step. In vector notation we denote $\mathbf{x} = x_{0...t}$.

Digital signals can be *sampled* at different frequencies. The *sample rate* of a signal is defined as the number of observations, or *samples*, are recorded per second. The sample rate determines the frequency range of the signal: the highest frequency that can be recovered (also referred to as the *Nyquist frequency*) is half of the sample rate.

2.1.2 The Discrete Fourier Transform

Any audio signal can be represented as a sum of sinusoids at various frequencies, phase offsets, and magnitudes. This representation can be derived through the Fourier transform. The focus of this thesis is on *digital* signal processing, so we will limit our discussion to the Discrete Fourier Transform (DFT).

The DFT is defined as

$$\text{DFT}(x)_k := \sum_{n=0}^{N-1} x_n e^{\frac{-i2\pi kn}{N}}, k \in \{0...N-1\} \tag{2.1}$$

where k is the *coefficient* index, n is the time step, N is the total number of time steps, and i is the imaginary unit.

The frequency of each coefficient k is dependent on both the sample rate and the number of time steps N:

$$f_k = \frac{kR}{N}, k \in \left\{0...\frac{N}{2}-1\right\} \tag{2.2}$$

where R is the sample rate. When the signal x is real (as opposed to complex), the second half of the DFT is the mirror conjugate of the first half and contains no additional information. For that reason, we disregard the second half and only use the coefficients $k \in \left\{0...\frac{N}{2}-1\right\}$. Frequencies are spaced equally from 0 Hz to the Nyquist frequency, $\frac{R}{2}$, where the 0 Hz component represents a constant offset.

2.1.3 The Short-Time Fourier Transform

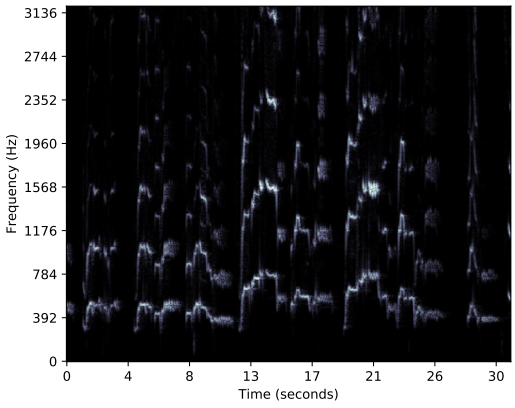


Figure 2.1: STFT example

In many applications we are interested in the frequency response of a signal over time. The Short-Time Fourier Transform (STFT) computes the DFT

over short windows of the signal, resulting in a matrix of frequency bins and time frames.

$$\text{STFT}(x)_{n,k} := \text{DFT}(w(x_{hn\ldots(hn+N)}))_k \tag{2.3}$$

where t is the time STFT frame, k is the DFT bin, $w(\cdot)$ is a window function (e.g. the Hamming or Hann window), and h is the hop length between windows..

Like the DFT, the STFT is complex-valued. In polar coordinates, the magnitude component of the STFT represents the amplitude of various frequencies at each time step, and the phase component describes the phase offset of each frequency. In musical applications it is common to discard the phase component and only employ the STFT magnitude component. This is also true in musical source separation, but as we shall see in Section 2.4.9, several examples exist of phase-aware source separation models.

2.1.4 Fundamental frequencies and harmonics

The saxophone spectrogram in Figure 2.1 has a number of parallel "lines" at roughly equal spacing, decreasing in intensity as the frequency increases. At first glance it may appear that this is a spectrogram of multiple instruments, but is in fact a natural phenomenon that occurs in all pitched musical instruments to varying degrees. The *timbre* of an instrument is defined by these *overtones* — frequencies that resonate with the *fundamental frequency* at different amplitudes. When we talk about the "pitch" of a sound, we actually refer to the fundamental frequency (commonly denoted f_0) of the sound.

2.2 Musical Source Separation

As was mentioned in the introduction, musical source separation is not entirely analogous to speech separation. However, our description of speech separation was somewhat simplified. In fact, the human voice has several characteristics that complicates the separation of voices (Diehl, 2008).

For example, human voices have a wide range of vocal registers. Studies have shown that it is easier to separate a female voice when the interfering voices

are male than it is to separate a female voice from other female voices, and vice versa (Darwin, Brungart, & Simpson, 2003).

This would suggest that separation of different musical instruments should be a simpler problem than separation of different speakers. A saxophone and a tambourine have completely different timbres, pitch ranges, amplitude envelopes, etc.

However, there are several complicating factors. While different speakers pace their words differently, musicians do their best to play in time with each other. The result is that note onsets often coincide, causing transients to be masked.

Similar masking issues arise in the frequency domain. In polyphonic music, the same notes are often played by several instruments. A guitar might play a G major chord (G/B/D) while a singer sustains a D note. We now have to listen for several sources at the same fundamental pitch.

Timbral masking is yet another complicating feature of music. While a saxophone and a tambourine sound very different, an alto saxophone and a distorted solo electric guitar occupy similar frequency ranges, and can sound almost identical.

2.3 Musical Applications

Applications of musical source separation can broadly be categorized as user-facing systems and as components of higher-level systems. In the following sections we present several examples of both categories.

2.3.1 User-facing Applications

Perhaps the most obvious application of source separation in music is Karaoke. The worldwide Karaoke industry has an estimated annual value of $10 billion US Dollars[1]. Popular Karaoke providers Sound Choice and Sunfly reportedly own catalogs of 16,500[2] and 18,000[3] tracks respectively, re-recorded by professional musicians employed by these companies. Since 2015, Sunfly has

[1] http://www.prweb.com/releases/2017/07/prweb14507690.htm
[2] https://pep.rocks/pep, retrieved December 2018
[3] https://www.sunflykaraoke.com/about-sunfly-karaoke, retrieved December 2018

grown their catalog by approximately 4,000 songs[4]. This means that, on average, Sunfly records, mixes, masters, and produces Karaoke videos for circa five tracks per day.

The Spotify music streaming service hosts a catalog in excess of 40 million songs (as of December, 2018[5]). Listening patterns on the popular streaming platform is heavily skewed to popular music, with 80% of streams come from the most popular 5% songs (TechDirt, 2010). With 40 million songs on the platform, that corresponds to around 200,000 songs. Sunfly's ability to generate five tracks per day is an impressive feat, but it is clear that manual ways of creating Karaoke content can not scale past the most mainstream top of the worldwide music catalog. Niche audiences in the "long tail"(Brynjolfsson, Hu, & Smith, 2006) stand little hope of being able to perform their favorite songs in a Karaoke venue.

Automatic source separation could revolutionize the Karaoke industry, letting users automatically remove or attenuate the vocals from any piece of recorded music. In addition to greater scale, the automatically generated Karaoke songs would sound just like the originals (except the vocals), as opposed to being re-recordings.

Source separation could also add additional layers of interactivity, by analyzing the original vocals, and compare them to the vocals sung by the karaoke performer. Pitch-tracking features that award points for singing correct pitches are already part of the popular Smule *Sing! Karaoke* application[6].

In a similar vein, the *Guitar Hero* suite of video games[7] let the user take the role of the guitarist (or other instrumentalist) of a rock band, who virtually performs in front of a live audience. The guitar part of a track is mapped to a sequence of key presses that the player must perform on a custom guitar-shaped game controller. As the player moves around the virtual stage, the game mixes the sounds of the other instrument to make the illusion seem even more realistic. Source separation combined with automatic pitch tracking should be able to produce similar experiences, but for any available track.

[4]Archived web site from October, 2015: https://web.archive.org/web/20151024223224/https://www.sunflykaraoke.com/about-sunfly-karaoke

[5]https://newsroom.spotify.com/company-info/, retrieved December 2018

[6]https://www.smule.com/

[7]https://www.guitarhero.com/

The term "edu-tainment" has become a popular way to describe educational tools that take inspiration from video games. This style of teaching is also making inroads in music education, for example with tools like Yousician[8]. Automatic source separation could enable musical instruments to be taught by isolating individual instruments from pieces that are already familiar to the student.

2.3.2 Using Separated Sources in Higher-level Systems

Musical source separation is one of many subtasks in the field of Music Information Retrieval (MIR). At the highest level, MIR is concerned with the automatic analysis, classification, and generation of music. The various tasks of MIR, e.g. chord recognition, fundamental frequency estimation, mood estimation, etc., are often tackled in isolation. It has long been argued that we, as a community, should take a more holistic approach, and let outputs of one task become inputs to other tasks, or even that several tasks could be approached in tandem (Liem, Müller, Eck, Tzanetakis, & Hanjalic, 2011). Source separation can be considered a fundamental building block, on top of which systems can be built for other MIR tasks.

For example, vocal pitch estimation should benefit from vocal source separation; automatic drum transcription from an isolated drum track should be a simpler problem than drum transcription from a mixture; chord estimation results might improve if the inputs are separated guitar and bass tracks, etc.

Individual source separated instrument tracks could also be used as inputs to music generation systems. "Mashups" are a genre of music in which the accompaniment from one song is combined with the vocals from a different song, resulting in surprising and often musically appealing outputs.[9]

Isolated instruments could also be used as training data for algorithmic composition and music synthesis, where the generative model is optimized to approximate the sound and musical qualities of existing, human-created music. It is arguably easier for a machine learning system to learn representations for individual instruments, than the greatly varying mixture.

[8] https://yousician.com

[9] See for example DJ Danger Mouse's "The Grey Album", that combines the vocals from Jay-Z's "Black Album" with The Beatles' "White Album".

Similarly, auto-accompaniment systems ("Band-in-a-box") can be learned in a leave-one-out fashion. An automatic accompaniment for guitar could be trained on source separated outputs to predict the most likely mixture of bass and percussion that best matches a solo guitar performance.

2.4 Existing Systems for Automatic Source Separation

2.4.1 Auditory Scene Analysis

A music source separator can be seen as an automatic "listening machine". The goal is to build a machine that is able to perceive sound like humans do, where mixtures of sources are consistently and coherently grouped. In order to build computational models of sound perception, we may want to take inspiration from physiological and neurological model of how humans recognize sound.

In 1990, as a result of decades of extensive research, Arthur Bregman proposed "Auditory Scene Analysis" (ASA) as a framework to understand how the brain processes incoming sound and creates the perception of distinct sound sources (Bregman, 1990). ASA draws heavily from Gestalt theory, a field of psychology that attempts to explain how our brain discerns patterns and groups visual objects into coherent groupings. Gestalt theory presents a number of principles for visual grouping: Proximity, similarity (of color, size, etc.), common fate (e.g. shared direction), continuity (the tendency to see continued structures, even in the presence of overlapping objects), closure (the ability to complete partial shapes), symmetry, parallelism, and disjoint allocation (visual elements belong only to single objects) (Wagemans et al., 2012).

Many similar effects can be found in sound perception (Deutsch, 1999). For example, a single monophonic melody played one note at a time is heard as a single source as long as the notes are close in pitch and time. If notes are added that are distant in pitch, we no longer hear a single source but multiple. The principle of continuity can be observed in that we hear the original melody continue "past" the outlier note. If we listen to the melody on headphones, and randomly assign notes left and right, we tend to group notes into multiple sources by spatial proximity.

These are examples of what Bregman refers to as *sequential integration*, the ability to attribute sequential sounds to the same source. But we are also able to perceive several sounds at the same instant in time, an effect called *simultaneous (spectral) integration* in the ASA literature. This is believed to be influenced by a number of factors, e.g. proximity of pitch, similarity of timbre and spatial origin. Many of these factors are themselves the result of complex processes. For example, in order to perceive proximity in pitch, the auditory system has to "explain away" overtones that are near multiples of the same fundamental frequency, collapsing the perception of overtones into a single frequency. Bregman refers to this effect as the *principle of harmonicity*.

The common fate principle can also be observed during spectral integration. Harmonics that belong to the same source tend to synchronously fluctuate in similar patterns, e.g. a violin with vibrato.

Gestalt-like grouping principles of auditory perception have been observed in infants (Demany, 1982) as well as animals (Fishman, Arezzo, & Steinschneider, 2004), and are sometimes collectively referred to as *primitive auditory scene analysis*. We also develop our ability to discern sources through learning over time. As we grow up, there is evidence that we internalize "schemas" or templates of sounds and patterns. These schemas can be single timbres (the sound of a spoken vowel), or longer sequences of sounds or musical notes. For example, it has been shown that familiar melodies are more easily heard as individual sources than unfamiliar melodies, in the presence of interfering noise (Dowling, 1973).

2.4.2 Computational Auditory Scene Analysis

The Auditory Scene Analysis framework provides a number of empirically grounded insights into how we perceive sound. While ASA is essentially a high-level approximation of auditory perception, it is still a complex model with many interconnected components and variables. This fact, combined with inherent ambiguities, leads to several trade-offs when we try to build computational models based on ASA. How closely do we try to emulate the ASA principles in software? How do we set the values of the many free variables?

The term Computational Auditory Scene Analysis was coined in (Brown

& Cooke, 1994). The authors take a rather literal approach to modeling ASA through a processing graph where each node is a model of acoustical, neurological, or psycho-perceptual auditory function.

In the later sections of this chapter we will describe several deep learning-based source separation methods, where heuristics and manual feature engineering has been replaced by automatically learned feature sets. These learned features tend to self-organize hierarchically, such that early network layers correspond to low-level audio features, and later layers map to courser, high-level features. It is therefore instructive to investigate the details of CASA, which also follows a hierarchical organization that progressively transforms low-level audio features into high-level audio "objects". Below is a summary of the CASA system.

First, the input audio signal is low-pass filtered to simulate filtering that occurs in our outer ear. The filtered signal is then fed through a series of gammatone filters to produce a two-dimensional time-frequency representation that resembles a spectrogram.

Taking inspiration from neural functions, the authors compute *auto-correlation maps* over the extracted time-frequency representation, for each spoken syllable of interest. They then compute a frequency-wise *cross-correlation map* over each auto-correlation map. Similar to the Gestalt principle of common fate, frequency bins of the auto-correlation map that have a cross-correlation above a certain threshold are grouped into *periodicity groups*.

In the next step, sequential integration is modeled as short-term frequency transition curves. The intuition is that spoken words advance through time by continuously sliding pitch, and that continuity helps us perceive words as emanating from the same source, invoking the Gestalt principles of proximity as well as continuity. The authors model pitch transitions using a bank of two-dimensional Gaussian filters rotated at different angles. The filters are convolved with the time-frequency representation from above, and the per time frame and frequency bin maxima are computed into a *frequency transition map*.

One key component of spectral integration in the ASA framework is onset times. When several sounds start at the same time, the brain tends to attribute them to the same source. In CASA, this behavior is modeled using per-frequency onset detectors, implemented as leaky integrators. This results

in a time-frequency *onset map*.

The periodicity groups, frequency transition map, and onset map are fed through a series of heuristic rules to form a set of *auditory elements*, a segmentation of the spectrogram into higher-level connected regions. These elements are then further grouped into the final source-specific spectrogram masks by estimated fundamental frequency, and shared onset and offset. The gammatone-filtered spectrogram is multiplied element-wise with each mask, and resynthesized back to the time domain.

Several other computational models of auditory scene analysis have been proposed. For example, (Ellis, 1996) takes a more data-driven approach. A hierarchical series of processing components, similar to the CASA model outlined above, compute a source separation hypothesis. But in contrast to (Brown & Cooke, 1994), Ellis' model then compares the predicted isolated sources from real isolated sources. The difference between the real and predicted sources is then fed back into the system, and is used to update the processing components to better match the real isolated sources.

REPET (Rafii & Pardo, 2013) makes the observation that, in music, the background is often repeating, whereas the foreground is dynamically changing over time. They exploit this fact for source separation. A beat tracking algorithm is first run over the spectrogram. Repeating groups of beats are subtracted out, and what remains is an isolated foreground signal.

In Kernel Additive Models (Liutkus, Fitzgerald, Rafii, Pardo, & Daudet, 2014), repetitions, as well as other local audio patterns, are modeled as 2-dimensional kernels. Sources are extracted by median filtering or convolving the mixture spectrogram with the pre-defined kernels. Kernels can be designed for locally smooth signals, such as vocals, as well as percussive and harmonic signals.

2.4.3 Matrix Decomposition Methods

We can pose the source separation problem as a decomposition of a multi-dimensional mixture signal $\mathbf{x}_i$ of into sources $\mathbf{y}_j$:

$$x_{i,t} = \sum_{j=0}^{K} a_{i,j} y_{j,t} \tag{2.4}$$

where t is the discrete time step, and $a_{i,j}$ is a projection weight from mixture dimension i to source j. In matrix notation:

$$\mathbf{x}_t = \mathbf{A}\mathbf{y}_t \tag{2.5}$$

where $\mathbf{x}_t \in \mathbb{R}^N$, $\mathbf{A} \in \mathbb{R}^{N \times K}$, and $\mathbf{y}_t \in \mathbb{R}^K$, N is the number of input dimensions and K is the number of sources. It should be noted that $\mathbf{x}_t$ can represent a multi-channel time-domain audio signal, a frequency domain spectrogram, or any other multi-dimensional time series.

By approximating an inversion of $\mathbf{A}$, $\mathbf{W} \approx \mathbf{A}^{-1}$ we can estimate the original sources:

$$\hat{\mathbf{y}}_t = \mathbf{W}\mathbf{x}_t \tag{2.6}$$

Independent Component Analysis (ICA), proposed in (Comon, 1994), makes the observation that a mixture of sources will be more Gaussian than individual sources. ICA attempts to optimize $\mathbf{W}$ such that the extracted sources $\mathbf{y}$ maximize non-Gaussianity. It can be shown that maximizing non-Gaussianity is equivalent to minimizing mutual information.

In classical ICA for audio source separation, the number of sources are required to be fewer than the number of input channels. That means that ICA is mostly applicable for multi-microphone recordings.

One simple way to handle the *overcomplete* case where $K > N$ is to perform dimensionality reduction on $\mathbf{x}$. This is the basic idea behind Independent Subspace Analysis (ISA) (Casey & Westner, 2000). In audio-based ISA, the input time-domain signal is first transformed to a frequency-domain spectrogram. The spectrogram is then dimensionality reduced using PCA, ICA is applied to extract low-dimensional source matrices, and source spectrograms are retrieved by inverting the PCA. Finally, the extracted source spectrograms are clustered according to KL-divergence.

A slightly different formulation of source separation decomposes a mixture into a linear mixture of a set of source *basis functions*, weighted over time:

$$\mathbf{x}_t = \sum_{i=0}^{K} \mathbf{b}_i g_{i,t} \tag{2.7}$$

where $g_{i,t}$ is the gain of source i at time t, and $\mathbf{b}_i$ is the basis function for source i.

In matrix notation:

$$\mathbf{X} = \mathbf{BG} \tag{2.8}$$

where $\mathbf{X} \in \mathbb{R}^{T \times N}$, $\mathbf{B} \in \mathbb{R}^{N \times K}$, and $\mathbf{G} \in \mathbb{R}^{K \times T}$, where T is the number of time steps, N is the number of input dimensions, and K is the number of basis functions.

Basis functions can be interpreted as per-source spectral prototypes. For example, if we hand craft basis functions corresponding to the spectral profiles of keys on a piano, and let $\mathbf{x}_t$ be spectrogram frames of a piano recording, the time-varying gain $g_{i,t}$ will correspond to notes being played on the piano. Hence, we have designed a piano note transcription system, by means of a note separation system (Smaragdis & Brown, 2003). Similarly, if we knew the prototypical spectral profiles for musical instruments, we should be able to decompose a musical mixture signal into individual source signals.

This problem is related to schema-based auditory scene analysis, described in 2.4.1. Spectral profiles can be considered schemas or templates of sound sources. In a process analogous to human schema learning, we can algorithmically learn a dictionary of spectral profiles from audio examples.

Non-negative Matrix Factorization (NMF) is an unsupervised method to learn source basis functions from a mixture. It has been applied extensively to musical source separation, e.g. in (B. Wang, Mary, Plumbley, & Mary, 2005).

In (FitzGerald, Lawlor, & Coyle, 2003), templates for individual drum sounds are extracted with ISA. Each drum sound is itself separated into a number of templates that together capture the various characteristics of that drum sound, the intuition being that a real drum can be hit in many ways. A full mixture of different drum sounds is then analyzed by the instrument-specific ISA models.

It can be argued that a mixture of drum sounds is one of the simpler forms of musical source separation, since drums are not pitched instruments. Drums are also fairly well separated in the frequency domain, with kick drums, snare drums, hi-hats, and cymbals occupying relatively different parts of the spectrogram, with comparatively little overlap of harmonics. These are indeed some of the factors that make musical source separation a more difficult task than speech separation.

Template learning for pitched instruments is discussed in (Vincent & Rodet, 2004). In contrast to the drum transcription system described above, separate ISA models are learned for each source and note pitch pair. By building per-genre dictionaries, (Laroche, Papadopoulos, Kowalski, & Richard, 2016) find that genre-specific dictionaries outperform global dictionaries.

Harmonic/Percussive Source Separation (HPSS) attempts to decompose a mixture spectrogram into two matrices representing harmonic and percussive components of the input spectrogram. A simple, yet effective algorithm was introduced by (FitzGerald et al., 2003), who applies median filtering over time to retrieve harmonic components, and similarly median filters over frequency to retrieve percussive components.

(Jeong & Lee, 2014) observes that vocals are neither exclusively harmonic nor percussive in nature, and thus decomposes the mixture spectrogram into harmonic, percussive, and vocal components. They do so by enforcing a sparsity constraint on the residual vocal component.

The inherent sparsity of the vocal signal is also exploited in (P.-S. Huang, Chen, Smaragdis, & Hasegawa-Johnson, 2012), which uses Robust PCA (RPCA) (Candes, Li, Ma, & Wright, 2010) to separate vocals from accompaniment. RPCA decomposes the mixture signal into a sparse matrix and a low-rank matrix. Instrumental music signals tend to be repetitive, and therefore their magnitude spectrograms can be considered low-rank matrices.

2.4.4 Pitch-informed Methods

In the template learning methods in (FitzGerald et al., 2003) and (Vincent & Rodet, 2004), the ISA-based source separators effectively double as drum and pitch transcription systems. Pitch tracking is also a key element in the CASA system described in 2.4.2 There are many examples in the literature where source separation has been approached in conjunction with pitch tran-

scription. Intuitively, it makes sense that a separated source should be easier to transcribe. And conversely, if we knew the vocal melody *a priori*, it should be possible to follow that melody during vocal source separation.

(Kashino & Murase, 1999) proposes a multi-agent system for musical source separation. Each agent adapts to process an individual instrument, using a bank of adaptively updated waveform templates, tuned to musical pitches. Agents cooperate via a "mediator", to group an auditory scene into segments labeled by instrument and pitch.

In (Durrieu, Richard, & David, 2008), the authors augment an NMF model with a vocal pitch tracker. The vocal source is estimated with a pitch-dependent Gaussian Mixture Model (GMM) and all non-vocal sources are modeled with NMF. The GMM and NMF models are combined into a single graphical model that is jointly optimized.

A related vocal separation algorithm is presented in (Virtanen, Mesaros, & Ryynänen, 2008). First, pitch transcription based on the most prominent detected fundamental frequency is performed on the input spectrogram. It is assumed that the vocal signal is responsible for the most prominent f_0, and the spectrogram partials that are found to originate in that f_0 are designated as the vocal source. It is then masked out of the input spectrogram, and NMF is performed on the remaining non-zero bins.

An iterative approach is presented in (Hsu, Wang, Jang, & Hu, 2012), where a rough pitch estimation is used to inform a rough source separation. This process is then iterated: the rough separation results produce a slightly better pitch estimation, which produces a slightly better separation, and so forth.

2.4.5 User-guided separation

Machine learning is often applied to problems that humans can solve trivially, but that are difficult to design procedural algorithms and heuristics for. For example, hand-written digit recognition is a domain where we, up until relatively recently, had to employ human data entryists to transfer numbers from paper to digital databases. Source separation is arguably an even more intuitive for human brain function than reading text. However, actually separating an audio mixture into several isolated audio files is not an easy task for humans to perform. While software packages exist for this task, e.g.

SADiE[10], they require painstaking human intervention.

Several user-guided source separation have been proposed to simplify the process of manual source separation. While these systems are not able to process audio at the scale of the fully automatic systems that are reviewed throughout this chapter, they often produce higher quality separation results than automatic systems.

The pitch-informed system in (Durrieu & Thiran, 2012) first estimates multiple fundamental frequencies in the mixture. A user then selects which fundamentals to extract sources for. In (Bryan, Mysore, & Wang, 2013), the user interface consists of a piano roll representation of separation results, that the user can manipulate to improve the automatic source separation.

2.4.6 Frequency-domain masking

Many of the deep learning architectures presented below in Section 2.4.7 produce "masks" that are applied to frequency-domain spectrograms. A mask is typically a matrix of the same shape as the target spectrogram, whose aim is to selectively "let pass through" only those components of the input signal corresponding to the particular source. A similar form of frequency-domain masking is known to be present in the auditory system of the human brain: a quiet sound becomes imperceptible if a different, louder sound is activated close to it in time and frequency (Moore, 2004), suggesting that each time-frequency component could be associated to a single, dominant source. This notion is formalized in (D. Wang, 2005): in the *Ideal Binary Mask* (IBM) each time-frequency (T-F) component is either 1 or 0, depending on the intensity of the target signal with respect to background noise. The IBM has been used in the context of neural network-based speech enhancement (P.-S. Huang, Kim, Hasegawa-Johnson, & Smaragdis, 2014; Narayanan & Wang, 2013) and musical source separation (Luo, Chen, Hershey, Roux, & Mesgarani, 2017b; Simpson, Roma, & Plumbley, 2015).

Given a set of source spectrograms $\mathbf{X}_i$, the Ideal Binary Mask $\mathbf{M}_s^{\text{IBM}}$ for source s is defined as:

$$\mathbf{M}_{s_{t,f}}^{\text{IBM}} = \begin{cases} 1, & \text{if } s = \arg\max_i X_{t,f}^i \\ 0, & \text{otherwise} \end{cases} \tag{2.9}$$

[10] http://www.sadie.com

where T and F are the number of time steps and frequency bins in the spectrogram, t and f are time and frequency indices, and s and i are source indices.

Given the input mixture spectrogram, $\mathbf{X}$, the isolated source spectrogram, $\hat{\mathbf{Y}}$, is estimated as

$$\hat{\mathbf{Y}} = \mathbf{X} \otimes \text{IB}\bar{\text{M}} \tag{2.10}$$

where $\otimes$ represents the Hadamard product.

The Hadamard product is simply an element-wise multiplication. Given two matrices of the same shape, $\mathbf{A}$ and $\mathbf{B}$, the Hadamard product is defined as:

$$\mathbf{C} = \mathbf{A} \otimes \mathbf{B} \tag{2.11}$$

$$C_{i,j} = A_{i,j} \cdot B_{i,j} \tag{2.12}$$

Although there are arguments in favor of binary masks – namely, that the network is forced to make hard decisions about the source to which each T-F component belongs, thus reducing interference (Grais, Roma, Simpson, & Plumbley, 2016) – it is becoming increasingly common for source separation models to estimate *soft* masks, to be applied to the mixture spectrogram. As the name implies, the components of a soft mask are not restricted to be binary-valued.

A soft mask, denoted by $\mathbf{M}$, may either be real-valued $M_{i,j} \in \mathbb{R}, 0 \leq M_{i,j} \leq 1$, or complex $M_{i,j} \in \mathbb{C}, 0 \leq |M_{i,j}| \leq 1$. The application of a soft mask is typically performed by multiplication to the input mixture spectrogram; in such case it should be noted that the estimation of the mask is invariant to differences in the scaling of the input signal. Soft ratio masks are applied the same way as binary masks:

$$\hat{\mathbf{Y}} = \mathbf{X} \otimes \mathbf{M} \tag{2.13}$$

A common post-processing technique involves the use of multi-channel Wiener filtering (Nugraha, Liutkus, & Vincent, 2018). Source spectrograms are initially estimated using some source separation algorithm (DNN, NMF, etc.), and then refined in an expectation-maximization setting that maximizes the consistency of source estimates. For an implementation of multi-channel Wiener filtering, refer to the Norbert Python package (Liutkus & Stöter, 2019).

2.4.7 Deep Learning Methods

Computation models of the neural networks that exist in the brain have been investigated since this 1950s (Rosenblatt, 1958). Since then, artificial neural networks have shifted away from simulation of biological systems, to become a useful computational technique in their own right. Deep learning represents the latest "wave" of interest in neural networks, propelled by advances in computer hardware and software (Bengio, 2016). For a historical perspective of deep learning, please refer to (Schmidhuber, 2015).

Deep Neural Networks (DNN) have are now responsible for state-of-the-art results in a wide range of applications of supervised and unsupervised machine learning, as well as reinforcement learning. For several years deep neural networks have surpassed human accuracy in image classification (Geirhos et al., 2017). Deep networks are also used in state-of-the-art results in semantic image segmentation (Tao, Sapra, & Catanzaro, 2020), object detection (Qiao, Chen, & Yuille, 2020), image generation (Ho, Jain, & Abbeel, 2020), image super-resolution (Y. Zhang et al., 2018), image captioning (Yun et al., 2019), as well as many other image processing tasks.

Tranformer-based architectures (Vaswani et al., 2017) have resulted in successful natural language processing processing models for common sense reasoning (Devlin, Chang, Lee, & Toutanova, 2019), machine translation (Edunov, Ott, Auli, & Grangier, 2018), question answering, and sentiment analysis (Raffel et al., 2019).

Deep networks are have also been shown to outperform previous methods in tasks as diverse as graph processing (R. Wang, Li, Hu, Du, & Zhang, 2020), time series forecasting (Lim, Arik, Loeff, & Pfister, 2019), programming code generation (Yu, Li, Zhang, Zhang, & Radev, 2018), reinforcement learning-based computer game play (Vinyals et al., 2017), and has also been instrumental in the development of self-driving car technology (Liang, Jiang, Niebles, Hauptmann, & Fei-Fei, 2019)

The Music Information Retrieval community has been swift to adopt deep learning techniques for many tasks. (Eric J. Humphrey, Bello, & LeCun, 2013) outlined several opportunities where DL could have a transformative effect on MIR. The authors argued that the hand-crafted features that dominated the field at the time were inherently suboptimal and lead to overly complicated architectures; that current methods were too localized in time

and failed to take advantage of long-term musical structure; that shallow models struggle to approximate the highly non-linear functions that underlie real-world musical audio signals; and that music itself exhibits deep hierarchical internal structure. They suggested that the MIR community embrace deep feature learning to address these issues.

We can get a rough idea of the adoption of deep learning in MIR by looking at the appearance of the string "deep" in the titles of papers submitted to the ISMIR conference between 2000 and 2018. Since 2004, around 100 papers per year have been accepted. From 2000 to 2015, 7 papers in total had the string "deep" in the title. In 2016, 2017, and 2018, there were 8, 6, and 7 "deep" papers per year, respectively.

There are many examples of deep learning-based methods in the larger speech and sound source separation communities. However, neural networks have been applied to source separation before the emergence of Deep Learning. Here follows a review of a few selected papers from the source separation literature that uses neural networks and deep learning. It can also be read as a brief (and rather incomplete) history of the evolution of neural networks over the past twenty years, through the lens of audio source separation.

One of the earliest source separation systems that employed neural networks is (Herault & Jutten, 1986), later refined in (Jutten & Hérault, 1991). In this work, a single-layer linear recurrent neural network is optimized using an unsupervised learning function. Another early unsupervised neural model is presented in (Burel, 1992), where a three-layer feed-forward network is trained to minimize a cost function based on statistics of the input signal, using backpropagation. Speech denoising, a problem related to segregation of vocals and music accompaniment, was addressed in 1988 using neural networks (Tamura & Waibel, 1988). A linear four-layer feed-forward network, operating in the time-domain, learns to remove synthetically added room noise from speech recordings.

In (Berenzweig & Ellis, 2001), a neural network-based method for Vocal Activity Detection (VAD) is proposed. VAD is similar to source separation, but instead of estimating a continuous source signal, the model predicts a binary vocal activation function over time. The authors extract audio features by using a multi-class neural network trained to predict spoken phonemes. These extracted features are then fed through a Hidden Markov Model to produce the vocal/music segmentation.

Deep Learning was applied to speech enhancement in (Lu, Matsuda, Hori, & Kashioka, 2012). Using an architecture common in the early Deep Learning literature, the authors build an bottle-necked autoencoder to reconstruct a clean speech signal. When presented with the Mel spectrogram of a noisy speech signal, the model removes noise and outputs a clean signal.

One of the first models to apply deep neural networks to musical source separation was presented in (Grais, Sen, & Erdogan, 2014). A five-layer feed-forward network is first initialized by layer-wise pre-training using Restricted Boltzmann Machines. It is then trained to separate artificially synchronized mixtures of speech and piano music.

Recurrent Neural Networks

Several Recurrent Neural Network (RNN) models have been explored for source separation and related fields. In (Maas et al., 2012), a Deep Recurrent Denoising Auto-Encoder is applied to speech enhancement. (Weninger, Hershey, Roux, & Schuller, 2014) trained a two-layer Long Short-Term Memory (LSTM) network to output spectral masks for each source.

(Erdogan, Hershey, Watanabe, & Roux, 2015) used a deep Bidirectional LSTM (BLSTM) model for speech separation, and uses a loss function that incorporates the phase difference between the estimated and actual sources. This system also incorporates speech recognition, by adding phoneme information as an additional input feature to the separation network.

An extension of Restricted Boltzmann Machines (RBM) to temporal data, RNN-RBM, is used in (Boulanger-Lewandowski, Mysore, & Hoffman, 2014) for separating vocals and accompaniment in music. Their RNN-RBM network is combined with unsupervised NMF, and they find that the joint model outperforms the individual components.

Another deep RNN for separating vocals from accompaniment is presented in (P. Huang, Kim, Hasegawa-Johnson, & Smaragdis, 2014). A three-layer RNN with ReLU non-linearities is trained to predict a soft spectrogram mask. When synthesizing the masked estimated source magnitude spectrogram, the phase from the original complex spectrogram is combined with the estimated magnitudes. Their system is trained on 1000 short clips of Chinese karaoke music, with durations between 4-13 seconds.

In (Mimilakis, Drossos, Virtanen, & Schuller, 2017), a recurrent encoder-decoder architecture is built with Bidirectional GRU units. Skip connections are added from the input to the decoder. To fine-tune the estimated source spectrogram, a highway network and a generalized Wiener filter are added after the decoder.

Convolutional Networks

The benefits of using recurrent networks over convolutional networks are generally cited as (theoretically) infinite memory, compared to the finite context window of convolutional networks. However, that apparent truism has been repeatedly challenged in non-recurrent architectures such as dilated convolutions (van den Oord et al., 2016), *"transformer networks"* (Vaswani et al., 2017), temporal convolutional networks (Bai, Kolter, & Koltun, 2018), among others. Convolutional networks have been successfully applied to musical source separation too, and below follows a selection of highlighted papers from the rather large literature.

A convolutional encoder-decoder architecture for source separation is introduced in (Chandna, Miron, Janer, & Gómez, 2017). After transforming the mixture signal to the frequency domain, the network consists of a vertical convolution (across frequency) layer followed by horizontal convolution (across time) layer. After a fully connected bottleneck layer, the decoder mirrors the encoder with horizontal transposed convolution and vertical transposed convolution layers. The output of the network is source-specific soft ratio mask, that is then multiplied element-wise with the input spectrogram.

The ratio mask is an estimate of the *Ideal Ratio Mask* (IRM), $\mathbf{M}^{\text{IRM}}$, which is defined as

$$\mathbf{M}^{\text{IRM}} = \mathbf{Y} \oslash \mathbf{X} \tag{2.14}$$

where $\mathbf{Y}$ is a spectrogram magnitude matrix corresponding to a single source, $\mathbf{X}$ is a mixture spectrogram containing $\mathbf{Y}$ and other sources, and $\oslash$ denotes element-wise division.

The hierarchical nature of music is explicitly modeled in (Grais, Wierstorf, Ward, & Plumbley, 2018). Three parallel convolutional encoder-decoder networks are jointly optimized to predict source spectrograms. The networks differ in receptive field sizes, thus allowing the network to learn dependencies at different time scales.

The *DenseNet* (G. Huang, Liu, Van Der Maaten, & Weinberger, 2017) is a neural network architecture consisting of blocks of fully connected layers where all layers in a block have all-to-all skip connections. Similar to the way convolutional networks have been constructed as encoder-decoders with a bottleneck layer, (Takahashi & Mitsufuji, 2017) builds an hourglass-like architecture of DenseNet blocks. In the encoder path the blocks are connected using pooled, and in the decoder using transpose convolutions. Several identical networks are applied to different frequency bands of the input, concatenated with a full-band network, and fed through a final DenseNet block to produce the spectrogram prediction.

Deep Clustering

Clustering has been applied to source separation, for example using self-organizing maps (Herrmann & Yang, 1996), density-based clustering (Van Hulle, 1999), and spectral clustering (Bach & Jordan, 2006). A deep neural network approach to clustering is introduced in (Hershey, Chen, Roux, & Watanabe, 2015).

We transpose $\mathbf{M}^s$ into an *indicator matrix* $\mathbf{I} \in \mathbb{Z}^{TF \times S}$ where S is the number of sources:

$$I_{\mathrm{ix}(t,f),s} = M^s_{t,f} \tag{2.15}$$

where the indexing function $\mathrm{ix}(t, f) = Tt + f$.

From the indicator matrix we derive an *affinity matrix* $\mathbf{A} \in \mathbb{Z}^{TF \times TF}$

$$\mathbf{A} = \mathbf{I}\mathbf{I}^T \tag{2.16}$$

An element in the affinity matrix $\mathbf{A}_{\mathrm{ix}(t_1,f_1),\mathrm{ix}(t_1,f_1)}$ will be 1 if $\mathbf{M}_{t_1,f_1} = \mathbf{M}_{t_2,f_2}$, otherwise 0.

The Deep Clustering algorithm uses a deep BLSTM network to produce an encoding vector for each time-frequency component, $\mathbf{V} \in \mathbb{R}^{TF \times D}$, where D is the encoding dimension. Encoding vectors are used to estimate the affinity matrix, $\hat{\mathbf{A}} = \mathbf{V}\mathbf{V}^T$, where $\hat{\mathbf{A}}$ is the estimated affinity matrix. The loss function for the network is defined as the squared Frobenius norm of the difference between the estimated affinity and real matrices

$$\mathrm{C}(\mathbf{A}, \hat{\mathbf{A}}) = \left\| (\hat{\mathbf{A}} - \mathbf{A}) \right\|_F^2 \tag{2.17}$$

After training, encoding vectors $\mathbf{V}$ are grouped into sources using k-means clustering, creating source-specific binary spectrogram masks.

One limitation of deep clustering is that the final clustering step is not part of the optimization procedure. To overcome this issue, the "Chimera" architecture is proposed in (Luo et al., 2017b). The Chimera network, like the eponymous mythological creature, has two heads. A deep BLSTM network produces an intermediate encoding that is then split into a deep clustering head, and a mask inference head. The deep clustering head has the same loss as in 2.17, and the mask inference head produces a spectrogram mask that optimizes similarity to the original source spectrogram. When trained independently, the mask inference head performs better than deep clustering. However, jointly training both objectives improve the performance of the mask inference head.

Generative Adversarial Networks

Generative Adversarial Networks (GAN) are a recent family of deep neural networks, originally proposed for realistic image generation (Goodfellow et al., 2014). Two networks, a *generator* network and a *discriminator* network, with different objectives are optimized independently, in tandem. The discriminator is trained to discriminate between real and generated images, while the generator is trained to produce images that the discriminator classifies as real. The generator has to adapt to a constantly improving discriminator by producing more and more realistic images, and not leave any "hints" that the discriminator can use to correctly classify generated images as generated.

The SEGAN applies a GAN architecture to speech enhancement (Pascual, Bonafonte, & Serrà, 2017). The generator consists of a 1D convolutional encoder-decoder. The input to the generator is speech with added noise and the discriminator is presented with clean speech.

GANs are used in a similar way for vocal separation in (Fan, Lai, & Jang, 2018) and (Stoller, Ewert, & Dixon, 2018). Here, the generator is conditioned on a musical mixture, and the discriminator is presented with examples of clean, isolated vocals.

Stacking networks

As deep neural networks have grown deeper and deeper with increasing numbers of layers, the risk of information being lost increases. Several methods have been proposed to facilitate the flow of information from lower layers to upper layers.

The U-Net (Ronneberger, Fischer, & Brox, 2015), introduced in detail in Chapter 4, is a popular architecture that concatenates the relatively unprocessed output from lower layers with the heavily processed inputs to the upper layers, by means of *skip connections*. Similar skip connections can be found in Highway Networks (Srivastava, Greff, & Schmidhuber, 2015), where skip connections are used to connect lower layers with upper layers to allow the flow of information.

In Recurrent Deep Stacking Networks (Palangi, Deng, & Ward, 2014) several RNN modules are stacked and the inputs are fed to each of the RNNs in the stack. This architecture was used for speech separation in (Z.-Q. Wang & Wang, 2017).

The DeepOtsu model (He & Schomaker, 2019) for image enhancement uses a stack of equivalent convolutional networks to iterative enhance the results of previous models. We investigate this iterative enhancement framework in the context of source separation in Chapter 5.

2.4.8 Training objectives

When the training target is a spectrogram, the L_1 and L_2 loss functions are commonly employed. The L_1 loss, sometimes called "mean absolute error", measures the average of the absolute deviation between a target y and a prediction $\hat{y}$:

$$L_1(y, \hat{y}) = \frac{1}{N} \sum_{i}^{N} \|y - \hat{y}\| \tag{2.18}$$

where N is the number of elements in y and $\hat{y}$.

The L_2 loss, or "mean squared error" is defined as:

$$L_1(y, \hat{y}) = \frac{1}{N} \sum_{i}^{N} (y - \hat{y})^2 \tag{2.19}$$

In the context of speech enhancement (Pandey & Wang, 2018) and (Z.-Q. Wang & Wang, 2017) both found that the L_1 loss consistently performed better than the L_2 loss when estimating spectrograms. It is believed that the L_2 loss tends to produce blurry images when applied to image generation tasks, since the penalty for pixel-wise error is higher.(Pathak, Krähenbühl, Donahue, Darrell, & Efros, 2016)

Since the standard evaluation method in many cases is the Signal-to-Distortion Ratio (SDR, see in Section 2.6), the authors of (Venkataramani, Casebeer, & Smaragdis, 2018) designed a loss function that directly optimized SDR. They somewhat unsurprisingly found that the SDR loss performed better than L_2 loss on the SDR metric.

The Phase-U-Net (Choi et al., 2019) utilizes a time-domain SDR loss, despite the fact that the network produces spectrogram outputs. Since the spectrograms are complex, L_1 and L_2 losses are undefined. They solve this by computing the SDR loss in the time-domain and backpropagating the loss through the ISTFT.

Multi-objective learning is becoming increasingly common for deep learning models, in which multiple loss functions are linearly weighted by fixed (or trained) hyperparameters, and combined to a single training objective. For example, the automatic *anime* sketch colorization model in (Y. Liu, Qin, Luo, & Wang, 2017) uses a four-term loss function to jointly optimize for pixel-level similarity, colorfulness, realness, and color smoothness.

In speech enhancement, (Y. Zhao, Xu, Giri, & Zhang, 2018) combine a differentiable modification to the Short-Time Objective Intelligability (STOI) measure with a spectrogram similarity measure in a single loss function. A combination of time and frequency domain loss function is presented in (Fu, yao Hu, Tsao, & Lu, 2017), as the objective for a complex valued spectrogram estimation model for speech enhancement. The authors construct a loss function out of three weighted terms: complex valued distance, spectrogram magnitude difference, and time domain L_2 difference. As in the Phase-U-Net, the time domain loss is backpropagated through the ISTFT. A similar model

was introduced in (Arik, Jun, & Diamos, 2019), for spectrogram inversion in speech generation. The model outputs time-domain samples, but the loss is computed in the frequency domain by means of a fully differentiable STFT. The loss function is a combination of four terms: Spectral convergence, magnitude spectrogram difference, instantaneous frequency difference, and phase difference.

2.4.9 Phase-aware methods

The majority of the deep learning architectures reviewed in Section 2.4.7 operate on the magnitude component of spectrograms. The phase component is mostly ignored during training, and at inference time the phase of the mixture is combined with the estimated magnitude before inverting the STFT back to the time domain. This is in spite of ample evidence that phase is highly important for accurate reconstruction, both from the image processing (Oppenheim & Lim, 1981), speech enhancement (Paliwal, Wójcicki, & Shannon, 2011), and source separation domains (Dubey, Kenyon, Carlson, & Thresher, 2017).

Several systems have been proposed that operate directly in the complex domain. Prior and parallel to the advent of deep learning, complex Nonnegative Matrix Factorization was explored in (King & Atlas, 2010) and (Magron, Badeau, & David, 2018).

It was shown in (Erdogan et al., 2015) that incorporating phase information in the mask function resulted in more accurate source estimation, even though the actual mask was real-valued. However, predicting phase directly appears to be intractable (Williamson, Wang, & Wang, 2016), presumably because of lack of inherent structure due to the wrap-around nature of phase. In order to mitigate this problem, phase unwrapping (the addition of appropriate multiples of 2π to individual parts of the phase component of a signal, so as to avoid discontinuities) has been applied in the context of source separation (Mayer, Williamson, Mowlaee, & Wang, 2017; Spoorthi, Gorthi, & Gorthi, 2019). Another potential solution is to discretize phase and treat phase prediction as a classification problem, rather than regression (Takahashi & Mitsufuji, 2017).

Complex inputs and outputs have been applied to deep neural networks as well. For example (Simpson, 2015), introduces a complex convolution oper-

ation. The ReLU non-linearity is adapted to the complex domain in (Lee, Wang, Wang, Wang, & Wu, 2017).

A deep feed-forward network using phase features is presented in (Muth et al., 2018). This network consists of two parallel feed-forward networks ("heads"); magnitude and phase features. Several different phase features are tested: instantaneous frequency, group delay, and raw phase. Instantaneous frequency and group delay both enhance the network performance compared to only using magnitude features.

The Phase-U-Net, introduced in (Choi et al., 2019) introduces a U-Net that is internally complex and outputs a constrained polar complex spectrogram mask. The mask is then applied to the mixture complex spectrogram mask. We expand on this model in Chapter 6 and apply it to multi-instrument source separation.

The majority of deep networks for musical source separation operate on time-frequency representations (spectrograms) of both the input and output signals. Usually separation is achieved using masks in the magnitude domain, as discussed above. However, a number of recent systems propose alternative models that explicitly attempt to estimate the time domain signals.

In (Rethage, Pons, & Serra, 2018), the authors construct a non-causal WaveNet (van den Oord et al., 2016) for speech enhancement entirely in the time domain. The Wave-U-Net (Stoller et al., 2018) is an extension of the WaveNet that adds skip connections in the time domain, analogous to the U-Net in the frequency domain. TasNet is a time-domain model that employs a sample-level convolutional encoder-decoder, with an LSTM in the bottleneck layer to capture long-term dependencies (Luo & Mesgarani, 2018).

2.5 Multi-instrument separation

So far, the systems we have reviewed have concerned the separation of vocals from non-musical noise or background music. Algorithms have also been developed specifically for the separation of multiple instruments. While the approaches to instrument separation typically differ little from vocal separation, the multi-instrument separation task provides a challenging testbed, that stressed the algorithms' abilities to function in environments where sources are heavily overlapping, and often ambiguous in timbre.

The annual Signal Separation Evaluation Campaign (SiSEC)(Vincent, Araki, & Bofill, 2009) has, since 2008, conducted standardized evaluation of multi-instrument separation algorithms submitted by researchers. The SiSEC campaign evaluates the performance of each algorithm on a test set consisting of musical stereo mixtures, and the constituent sources bass, guitar, vocals, and "other", where the "other" category is the sum of all sources that are not bass, guitar, or vocals. SiSEC has been host to many promising algorithms; as of 2018 most algorithms were based on deep learning.(Stöter, Liutkus, & Ito, 2018) The highest scoring algorithm in 2018 (Takahashi & Mitsufuji, 2017), previously described in section 2.4.7 trained individual networks per-source, in a fully convolutional setting using DenseNets.

Joint, concurrent learning of all instrument sources with a shared deep network has been explored in (Lluís, Pons, & Serra, 2018) and (J.-Y. Liu & Yang, 2018). The former builds a non-causal WaveNet-like (van den Oord et al., 2016) model, that predicts vocals, bass, and percussion as three separate outputs from the same model; the network is optimized by maximizing the similarity between each target source and estimated source, while minimizing the similarity between the estimated source and the other sources' targets. The latter model builds a U-Net like architecture, where skip connections are long-term dependency-preserving GRUs; the output is the concatenation of vocals, bass, drums, and "other" sources.

Many multi-instrument separation systems are trained on the DSD100 dataset (described in Section 2.7.4. We mentioned in Section 2.4.7 how "data hungry" deep networks are, and deep source separation models are no exceptions. Unfortunately DSD100 is a relatively small training dataset. Attempts have been made to remedy this through data augmentation: (Uhlich et al., 2017) proposes to increase the training dataset size by swapping source stereo channels, randomly scaling the amplitudes of the sources, randomly chunking into new sequences, and randomly mixing sources from different tracks. This increases the dataset size, but has the potential drawback of skewing the dataset to a less realistic distribution. Special care has to be taken when mixing instruments from different sources to preserve the correlation between sources that makes multi-source separation a challenging problem.

A multi-source classical music separation system using data augmentation is presented in (Miron, Janer, & Gómez, 2017). A set of classical scores are augmented through tempo changes, key shifts, dynamic range alterations,

etc., and then synthesized using virtual software instruments.

2.6 Evaluation of Musical Source Separation Systems

The evaluation of source separation systems is an inherently difficult task, for the same reasons that the development of automatic source separation systems is difficult. In order to give an accurate score to a particular source separation algorithm, we ideally need a perceptual model of how well humans perceive sources to be separated. The same kind of modeling of the human auditory system that we undertake as we develop source separation algorithms, would have to be performed in order to make evaluation consistent with human perception.

While we have been able to collect and create training datasets for source separation, no such datasets exist for the evaluation of source separation systems. For that reason, we are forced to resort to heuristic evaluation metrics.

Early source separation systems used simple distance functions such as the Euclidean distance, either in the time domain or the frequency domain, to compare their estimations to true isolated sources. This method has several shortcomings.

Firstly, it does not take into account the mixture from which the source is isolated. For example, say that we attempt to separate vocals from musical accompaniment. If we accurately estimate the vocals we will get the same score if the accompaniment is a quiet guitar or a loud heavy metal band — a far more difficult condition for source separation. In the latter case the algorithm has to remove more of the interference than in the former, but this is not rewarded by source-specific evaluation metrics.

Secondly, it does nothing to quantify the sound quality of the output. An algorithm that aims to isolate a high pitch voice from a low pitch voice might simply apply a steep high-pass filter, removing any frequency components below the average pitch of the interfering speaker. That algorithm would receive high scores if the metric was a time-domain L_1 loss, but the result would sound muffled or possibly even be unintelligible. If the sources were

musical instruments, say a viola and a violin, attempting to remove the viola through high-pass filtering would distort the output signal, leaving a timbre that no longer was characteristic of the violin.

Thirdly, many source separation algorithms introduce artifacts, unintended sonic effects that are often small in magnitude, but immediately noticed by human hearing. A simple distance-based evaluation function would not penalize such artifacts more than inaudible deviations in other parts of the output.

Several attempts have been made to address these issues.

2.6.1 BSS Eval

The *BSS Eval* family of metrics, introduced in (Vincent, Gribonval, & Févotte, 2006), produces not one, but four metrics relating to the distortion, interference, noise, and artifacts present in the source estimate. BSS Eval decomposes the estimated source into the true source and error components representing interference, noise, and artifacts.

$$\hat{s} = s + e_{\text{interf}} + e_{\text{noise}} + e_{\text{artif}} \tag{2.20}$$

where $\hat{s}$ is the estimated source, s is the part of $\hat{s}$ coming from the target source, e_{interf} is the interference error, e_{noise} is the noise error, and e_{artif} is the artifact error.

The Signal-To-Distortion Ratio (SDR), Signal-to-Interference Ratio (SIR), and Signal-to-Noise Ratio (SNR), and Signal-to-Artifact Ratio (SAR) are defined as

$$\mathrm{SDR}(\hat{s}, s) := 10 \log_{10} \frac{\|s\|^2}{\|e_{\text{interf}} + e_{\text{noise}} + e_{\text{artif}}\|^2} \tag{2.21}$$

$$\mathrm{SIR}(\hat{s}, s) := 10 \log_{10} \frac{\|s\|^2}{\|e_{\text{interf}}\|^2} \tag{2.22}$$

$$\mathrm{SNR}(\hat{s}, s) := 10 \log_{10} \frac{\|s + e_{\text{interf}}\|^2}{\|e_{\text{noise}}\|^2} \tag{2.23}$$

$$\mathrm{SAR}(\hat{s}, s) := 10 \log_{10} \frac{\|s + e_{\text{interf}} + e_{\text{noise}}\|^2}{\|e_{\text{artif}}\|^2} \tag{2.24}$$

$$\tag{2.25}$$

Additionally, we define the Normalized Signal-to-Distortion Ratio (NSDR) ("MIREX 2014:Singing Voice Separation", 2014) as

$$\mathrm{NSDR}(\hat{s}, s) = \mathrm{SDR}(\hat{s}, s) - \mathrm{SDR}(m, s) \tag{2.26}$$

where m is the mixture source.

2.6.2 PEASS

The motivation behind the *PEASS* family of evaluation metrics (Vincent, 2012) is that BSS Eval correlates poorly with human judgments. The authors attempt to remedy this by introducing a new set of metrics based on a gammatone filter decomposition of the original signal. Their method produces metrics that correlate very with human perception, but it has seen limited adoption due to its high computational cost.

2.7 Training Datasets

Below follows a list of publicly available datasets that are commonly used for training music source separation models. The datasets are summarized in Table 2.1.

Dataset	Genre	Channels	Sample rate	Sources	Tracks	Length
MIR-1K	Karaoke	Mono	16kHz	Vox+Accomp.	1000 snippets	2:13h
CCMixter	Various	Stereo	44.1kHz	Vox+Accomp.	50 full tracks	3:12h
iKala	Karaoke	Mono	44.1kHz	Vocal+Accomp.	126 snippets	2:06h
DSD100	Various	Stereo	44.1kHz	Dr.+Bass+Vox+Other	100 full tracks	7h
MedleyDB	Various	Stereo	44.1kHz	82 instruments	122 full tracks	7:17h
MedleyDB 2.0	Various	Stereo	44.1kHz	47 instruments	74 full tracks	6:32h
MedleyDB 1+2	Various	Stereo	44.1kHz	82 instruments	196 full tracks	13:49h
MUSDB18	Various	Stereo	44.1kHz	Dr.+Bass+Vox+Other	150 full tracks	10h

Table 2.1: Music source separation datasets

2.7.1 MIR-1K

The MIR-1K dataset (Hsu & Jang, 2010) contains 1000 clips of Chinese karaoke songs. The dataset is structured as a number of 16kHz stereo audio files, where one channel contains accompaniment, and the other channel is a solo voice, which means that both vocal and accompanying instrumental are mono sources. Singers are amateurs, recruited from the authors' research lab. Each clip is between 4 and 13 seconds, with a total dataset length of 133 minutes.

2.7.2 CCMixter

The CCMixter dataset (Liutkus et al., 2014) is comprised of 50 full-length, 44.1kHz stereo mixes, in various different genres, downloaded from the CCMixter web site[11]. CCMixter is an online community where professional and hobbyist electronic music producers share and collaborate on mixes. The dataset has a total duration of 3:12 hours.

2.7.3 iKala

iKala was[12] an online karaoke service, based in China. The iKala dataset (Chan et al., 2015), like to MIR-1K, consists of 44.1kHz mono accompaniment and vocal recordings for Chinese karaoke songs. The accompaniment was provided by iKala, and the vocal tracks are recorded by professional singers. The dataset contains 252 30-second snippets, i.e. a total dataset size of 126

[11] http://ccmixter.org/
[12] http://mac.citi.sinica.edu.tw/ikala/, retrieved 2019-01-17

minutes. Unfortunately, the iKala dataset is no longer available for download, due to legal constraints.

2.7.4 DSD100

The DSD100 dataset (Liutkus et al., 2014) (sometimes also referred to as the "SISEC dataset") consists of 100 full-length tracks in different genres, collected from the "Mixing Secrets" multi-track library[13]. Tracks are provided as isolated stereo stems for drums, bass, vocals, and "other", where "other" is a summed mix of all remaining sources. Track lengths range from 2:22 to 7:10, with a total duration of 7 hours of audio.

2.7.5 MedleyDB

MedleyDB (Rachel M. Bittner et al., 2014) is a dataset of professionally recorded multi-tracks in several different genres. The dataset contains 122 full length stereo tracks at 44.1kHz, ranging from approximately 3 to 5 minutes, at a total of 7:17 hours of audio. Isolated sources for 82 different instruments are included, along with pitch ground truth and other metadata.

2.7.6 MedleyDB 2.0

A second version of MedleyDB was presented in (R. Bittner, Wilkins, Yip, & Bello, 2016). The data format is unchanged from the original MedleyDB, but with new tracks. MedleyDB 2.0 contains 47 unique instruments, and new 74 tracks at a total of 6:32 hours.

2.7.7 MUSDB18

MUSDB18 is a concatenation of the 100 tracks from DSD100, 46 tracks from MedleyDB, and 4 new professionally recorded multi-tracks. This dataset, like DSD100, contains isolated sources for drums, bass, vocals, and "other". Instruments from MedleyDB have been mapped into these categories. The total length of data is approximately 10 hours of stereo audio at 44.1kHz.

[13] http://www.cambridge-mt.com/ms-mtk.htm, retrieved 2019-01-17

2.8 Summary

We began this chapter by introducing some of the digital signal processing prerequisites of source separation. We then introduced automatic musical source separation by listing some current and potential future applications of the technology, both user facing applications such as karaoke, as well as source separation as a building block for other MIR tasks.

This was followed by an extensive literature review of musical source separation and related domains: from Bregman's Auditory Scene Analysis and its software approximations, to matrix decomposition methods like ICA and NMF, to methods that incorporate pitch information and user inputs. We then listed several masking schemes common to source separation in the spectral domain. In the following section, we presented a number of deep learning-based methods, starting with early feed-forward networks, and moving on to more complicated architectures, such as RNNs, convolutional networks, deep clustering, GANs, and stacking networks.

We then introduced a number of training objectives that have been employed by machine learning-based source separation models, both in the frequency and time-domain. Phase-aware methods were then explored — frequency-domain models that incorporate phase information, followed by a review of multi-instrument separation systems.

Finally, we discussed several evaluation metrics for source separation, as well as a number of publicly available datasets.

Chapter 3

Data Mining Source Separation Datasets from Large-Scale Music Catalogs

In this chapter we will develop a methodology for extracting meaningful training examples for vocal source separation, from a large commercial music database. Our method is primarily based around music metadata, with a final content-based post-processing step. Music metadata is inherently noisy, and we will describe some of the sources for this noise and ways to make our algorithms robust to noise.

In addition to noisy metadata and inconsistencies, commercial music databases are large, with hundreds of millions of data rows. This allows us to create the kinds of large datasets that are required for effective training of modern deep learning architectures, but also complicates the data engineering process: these databases are too large to fit in the memory of single computers, and we have to resort to distributed computing in order to process the data. This also means that our algorithms need to be designed for a distributed setting. Fortunately, modern distributed computing software can abstract the intrinsic complexities of distributed data processing. We will begin this chapter with a brief history and overview of distributed data processing.

3.1 Brief History of Distributed Data Processing

The rise of deep learning has gone hand-in-hand with the increased availability of large training datasets (Goodfellow, Bengio, & Courville, 2018). Deep neural network models have huge capacity in terms of trainable variables, but are also prone to overfitting if presented with small datasets. Only when deep models are trained with large datasets is deep learning able to achieve state-of-the-art results. This is reflected in the exponential increase in the number of examples available in datasets released over the past decade. A subset of machine learning dataset sizes over the last century can be found in Figure 3.1 (figure from (Goodfellow et al., 2018)).

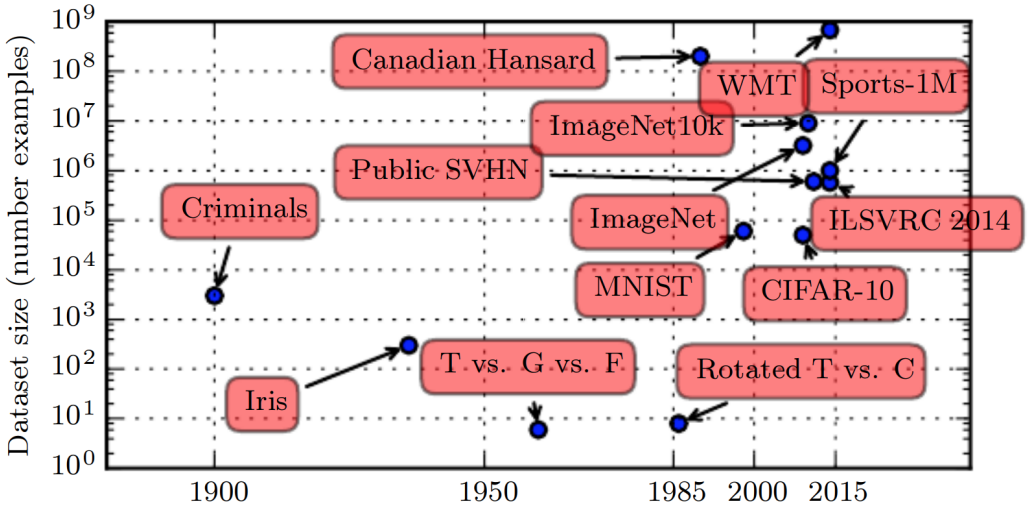


Figure 3.1: Machine learning dataset sizes over time (from (Goodfellow, Bengio, & Courville, 2018))

The dependency on deep learning and big data is symmetric. Deep learning models need to be fed with big data to function, but big data also require models with enough capacity to capture all the information hidden in these big datasets.

Over the past couple of decades, the Internet has allowed businesses to log and track more and more of their customers' behavior. This has lead to data-driven decision making, highly targeted marketing, algorithmic recom-

mendation systems, etc., increasing both the profitability and productivity of the companies who have embraced "big data" (McAfee & Brynjolfsson, 2012).

However, simply logging every user interaction into a mountain of data does not create value on its own. In order to gain understanding and make predictions from this data we need algorithms that are able to learn from such large datasets. Deep learning has emerged as one of the most powerful big data processing tools, causing top technology companies in the world to invest heavily in deep learning research and further fueling growth of the field.

While deep learning models are able to learn from big data, it does not provide any solutions to storage, preprocessing, and exploratory analysis of such datasets. In the remained of this section we turn to this important, but often overlooked, prerequisite task.

Data preprocessing is often one of the most time-consuming activities in machine learning. Unfortunately, when dealing with gigabytes, terabytes, and even petabytes of data, preprocessing tends to become exponentially more complex.

Modern, high-end SSD hard drives are able to read contiguous data at around 500 MB/sec, meaning that one terabyte of data can be read in 30 minutes on a single computer.[1] It is clear that processing terabyte or petabyte-scale datasets in a tractable amount of time requires some means of reading data from multiple disks in a distributed fashion.

The same is true for the computational aspect of data processing: given the size of our data, processing on a single server — even a high performance server with tens or even hundreds of CPU cores — has become infeasible.

Distributed storage and computation has been extensively studied in the software engineering literature, and many systems have been used in industry over the years (Fox et al., 1989). In early systems it was up to the the programmer to synchronize processes on distributed compute nodes, as well as handle message passing between nodes. Several architectures were proposed to improve this difficult and error-prone paradigm. MPI (Snir,

[1]This is a highly optimistic approximation, since data is rarely completely contiguous but fragmented across the disk, forcing the disk to seek with considerable slowdown in read performance.

Otto, Huss-Lederman, Walker, & Dongarra, 1996) standardized the message passing interface, but the programmer was still responsible for most of the delicate synchronization work.

Google addressed this in their seminal MapReduce paper (Dean & Ghemawat, 2008). By providing a highly restricted API, consisting mainly of two operations: "map" and "reduce", they decoupled data processing operations from cluster management, message passing, synchronization, and fault tolerance. MapReduce was built on top of the Google File System (GFS) (Ghemawat, Gobioff, & Leung, 2003), a highly distributed data storage system. Both MapReduce and GFS were designed to be massively scalable, and deployed on large numbers of consumer-grade computers. An open-source re-implementation of MapReduce and GFS called Hadoop (White, 2010) quickly gained adoption in industry, outside of Google.

A decade after MapReduce, Google released DataFlow (Akidau et al., 2015) (later re-branded as "Apache Beam"). The DataFlow API has a larger set of primitive operations than MapReduce, while maintaining the strong encapsulation of the underlying distributed computing model.

For many smaller data analysis tasks, the overhead of developing a MapReduce/DataFlow pipeline can seem excessive. This use case was explored by researchers at Facebook, leading to the release of Hive (Thusoo et al., 2010), a distributed database on top of Hadoop with a SQL-like interface. Around the same time, Google developed their own SQL-based query system for petabyte-scale data (Melnik et al., 2010). This system was later publicly released as BigQuery, a cloud-hosted data warehouse.

3.2 Big Datasets in Music Information Retrieval

Datasets for music information retrieval have seen a similar evolution to the general machine learning field. Figure 3.2 shows approximations of the total length of audio for MIR datasets that include audio data, since the year 2000.

This chart was created using the list of datasets maintained on the web site www.audiocontentanalysis.org/data-sets (Lerch, 2019). Release years and audio lengths were taken from the linked dataset web sites, when they were

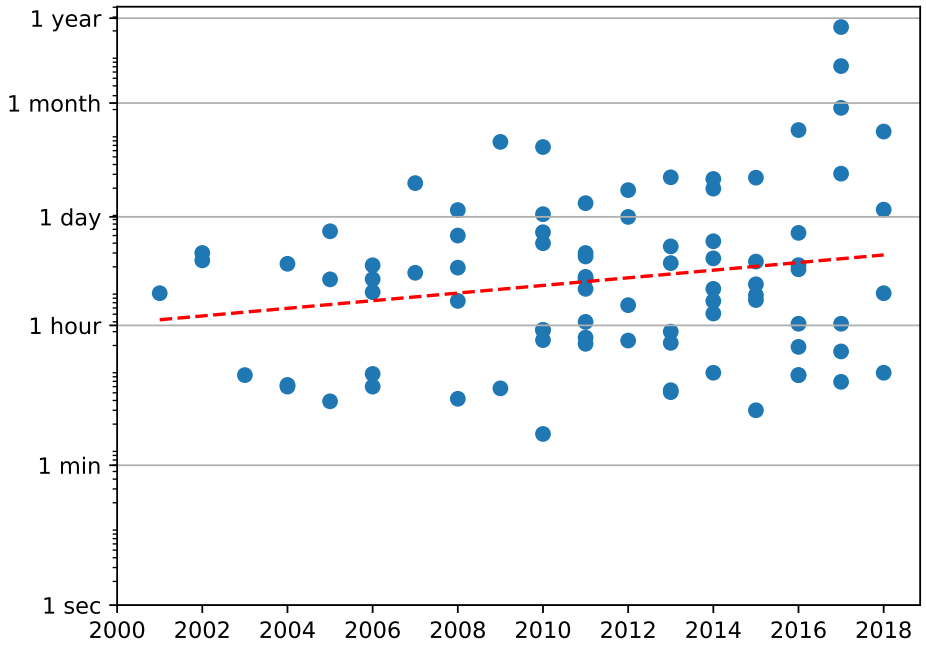


Figure 3.2: MIR dataset estimated audio length over time

accessible. Audio lengths are mostly exact, but were estimated in some cases where exact numbers were not easily accessible. For example, large datasets containing full-length tracks were assumed to have an average length of 3:30. Datasets of single notes, one-shots, chords, etc., were assumed to have an average length of one second.

We have overlaid the (logarithmic) plot with a linear trendline, showing the exponential trend of dataset growth. A similar trend can be observed in absolute storage sizes. In Figure 3.3 we have plotted dataset sizes in bytes over release year, not limited to datasets with audio.

The five largest MIR datasets, in bytes, are:

- FMA-full (Defferrard, Benzi, Vandergheynst, & Bresson, 2017), 879GB. A dump of the entire Free Music Archive.

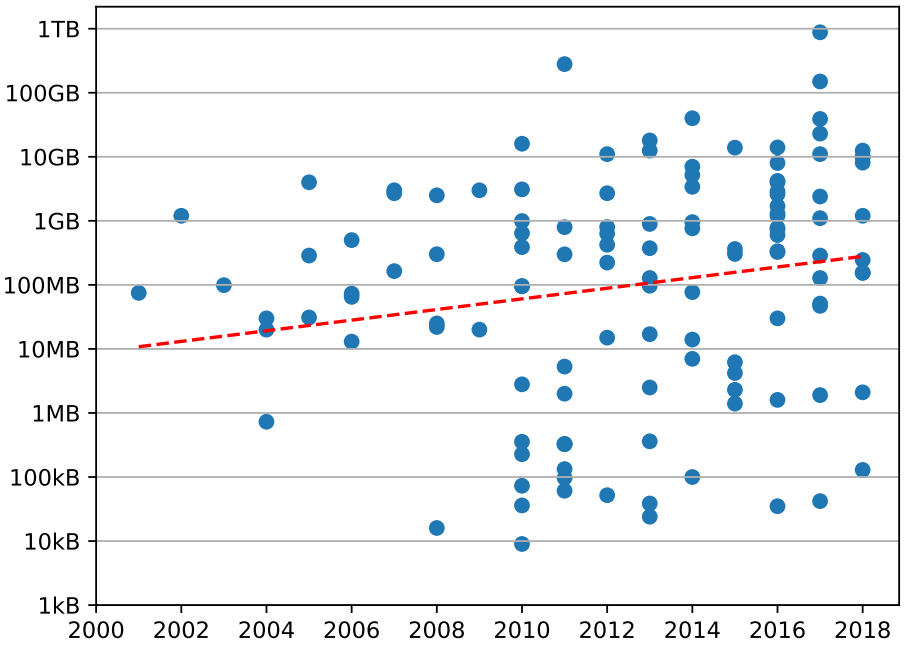


Figure 3.3: MIR dataset size in bytes

- The Million Song Dataset (Bertin-Mahieux, Ellis, Whitman, & Lamere, 2011), 280GB. Metadata and audio features for 1,000,000 songs.
- The Stanford Digital Archive of Mobile Performances (Smule, 2017). 34,000 amateur acapella recordings.
- MedleyDB (Rachel M. Bittner et al., 2014), 40GB. 122 multi-track recordings.
- NMED-T (Losorelli, Nguyen, Dmochowski, & Kaneshiro, 2017), 39GB. EEG data collected during music listening.

3.3 Structure of Music Data

Music data processing has additional layers of complexity, compared to many other types of data. A structured music database will have many types of entities, often with idiosyncratic relationships. Commercial music databases differ between organizations, but the open MusicBrainz database has a schema that is representative of contemporary digital streaming services. A visualization of the MusicBrainz database schema can be found in Figure 3.4[2].

The main musical entities in the MusicBrainz schema are:

- Artist
- Label
- Work
- Recording
- Release group
- Release
- Track

The Artist and Label entities unambiguously represent music artist and labels. A Work is what we would colloquially refer to as a "song" or "composition". Recorded music is complicated by the fact that a Work usually has multiple Recordings. A Recording is one particular recording of a Work (e.g., a live version would be a different Recording than the studio version of the same song).

A Release Group is what we refer to as "album", "single", "EP", etc. Albums are often released on different dates in different locales, sometimes with slightly different track lists. Single instances of Release Groups are called Releases. Tracks relate to Recordings like Releases relate to Release Groups, one Recording has several instantiations as Tracks across different Releases.

In the following text we will use the terms (with equivalent MusicBrainz terms in parenthesis):

[2]Via https://musicbrainz.org/doc/MusicBrainz_Database/Schema, retrieved 2019-01-15

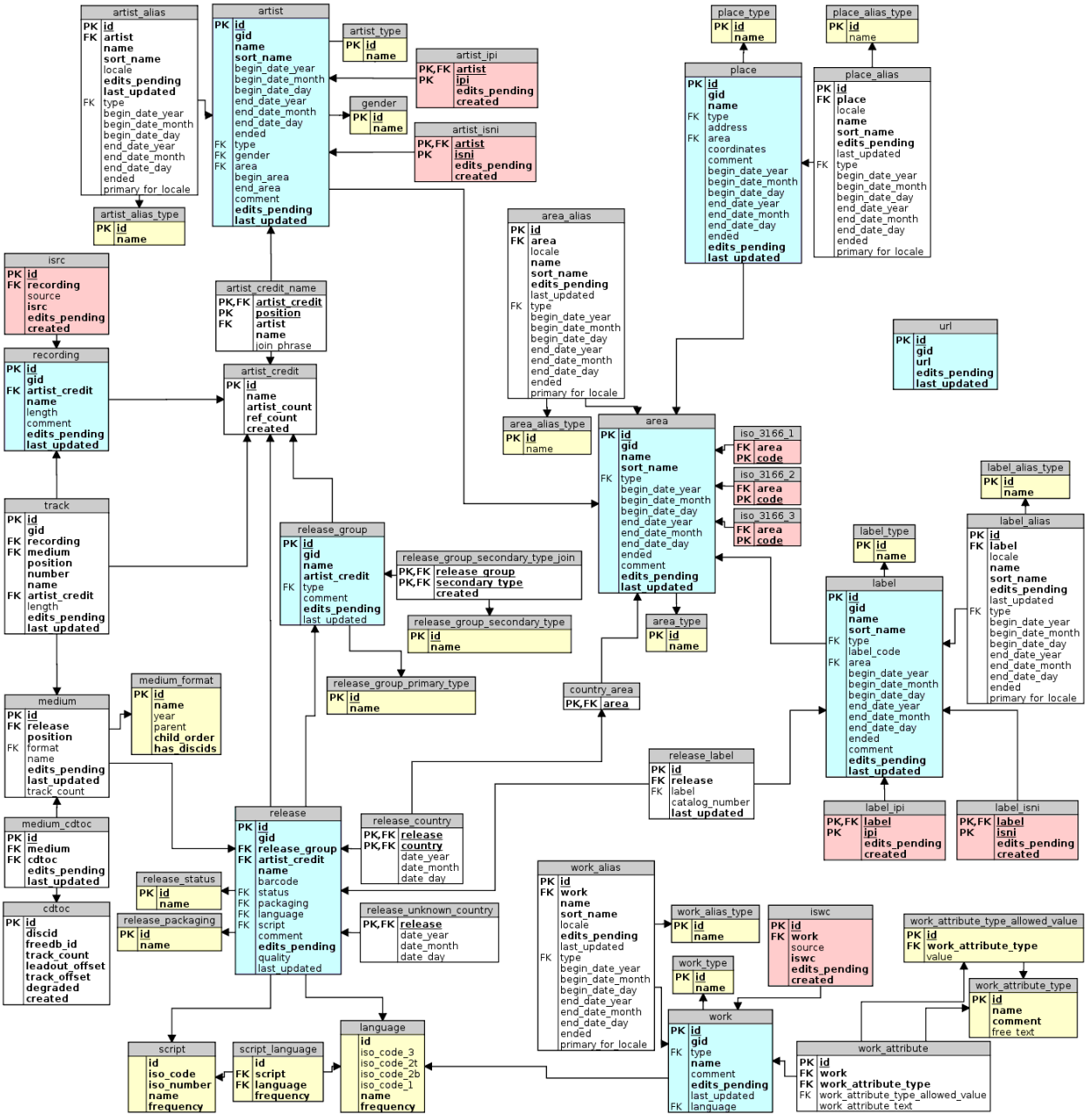


Figure 3.4: MusicBrainz database schema

- Artist (Artist)
- Track (Track)
- Recording (Recording)
- Song (Work)
- Album (Release group)

We will primarily focus on individual songs, so we will not need to differentiate between Releases and Release Groups.

3.4 Metadata Quality

Music databases are generally filled with misspellings, ambiguous transliteration, inconsistent letter casing, aliases, and other metadata quality issues that make querying these databases particularly challenging. For example, MusicBrainz has 14 aliases for The Supremes[3], including "The Surpremes", "Supremes", "Diana Ross & The Supremes", and "Diana Ross and The Supremes". The Russian composer Tchaikovsky has 79 aliases[4], mostly due to different Cyrillic transliterations.

(Freed, 2006) analyzed the causes for incorrect and ambiguous metadata, in the context of 1930s blues singer Skip James. The author tracked the mutations of track and album titles over time, and found many causes for metadata errors, such as malfunctioning printers, truncation due to sleeve space constraints, inconsistent use of numerals and letters, etc. Some artists even deliberately obfuscate metadata as a creative device. For example, the second track on Aphex Twin's EP "Window Licker" is named

$$\Delta M_i^{-1} = -\alpha \sum_{n=1}^{N} D_i[n] \left[\sum_{j \in C[i]} F_{ji}[n-1] + Fext_i[n^{-1}] \right]$$

which most music services do not even try to render (Lamere, 2008).

[3] https://musicbrainz.org/artist/c1aa2ec9-53e7-4d90-8d36-bac75832e986/aliases, retrieved 2019-01-15

[4] https://musicbrainz.org/artist/9ddd7abc-9e1b-471d-8031-583bc6bc8be9/aliases, retrieved 2019-01-15

3.5 Mining Ground Truth for Music Source Separation from Commercial Music Databases

Despite the complex structure and inherent noise of music databases, they still present a promising target for mining ground truth for music information retrieval tasks. Spotify has 40+ million songs, each song have several recordings, which in turn have several tracks. Given associated audio and metadata, one can assume that many potential MIR training datasets could be created by filtering and transforming music catalogs such as this.

The metadata matching method described here was my contribution to (E. Humphrey, Montecchio, Bittner, Jansson, & Jehan, 2017), published at ISMIR 2017. It was further refined in (Jansson et al., 2017) to also include a spectrogram-based post-processing step, which differs from Humphrey, et al.'s fingerprint-based post-processing technique.

3.5.1 Instrumental Versions of Popular Music

It is a relatively common practice for artists to release instrumental versions of songs ("instrumentals"), or sometimes even entire albums. Instrumentals, like CD booklets, provide the listener with additional media. By making it easier for the listener to sing along with the background accompaniment, or use the instrumental as less distracting background music, the instrumental brings a new level of interactivity to the listening experience.

The release of instrumental versions of tracks is not a new phenomenon, but have seen a significant increase since the 2000s. Figure 3.5 shows the relative prevalence of instrumental versions compared to the full catalog, from 1950 to 2018.

If our dataset contains both original and instrumental versions of songs, we should be able to create pairs of originals and instrumentals, creating a training dataset for vocal source separation. If we assume that the instrumental is simply the original with the vocal subtracted, we should be able to subtract the instrumental from the original to retrieve the vocal track, to use for training a vocal isolation model. Below follows a detailed explanation of our method, originally introduced in (E. Humphrey et al., 2017).

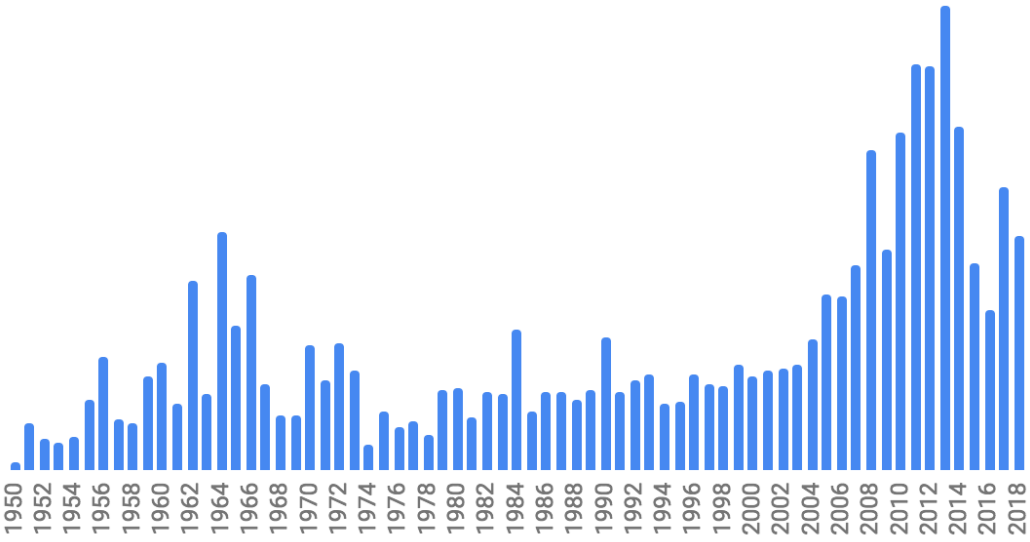


Figure 3.5: Relative proportion of releases that had instrumental versions, 1950–2018. The Y-axis has purposely been omitted for data anonymization reasons.

3.5.2 Mining Instrumentals

The first step in building a dataset of original mixtures, instrumentals, and isolated vocal tracks, is to define a function that lets us pair the original recording with the instrumental version. This is not a trivial task, both because of the size of the dataset, and the metadata noise that is inherent to music datasets.

By loading our data into BigQuery (described in Section 3.1), we are able to process this large music dataset in a matter of minutes. This enables us to construct a data processing pipeline iteratively and interactively, by testing parts of queries one step at a time, instead of having to meticulously prepare the program in advance[5].

While BigQuery largely mitigates the complexities of handing billions of database records, we still have to develop some way to address noisy meta-

[5]The difference in developer experience between older systems like Hive and modern tools like BigQuery is comprable to the difference between the hours-long compilation cycle of large C++ programs, and the immediate feedback provided by the Python "Read-Eval-Print-Loop."

data. In an ideal scenario, with perfect metadata, a simple algorithm for finding pairs of original mixtures and instrumentals would be:

1. Find a set of tracks that are not instrumentals,

   $M = \{t \in T \mid \neg\text{is\_instrumental}(t)\}$, where T is the set of all tracks

2. Find a set of tracks that are instrumentals,

   $I = \{t \in T \mid \text{is\_instrumental}(t)\}$

3. Join the sets into a set of pairs,

   $P = \{(m, i) \in M \times I \mid \text{is\_instrumental\_version\_of}(m, i)\}$

We define $\text{is\_instrumental}(t)$ as

$$\begin{aligned}\text{is\_instrumental}(t) := \text{regex\_match}(/\texttt{instrumental|karaoke}/,\\ \text{lowercase}(\text{title}(t)))\end{aligned} \tag{3.1}$$

where $/\cdot/$ represents a regular expression pattern (Friedl, 2006), $\text{regex\_match}(r, s)$ evaluates to true if the regular expression pattern r matches the string s, and $\text{lowercase}(s)$ returns the string s with all characters lowercased. In this case it returns true if and only if the lowercased title of t contains either of the strings "instrumental" or "karaoke".

This purely text-based method of applying the instrumental label has the disadvantage that non-instrumentals that have the string "instrumental" or "karaoke" in the title will falsely be classified as instrumental. We acknowledge this limitation, but we find it an acceptable trade-off, given that the alternative would have been to train a content-based instrumental classifier.

In an idealized scenario with clean metadata, the function that matches original mixes to instrumentals, $\text{is\_instrumental\_version\_of}(m, i)$, could be expressed as

$$\begin{aligned}\text{is\_instrumental\_version\_of}(m, i) :=& (\text{artist\_id}(m) = \text{artist\_id}(i))\\ &\wedge |\text{duration}(m) - \text{duration}(m)| < 1.0\\ &\wedge (\text{title}(m) = \text{title\_sans\_instrumental}(i))\end{aligned} \tag{3.2}$$

Here we make the assumption that two tracks by the same artist, with the same title (except for the string "instrumental" or "karaoke"), where the difference in duration is less than one second, are derived from the same recording. In this ideal scenario, we assume that the string "instrumental"/"karaoke" is always prefixed by " - ", .e.g. the instrumental version of "My Song Title" would be "My Song Title - Instrumental". Then we can define title_sans_instrumental(i) as

$$\text{title\_sans\_instrumental}(i) = \text{regexp\_remove}(/\texttt{ - (instrumental|karaoke)}/, \text{title}(m)) \tag{3.3}$$

where regexp_remove(r, s) removes the part of s that matches the regular expression r.

Unfortunately, as we saw in the previous section, actual music metadata is not as clean and consistent as the matching logic above requires.

Fuzzy title matching

In order to work in real-world scenarios, our matching logic should have the following properties:

- Tolerance to use of various non-alphanumeric symbols
- Equivalence of numerals and spelled out numbers
- Basic transliteration of non-English characters
- Robustness to text patterns idiomatic to music titles

We satisfy these constraints by redefining is_instrumental_version_of(x) as

$$\begin{aligned}\text{is\_instrumental\_version\_of}(m, i) := &(\text{artist\_id}(m) = \text{artist\_id}(i)) \\ &\wedge |\text{duration}(m) - \text{duration}(m)| < 1.0 \\ &\wedge (\text{fuzzy}(\text{title}(m)) = \text{fuzzy}(\text{title}(i)))\end{aligned} \tag{3.4}$$

where fuzzy is defined as the following function composition:

$$\begin{aligned}\text{fuzzy} := \text{latinize} \\ &\ggg \text{lowercase} \\ &\ggg \text{trim} \\ &\ggg \text{strip\_comments} \\ &\ggg \text{normalize\_ampersand} \\ &\ggg \text{normalize\_numerals} \\ &\ggg \text{strip\_leading\_the} \\ &\ggg \text{strip\_non\_alphanumeric}\end{aligned} \tag{3.5}$$

where $\ggg$ denotes left-to-right function composition, $(f \ggg g)(x) = g(f(x))$. For example, (latinize $\ggg$ lowercase)(s) can be read as "First latinize s, then lowercase the latinized string".

The function latinize(s) replaces non-ASCII characters in s with ASCII equivalents. For example, latinize(`Våffeljärnet`) = `Vaffeljarnet`.

The trim(s) function removes any leading and trailing whitespace from s.

strip_comments(s) is defined as

$$\text{strip\_comments}(s) := \text{regex\_split}(/\texttt{\textbackslash(|\textbackslash[| - }/, s)[0] \tag{3.6}$$

where regex_split(r, s) splits the string s based on the regular expression pattern r. In this case, this splits s at the substrings `(`, `[`, and ` - `. This is based on the observation that music producers often include comments in parenthesis, square brackets, and after a dash. For example:

$$\begin{aligned}\text{strip\_comments}(\texttt{Taki Taki (with Selena Gomez)}) &= \texttt{Taki Taki} \\ \text{strip\_comments}(\texttt{Don't Stop Me Now - Remastered}) &= \texttt{Don't Stop Me Now}\end{aligned} \tag{3.7}$$

The normalize_ampersand(s) function replaces all instances of ` & ` in s with the string ` and `. Similarly, normalize_numerals(s) replaces the strings `zero`, `one`, ..., `nine` with the numerals `0`, `1`, ..., `9`.

x	fuzzy(x)
`Seasons in the Sun`	`seasonsinthesun`
`Gold (Stupid Love)`	`gold`
`Don't Follow`	`dontfollow`
`Swim - RAMI Remix`	`swim`
`The Zone`	`zone`
`Sácate Uno`	`sacateuno`
`One For My Baby`	`1formybaby`
`Fear & Delight`	`fearanddelight`
`Ahora No [feat. Fiestero]`	`ahorano`

Figure 3.6: Examples outputs of the fuzzy($\cdot$) function

strip_leading_the(x) strips the string "the" from the beginning of x. Finally, strip_non_alphanumeric(x) deletes any non-alphanumeric characters, i.e. any characters that do not match the regular expression `/[a-z0-9]/`. This removes all spaces, as well as any punctuation.

Table 3.5.2 shows several applications of the fuzzy($\cdot$) function.

Candidate Pruning

After applying the fuzzy string matching algorithm 3.5.2, we end up with a set of candidate vocal mix/instrumental pairs. However, due to inconsistent and erroneous metadata, not all pairs have one vocal mix and one instrumental. Specific error cases include:

- Instrumentals that are not explicitly labeled as "instrumental".
- Tracks that have "instrumental" in their titles but are not actually instrumental.
- Tracks that have been mis-labeled with the title of a different track.
- Candidate pairs that are not derived from the same original recording, despite having approximately equal title and duration.

These errors could have been avoided if we had taken a content-based approach rather than the string-matching based heuristic outlined above. That would have required many-to-many comparisons between tens of millions of audio signals, an extremely computationally intensive task. However, now

that we have identified a set of pairs, some of which are false positives, we can apply content-based methods as a filter to prune the candidate pool. We implement the filtering system as a DataFlow pipeline (as introduced in Section 3.1).

We take a fairly cautious approach to filtering, optimized for high quality rather than quantity, since the candidate pool is large enough to begin with. Pruning is performed by filtering out pairs who's audio spectrograms are either too different, or too similar.

First we strip out any silent frames from the beginning and end of the audio signals for the two tracks in the pair. We then convert the audio to a mixture spectrogram $\mathbf{X}$ and a $\mathbf{Y}$ instrumental spectrogram. We compute the spectrogram difference for pair i as

$$d_i = \sum |\mathbf{X} - \mathbf{Y}| \tag{3.8}$$

In the absence of validation data, we empirically define a minimum difference d_{min} and maximum difference d_{max}. Any pair i where $d_{min} \leq d_i < d_{max}$ is kept in the final dataset. These thresholds were chosen by manually checking a random sample of 100 pairs, such that all pairs consisted of one vocal mixture track and one instrumental.

After this process is complete, we arrive at a dataset with 20,000 examples.

3.6 Summary

This chapter introduced a novel method for extracting structured source separation training data from large unstructured commercial music catalogs. Processing these terabyte-scale datasets would not be possible without recent developments in distributed data storage and processing, and we began this chapter by presenting a number of such tools. We showed how dataset sizes have grown exponentially in the past decades, in machine learning generally, as well as in MIR specifically. We then used MusicBrainz as an example of the schematic complexities of music databases, and highlighted some typical types of noise that are common in such databases. For the remainder of the chapter we gradually derived an algorithm for finding pairs of original

mixtures and instrumentals in music databases, taking into account various types of noise and metadata inconsistencies.

Chapter 4

Vocal Separation from Commercial Recordings

4.1 Motivation

Vocals are prominent in the music of practically every known culture. It has even been theorized that music emerged from early humans using the voice to convey emotions (Juslin, 2003). While vocals have been extensively studied in the MIR literature[1], very little research has investigated the relative importance of vocals to other aspects of modern popular music.

To remedy this, we performed an interdisciplinary study in collaboration with Andrew Demetriou at the University of Delft (Demetriou, Jansson, Kumar, & Bittner, 2018). Our belief was that vocals mattered more than other aspects of music in the minds of music listeners, but we wanted to collect unbiased evidence.

We first carried out a quantitative analysis of several proprietary data sources at Spotify, expecting to find significant evidence of the importance of vocals. We started by looking at playlist tags and search queries. These datasets are composed of user-provided descriptions of the music they listen to. We grouped playlist tags and search queries into semantic categories, and compared their relative frequencies. Surprisingly, this experiment yielded a neg-

[1] A review of singing-voice analysis can be found in (Eric J Humphrey et al., 2018)

ative result: vocals were not commonly used by end-users to describe their music collections (Figure 4.1, top and middle).

We then turned our attention to how professional music writers describe music. Artist biographies on Spotify are authored by the artist themselves, or by professional writers. We analyzed these biographies, comparing the term frequencies to Wikipedia. The top terms, unambiguously music-related, were clustered into groups, as in the playlist tag analysis. Our hypothesis was that vocal-related terms would appear at or near the top of the list of terms. But again, the result was negative. Vocal terms appeared in the list of top terms, but not near the top of the list (Figure 4.1, bottom).

Perhaps our hypothesis was invalid, and users are not interested in vocals. Or, perhaps vocals are so ubiquitous that they are not explicitly mentioned in playlist titles and artist biographies. Instead of searching for answers quantitatively in data, we decided to ask listeners directly.

Two surveys were sent out to 50,000 random Spotify users in English-speaking countries (different cohorts for different surveys). We received 626 responses to the first survey and 531 responses to the second.

In the first survey, we asked the open ended question, "When you listen to music, what things about the music do you notice?" The answers from this survey were printed on physical cards and sorted by the research team into emerging categories, such as "Genre", "Musicianship", "Lyrics", "Vocals", etc.

The second survey then presented the users with a shuffled list of the categories derived from the first survey. The respondents where asked to rank these categories based on how important they were to the users' perception of a song.

This study, in contrast to the two data analyses we had performed previously, provided strong corroboration of our initial hypothesis. The most important category, in the minds of listeners, was "Emotion/mood". This is consistent with research from music psychology which suggests that the primary purpose of music is communication of emotions (Juslin, 2003).

The second and third top-most categories were "Voice" and "Lyrics". This result shows that vocals are in fact highly salient in the minds of listeners. Statistical significance was verified using Robust Rank Aggregation (Kolde,

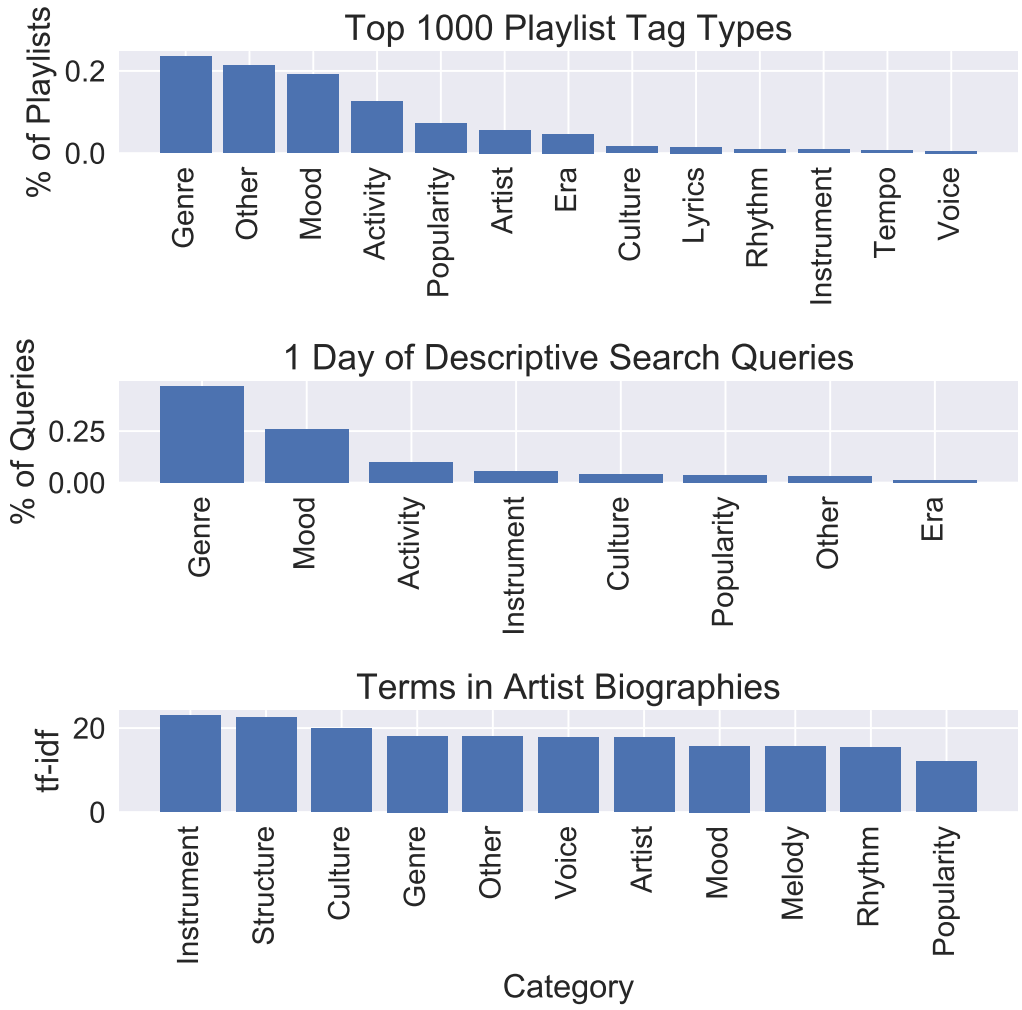


Figure 4.1: (Top) Percentage of playlists containing one of the top 1000 tags corresponding to each tag category. (Middle) Percentage of descriptive search queries corresponding to each tag category, sampled from one day of search data. (Bottom) tf-idf for each term category in artist biographies compared with wikipedia term frequencies. Adapted from (Demetriou, Jansson, Kumar, & Bittner, 2018)

Laur, Adler, & Vilo, 2012). The full ordered list of categories can be found in Table 4.1.

Broad Semantic Cat	Description	Borda score	p-value
Emotion/mood	How it makes you feel - the emotions/mood	4641	**<0.001**
Voice	Voice/vocals	3688	**<0.001**
Lyrics	Lyrics	3656	**<0.001**
Beat/rhythm	Beat/rhythm	3460	**<0.001**
Structure/Complexity	How it's composed, the hook, the structure	2677	1.000
Musicianship	Skill of the musicians, musicianship	2583	1.000
Melody	The main melody	2577	1.000
Sound	The "sound", or the recording quality	2406	1.000
Specific Artist	The specific artist	2349	1.000
Genre	The specific genre	2293	1.000
Instrumentation	The musical instruments (e.g. drums, bass, guitar)	2084	1.000
Tempo/BPM	How fast or slow the song is	1828	1.000
Harmony	Harmony	1763	1.000
Chords	The chords	1086	1.000
Popularity/Novelty	How popular or unique it is	777	1.000

Table 4.1: Broad semantic categories and their clarifying descriptions created during Study 1, ordered by rankings from Study 2. The Borda scores and p-values (as reported by Robust Rank Aggregation) from Study 2 are reported in columns 3 and 4, respectively. Statistically significant p-values are shown in bold. p-values of 1.000 indicate that the ranking is no different from random. Adapted from (Demetriou, Jansson, Kumar, & Bittner, 2018).

The importance of vocals and lyrics motivates further work on vocal processing and analysis. Listeners self-report that they care more about the vocals than any other aspect of music, with the exception of the overall mood. This suggests that altering the vocals of a piece of music would have a high impact on listeners' impression perception of the piece. It also indicates that in order to better understand music perception, we should study the contents of the vocal signal. Both vocal processing and vocal signal analysis require the ability to isolate the vocal signal from the accompanying instrument sources, which brings us to the main focus of this chapter: Vocal source separation.

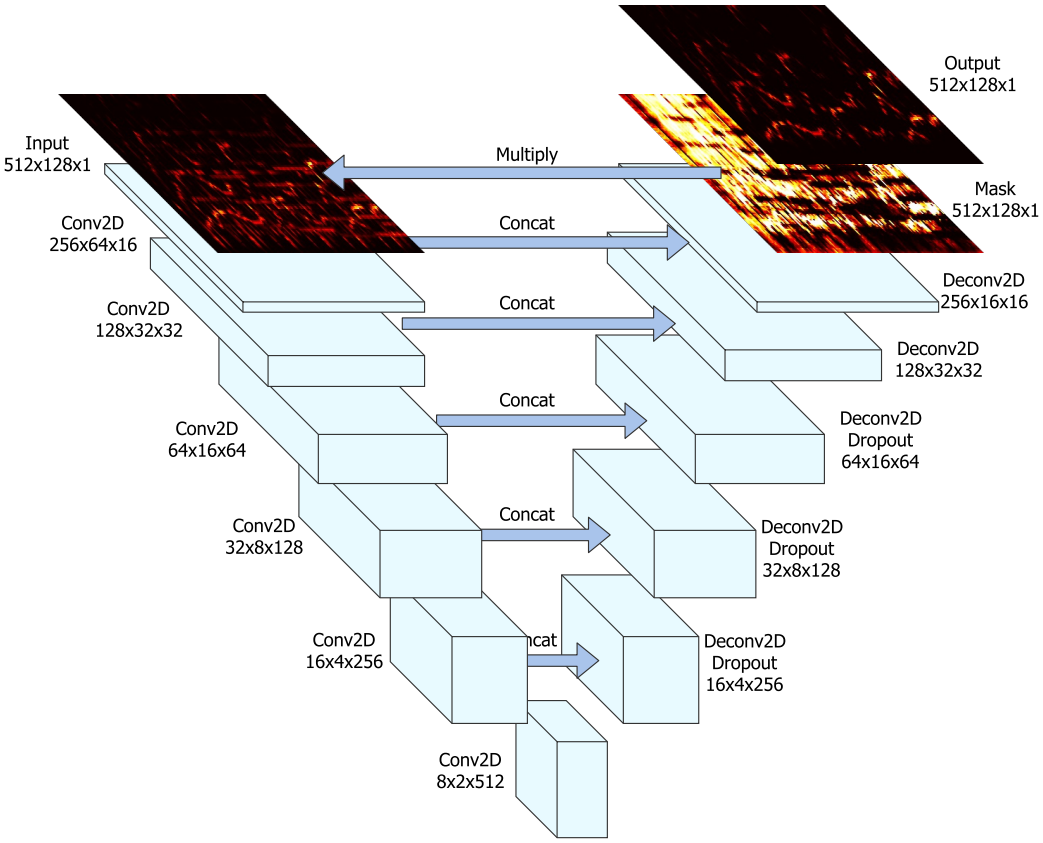


Figure 4.2: Network Architecture

4.2 Methodology

The remainder of this chapter is adapted from our paper (Jansson et al., 2017), presented at the ISMIR 2017 conference in Suzhou, China.

4.2.1 Intuition

This work adapts the *U-Net* (Ronneberger et al., 2015) architecture to the task of vocal separation. The architecture was introduced in biomedical imaging, to improve precision and localization of microscopic images of neuronal structures.

The U-Net builds upon the *fully convolutional network* (Long, Shelhamer, & Darrell, 2015) and is similar to the *deconvolutional network* (Noh, Hong, & Han, 2015). In a deconvolutional network, a stack of convolutional layers — where each layer halves the size of the image but doubles the number of channels — encodes the image into a small and deep representation. That encoding is then decoded to the original size of the image by a stack of upsampling layers. Upsampling can be performed in several ways (linear interpolation, max-unpooling, etc.). In the work presented below we use strided transpose convolutions (Dumoulin & Visin, 2016).

U-Nets have has been used in many domains outside of medical applications, for example in (Isola, Zhu, Zhou, & Efros, 2017) who uses U-Nets to perform pixel-to-pixel translations of images. The image translations in Isola, et al., ranged from image colorization to generating maps from aerial images. While their architecture used U-Nets as part of a generative adversarial network, the U-Net itself was found to yield a higher level of detail compared to deconvolutional encoder-decoders. For this reason we decided to investigate the use of U-Nets for vocal source separation. We treat source separation in the time-frequency domain as an image-to-image translation, where we map mixture spectrograms to isolated source spectrograms.

In the reproduction of a natural image, displacements by just one pixel are usually not perceived as major distortions. In the frequency domain however, even a minor linear shift in the spectrogram has disastrous effects on perception: this is particularly relevant in music signals, because of the logarithmic perception of frequency. Moreover, a shift in the time dimension can become audible as jitter and other artifacts. Therefore, it is crucial that

the reproduction preserves a high level of detail. The U-Net adds additional skip connections between layers at the same hierarchical level in the encoder and decoder. This allows low-level information to flow directly from the high-resolution input to the high-resolution output.

4.2.2 Architecture

Let $\mathbf{X}$ denote the magnitude of the spectrogram of the original, mixed signal, that is, of the audio containing both vocal and instrumental components. Let $\mathbf{Y}$ denote the magnitude of the spectrograms of the target audio; the latter refers to either the vocal ($\mathbf{Y}_v$) or the instrumental ($\mathbf{Y}_i$) component of the input signal. The estimated magnitude spectrogram is denoted by $\hat{\mathbf{Y}}$. Vocal and instrumental estimates are denoted by $\hat{\mathbf{Y}}_v$ and $\hat{\mathbf{Y}}_i$ respectively.

The goal of the neural network architecture is to predict the vocal and instrumental components of its input indirectly: the output of the final decoder layer is a soft mask that is multiplied element-wise with the mixed spectrogram to obtain the final estimate. Given that the network outputs a mask $\mathbf{M}$, the predicted source spectrogram is computed as

$$\hat{\mathbf{Y}} = \mathbf{X} \otimes \mathbf{M} \tag{4.1}$$

where $\otimes$ denotes element-wise multiplication.

Figure 4.2 outlines the network architecture. In this work, we choose to train two separate models for the extraction of the instrumental and vocal components of a signal, to allow for more divergent training schemes for the two models in the future.

Training

The loss function used to train the model is the L_1 norm of the difference of the target spectrogram and the masked input spectrogram:

$$\mathcal{L} = |\mathbf{Y} - \hat{\mathbf{Y}}|_1 \tag{4.2}$$

As discussed in Section 2.4.8, the L_1 loss has been shown to yield better results in spectrogram reproduction, as well as "crisper" outputs when applied in the image domain.

Two independent U-Nets are trained to predict vocal and instrumental spectrogram masks, respectively. The reason for this choice, as opposed to treating estimated vocal spectrograms as the inverse of the estimated instrumental, is two-fold: We initially planned to replicate the GAN architecture from (Isola et al., 2017), which would have meant that the instrumental and vocal model would diverge. Even though we decided against implementing the GAN architecture, the vocal and instrument training data prevented us from training a single network for both tasks, since we apply half-wave rectification to the vocal residual.

Network Architecture Details

Our implementation of U-Net is similar to that of (Isola et al., 2017). Each encoder layer consists of a strided 2D convolution of stride 2 and kernel size 5x5, batch normalization, and *leaky rectified linear units* (ReLU) with leakiness 0.2. The leaky ReLU is defined as

$$\text{lrelu}(x) := \begin{cases} x & \text{if } x > 0 \\ \lambda x & \text{otherwise} \end{cases} \tag{4.3}$$

where λ is the leakiness factor.

In the decoder we use strided deconvolution (sometimes referred to as transposed convolution) with stride 2 and kernel size 5x5, batch normalization, plain ReLU, and use 50% dropout to the first three layers, as in (Isola et al., 2017). The plain ReLU is defined as

$$\text{relu}(x) := \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

In the final layer we apply a sigmoid activation function, $\sigma(\cdot)$, element-wise to the output matrix:

$$\sigma(x) := \frac{1}{1 + e^{-x}} \tag{4.5}$$

While (Isola et al., 2017) used tanh activation in the final layer, we opted for the sigmoid function since the spectrogram output should fall in the range $[0, 1]$.

The model is trained using the ADAM (Kingma & Ba, 2014) optimizer with a learning rate of 0.0002 and a β_1 value of 0.5.

Given the heavy computational requirements of training such a model, we first downsample the input audio to 8192 Hz in order to speed up processing. We then compute the Short Time Fourier Transform with a window size of 1024 and hop length of 768 frames, and extract patches of 128 frames (roughly 11 seconds) that we feed as input and targets to the network. The magnitude spectrograms are normalized to the range $[0, 1]$.

Audio Signal Reconstruction

The neural network model operates exclusively on the magnitude of audio spectrograms. The audio signal for an individual (vocal/instrumental) component is rendered by constructing a spectrogram: the output magnitude is given by applying the mask predicted by the U-Net to the magnitude of the original spectrum, while the output phase is that of the original spectrum, unaltered. Experimental results presented below indicate that such a simple methodology proves effective.

4.2.3 Dataset

Our dataset is collected from the Spotify music collection, using the methodology outlined in Chapter 3. The final dataset contains approximately 20,000 track pairs, resulting in almost two months worth of continuous audio. To the best of our knowledge, this is the largest training data set ever applied to musical source separation.

Table 4.2 shows the relative distribution of the most frequent genres in the dataset, obtained from the catalog metadata.

We split the data into 90% training and 10% validation. For testing we use different datasets entirely (described below in Section 4.3), ensuring that the test set is completely out-of-distribution with respect to the training set.

4.3 Evaluation

We compare the proposed model to the *Chimera* model (Luo, Chen, Hershey, Roux, & Mesgarani, 2017a) that produced the highest evaluation scores in

Genre	Percentage
Pop	26.0%
Rap	21.3%
Dance & House	14.2%
Electronica	7.4%
R&B	3.9%
Rock	3.6%
Alternative	3.1%
Children's	2.5%
Metal	2.5%
Latin	2.3%
Indie Rock	2.2%
Other	10.9%

Table 4.2: Training data genre distribution

the 2016 MIREX Source Separation campaign ("MIREX 2016:Singing Voice Separation Results", 2016); we make use of their web interface[2] to process audio clips. It should be noted that the Chimera web server is running an improved version of the algorithm that participated in MIREX, using a hybrid "multiple heads" architecture that combines deep clustering with a conventional neural network (Luo et al., 2017a).

For evaluation purposes we built an additional baseline model; it resembles the U-Net model but without the skip connections, essentially creating a convolutional encoder-decoder, similar to the "Deconvnet" (Noh et al., 2015).

Both the U-Net model and the baseline model were trained on the large, proprietary Spotify dataset, whereas the Chimera model was trained on the smaller DSD100 dataset.

We evaluate the three models on the standard iKala (Chan et al., 2015) and MedleyDB dataset (Rachel M. Bittner et al., 2014). The iKala dataset has been used as a standardized evaluation for the annual MIREX campaign for several years, so there are many existing results that can be used for comparison. MedleyDB on the other hand was recently proposed as a higher-quality, commercial-grade set of multi-track stems. We generate isolated instrumental and vocal tracks by weighting sums of instrumental/vocal stems

[2] danetapi.com/chimera

	U-Net	Baseline	Chimera
NSDR Vocal	**11.094**	8.549	8.749
NSDR Instrumental	**14.435**	10.906	11.626
SIR Vocal	**23.960**	20.402	21.301
SIR Instrumental	**21.832**	14.304	20.481
SAR Vocal	**17.715**	15.481	15.642
SAR Instrumental	**14.120**	12.002	11.539

Table 4.3: iKala mean scores

	U-Net	Baseline	Chimera
NSDR Vocal	**8.681**	7.877	6.793
NSDR Instrumental	**7.945**	6.370	5.477
SIR Vocal	**15.308**	14.336	12.382
SIR Instrumental	**21.975**	16.928	20.880
SAR Vocal	**11.301**	10.632	10.033
SAR Instrumental	**15.462**	15.332	12.530

Table 4.4: MedleyDB mean scores

by their respective mixing coefficients as supplied by the MedleyDB Python API[3]. We limit our evaluation to clips that are known to contain vocals, using the melody transcriptions provided in both iKala and MedleyDB.

The following BSS Eval functions are used to measure performance: Signal-To-Distortion Ratio (SDR), Signal-to-Interference Ratio (SIR), Signal-to-Artifact Ratio (SAR), and Normalized Signal-to-Distortion Ratio (NSDR). We compute performance measures using the *mir_eval* toolkit (Raffel et al., 2014).

Table 4.3 and Table 4.4 show that the U-Net significantly outperforms both the baseline model and Chimera on all three performance measures for both datasets. In Figure 4.3 we show an overview of the distributions for the different evaluation measures.

Assuming that the distribution of tracks in the iKala hold-out set used for MIREX evaluations matches those in the public iKala set, we can compare our results to the participants in the 2016 MIREX Singing Voice Separa-

[3] github.com/marl/medleyDB

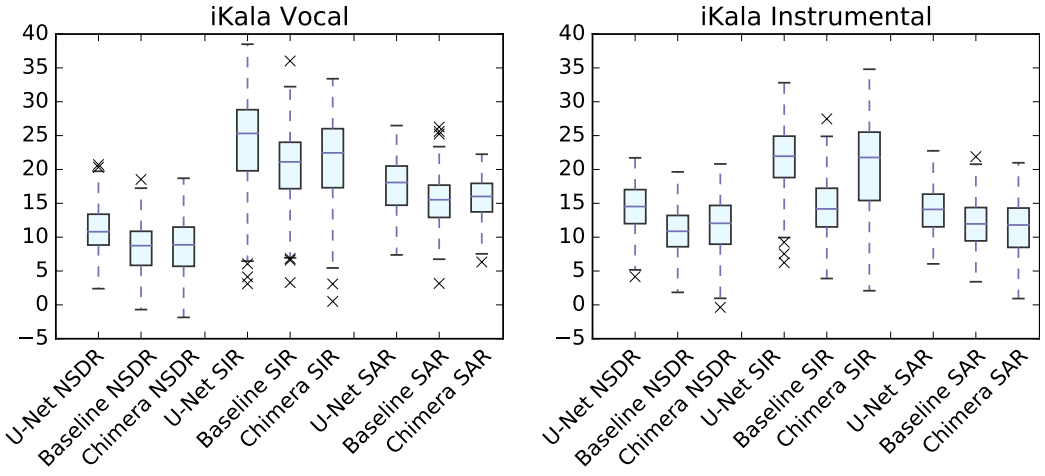


Figure 4.3: iKala vocal and instrumental scores. The top and bottom of boxes represent first (Q1) and third quantiles (Q3), with the middle line corresponding to the median. "Whiskers" are set at $Q1 - 1.5 * IQR$ and $Q3 + 1.5 * IQR$, where IQR is the inter-quantile range, Q3-Q1. Crosses outside of the whiskers indicate outliers.

tion task.[4] "LCP2", "LCP1", and "MC2" refer to the top-scoring models that were submitted to MIREX 2016. Table 4.5 and Table 4.6 show NSDR scores for our models compared to the best performing algorithms of the 2016 MIREX campaign.

In order to assess the effect of the U-Net's skip connections, we can visualize the masks generated by the U-Net and baseline models. From Figure 4.4 it is clear that while the baseline model captures the overall structure, there is a lack of fine-grained detail observable.

4.3.1 Subjective Evaluation

Emiya et al. introduced a protocol for the subjective evaluation of source separation algorithms (Emiya, Vincent, Harlander, & Hohmann, 2011). They suggest asking human subjects four questions that broadly correspond to the SDR/SIR/SAR measures, plus an additional question regarding the overall sound quality.

[4] http://www.music-ir.org/mirex/wiki/2016:Singing_Voice_Separation_Results

Model	Mean	SD	Min	Max	Median
U-Net	**14.435**	3.583	4.165	21.716	**14.525**
Baseline	10.906	3.247	1.846	19.641	10.869
Chimera	11.626	4.151	-0.368	20.812	12.045
LCP2	11.188	3.626	2.508	19.875	11.000
LCP1	10.926	3.835	0.742	19.960	10.800
MC2	9.668	3.676	-7.875	22.734	9.900

Table 4.5: iKala NSDR Instrumental, MIREX 2016

Model	Mean	SD	Min	Max	Median
U-Net	**11.094**	3.566	2.392	20.720	**10.804**
Baseline	8.549	3.428	-0.696	18.530	8.746
Chimera	8.749	4.001	-1.850	18.701	8.868
LCP2	6.341	3.370	-1.958	17.240	5.997
LCP1	6.073	3.462	-1.658	17.170	5.649
MC2	5.289	2.914	-1.302	12.571	4.945

Table 4.6: iKala NSDR Vocal, MIREX 2016

As we asked these four questions to subjects without music training, our subjects found them ambiguous, e.g., they had problems discerning between the absence of artifacts and general sound quality. For better clarity, we distilled the survey into the following two questions in the vocal extraction case:

- Quality: "Rate the vocal quality in the examples below."
- Interference: "How well have the instruments in the clip above been removed in the examples below?"

For instrumental extraction we asked similar questions:

- Quality: "Rate the sound quality of the examples below relative to the reference above."
- Extracting instruments: "Rate how well the instruments are isolated in the examples below relative to the full mix above."

Data was collected using CrowdFlower[5], an online platform where humans

[5] www.crowdflower.com

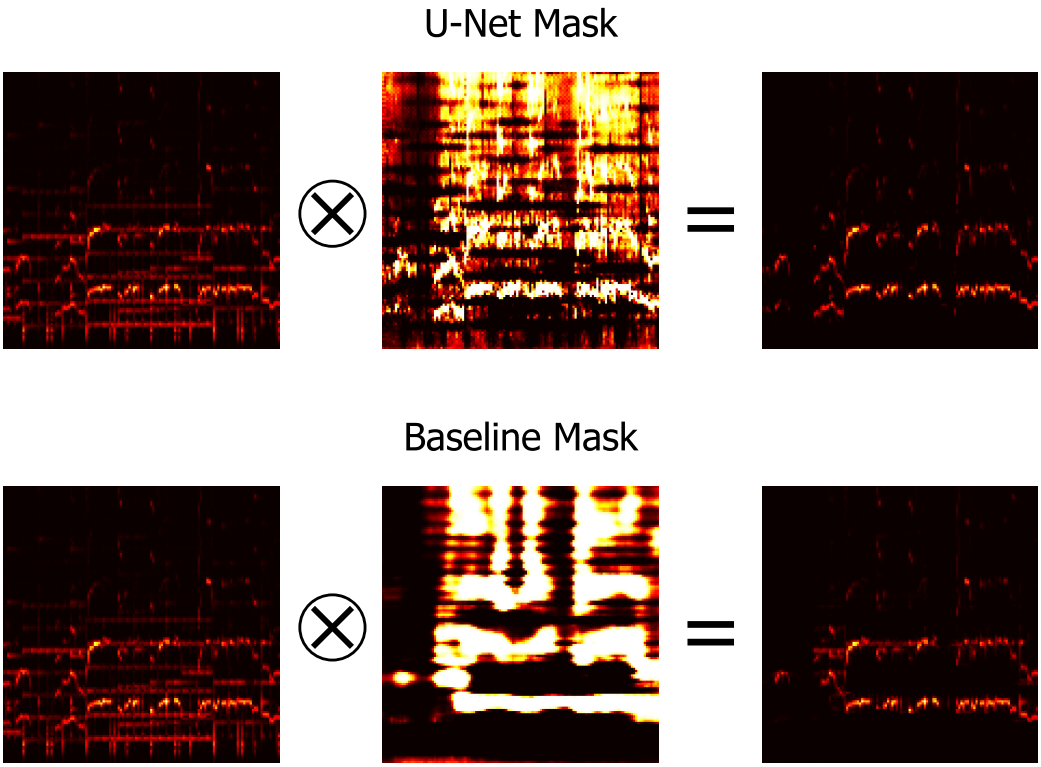


Figure 4.4: U-Net and baseline masks

carry out micro-tasks, such as image classification, simple web searches, etc., in return for small per-task payments.

In our survey, CrowdFlower users were asked to listen to three clips of isolated audio, generated by U-Net, the baseline model, and Chimera. The order of the three clips was randomized. Each question asked one of the Quality and Interference questions. In the Interference question we also included a reference clip. The answers were given according to a 7 step Likert scale(Likert, 1932), ranging from "Poor" to "Perfect". Figure 4.5 is a screen capture of a CrowdFlower question.

To ensure the quality of the collected responses, we interspersed the survey with "control questions" that the user had to answer correctly according to a predefined set of acceptable answers on the Likert scale. Users of the platform are unaware of which questions are control questions. If they are

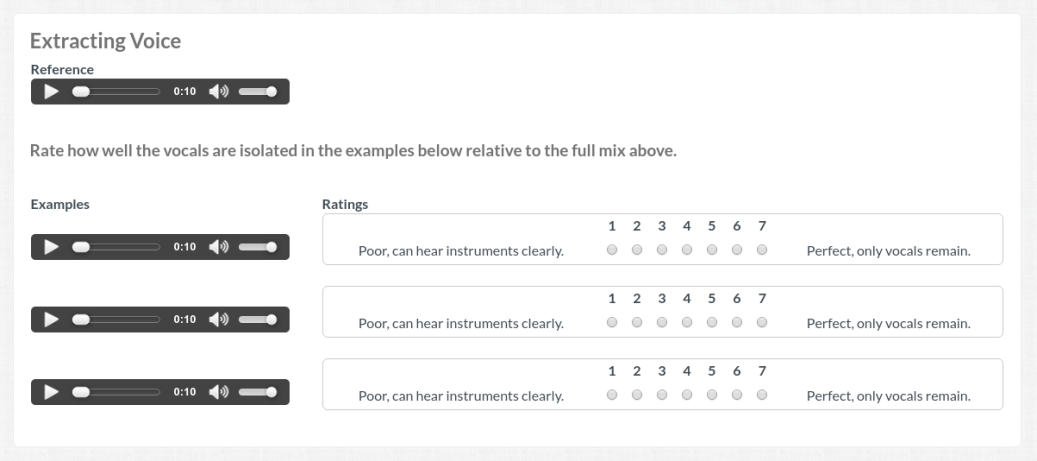


Figure 4.5: CrowdFlower example question

answered incorrectly, the user is disqualified from the task. A music expert external to our research group was asked to provide acceptable answers to a number of random clips that were designated as control questions.

For the survey we used 25 clips from the iKala dataset and 42 clips from MedleyDB. We had 44 respondents and 724 total responses for the instrumental test, and 55 respondents supplied 779 responses for the voice test[6].

Figure 4.6 shows mean and standard deviation for answers provided on CrowdFlower. The U-Net algorithm outperforms the other two models on all questions, though the differences are not statistically significant.

4.4 Conclusion and Future Work

We have explored the U-Net architecture in the context of singing voice separation, and found that it brings clear improvements over the state-of-the-art. The benefits of low-level skip connections were demonstrated by comparison to plain convolutional encoder-decoders.

A factor that we feel should be investigated further is the impact of large training data: work remains to be done to correlate the effects of the size of

[6]Some of the audio clips we used for evaluation can be found on http://mirg.city.ac.uk/codeapps/vocal-source-separation-ismir2017

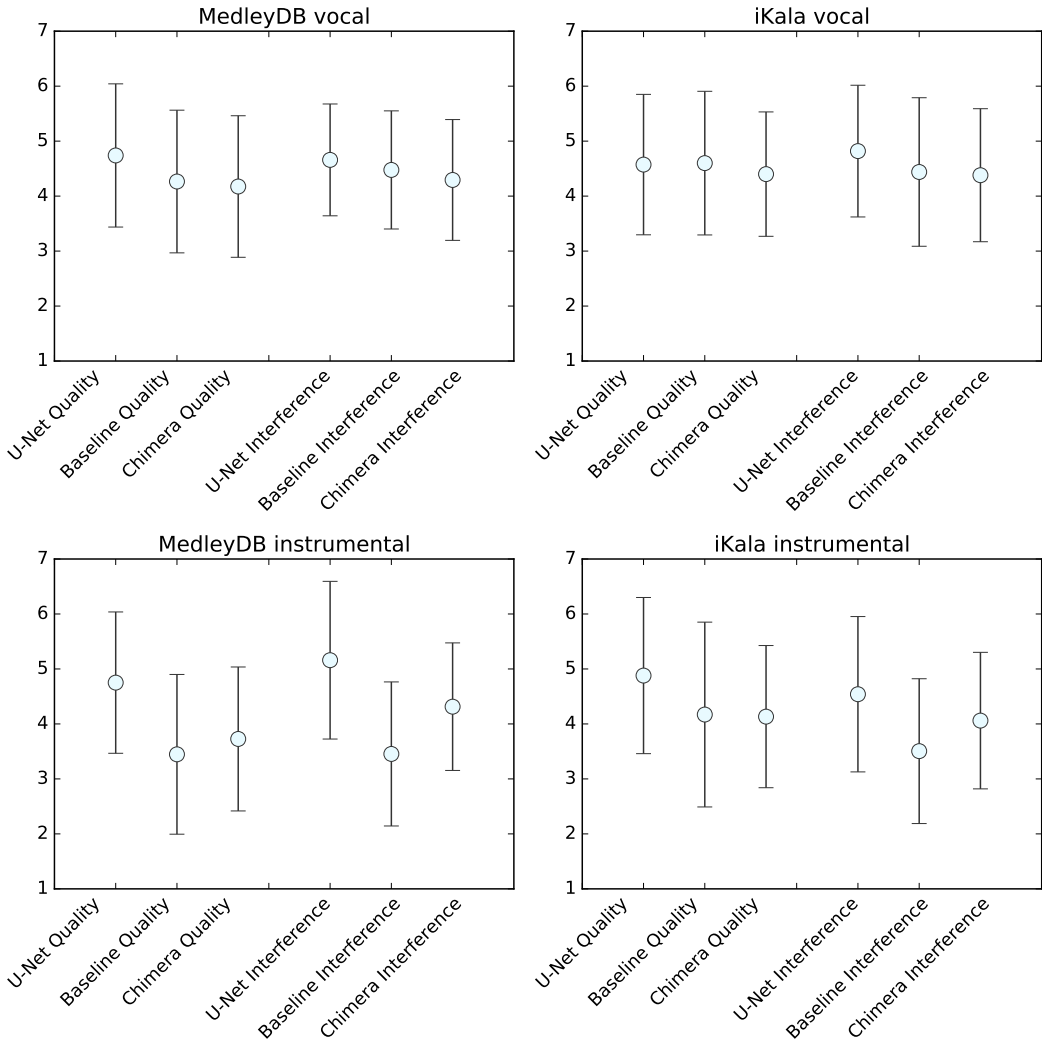


Figure 4.6: CrowdFlower evaluation results. Circles represent mean values, and lines extend one standard deviation in each direction.

the training dataset to the quality of source separation.

We have observed some examples of poor separation on tracks where the vocals are mixed at lower-than-average volume, uncompressed, suffer from extreme application of audio effects, or otherwise unconventionally mixed. Since the training data consisted exclusively of commercially produced recordings, we hypothesize that our model has learned to distinguish the kind of voice typically found in commercial pop music. We plan to investigate this further by systematically analyzing the dependence of model performance on the mixing conditions.

Finally, subjective evaluation of source separation algorithms is an open research question. Several alternatives exist to 7-step Likert scale, e.g. the ITU-R scale (Thiede et al., 2000). Tools like CrowdFlower allow us to quickly roll out surveys, but care is required in the design of question statements.

4.4.1 Recent developments

Since the publication of the work that we describe in this section, a number of articles and been published that build on this work. Below follow summaries of some notable examples:

(Stoller et al., 2018) trained a model for source separation in the time-domain, by combining the Wavenet (van den Oord et al., 2016) with the U-Net. This *Wave-U-Net* architecture has also been applied to speech enhancement (Macartney & Weyde, 2018) and lyric alignment (Stoller, Durand, & Ewert, 2019).

In (Choi et al., 2019), the authors train a speech enhancing U-Net in the complex domain. We investigate this architecture for music separation in Chapter 6.

Researchers at Deezer have developed an open-source software library for vocal source separation, "Spleeter" (Hennequin, Khlif, Voituret, & Moussallam, 2020), which has similarities with the model described in this chapter.

(Meseguer-Brocal & Peeters, 2019) introduced an instrument-specific conditioning vector for multi-instrument separation. This conditioning vector represents the embedding of a single instrument, and can be understood as a query key which the network uses to select the instrument signal to output.

Chapter 5

Joint Vocal Removal and Vocal Pitch Tracking

5.1 Introduction

In the previous chapter we developed a method for isolating vocals from a musical mixture. Given this newly improved capability, we would like to investigate potential applications. In the introduction we hypothesized that automatic transcription systems could benefit from vocal separation, based on the intuition that a less noisy input signal should result in less noisy transcriptions. In this chapter we will discuss ways of combining automatic vocal separation with the estimation of vocal fundamental frequency, f_0. This chapter is based on our paper (Jansson, Bittner, Ewert, & Weyde, 2019), presented at the 2019 EUSIPCO conference. We will tackle the challenging multi-f_0 case, where the f_0 estimator can predict one or many simultaneous pitches at each time frame.

Interdependencies between the two tasks have been demonstrated in literature. For example, (Virtanen et al., 2008) reported improved performance on one task by integrating information obtained via a method designed for the other. These dependencies can be modeled in different ways. For example, the fundamental frequency can be estimated and employed as side information in the separation process (Li & Wang, 2007; Virtanen et al., 2008). Alternatively, an estimate of the clean singing voice can be used as

input to simplify the estimation of the fundamental frequency of the voice (Durrieu, Richard, David, & Févotte, 2010). Given that both directions were successfully exploited in the past, it remains unclear how these dependencies should be modeled, especially given that prior work typically solves one task independently of the other and conditions the other on the resulting point estimate — eliminating the potential benefits of circular influence. Attempts have been made to learn both tasks iteratively, in an alternating fashion (Hsu et al., 2012). Learned joint pitch and separation models have been proposed for speech (X. Zhang, Zhang, Nie, Gao, & Liu, 2016). However, we are not aware of prior work in music that jointly performs vocal separation and vocal melody estimation.

Another important aspect of combining models for different tasks is the issue of a resulting mismatch between the data distributions at training and test time. Consider performing vocal f_0 estimation by first applying a source separation algorithm to a mixed signal and then running a standard pitch tracker. Pitch trackers are typically designed based on two assumptions: the signal being pitch tracked has little to no noise or interference, and is monophonic. Due to these assumptions, pitch tracker performance can suffer from artifacts introduced by source separation, which can include residual sounds that are pitched or other interference in the background. Figure 5.1 shows the performance of three different pitch tracking algorithms — Crepe (Kim, Salamon, Li, & Bello, 2018), pYIN (Mauch & Dixon, 2014), and Deep Salience (Rachel M. Bittner, McFee, Salamon, Li, & Bello, 2017) on both clean and source-separated vocals in the iKala dataset (Chan et al., 2015).

In this work we use U-Nets for both vocal separation and fundamental frequency estimation. U-Nets were first proposed for f_0 estimation in (Doras, Esling, & Peeters, 2019).

The Overall Accuracy (`OA`) and Raw Pitch Accuracy (`RPA`) metrics (see Section 5.4 for details about these metrics) considerably decrease for all three algorithms when applied to source separated instead of clean vocals. Additionally, many source separation systems which isolate the singing voice will isolate *all* singing voices, which can break the monophonic requirement and severely reduce the accuracy of pitch trackers.

Given this interdependent nature of the two tasks, it is an open question how to design a joint estimator, and whether such a system would actually yield benefits. As a first contribution in this chapter, we demonstrate that

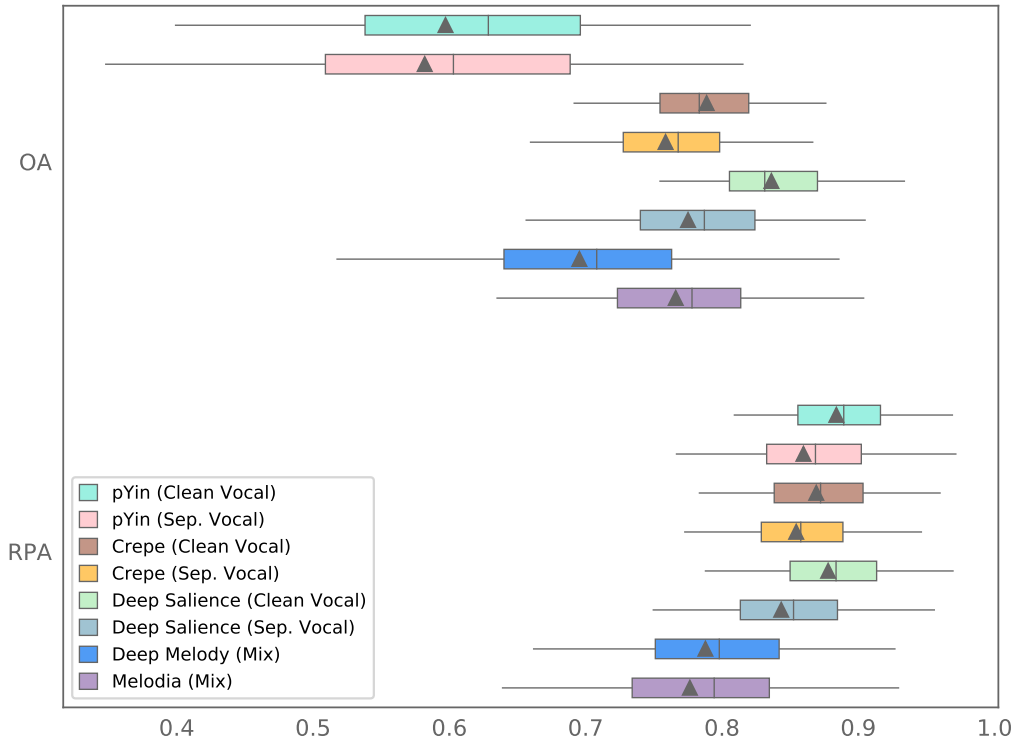


Figure 5.1: Performance of pYIN (Mauch & Dixon, 2014), Crepe (Kim, Salamon, Li, & Bello, 2018), and Deep Salience (Rachel M. Bittner et al., 2014) on the iKala (Chan et al., 2015) dataset. pYIN and Crepe are run on clean iKala vocals and on vocals computed by running a source separation algorithm ("Source Only" in Figure 5.3) from iKala mixtures as input. Deep Melody and Melodia are run on iKala mixtures as input. Boxplots show the distribution of `OA` and `RPA` over each track in the dataset. The left and right borders of boxes indicate first (Q1) and third (Q3) quartile, respectively; left and right "whiskers" show $Q1-(Q3-Q1)$ and $Q3+(Q3-Q1)$; the median is notated by a straight line through the box; averages are notated by triangles.

incorporating "oracle" (ground truth) information for both pitch and separated vocals can indeed improve the learned results for the other task. As a second contribution, we then design, implement and evaluate different model architectures that estimate pitch and vocals individually or jointly in a variety of ways, each reflecting a different perspective on the interdependency of

the two tasks. Finally, inspired by end-to-end unfolding techniques for representing iterative re-estimation processes of dependent components inside a network (Wisdom, Hershey, Le Roux, & Watanabe, 2016) as well as stacking networks (Park, Kim, Lee, & Kwak, 2018), we propose an architecture which resolves the task dependencies within a sequential re-estimation model.

Several of the models presented in this chapter can be classified as "multi-task learners". An overview of multi-task learning with deep neural networks can be found in (Ruder, 2017).

5.2 Input and Output Representations

We make use of an internal dataset of roughly 2500 pairs of music audio signals $\mathbf{x}$ and corresponding isolated vocal audio signal $\mathbf{y}$ from a number of musical genres, including pop, rock and rap vocals. All singing voices present in the mixture $\mathbf{x}$ are included in the isolated vocals signal $\mathbf{y}$, meaning that $\mathbf{y}$ may — and often does — contain more than one active voice at a time (e.g. a lead singer and background harmonies). $\mathbf{x}$ and $\mathbf{y}$ are converted to mono with a sample rate of 22050 Hz. Let $\mathbf{X}$ and $\mathbf{Y}$ be the magnitude of the Short Time Fourier Transform (STFT) spectrogram of $\mathbf{x}$ and $\mathbf{y}$ respectively. STFTs are computed with a hop size of 256 and with 1024 points in the FFT, as shown in Figure 5.2 (left) and (middle) respectively.

Our dataset does not contain ground truth vocal f_0 annotations. As a proxy for ground truth f_0, we run the Deep Salience multiple-f_0 estimation model (Rachel M. Bittner et al., 2017) on the isolated vocals $\mathbf{y}$ in our training set. Deep Salience predicts a matrix $\mathbf{S}$ of f_0 salience values — i.e. the likelihood of an f_0 value being present over a grid of time-frequency points (see Figure 5.2, right). Note that this algorithm does not assume the audio is monophonic — if multiple pitches are present at the same time, there can be multiple high-likelihood f_0 bins.

$\mathbf{S}$ is the target output for the vocal f_0 estimation (pitch) component of our models. Note that we are training a model to reproduce the output of another trained model (Deep Salience), similar to a teacher-student training paradigm (Hinton, Vinyals, & Dean, 2015). One notable difference is we are training our model to produce $\mathbf{S}$ given *mixtures* as input, while the pre-trained Deep Salience model is given *isolated vocals* as input. This also

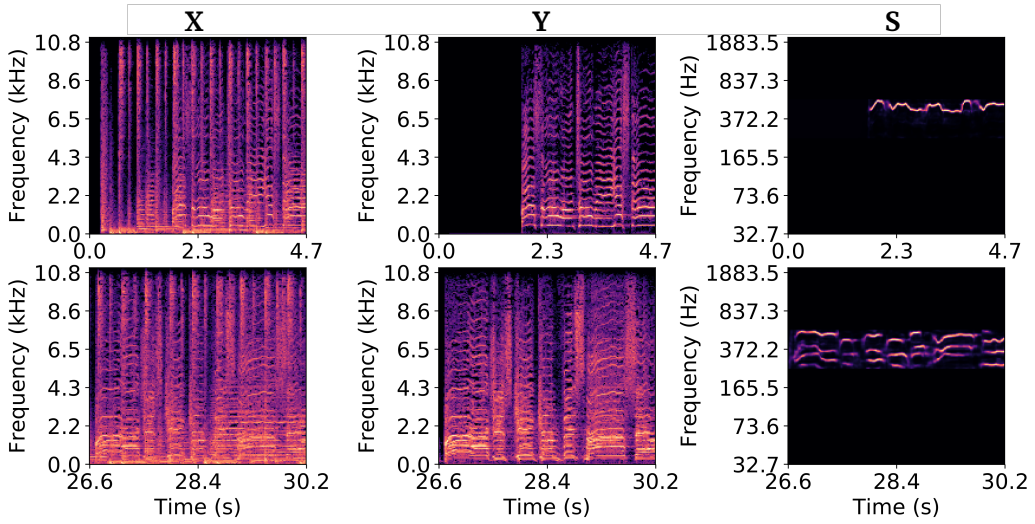


Figure 5.2: An example of the input and output representations used for training. (Left) Input magnitude STFT of the mixture audio. (Middle) Target magnitude STFT of the isolated vocal audio. (Right) Target vocal salience produced by the Deep Salience algorithm (Rachel M. Bittner, McFee, Salamon, Li, & Bello, 2017). The top row shows an example where there is one solo singer, while in the bottom row, three singers are singing in harmony.

means that the performance of our model will likely be upper bounded by the performance of Deep Salience on isolated vocals.

The largest public datasets with mixtures, corresponding isolated vocals and annotated vocal f_0 are iKala (Chan et al., 2015) ($\approx$ 2 hours) and MedleyDB (Rachel M. Bittner et al., 2014) ($\approx$ 3 hours, since only half of the tracks contain vocals). Because Deep Salience was trained using MedleyDB, we evaluate the performance of our models on iKala. We compare vocal f_0 outputs with iKala's f_0 annotations using the `mir_eval` (Raffel et al., 2014) implementation of standard melody metrics (Salamon, Gómez, Ellis, & Richard, 2014). Vocal source separation outputs are evaluated using the Signal-to-Distortion Ratio (SDR) metric[1] from the `mir_eval` implementation of *BSS Eval* (Vincent et al., 2006).

[1]Signal-to-Interference and Signal-to-Artifact Ratios followed the same trends as SDR and are therefore not included.

5.3 Model Overview and Training

The models presented in the subsequent sections are composed of one or more U-Nets. We use the exact same U-Net architecture as in Chapter 4 for estimating both vocal separation $\hat{\mathbf{Y}}$ and vocal f_0 salience $\hat{\mathbf{S}}$. In all experiments we optimize the model weights using the ADAM (Kingma & Ba, 2014) optimizer, using the same parameters as in Chapter 4.

The vocal separation network produces a soft ratio mask $\mathbf{M}$ from where we derive the vocal magnitude spectrogram

$$\hat{\mathbf{Y}} = \mathbf{M} \odot \mathbf{X} \tag{5.1}$$

$\hat{\mathbf{Y}}$ is optimized by minimizing the loss L_1 loss:

$$\mathcal{L}_v = \left\| \hat{\mathbf{Y}} - \mathbf{Y} \right\|_1 \tag{5.2}$$

The isolated vocal signal $\hat{y}$ is synthesized by applying the phase of the original complex mixture spectrogram to the estimated magnitude spectrogram, and transforming to the time-domain by means of the Inverse Short Time Fourier Transform (ISTFT). The salience network outputs $\hat{\mathbf{S}}$ directly and is optimized with L_2 loss:

$$\mathcal{L}_s = \left\| \hat{\mathbf{S}} - \mathbf{S} \right\|_2 \tag{5.3}$$

The below experiments which jointly estimate vocal source and salience are optimized by summing the vocal and salience losses:

$$\mathcal{L}_{\text{joint}} = \mathcal{L}_v + \mathcal{L}_s \tag{5.4}$$

In the case of monophonic targets (e.g. as in iKala), f_0 time series are generated from $\hat{\mathbf{S}}$ by returning the frequency with maximum likelihood at each time frame. The voicing (when the voice is active/inactive) is determined by a simple threshold on the maximum likelihood at each time frame; frames where the likelihood falls below the threshold are reported as "unvoiced". In the results from our models, we fix the voicing threshold to 0.4. For the comparison models (Crepe, Melodia, etc.), we compute performance for a full grid of possible thresholds, and report the performance for the threshold that maximizes performance in each case ("oracle" threshold).

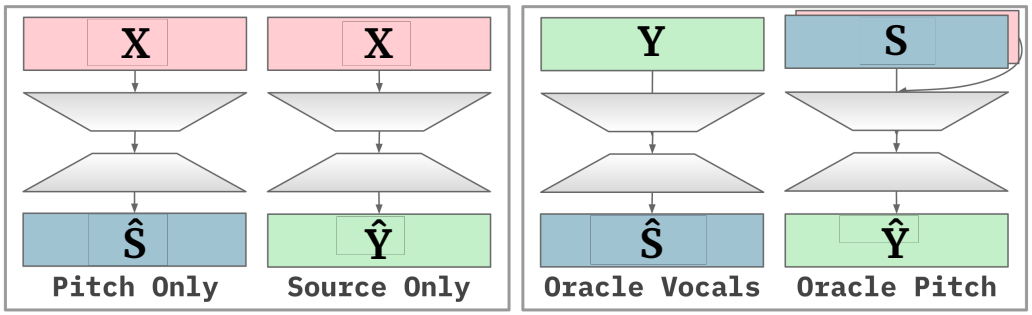


Figure 5.3: (Left) Baseline models take the mixture magnitude STFT $\mathbf{X}$ as input, and output vocal f_0 salience $\hat{\mathbf{S}}$ in "Pitch Only", and the vocal magnitude STFT $\hat{\mathbf{Y}}$ "Source Only". (Right) Oracle models which are given "perfect" input information. "Oracle Source" is given isolated vocals magnitude STFTs $\mathbf{Y}$ as input and trained to output $\mathbf{S}$. "Oracle Pitch" is given $\mathbf{X}$ and oracle vocal f_0 salience $\mathbf{S}$ as input and trained to output $\mathbf{Y}$.

5.4 Evaluation measures

In addition to the source separation metrics described in Section 2.6.1, we evaluate the f_0 estimation using the objective measures Raw Pitch Accuracy (RPA) and Overall Accuracy (OA) (Salamon et al., 2014). Note that the ground truth vocal f_0 is undefined during parts of the track where the singers are silent. This is reflected in the definition of the evaluation measures.

- Raw Pitch Accuracy is defined as the proportion of frames in which the ground truth has active vocals, that the estimated melody is correct within half a semitone.
- Overall Accuracy is defined as the proportion of frames that are correctly labeled, where correct labeling is defined differently for frames with and without vocal activity. For frames that have active ground truth vocals, correct is defined as in RPA to accuracy within half a semitone. Unvoiced frames are marked as correct if the algorithm marked them as unvoiced.

5.5 Baselines and Oracle Experiments

As a baseline, we first train separate models, shown in Figure 5.3 (left): `Pitch only` which estimates f_0 salience given mixtures as inputs, and `Source only` which performs vocal source separation. These models are completely independent and do not share weights.

As an upper bound, we train models that receive *oracle* information — `Oracle Vocals` which estimates vocal salience $\hat{\mathbf{S}}$ given ground truth vocals $\mathbf{Y}$ as input, and `Oracle Pitch` which estimates the vocal spectrogram magnitudes $\hat{\mathbf{Y}}$ given the mixture $\mathbf{X}$ and ground truth f_0 salience $\mathbf{S}$ as inputs, as shown in Figure 5.3 (right). `Oracle Vocals` tells us how well we can estimate f_0 performance given perfect information, i.e. the performance reported provides an estimate for the upper bound achievable with this architecture and number of parameter. `Oracle Pitch` tells us how much it helps source separation performance to have f_0 salience as side information, and similarly gives us an upper bound on source separation performance.

The results in Figure 5.4a clearly show that a vocal pitch estimator trained on clean vocals $\mathbf{Y}$ (`Oracle Vocals`) performs better than one trained on mixtures $\mathbf{X}$ (`Pitch only`). This is consistent with Figure 5.1 where pitch trackers with clean vocals as input outperform pitch trackers that operate on mixtures. A similar effect can be seen for vocal separation in Figure 5.4b, where the inclusion of ground truth pitch salience $\mathbf{S}$ improves the source separation metric.

5.6 Joint Models

In the following section, we present a series of different architectures that perform both vocal source separation and f_0 salience estimation. Figure 5.5 gives an overview of the architectures we compare; they share information in various ways, either through weight sharing (treating the problem in a standard multitask setup) or by directly giving the outputs of one stage of the model as input to the next.

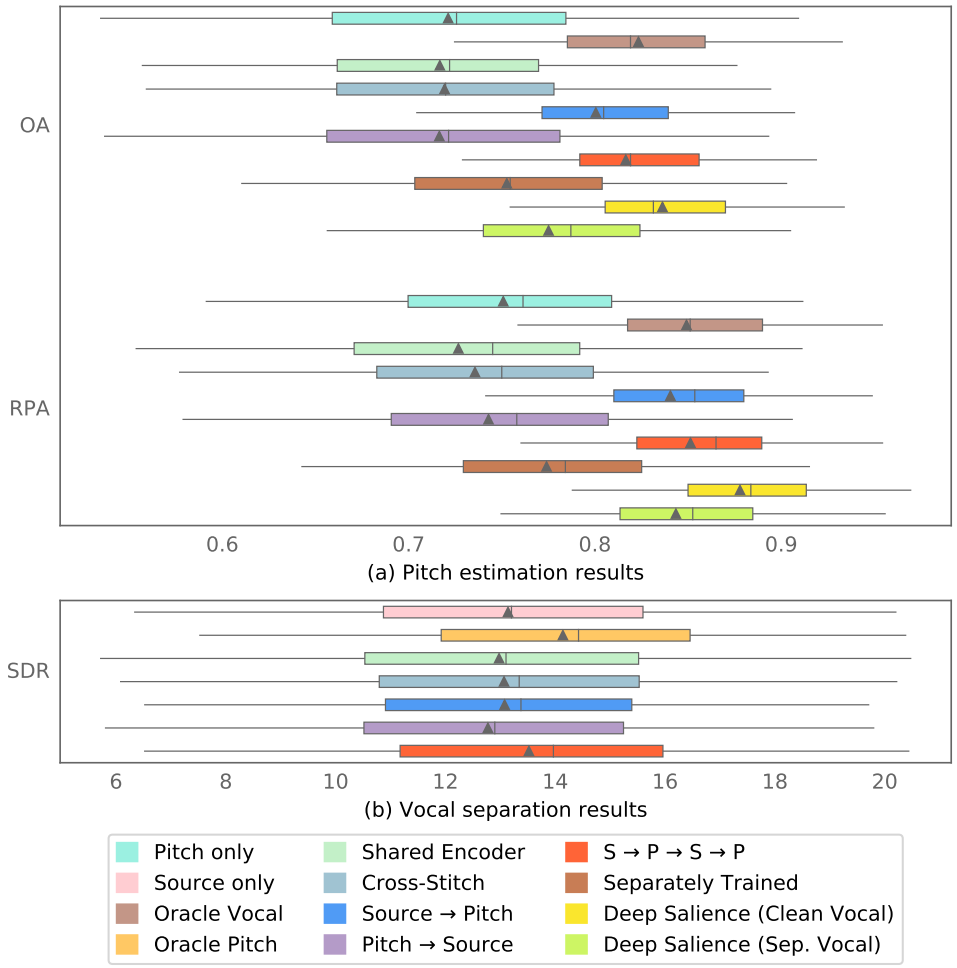


Figure 5.4: Performance comparison of our experiments, oracle, and baseline models, evaluated on iKala. (Top) Single-f_0 metrics. (Bottom) Vocal source separation metrics. Refer to Figure 5.1 for an explanation of the elements in the plot.

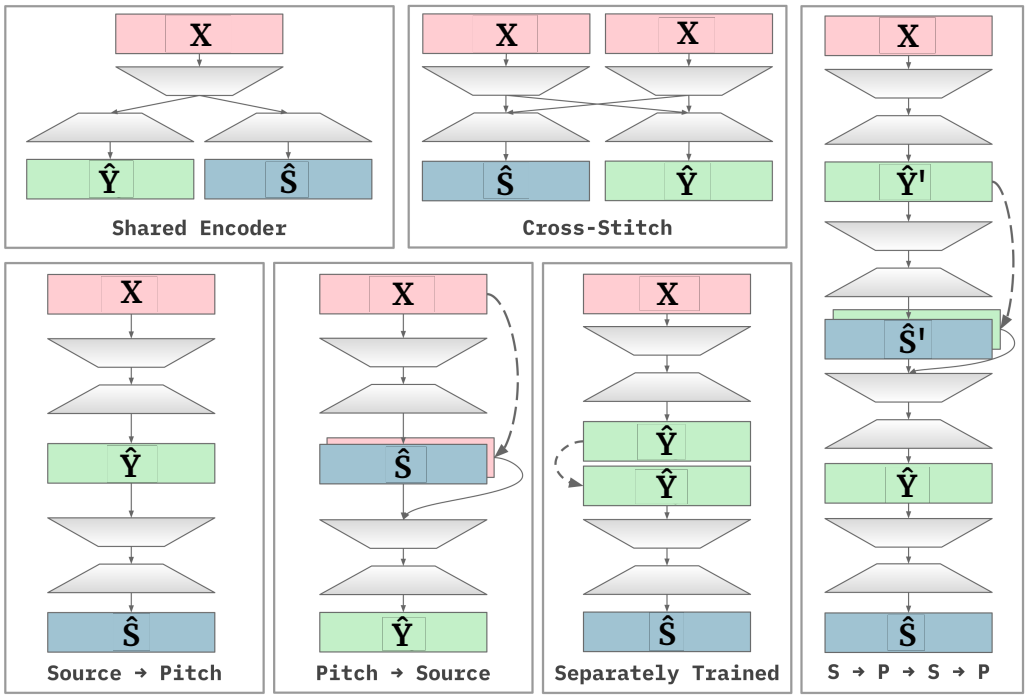


Figure 5.5: Joint U-Net models. Each model takes the magnitude spectrogram of the mixture $\mathbf{X}$ as input and outputs estimates of the vocal magnitude spectrogram $\hat{\mathbf{Y}}$ and the vocal f_0 salience $\hat{\mathbf{S}}$. In `Pitch→Source` model, $\mathbf{X}$ is given as additional input to the vocal source separation portion of the model (indicated by a dotted line), and similarly in the `S→P→S→P` model, the first vocal estimate $\hat{\mathbf{Y}}'$ is given as additional input to the second vocal source separation model. In `Separately Trained`, each network is trained separately, first optimizing $\hat{\mathbf{Y}}$, and then using the optimized $\hat{\mathbf{Y}}$ as input to a second model that outputs $\hat{\mathbf{S}}$.

5.6.1 Conventional Multitask Models

We first experiment with architectures that share weights for both tasks. The `Shared Encoder` model, Figure 5.5 (top left), shows the simplest such architecture, which has one encoder that is shared and separate decoders for each task. The results for this experiment show that the shared encoder model is inferior to disjoint models for both vocal f_0 estimation (Figure 5.4a) and vocal separation (Figure 5.4b). One potential explanation for this result is that the shared encoder reduces the number of parameters in the network

by 25%.

In the `Cross-Stich` model, shown in Figure 5.5 (top right), each output has separate encoder-decoders, but the encoders are concatenated before being passed to the individual decoders. However, the skip connections are task-specific. This model has a capacity equivalent to that of the two baseline models, and should not suffer from the same potential capacity issue of the Shared Encoder model. `Cross-Stich` performs slightly better than `Shared Encoder`, but is still worse than the baseline models on both tasks.

5.6.2 Stacked Models

In Figure 5.5, (bottom left and middle), the tasks are learned in a cascaded manner. In the `Source`→`Pitch` model, a first U-Net computes $\hat{\mathbf{Y}}$ given $\mathbf{X}$, and a second U-Net computes $\hat{\mathbf{S}}$ given $\hat{\mathbf{Y}}$ as input. `Pitch`→`Source` is similar but in the reverse order, and with the addition of concatenating $\mathbf{X}$ and $\hat{\mathbf{S}}$ as input to the second U-Net model. This concatenation was added because there is not enough information in $\hat{\mathbf{S}}$ alone to compute $\hat{\mathbf{Y}}$ — the mixture information is needed as well.

As shown in Figure 5.4a, `Source`→`Pitch` outperforms all of `Pitch only`, `Shared Encoder` and `Cross-Stich` for pitch estimation. This is perhaps unsurprising, since the baseline model trained on clean vocals (`Oracle Vocals`) performs better than the baseline trained on mixtures (`Pitch only`). However, `Pitch`→`Source` does not result in a similar improvement for vocal separation, even though `Oracle Pitch` saw significant improvements. We hypothesize that the lack of accuracy of pitch estimates from mixture (see Figure 5.4a) prevents improvements of the vocal separation on the level of `Oracle Pitch`.

5.6.3 Stacked Refinement

In our final experiment, we attempt to further refine the results of `Source`→`Pitch`. *Stacked Refinement* (He & Schomaker, 2019) is an architectural pattern in which a network module is repeated, feeding the output of a module as input to an identically designed module, with different weights. We adapt this pattern to our domain by extending `Source`→`Pitch` to a "Source → Pitch → Source → Pitch" (`S`→`P`→`S`→`P`) network 5.5 (right). We notate the output of the first source network $\hat{\mathbf{Y}}'$ and the output of the first pitch network $\hat{\mathbf{S}}'$. The

second source network is fed a concatenation of $\hat{\mathbf{Y}}'$ and $\hat{\mathbf{S}}'$, and it outputs $\hat{\mathbf{Y}}$, which it then feeds into the second pitch network to output $\hat{\mathbf{S}}$. This allows higher level modules to learn a refinement function from initial predictions to cleaner predictions.

The scores for the resulting model are plotted in Figure 5.4a and Figure 5.4b. The stacked refinement model surpasses the performance of all of our other models for both vocal source separation and vocal melody estimation.

The second source separation network is presented with an adequate pitch estimation, which provides additional guidance to refine the vocal source estimate. It is a somewhat surprising result that the difference for melody estimation is higher than the difference for vocal separation, since that implies that the inputs to the first and second melody estimation networks have little difference. We hypothesize that the doubled network capacity might be an important additional factor in explaining this result.

5.7 Joint vs. Separate Training

We saw in the previous section that `Source`→`Pitch` performs better than the other configurations of the same network capacity. In order to test if the joint training is necessary, we take the output $\hat{\mathbf{Y}}$ of `Source only` as input to a model which outputs $\hat{\mathbf{S}}$, as shown in Figure 5.5 (`Separately Trained` bottom, 2nd from right). The results, plotted as `Separately Trained` in Figure 5.4a, show that joint optimization is indeed beneficial for vocal melody estimation. This is possibly because our pitch model is "borrowing" capacity from the separation model, or alternatively because the separation results become more tailored towards pitch estimation.

5.8 Discussion

5.8.1 Vocal Melody Estimation from Mixtures

The fusion of vocal source separation and vocal salience estimation results in system that is able to estimate vocal melody from musical mixtures. As we saw in Section 5.1, vocal pitch trackers that take isolated vocals as input have higher accuracy than systems that estimate vocal melody directly from a mixture. We now pose the question of whether our jointly trained vocal

Experiment	OA	RPA
Melodia	0.766	0.776
Deep Melody	0.695	0.788
Pitch only	0.721	0.751
Shared Encoder	0.717	0.727
Cross-Stitch	0.719	0.736
Source → Pitch	0.800	0.841
Pitch → Source	0.716	0.743
S → P → S → P	**0.817**	**0.851**

Table 5.1: Vocal Melody Estimation Results

separator and multi-f_0 estimator can predict single-f_0 vocal melody as well as pitch trackers that were explicitly trained to predict single-f_0 from mixtures.

Comparing our best model (S→P→S→P) to Melodia (Salamon et al., 2014) and Deep Melody (Rachel M. Bittner et al., 2017), shows that our model does indeed perform better than both other models on single-f_0 vocal pitch estimation, on the iKala dataset (Table 5.1). All three results were obtained by presenting the models with mixture inputs.

5.8.2 Qualitative Analysis

To build an intuition of the characteristics of the model's pitch estimates, we zoom in on a few bars of the 1965 pop song "Turn! Turn! Turn!" by The Byrds. This song is interesting for our multi-f_0 model since its vocals alternate between unison and multi-part harmony. Figure 5.6 shows the final pitch estimation using S→P→S→P, with a voicing threshold set to 0.4. Since we do not have vocal melody ground truth for this track, we only show the model outputs. While there are a few scattered false positives, the majority of the pitch estimates appear to belong to vocal notes. The likelihoods for sustained notes sometimes drop below the voicing threshold, leaving only activations at the initial note onset transient.

It is also instructive to visualize the raw estimated pitch salience, before applying the voicing threshold. Figure 5.7 show vocal salience matrices, estimated on the first phrase of the example above, using our model (left) and Crepe (right).

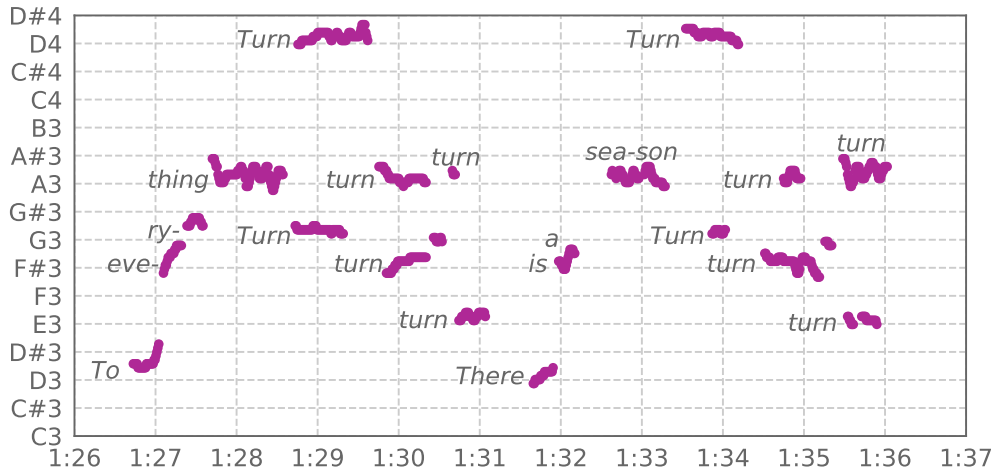


Figure 5.6: Vocal f_0 estimation from S→P→S→P on an excerpt from the pop song "Turn! Turn! Turn!" by The Byrds.

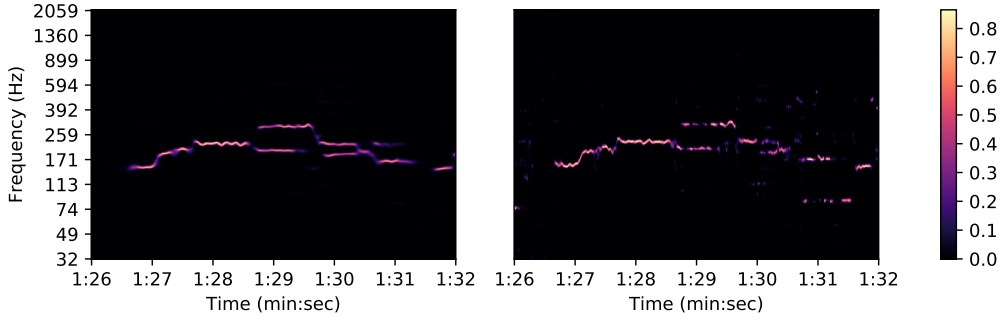


Figure 5.7: Estimated salience matrix $\hat{\mathbf{S}}$. Left: our S→P→S→P model, predicted from mixture $\mathbf{X}$. Right: Crepe, predicted from estimated vocal source $\hat{\mathbf{Y}}$. The excerpts begins with one voice, and a second voice enters at 1:29.

The most striking difference is our model's ability to predict multiple simultaneous vocal parts, while Crepe oscillates between the two notes. Despite the fact that our model is presented with a full mixture, and Crepe with vocals isolated by the source separation output of our model, our model produces a cleaner salience matrix, with fewer artifacts. It appears that during the joint training procedure, the vocal melody estimation module learns to ignore source separation artifacts, despite the fact that artifacts are present in this excerpt. It should also be noted that our model is less precise than Crepe, lacking some of the fine-grained sharpness of the vibratos that Crepe more accurately captures.

5.9 Conclusions and Future Work

In this work, one of the biggest challenges was in assessing *why* certain models performed better or worse than others, and in this chapter we suggested hypotheses with possible reasons. Future work includes testing these specific hypotheses to obtain better insights about the specific advantages and disadvantages of these architectures. Beyond this, we would like to explore applying joint f_0 and source separation models to other types of pitched sources beyond the singing voice, such as the piano. We think the same ideas could also be applied to drum separation and drum transcription.

Overall, we explored a number of different architectures for jointly separating vocals and estimating fundamental frequency in a single data driven system based on deep U-Net neural network architectures. We compared single-task models with single-task models given oracle information from the other task, and obtained ideal upper and lower bounds on performance. We saw that including oracle information improves performance, in particular for vocal-f_0 information. A joint stacked model that first performs vocal source separation followed by vocal f_0 estimation approaches the performance of the oracle models, and outperformed conventional multitask architectures, which underscores the value of incorporating domain knowledge when designing models. Additionally, we showed that the model achieves state-of-the-art results for vocal-f_0 estimation on the iKala dataset. Finally, we highlighted the importance of performing *polyphonic*, rather than monophonic vocal-f_0 estimation for many real-world cases.

Chapter 6

Multi-instrument separation with complex masks

6.1 Introduction

In the preceding chapters, we have developed source separation methods that operate solely on the magnitude component of the STFT spectrogram. While this makes it possible to translate algorithms from the image domain to the source separation task, by treating spectrograms as images, this training scheme also comes with significant drawbacks. Typically, masks are trained to "let through" spectral magnitudes that correspond to some particular instrument. Yet, when we apply the ISTFT to re-synthesize the estimated isolated instrument, reproduction will never be perfect.

This is due to phase artifacts: sources that overlap in time and frequency with the target source leave audible traces in the masked signal. These traces are due to the phase of the interfering source being preserved after masking. This effect is the more pronounced the greater the spectral overlap between sources is, and is especially problematic in the music domain: due to the very nature of (western) music, musical instruments and voices are synchronous in time by following a shared meter (unlike speech), and their spectral components tend to occupy the same frequency ranges when sounding together in consonance (Schönberg, 1922).

The complex ideal ratio mask (cIRM) can be helpful in illustrating this effect.

The cIRM, $\mathbf{M}_{\text{cIRM}}$, given an isolated source complex spectrogram $\mathbf{Y}$ and a mixture complex spectrogram $\mathbf{X}$ is defined as:

$$\mathbf{M}_{\text{cIRM}} = \mathbf{Y} \oslash \mathbf{X} \tag{6.1}$$

where $\oslash$ denotes element-wise division.

From the definition of complex division,

$$|\mathbf{M}_{\text{cIRM}}| = |\mathbf{Y}| \oslash |\mathbf{X}| \tag{6.2}$$
$$\angle\mathbf{M}_{\text{cIRM}} = \angle\mathbf{Y} - \angle\mathbf{X} \tag{6.3}$$

where $|\cdot|$ denotes complex magnitude and $\angle$ denotes complex phase or angle.

Hence, the cIRM magnitude is simply the ideal ratio mask for magnitude spectrograms that we have estimated in previous chapters, and the phase of the cIRM is the difference between the phase of source and the mixture. Figure 6.1 shows the mixture of a vocal source and a percussion source, and the magnitude and phase components of cIRM that recovers the percussion source from the mixture.

In the bottom right of Figure 6.1 is a plot of the ideal phase mask, multiplied by the percussion source. This shows clearly how vocal components are present in the percussion mask, especially during times when the sources have high degrees of spectral overlap. Copying the phase from the mixture to the isolated source estimate, as we have done in previous chapters, results in audible phase artifacts due to unaltered phase differences between the mixture and the source.

In this chapter we investigate using complex-valued masks as a drop-in replacement for magnitude-domain masks within the context of multi-instrument source separation, in an attempt to reduce the effect of phase artifacts from interfering instruments. Additionally, we examine several loss functions, in both time and frequency domains.

6.2 Datasets

The proposed model is trained on a dataset of about 2000 professional quality recordings. For each recording, five *stems* are available, grouped by instrument: "Vocals", "Guitar", "Bass", "Percussion", and "Other". Each record-

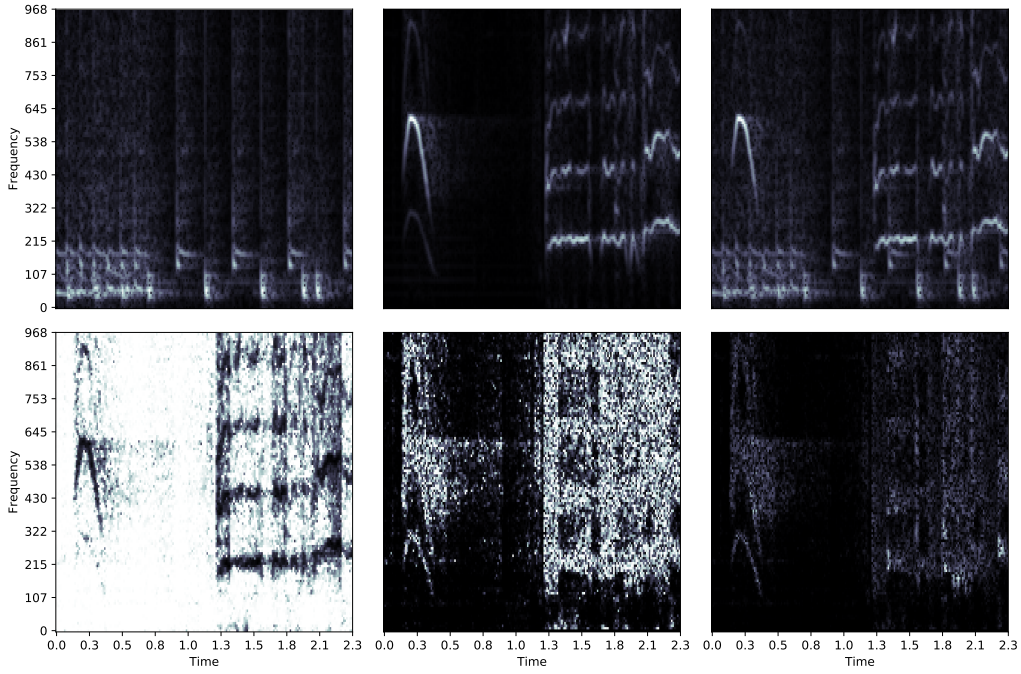


Figure 6.1: Complex Ideal Ratio Masks. **Top left**: Percussion source; **Top middle**: Vocal source; **Top right**: Mixture; **Bottom left**: Ideal magnitude mask; **Bottom middle**: Ideal phase mask; **Bottom right**: Ideal phase mask weighted by percussion source.

ing is, by design, the (unweighted) sum of its five stems. The distribution of genres in the dataset is listed in table 6.1.

To increase the size of the training set, a series of augmentation steps is performed on the stems. Two classes of augmentations are performed: *stem-level* and *track-level* changes. Stem-level augmentations are applied to single stems, and include random volume adjustments and equalization, as well as randomized audio effects such as chorus, phasers, flangers, tremolo, etc. Track-level augmentations are applied to the entire mixture, after each stem has been augmented individually, including time stretching, pitch shifting, and resampling. All augmentations were implemented using the PySOX library.(Rachel Bittner, Humphrey, & Bello, 2016). Ranges of acceptable random parameterizations were chosen empirically such that the resulting mixture was still plausible from a musical standpoint.

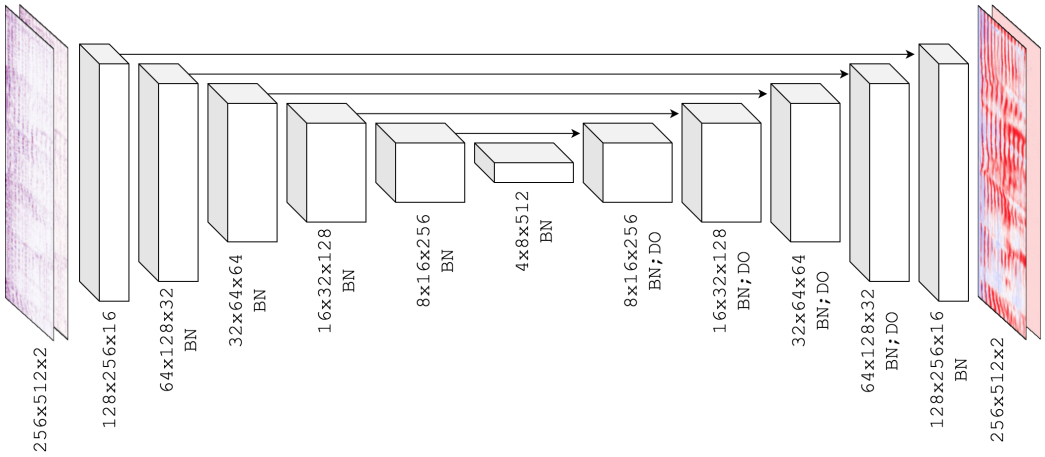


Figure 6.2: Architecture diagram of the proposed model.

For validation and testing, the MUSDB 2018 dataset (Stöter et al., 2018) is used; in particular, the training split (100 tracks) is used to track the progress of the model during training, and the test split (50 tracks) to evaluate the final results. MUSDB contains only 4 instrument types, namely "Vocals", "Bass", "Percussion", and "Other". When running validation and test results, we group our model's outputs for "Guitar" into the "Other" category.

6.3 Model

6.3.1 Architecture

The proposed model extends the U-Net source separation model detailed in Chapter 4; pictured in Figure 6.2, it consists of 6 downsampling "encoder" layers, 6 upsampling "decoder" layers, and skip connections between corresponding encoder and decoder layers. Each encoder layer comprises a strided convolution (with kernel size 5×5 and stride 2×2) followed by batch normalization (except in the first layer), and a leaky ReLU activation function. In the decoder, each layer consists of a strided transpose convolution (with kernel size 5×5 and stride 2×2, as in the encoder); batch normalization is applied to all decoder layers except the last, as well as 50% dropout in the first five layers, and a ReLU activation function in all layers except the last.

Genre	Percentage
Rock	26.2%
Alternative	20.8%
Metal	14.6%
Indie Rock	9.1%
Punk	9.0%
Pop	8.0%
Electronica	2.5%
R&B	2.2%
Dance & House	1.3%
Rap	1.0%
Other	5.3%

Table 6.1: Stems data genre distribution

6.3.2 Input and Output Representation

For multi-source separation, independent models are trained for vocals, drums, guitar, and bass. The "Other" source is estimated as the residual difference between mixture and the sum of the four estimated sources. Each source-specific network is presented with mixture spectrograms, extracted from 22050 Hz mono audio signals, with overlapping windows of size 1024, and hop size 256. During training, we slice the spectrograms into *patches* of 256 frames, and feed the network batches of 16 patches each. This is not necessary during inference, since the network is fully convolutional and can be applied to the spectrogram of the full length of the signal.

The network is trained to output a mask that is then applied to the mixture spectrogram to produce the source spectrogram estimate. We evaluate two variations of this architecture: non-complex, where the input is a single mono channel representing the magnitude of the mixture spectrogram; and complex, where the input is the complex spectrogram. In the non-complex case, the mask is only applied to the magnitude component of the mixture, whereas the complex experiments apply the mask in the complex domain. After mask application, we synthesize the estimated source signal by means of the Inverse Short-Time Fourier Transform (ISTFT).

In the complex case, the complex single-channel mono mixture spectrogram matrix is transformed into a three-dimensional tensor where the third di-

mension has two channels representing the real and imaginary components of the complex mixture spectrogram. After having been transformed by an internally real-valued neural network, the three-dimensional network output is then transformed back to a complex two-dimensional matrix by treating the two channels of the third dimension as real and imaginary. Mathematically, let $O^{\mathbb{C}\text{-Network}}_{i,j,c}$ denote the real-valued final network layer output, where i and j represents width and height indices, and c represents the channel index, where $c = 0$ and $c = 1$ represents real and imaginary components. We then transform this real-valued $\mathbf{O}^{\mathbb{C}\text{-Network}}$ into a complex output $\mathbf{O}^{\mathbb{C}}$ using

$$O^{\mathbb{C}}_{i,j} = O^{\mathbb{C}\text{-Network}}_{i,j,0} e^{iO^{\mathbb{C}\text{-Network}}_{i,j,1}} \tag{6.4}$$

6.3.3 Masking

In the remainder of this section, $\mathbf{O}^{\mathbb{C}}$ and $\mathbf{O}^{\mathbb{R}}$ denotes the outputs by the complex and non-complex networks; $\mathbf{M}^{\mathbb{C}}$ and $\mathbf{M}^{\mathbb{R}}$ denotes complex and non-complex masks; $\mathbf{X}$ denotes the spectrogram of the input mixture to be separated by the model; for individual stems (individual instruments or voice), $\mathbf{Y}$ and $\hat{\mathbf{Y}}$ denote the spectrograms of the recorded and estimated stems, respectively, while $\mathbf{y}$ and $\hat{\mathbf{y}}$ denote their time-domain equivalent.

Non-complex masking

The non-complex mask is computed as

$$\mathbf{M}^{\mathbb{R}} = \sigma\left(\mathbf{O}^{\mathbb{R}}\right) \tag{6.5}$$

where σ is the sigmoid function, constraining the (real-valued) network output to the $(0, 1)$ range. It is multiplied element-wise (denoted by $\otimes$) with the mixture spectrogram magnitude to obtain the estimated source spectrogram magnitude. The phase is copied from the mixture spectrogram into the spectrogram of the estimated stem, unaltered.

$$\hat{\mathbf{Y}} = \mathbf{M}^{\mathbb{R}} \otimes |\mathbf{X}| \otimes e^{i\angle\mathbf{X}} \tag{6.6}$$

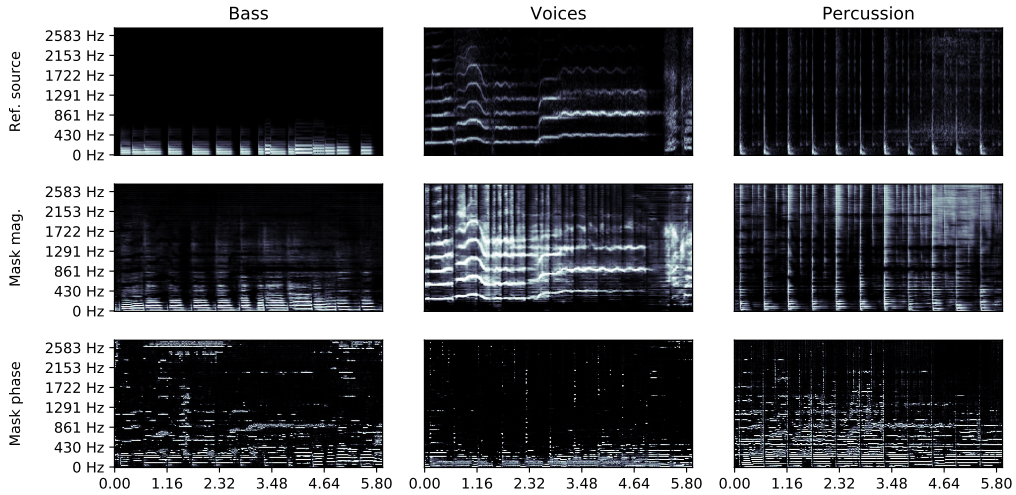


Figure 6.3: Magnitude and phase of learned complex masks (Complex-SDR) for Bass, Voices, and Percussion on a test example from MUSDB18, and the reference STFTs for each source. **Top Row:** Magnitude (dB) of the ideal STFTs. **Middle Row:** Magnitude of the learned complex masks. **Bottom Row:** Absolute value of the phase of the learned complex masks.

Complex masking

In the complex case we follow (Choi et al., 2019), by estimating a complex-domain mask that is constrained in magnitude but not in phase. The tanh magnitude constraint ensures that $\mathbf{M}^{\mathbb{C}}$ is bounded to the unit circle in the complex plane.

$$|\mathbf{M}^{\mathbb{C}}| = \tanh(|\mathbf{O}^{\mathbb{C}}|) \tag{6.7}$$

$$\angle\mathbf{M}^{\mathbb{C}} = \mathbf{O}^{\mathbb{C}} \oslash |\mathbf{O}^{\mathbb{C}}| \tag{6.8}$$

where $\oslash$ denotes element-wise division.

The complex mask is then applied element-wise, as

$$\hat{\mathbf{Y}} = |\mathbf{M}^{\mathbb{C}}| \otimes |\mathbf{X}| \otimes e^{i(\angle\mathbf{M}^{\mathbb{C}} + \angle\mathbf{X})} \tag{6.9}$$

The mask applied determines a magnitude rescaling of the mixture STFT

and a rotation (addition) in phase. This allows the model to not only remove interfering frequency bins, but also to cancel the phase from other sources.

6.3.4 Loss

We evaluate three different loss functions: *Magnitude loss*, *SDR loss*, and the combination *SDR + Magnitude loss*.

The *Magnitude loss* is optimized by minimizing the L_1 distance between the spectrogram magnitudes of the estimated and target components:

$$\mathcal{L}_{\text{Mag}} = |\mathbf{Y} - \hat{\mathbf{Y}}|_1 \tag{6.10}$$

The time-domain *SDR loss* (Choi et al., 2019) is defined as:

$$\mathcal{L}_{\text{SDR}} = -\frac{\mathbf{y} \cdot \hat{\mathbf{y}}}{\|\mathbf{y}\|\|\hat{\mathbf{y}}\|} \tag{6.11}$$

which smoothly upper bounds the SDR metric we are interested in optimizing.

The hybrid time-domain and frequency-domain *SDR + Magnitude loss* is defined as:

$$\mathcal{L}_{\text{SDR+Mag}} = \alpha\mathcal{L}_{\text{SDR}} + \beta\mathcal{L}_{\text{Mag}} \tag{6.12}$$

where α and β are weighting factors for the two losses. We use $\alpha = 1$ and $\beta = 1$ in our experiments.

We train our models in an end-to-end fashion using TensorFlow (Abadi et al., 2016), taking advantage of its built-in short-time Fourier transform functions, which allow back-propagation to be computed through the forward and backward transforms themselves. All losses were optimized using Adam (Kingma & Ba, 2014), as in previous chapters.

6.4 Results

We test three configurations of our system: "Non-complex", "Complex-SDR", and "Complex-SDR+Mag", outlined in Table 6.2. Objective evaluation is performed on the test portion of the MUSDB18 dataset, using the `museval` toolkit (Stöter et al., 2018).

Experiment	Mask	Loss
Non-complex	Non-complex	Magnitude
Complex-SDR	Complex	SDR
Complex-SDR+Mag	Complex	SDR+Magnitude

Table 6.2: Experimental configurations.

Objective scores, obtained using the tools detailed in Section 2.6.1, are shown in Figure 6.4. There are slight differences between the results of the different models when considering different metrics, however the complex models achieve higher Signal to Distortion Ratio (commonly considered to be the most important metric) than the non-complex model for most sources. Although these results do not compare favorably to State-of-the-Art approaches, such as those reported in the yearly SiSEC evaluation campaign (Stöter et al., 2018), they do yield insights into the design of such systems, by providing a unified comparison not only in terms of dataset, but also through the use of a single model in which individual aspects (masking, loss function) are controlled.

We also collected subjective judgments from a number of individuals, using a methodology similar to that of (Cartwright, Pardo, & Mysore, 2018). A questionnaire was prepared, in which subjects were prompted to compare audio recordings in pairs, each pair presenting the outputs of two different model configurations applied the same short audio excerpt. As a reference, evaluators were given both the original mixture and the target single source signal.

Two questions were asked per pair of audio excerpts:

- "Quality: Which one has better sound quality?"
- "Separation: Which one has better isolation from the other instruments in the original mix?"

For each question, the test subject could select either of the two audio excerpts, as well as a third option "I don't know". The results from the qualitative test are shown in Figure 6.5. Because of the difficulty of the task, we were unable to employ online "crowd worker" platforms, and had to rely on a small number of individuals trained in audio engineering. This limited the size of the test to 210 examples. The results show a slight preference for

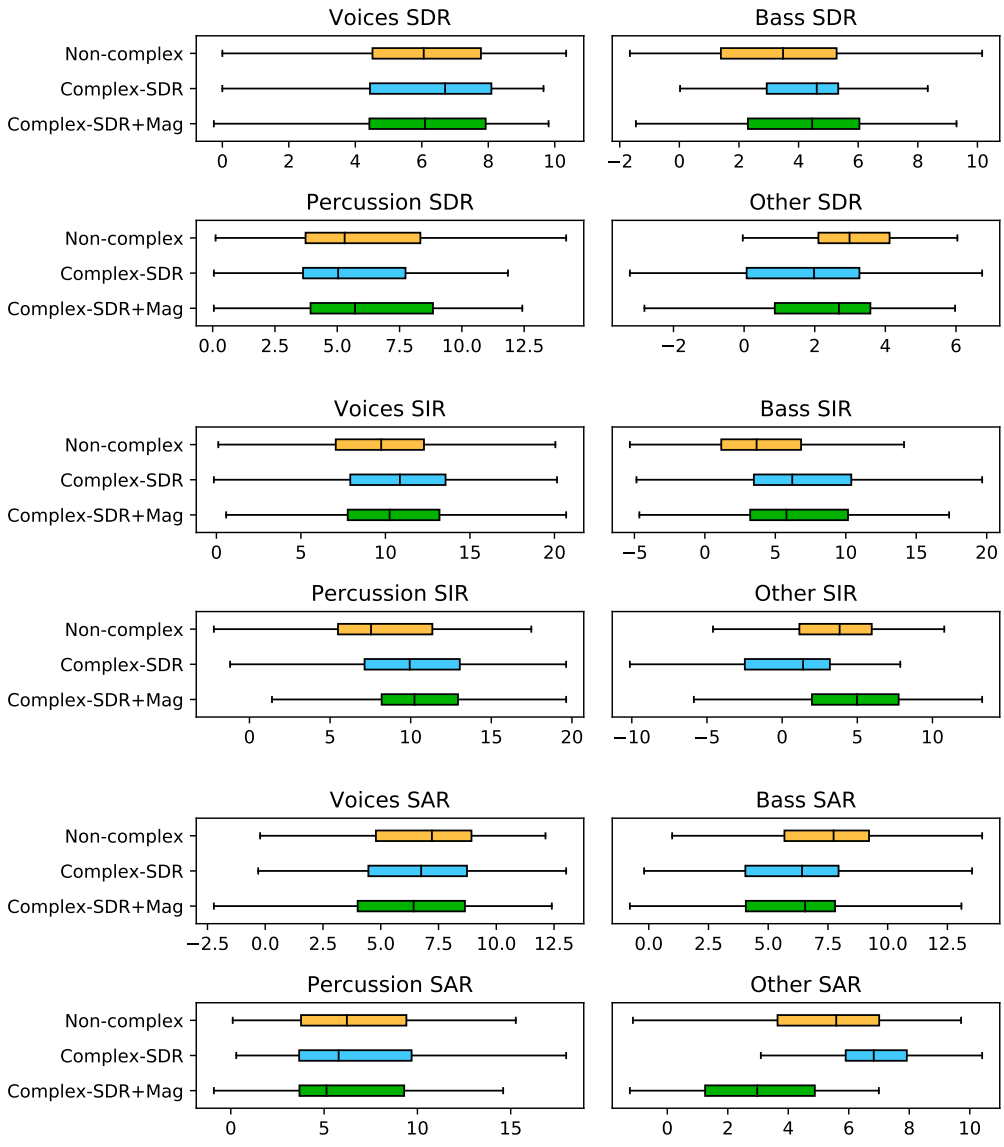


Figure 6.4: SDR (Signal to Distortion ratio), SIR (Signal to Interference ratio), and SAR (Signal to Artifacts Ratio) across experimental conditions.

the complex models over the non-complex baseline, with a mostly equal split between SDR loss and SDR+Magnitude loss.

6.4.1 Qualitative Analysis

The biggest audible difference between the audio output by the complex masking model and the non-complex masking model is the amount of phase distortion, in particular for bass and percussion. When using real masks, both bass and percussion tend to sound very tinny, and the phase of other sources is often captured in the sound of the reconstructed source, such as hearing faint voices in a percussion signal. The predicted vocals sound similar in both types of model, and the overall quality is rather good.

6.4.2 Analysis of learned masks

Figure 6.3 displays the learned masks from the Complex-SDR model on a test track from MUSDB18, and the STFTs of the clean reference sources. The middle and bottom rows show the magnitude and (absolute) phase of the learned masks respectively. Several interesting properties can be observed. As expected, the magnitude of the learned masks looks like that of a typical soft mask, where the time frequency bins of the target source have values close to 1 and other bins have values close to 0. For bass, harmonic patterns with few transients in the low frequency range are noticeable; for voice, harmonic patterns tend to inhabit a higher frequency range and exibit stronger transients, e.g., at the starts of words; percussions exibit more regular transients.

The phase of the mask determines how much the phase of the mixture should be rotated: higher values indicate places where the phase of the mixture is substantially different from the phase of the source (e.g., because it overlaps with another source). In this example, we see that for percussions there are large phase corrections being made for horizontal patterns, likely canceling the phase from harmonic sources in the mixture. Additionally, the phase is corrected the instant before an onset, which likely makes the onset sound more crisp than if it were smeared with phase from other sources. For the vocal signal, most of the phase corrections are made where the bass and percussion masks are active, indicating that the phase of the bass and percussion is being canceled out for the vocal mask. Interestingly, for the phase of the bass mask, close inspection reveals that in the low frequency regions, the frequency bands that have large values from the mask and for the phase are interleaved, implying that the phase of closely overlapping low frequency sources is being canceled.

6.4.3 Benefits of Hybrid Loss

Empirical observations suggest that while the complex mask alleviates the phase artifact problem, the SDR loss has a tendency to distort the spectral envelope of the estimated source. It was this observation that led us to implement the hybrid SDR+Magnitude loss; we hypothesize that incorporating the magnitude loss should improve the accuracy of the overall frequency response. In order to assess this hypothesis, Figure 6.6 plots the average frequency spectrum of the MUSDB18 test set, per source, for the target source and our experiments. For every source except voice, the SDR+Magnitude loss results in a frequency response that is more similar to that of the SDR loss.

Figure 6.7 visualizes SDR and magnitude loss for both complex models at training time (even though only the SDR+Magnitude loss model attempts to minimize magnitude loss). The plots show that the SDR loss causes, predictably, magnitude differences to decrease, however not to the extent that the explicit optimization of the magnitude does. Notably, the combined SDR+Magnitude loss does not degrade the SDR component.

6.5 Conclusions

In this chapter we explored the use of complex masks for multi-instrument source separation using a U-Net architecture. It was found that learned complex masks perform better than non-complex masks perceptually, and marginally better in terms of SDR. An analysis of the learned masks found interesting properties of phase masking for different sources, highlighting a particularly strong effect when masking percussion and bass.

Additionally, the use of a hybrid SDR + magnitude loss, was found to yield better average frequency spectra, in particular matching better in the high frequency range. In future work we plan to incorporate additional perceptually-informed loss functions.

Finally, a practical implication of the proposed approach is that the constrained complex mask is a viable low effort, drop-in replacement for common magnitude-domain masks.

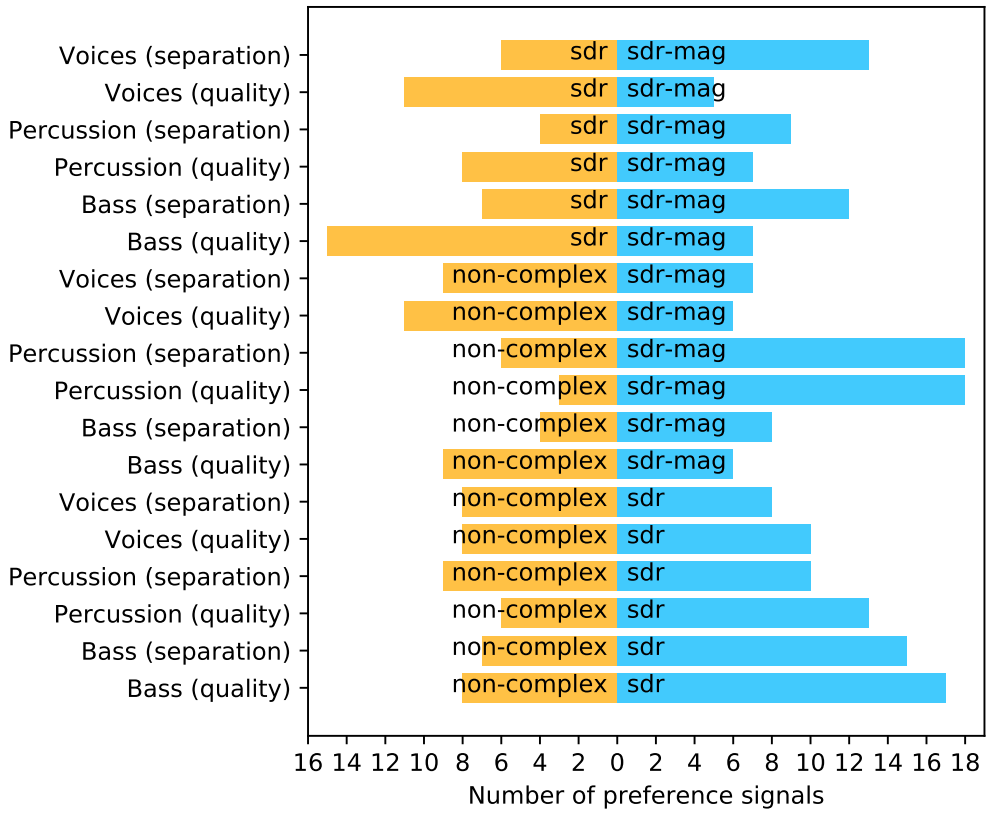


Figure 6.5: Pairwise perceptual preference ratings. The length of the yellow and blue bars represent the number of ratings that favored a particular example. For example, out of 21 examples of "Percussion (quality)", 18 ratings favored the SDR+magnitude model, whereas 3 ratings favored the non-complex model.

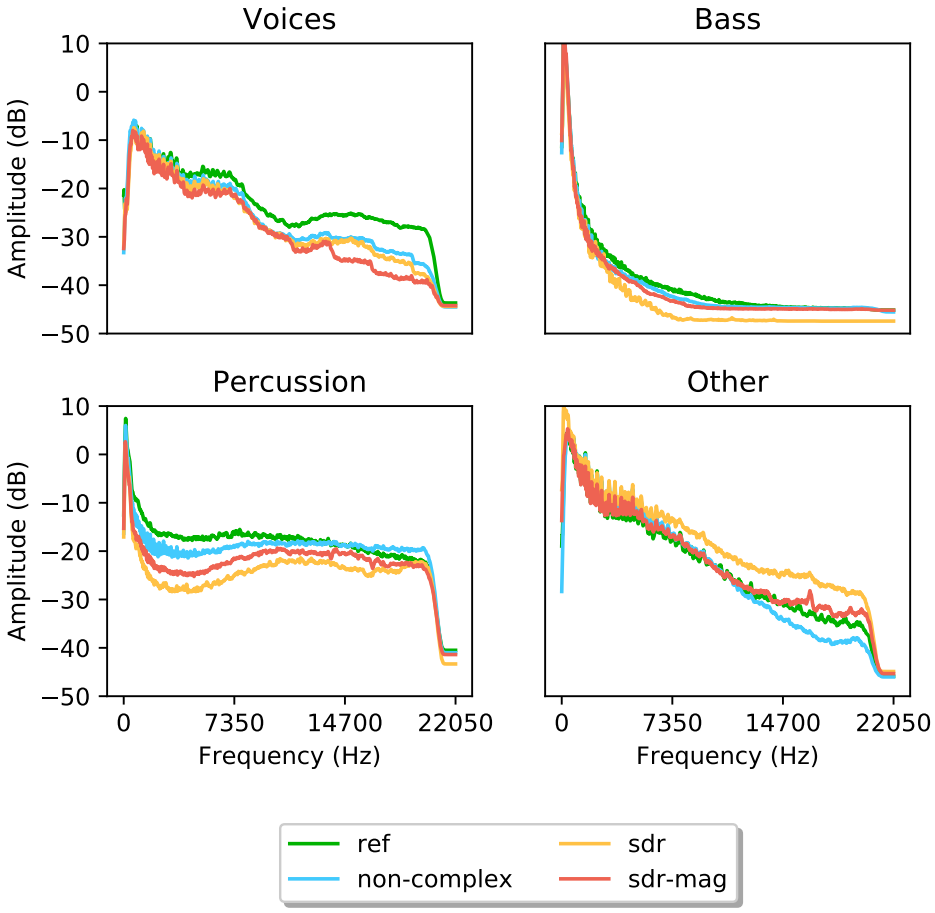


Figure 6.6: Average frequency spectrum per instrument type of the audio outputs of different models against the reference.

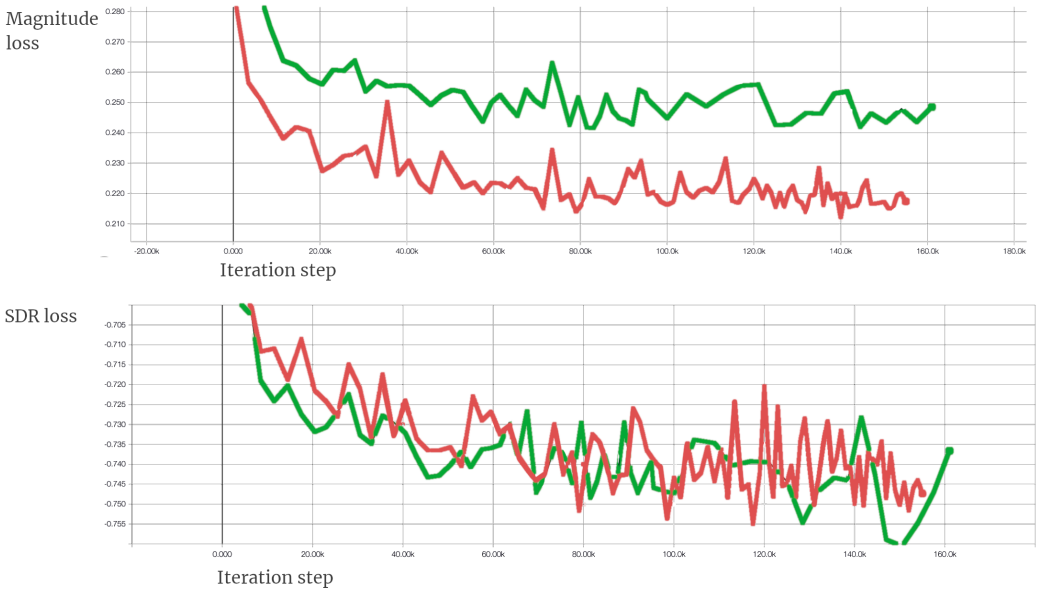


Figure 6.7: Training loss over time. Green line: Complex-SDR model. Red line: Complex-SDR+Magnitude model.

Chapter 7

Conclusion

7.1 Summary

In this thesis we have presented several ways of leveraging large-scale music databases for musical source separation.

We have shown how music data presents many challenges, both in terms of the complexity of music data modeling and the prevalence of inconsistent and incorrect metadata, but also in terms of scale. We have seen how the terabyte scale of contemporary music databases forces us to formulate our problems in way that lends itself to distributed processing.

We have seen how modern distributed data processing frameworks can be deployed for practical, real-world music and audio processing tasks. Using these tools has allowed us to mine commercial music catalogs for meaningful training data for source separation. We have compiled the — to the best of our knowledge — largest dataset to date, of music mixtures mapped to isolated vocals and accompaniment.

Guided by the observation that artists often release instrumental mixes as "B-sides", we designed a matching algorithm to retrieve pairs of instrumentals and original mixtures. This algorithm was efficiently implemented and executed on modern distributed systems, allowing us to quickly retrieve tens of thousands of pairs.

Having a dataset of this size allowed us to build deep neural network models

with high capacity for vocal source separation. While previous attempts to leverage deep learning for music source separation has fallen short of the full potential of these models, it appears that our dataset is sufficiently large. We show new state-of-the-art results on vocal source separation, both on objective and crowdsourced subjective evaluation metrics.

Our neural network architecture, the *U-Net*, was adapted from the image segmentation domain. Two-dimensional mixture spectrograms were used as input "images", and vocal activation masks were analogous to soft segmentation maps.

We then applied source separation as a preprocessing step to vocal fundamental frequency estimation, showing that joint learning of the two tasks resulted in better pitch estimation than a separately learning the two tasks. This work emphasized the importance of choosing the right architecture for joint learning tasks, as several common multi-task architectures failed to improve performance. The most successful architecture was one that stacked a source separation training model followed by a fundamental frequency estimation model, followed by another source separation and fundamental frequency model pair for fine-tuning.

We then conducted research on adaptations of this architecture for multi-source separation. Phase artifacts were a common cause of noise in the magnitude spectrogram-based models for vocal separation. We hypothesized that this problem would become more pronounced as we introduced additional instrument sources, since many instruments have overlapping spectra.

In order to remedy this problem we extended the source separating U-Net to the complex domain, directly estimating magnitude and phase mask coefficients. Qualitatively we saw the phase component of the mask effectively "filter" out the phase of the interfering instruments. In listening tests we found that the resulting synthesized instrument sources suffered from phase artifacts to a lesser degree than the baseline model.

For the complex architecture we used a joint loss function, that simultaneously operated in the time and frequency domains. The time-domain loss was used instead of predicting complex spectrograms directly, and the additional magnitude spectrogram loss worked to preserve the overall frequency characteristics.

7.2 Discussion and future work

Musical source separation is a challenging task for several compounding reasons.

In the commonly used time-frequency representation, sources tend to overlap. The analogy to image segmentation is imperfect, since each "pixel" can be attributed to several sources to varying degrees. Another difference is that objects in images tend to be connected in space, whereas in a spectrogram, instrument activations are represented as disjointed harmonics.

While this thesis has made use of several techniques from image segmentation and recognition, these issues suggest that we ought to invest research into alternative representations that make better use of convolutional neural work properties. The Harmonic Constant-Q Transform (Rachel M. Bittner et al., 2017) (HCQT) is a three-dimensional time-frequency representation where harmonics are stacked in the third dimension, creating continuously connected "shapes" of musical events. While the HCQT was designed to preserve musical pitch information, similar representations should be investigated for musical source separation.

We might also consider replacing the convolution operator with some operator that does not depend on the connectivity of elements. Several novel transforms have been proposed, such as Deformable Convolutional Networks (Dai et al., 2017) Non-local Neural Networks (X. Wang, Girshick, Gupta, & He, 2018), and Point-wise Spatial Attention Networks (H. Zhao et al., 2018) have been proposed to overcome the locality limitations of convolutional network. These methods, which have yet to be applied to musical source separation, could potentially offer improvements over traditional convolutions.

When we train the U-Net model we split the full spectrogram into patches, and train each patch independently. Patches are usually on the order of a few seconds in length, and while this reduces training complexity, it also introduces several limitations. Crucially, we are not able to make use of information from other parts of the song. Several methods have been proposed in other domains to manage such long-range dependencies. In future work we plan to explore techniques such as hybrid convolutional-recurrent models (Shi et al., 2015), Atrous convolution (Chen, Papandreou, Kokkinos, Murphy, & Yuille, 2018), and multi-scale convolutional neural networks (Nah,

Kim, & Lee, 2017).

In Chapter 5 we saw how multi-task learning can improve accuracy in individual tasks. While we did not see an improvement in the source separation task in this scenario, intuition suggests that there exists cases where multi-task learning and/or additional metadata can lead to better separation results. For example incorporating or jointly learning musical genre could feasibly lead to improvements, since different genres have different instrumentation, and different instrument "sounds". Other candidates include release year, artist country of origin, and gender of singer.

The musical multi-source separation task itself is often ill-defined, due to ambiguity in instrument labeling. Are vocoded vocals still considered vocals, or perhaps synthesizer? In modern pop and dance music, most instrument sounds are synthesized or drastically effected. This will have to be addressed if we are to build a multi-instrument source separator that generalizes to many different styles of music.

Parallel to model improvements is future work on dataset mining and creation — large models need large datasets. This is especially important as we start incorporating additional metadata such as genre. Ideally we should have sufficient numbers of examples for each genre.

The most scalable way to increase the dataset size is through augmentation. Given a set of isolated stems, we should be able to automatically mix stems together to new songs, under certain constraints. These constraints should enforce realism, such that stems are correlated and overlapping. It is properties such as these that make musical source separation a particularly challenging problem, and if the augmentation system does not generate music with these properties, the trained model will not generalize to real music. In future work we plan to generate training data by automatically mixing under constraints such as tempo and beat alignment, key matching, chord alignment, and genre matching.

While labeled source separation data is scarce, unlabeled music data is ubiquitous. The challenge is how to make use of this vast dataset to train a source separation model. This problem is by no means unique to our task, but is prevalent in many machine learning fields. Several kinds of unsupervised learning architectures have been developed in order to address this problem, many of which can be adapted to source separation. Similar to how

the U-Net model has been used in fully convolutional image segmentation systems, the W-Net (Xia & Kulis, 2017) is an unsupervised auto-encoder consisting of two stacked U-Nets, with the middle layer producing an image segmentation map. *Co-training* has also been used for image segmentation (Peng, Estradab, Pedersoli, & Desrosiers, 2019) as a way to augment a small labeled dataset with a large unlabeled dataset from the same distribution. These models should be applicable to source separation, potentially with additional task-specific post-processing.

In the absence of true training data such as stems and instrumentals, we may still be able to learn from weakly labeled data in a semi-supervised fashion. The Discogs database[1] contains instrumentation information for a large number of tracks. We intend to make use of the semi-supervised algorithm in (Xiao, Wei, Liu, Zhang, & Feng, 2018) to learn source separation masks from these track-level instrument labels in conjunction with a small strongly labeled dataset.

On the application side, we see plenty of popular games and features that put isolated instrument sources and/or vocals at center stage. Through automatic source separation, we will be able to extend these experiences beyond the most mainstream content, creating personalized features that serve diverse tastes in music.

Digital music is becoming ubiquitous, yet the modes of music consumption have barely changed since the beginning of recorded music. We are finally starting to develop technologies that will allow new, creative, and unconventional ways of interacting with music, beyond simply pressing play on the latest releases. And we believe that automatic source separation has the potential to be a driving force in this transformation of music consumption and creation.

[1] https://www.discogs.com/

Bibliography

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... Zheng, X. (2016). TensorFlow: A System for Large-Scale Machine Learning. In K. Keeton & T. Roscoe (Eds.), *OSDI* (pp. 265–283). USENIX Association.

Akidau, T., Bradshaw, R., Chambers, C., Chernyak, S., Fernández-Moctezuma, R., Lax, R., ... Whittle, S. (2015). The Dataflow Model: A Practical Approach to Balancing Correctness, Latency, and Cost in Massive-Scale, Unbounded, Out-of-Order Data Processing. *PVLDB*, *8*(12), 1792–1803.

Arik, S. Ö., Jun, H., & Diamos, G. F. (2019). Fast Spectrogram Inversion Using Multi-Head Convolutional Neural Networks. *IEEE Signal Processing Letters*, *26*(1), 94–98.

Bach, F. R., & Jordan, M. I. (2006). Learning Spectral Clustering, With Application To Speech Separation. *Journal of Machine Learning Research*, *7*, 1963–2001.

Bai, S., Kolter, J. Z., & Koltun, V. (2018). An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv preprint arXiv:1803.01271*.

Bengio, Y. (2016). Machines Who Learn. *Scientific American*, *314*(6), 46–51.

Berenzweig, A., & Ellis, D. (2001). Locating Singing Voice Segments Within Music Signals. In *Proceedings of IEEE WASPAA*. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), New Platz, NY.

Bertin-Mahieux, T., Ellis, D. P. W., Whitman, B., & Lamere, P. (2011). The Million Song Dataset. In *Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011* (pp. 591–596).

Bittner, R. [R.], Wilkins, J., Yip, H., & Bello, J. (2016). MedleyDB 2.0: New Data and a System for Sustainable Data Collection. In *Proceedings of the International Society for Music Information Retrieval Conference Late Breaking and Demo Papers*.

Bittner, R. M. [Rachel M.], McFee, B., Salamon, J., Li, P., & Bello, J. P. (2017). Deep Salience Representations for F0 Estimation in Polyphonic Music. In S. J. Cunningham, Z. Duan, X. Hu, & D. Turnbull (Eds.), *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017* (pp. 63–70).

Bittner, R. M. [Rachel M.], Salamon, J., Tierney, M., Mauch, M., Cannam, C., & Bello, J. P. (2014). MedleyDB: A Multitrack Dataset for Annotation-Intensive MIR Research. In *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014* (pp. 155–160).

Bittner, R. [Rachel], Humphrey, E., & Bello, J. (2016). PySox: Leveraging the Audio Signal Processing Power of Sox in Python. In *Proceedings of the International Society for Music Information Retrieval Conference Late Breaking and Demo Papers*.

Boulanger-Lewandowski, N., Mysore, G. J., & Hoffman, M. D. (2014). Exploiting Long-Term Temporal Dependencies in NMF Using Recurrent Neural Networks with Application to Source Separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014* (pp. 6969–6973). IEEE.

Bregman, A. S. (1990). *Auditory scene analysis: The Perceptual Organization of Sound*. Boston, MA, USA: MIT Press.

Brown, G. J., & Cooke, M. (1994). Computational Auditory Scene Analysis. *Computer Speech & Language*, *8*(4), 297–336.

Bryan, N. J., Mysore, G. J., & Wang, G. (2013). Source Separation of Polyphonic Music with Interactive User-Feedback on a Piano Roll Display. In A. de Souza Britto Jr., F. Gouyon, & S. Dixon (Eds.), *Proceedings of the 14th International Society for Music Information Retrieval Conference, ISMIR 2013, Curitiba, Brazil, November 4-8, 2013* (pp. 119–124).

Brynjolfsson, E., Hu, Y. J., & Smith, M. D. (2006). From Niches to Riches: The Anatomy of the Long Tail. *Sloan Management Review*, *47*(2), 67–71.

Burel, G. (1992). Blind Separation of Sources: A Nonlinear Neural Algorithm. *Neural Networks*, *5*(6), 937–947.

Candes, E., Li, X., Ma, Y., & Wright, J. (2010). Robust Principal Component Analysis?: Recovering Low-rank Matrices from Sparse Errors. *2010 IEEE Sensor Array and Multichannel Signal Processing Workshop*.

Cartwright, M., Pardo, B., & Mysore, G. J. (2018). Crowdsourced Pairwise-Comparison for Source Separation Evaluation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018* (pp. 606–610). IEEE.

Casey, M. A., & Westner, A. (2000). Separation of Mixed Audio Sources By Independent Subspace Analysis. In *Proceedings of the 2000 International Computer Music Conference, ICMC 2000, Berlin, Germany, August 27 - September 1, 2000*, Michigan Publishing.

Chan, T.-S., Yeh, T.-C., Fan, Z.-C., Chen, H.-W., Su, L., Yang, Y.-H., & Jang, R. (2015). Vocal Activity Informed Singing Voice Separation with the iKala Dataset. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015, South Brisbane, Queensland, Australia, April 19-24, 2015* (pp. 718–722). IEEE.

Chandna, P., Miron, M., Janer, J., & Gómez, E. (2017). Monoaural Audio Source Separation Using Deep Convolutional Neural Networks. In P. Tichavský, M. Babaie-Zadeh, O. J. J. Michel, & N. Thirion-Moreau (Eds.), *Latent Variable Analysis and Signal Separation - 13th International Conference, LVA/ICA 2017, Grenoble, France, February 21-23, 2017, Proceedings* (Vol. 10169, pp. 258–266). Lecture Notes in Computer Science.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell. 40*(4), 834–848.

Choi, H.-S., Kim, J.-H., Huh, J., Kim, A., Ha, J.-W., & Lee, K. (2019). Phase-aware Speech Enhancement with Deep Complex U-Net. *CoRR*, *abs/1903.03107*.

Comon, P. (1994). Independent Component Analysis, A New Concept? *Signal Processing*, *36*(3), 287–314.

Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., & Wei, Y. (2017). Deformable Convolutional Networks. *CoRR*, *abs/1703.06211*.

Darwin, C. J., Brungart, D. S., & Simpson, B. D. (2003). Effects of Fundamental Frequency and Vocal-tract Length Changes on Attention to One of Two Simultaneous Talkers. *The Journal of the Acoustical Society of America*, *114*(5), 2913.

Dean, J., & Ghemawat, S. (2008). MapReduce: Simplified Data Processing on Large Clusters. *Communications of the ACM*, *51*(1), 107–113.

Defferrard, M., Benzi, K., Vandergheynst, P., & Bresson, X. (2017). FMA: A Dataset for Music Analysis. In *18th International Society for Music Information Retrieval Conference*.

Demany, L. (1982). Auditory Stream Segregation in Infancy. *Infant Behavior and Development*, *5*(2-4), 261–276.

Demetriou, A., Jansson, A., Kumar, A., & Bittner, R. M. (2018). Vocals in Music Matter: the Relevance of Vocals in the Minds of Listeners. In E. Gómez, X. Hu, E. Humphrey, & E. Benetos (Eds.), *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017* (pp. 514–520).

Deutsch, D. (1999). Grouping Mechanisms in Music. In *The Psychology of Music* (pp. 299–348). Elsevier.

Deville, Y., Jutten, C., & Vigario, R. (2010). Overview of Source Separation Applications. *Handbook of Blind Source Separation*, 639–681.

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*.

Diehl, R. L. (2008). Acoustic and Auditory Phonetics: The Adaptive Design of Speech Sound Systems. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *363*(1493), 965–978.

Doras, G., Esling, P., & Peeters, G. (2019). On the Use of U-Net for Dominant Melody Estimation in Polyphonic Music. In *2019 International Workshop on Multilayer Music Representation and Processing (MMRP)* (pp. 66–70). IEEE.

Dowling, W. J. (1973). Rhythmic Groups and Subjective Chunks in Memory for Melodies. *Perception & Psychophysics*, *14*(1), 37–40.

Dubey, M. L., Kenyon, G. T., Carlson, N., & Thresher, A. (2017). Does Phase Matter for Monaural Source Separation? *CoRR*, *abs/1711.00913*.

Dumoulin, V., & Visin, F. (2016). A Guide to Convolution Arithmetic for Deep Learning. *arXiv preprint arXiv:1603.07285*.

Durrieu, J.-L., Richard, G., & David, B. (2008). Singer Melody Extraction in Polyphonic Signals Using Source Separation Methods. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2008, March 30 - April 4, 2008, Caesars Palace, Las Vegas, Nevada, USA* (pp. 169–172). IEEE.

Durrieu, J.-L., Richard, G., David, B., & Févotte, C. (2010). Source/Filter Model for Unsupervised Main Melody Extraction From Polyphonic Audio Signals. *IEEE Trans. Audio, Speech & Language Processing*, *18*(3), 564–575.

Durrieu, J.-L., & Thiran, J.-P. (2012). Musical Audio Source Separation Based on User-Selected F0 Track. In F. J. Theis, A. Cichocki, A. Yeredor, & M. Zibulevsky (Eds.), *Latent Variable Analysis and Signal Separation - 10th International Conference, LVA/ICA 2012, Tel Aviv, Israel, March 12-15, 2012. Proceedings* (Vol. 7191, pp. 438–445). Lecture Notes in Computer Science. Springer.

Edunov, S., Ott, M., Auli, M., & Grangier, D. (2018). Understanding Back-translation at Scale. *arXiv preprint arXiv:1808.09381*.

Ellis, D. P. W. (1996). *Prediction-driven Computational Auditory Scene Analysis.* (Doctoral dissertation, Massachusetts Institute of Technology, Cambridge MA, USA).

Emiya, V., Vincent, E., Harlander, N., & Hohmann, V. (2011). Subjective and Objective Quality Assessment of Audio Source Separation. *IEEE Transactions on Audio, Speech, and Language Processing*, *19*(7), 2046–2057.

Erdogan, H., Hershey, J. R., Watanabe, S., & Roux, J. L. (2015). Phase-sensitive and Recognition-Boosted Speech Separation using Deep Recurrent Neural Networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015, South Brisbane, Queensland, Australia, April 19-24, 2015* (pp. 708–712). IEEE.

Fan, Z.-C., Lai, Y.-L., & Jang, J.-S. R. (2018). SVSGAN: Singing Voice Separation Via Generative Adversarial Network. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018* (pp. 726–730). IEEE.

Fishman, Y. I., Arezzo, J. C., & Steinschneider, M. (2004). Auditory Stream Segregation in Monkey Auditory Cortex: Effects of Frequency Separa-

tion, Presentation Rate, And Tone Duration. *The Journal of the Acoustical Society of America*, *116*(3), 1656–1670.

FitzGerald, D., Lawlor, R., & Coyle, E. (2003). Prior subspace analysis for drum transcription. In *Proceedings of the 114th AES Conference, Amsterdam, Netherlands, 2003*, Dublin Institute of Technology.

Fox, G., Johnson, M., Lyzenga, G., Otto, S., Salmon, J., Walker, D., & White, R. L. (1989). Solving Problems On Concurrent Processors Vol. 1: General Techniques and Regular Problems. *Computers in Physics*, *3*(1), 83.

Freed, A. (2006). Music MetaData Quality: A Multiyear Case Study using the Music of Skip James. In *Audio Engineering Society Convention 121*.

Friedl, J. E. (2006). *Mastering Regular Expressions*. " O'Reilly Media, Inc.".

Fu, S.-W., yao Hu, T., Tsao, Y., & Lu, X. (2017). Multi-Metrics Learning for Speech Enhancement. *CoRR*, *abs/1704.08504*.

Geirhos, R., Janssen, D. H. J., Schütt, H. H., Rauber, J., Bethge, M., & Wichmann, F. A. (2017). Comparing Deep Neural Networks Against Humans: Object Recognition when the Signal Gets Weaker. *arXiv preprint arXiv:1707.06969*.

Ghemawat, S., Gobioff, H., & Leung, S.-T. (2003). The Google File System. In M. L. Scott & L. L. Peterson (Eds.), *Proceedings of the 19th ACM Symposium on Operating Systems Principles 2003, SOSP 2003, Bolton Landing, NY, USA, October 19-22, 2003* (pp. 29–43). ACM.

Goodfellow, I., Bengio, Y., & Courville, A. (2018). *Deep Learning: Das umfassende Handbuch* [Grundlagen, aktuelle verfahren und algorithmen, neue forschungsansätze]. Frechen: MITP.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative Adversarial Nets. In *Advances in neural information processing systems* (pp. 2672–2680).

Grais, E. M., Roma, G., Simpson, A. J. R., & Plumbley, M. D. (2016). Combining Mask Estimates for Single Channel Audio Source Separation Using Deep Neural Networks. In N. Morgan (Ed.), *INTERSPEECH 2013, 14th Annual Conference of the International Speech Communication Association, Lyon, France, August 25-29, 2013* (pp. 3339–3343). ISCA.

Grais, E. M., Sen, M. U., & Erdogan, H. (2014). Deep Neural Networks for Single Channel Source Separation. In *IEEE International Conference*

on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014 (pp. 3734–3738). IEEE.

Grais, E. M., Wierstorf, H., Ward, D., & Plumbley, M. D. (2018). Multi-Resolution Fully Convolutional Neural Networks for Monaural Audio Source Separation. In Y. Deville, S. Gannot, R. Mason, M. D. Plumbley, & D. Ward (Eds.), *Latent Variable Analysis and Signal Separation - 14th International Conference, LVA/ICA 2018, Guildford, UK, July 2-5, 2018, Proceedings* (Vol. 10891, pp. 340–350). Lecture Notes in Computer Science. Springer.

He, S., & Schomaker, L. (2019). DeepOtsu: Document Enhancement and Binarization Using Iterative Deep Learning. *Pattern Recognition*.

Hennequin, R., Khlif, A., Voituret, F., & 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.

Herault, J., & Jutten, C. [C.]. (1986). Space or Time Adaptive Signal Processing by Neural Network Models. *AIP Conference Proceedings*.

Herrmann, M., & Yang, H. (1996). Perspectives and Limitations of Self-Organizing Maps in Blind Separation of Source Signals. In *Progress in Neural Information Processing. Proceedings of the International Conference on Neural Information Processing* (Vol. 2, p. 1211). Citeseer.

Hershey, J. R., Chen, Z., Roux, J. L., & Watanabe, S. (2015). Deep Clustering: Discriminative Embeddings for Segmentation and Separation. *CoRR*, *abs/1508.04306*.

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531*.

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models. *arXiv preprint arXiv:2006.11239*.

Hsu, C.-L., & Jang, J.-S. R. (2010). On the Improvement of Singing Voice Separation for Monaural Recordings Using the MIR-1K Dataset. *IEEE Trans. Audio, Speech & Language Processing*, *18*(2), 310–319.

Hsu, C.-L., Wang, D., Jang, J.-S. R., & Hu, K. (2012). A Tandem Algorithm for Singing Pitch Extraction and Voice Separation From Music Accompaniment. *IEEE Trans. Audio, Speech & Language Processing*, *20*(5), 1482–1491.

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017* (pp. 2261–2269).

Huang, P., Kim, M., Hasegawa-Johnson, M., & Smaragdis, P. (2014). Singing-Voice Separation from Monaural Recordings using Deep Recurrent Neural Networks. In *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014* (pp. 477–482).

Huang, P.-S., Chen, S. D., Smaragdis, P., & Hasegawa-Johnson, M. (2012). Singing-voice Separation from Monaural Recordings using Robust Principal Component Analysis. *2012 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2012, Kyoto, Japan, March 25-30, 2012*.

Huang, P.-S., Kim, M., Hasegawa-Johnson, M., & Smaragdis, P. (2014). Deep Learning for Monaural Speech Separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014* (pp. 1562–1566). IEEE.

Humphrey, E. J. [Eric J.], Bello, J. P., & LeCun, Y. (2013). Feature Learning and Deep Architectures: New Directions for Music Informatics. *J. Intell. Inf. Syst. 41*(3), 461–481.

Humphrey, E. J. [Eric J], Reddy, S., Seetharaman, P., Kumar, A., Bittner, R. M. [Rachel M], Demetriou, A., ... Lehner, B., et al. (2018). An Introduction to Signal Processing for Singing-Voice Analysis: High Notes in the Effort to Automate the Understanding of Vocals in Music. *IEEE Signal Processing Magazine*, *36*(1), 82–94.

Humphrey, E., Montecchio, N., Bittner, R., Jansson, A., & Jehan, T. (2017). Mining Labeled Data from Web-Scale Collections for Vocal Activity Detection in Music. In *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*.

Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2017). Image-to-Image Translation with Conditional Adversarial Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jansson, A., Bittner, R. M., Ewert, S., & Weyde, T. (2019). Joint Singing Voice Separation and F0 Estimation with Deep U-Net Architectures. In *Proceedings of the 27th European Signal Processing Conference*.

Jansson, A., Humphrey, E. J., Montecchio, N., Bittner, R. M., Kumar, A., & Weyde, T. (2017). Singing Voice Separation with Deep U-Net Convolutional Networks. In S. J. Cunningham, Z. Duan, X. Hu, & D. Turnbull (Eds.), *Proceedings of the 18th International Society for Music Informa-*

tion Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017 (pp. 745–751).

Jeong, I.-Y., & Lee, K. (2014). Vocal Separation from Monaural Music Using Temporal/Spectral Continuity and Sparsity Constraints. *IEEE Signal Processing Letters*, *21*(10), 1197–1200.

Juslin, P., Patrik N.and Laukka. (2003). Communication of Emotions in Vocal Expression and Musical Performance: Different Channels, Same Code? *Psychological Bulletin*, *129*, 770–814.

Jutten, C. [Christian], & Hérault, J. (1991). Blind Separation of Sources, Part I: An Adaptive Algorithm Based on Neuromimetic Architecture. *Signal Processing*, *24*(1), 1–10.

Kashino, K., & Murase, H. (1999). A Sound Source Identification System for Ensemble Music Based on Template Adaptation and Music Stream Extraction. *Speech Communication*, *27*(3-4), 337–349.

Kim, J. W., Salamon, J., Li, P., & Bello, J. P. (2018). CREPE: A Convolutional Representation for Pitch Estimation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018* (pp. 161–165). IEEE.

King, B., & Atlas, L. E. (2010). Single-channel Source Separation using Simplified-training Complex Matrix Factorization. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2010, 14-19 March 2010, Sheraton Dallas Hotel, Dallas, Texas, USA* (pp. 4206–4209). IEEE.

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*.

Kolde, R., Laur, S., Adler, P., & Vilo, J. (2012). Robust Rank Aggregation for Gene List Integration and Meta-analysis. *Bioinformatics*, *28*(4), 573–580.

Lamere, P. (2008). How Hard Can Music Metadata Be? Retrieved January 15, 2019, from http://static.echonest.com/DukeListens/how_hard_can_music_metadata.html

Laroche, C., Papadopoulos, H., Kowalski, M., & Richard, G. (2016). Genre Specific Dictionaries for Harmonic/Percussive Source Separation. In M. I. Mandel, J. Devaney, D. Turnbull, & G. Tzanetakis (Eds.), *Proceedings of the 17th International Society for Music Information Retrieval Conference, ISMIR 2016, New York City, United States, August 7-11, 2016* (pp. 407–413).

Lee, Y.-S., Wang, C.-Y., Wang, S.-F., Wang, J.-C., & Wu, C.-H. (2017). Fully Complex Deep Neural Network for Phase-incorporating Monaural Source Separation. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017* (pp. 281–285). IEEE.

Lerch, A. (2019). Audio Content Analysis: Data-sets. Retrieved January 15, 2019, from https://www.audiocontentanalysis.org/data-sets

Li, Y., & Wang, D. (2007). Separation of Singing Voice From Music Accompaniment for Monaural Recordings. *IEEE Transactions on Audio, Speech, and Language Processing*, *15*(4), 1475–1487.

Liang, J., Jiang, L., Niebles, J. C., Hauptmann, A. G., & Fei-Fei, L. (2019). Peeking Into the Future: Predicting Future Person Activities and Locations in Videos. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*.

Liem, C. C. S., Müller, M., Eck, D., Tzanetakis, G., & Hanjalic, A. (2011). The Need for Music Information Retrieval with User-centered and Multimodal Strategies. In C. C. S. Liem, M. Müller, D. Eck, & G. Tzanetakis (Eds.), *MIRUM '11: Proceedings of the 1st international ACM workshop on Music information retrieval with user-centered and multimodal strategies* (pp. 1–6). ACM.

Likert, R. (1932). A Technique for the Measurement of Attitudes. *Archives of psychology*.

Lim, B., Arik, S. O., Loeff, N., & Pfister, T. (2019). Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting. *arXiv preprint arXiv:1912.09363v2*.

Liu, J.-Y., & Yang, Y.-H. (2018). Denoising Auto-Encoder with Recurrent Skip Connections and Residual Regression for Music Source Separation. In M. A. Wani, M. M. Kantardzic, M. S. Mouchaweh, J. Gama, & E. Lughofer (Eds.), *17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018, Orlando, FL, USA, December 17-20, 2018* (pp. 773–778). IEEE.

Liu, Y., Qin, Z., Luo, Z., & Wang, H. (2017). Auto-painter: Cartoon Image Generation from Sketch by Using Conditional Generative Adversarial Networks. *CoRR*, *abs/1705.01908*.

Liutkus, A., Fitzgerald, D., Rafii, Z., Pardo, B., & Daudet, L. (2014). Kernel Additive Models for Source Separation. *IEEE Transactions on Signal Processing*, *62*(16), 4298–4310.

Liutkus, A., & Stöter, F.-R. (2019). sigsep/norbert: First Official Norbert Release (Version v0.2.0). Zenodo.

Lluís, F., Pons, J., & Serra, X. (2018). End-to-end Music Source Separation: Is It Possible in the Waveform Domain? *CoRR*, *abs/1810.12187*.

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully Convolutional Networks for Semantic Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3431–3440).

Losorelli, S., Nguyen, D. T., Dmochowski, J. P., & Kaneshiro, B. (2017). Naturalistic Music EEG Dataset - Tempo (NMED-T). Stanford Digital Repository. Retrieved January 15, 2019, from http://purl.stanford.edu/jn859kj8079

Lu, X., Matsuda, S., Hori, C., & Kashioka, H. (2012). Speech Restoration Based on Deep Learning Autoencoder with Layer-wised Pretraining. In *INTERSPEECH 2012, 13th Annual Conference of the International Speech Communication Association, Portland, Oregon, USA, September 9-13, 2012* (pp. 1504–1507). ISCA.

Luo, Y., Chen, Z., Hershey, J. R., Roux, J. L., & Mesgarani, N. (2017a). Deep clustering and conventional networks for music separation: Stronger together. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017* (pp. 61–65). IEEE.

Luo, Y., Chen, Z., Hershey, J. R., Roux, J. L., & Mesgarani, N. (2017b). Deep Clustering and Conventional Networks for Music Separation: Stronger Together. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017* (pp. 61–65). IEEE.

Luo, Y., & Mesgarani, N. (2018). TaSNet: Time-Domain Audio Separation Network for Real-Time, Single-Channel Speech Separation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018* (pp. 696–700). IEEE.

Maas, A. L., Le, Q. V., O'Neil, T. M., Vinyals, O., Nguyen, P., & Ng, A. Y. (2012). Recurrent Neural Networks for Noise Reduction in Robust ASR. In *INTERSPEECH 2012, 13th Annual Conference of the International Speech Communication Association, Portland, Oregon, USA, September 9-13, 2012* (pp. 22–25). ISCA.

Macartney, C., & Weyde, T. (2018). Improved Speech Enhancement with the Wave-U-Net. *arXiv preprint arXiv:1811.11307*.

Magron, P., Badeau, R., & David, B. (2018). Model-Based STFT Phase Recovery for Audio Source Separation. *IEEE/ACM Trans. Audio, Speech & Language Processing*, *26*(6), 1091–1101.

Mauch, M., & Dixon, S. (2014). pYIN: A Fundamental Frequency Estimator using Probabilistic Threshold Distributions. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014* (pp. 659–663). IEEE.

Mayer, F., Williamson, D. S., Mowlaee, P., & Wang, D. (2017). Impact of Phase Estimation on Single-channel Speech Separation Based on Time-frequency Masking. *The Journal of the Acoustical Society of America*, *141*(6), 4668–4679.

McAfee, A., & Brynjolfsson, E. (2012). Big Data: The Management Revolution. *Harvard Business Review*, *90*(10), 60–68.

Melnik, S., Gubarev, A., Long, J. J., Romer, G., Shivakumar, S., Tolton, M., & Vassilakis, T. (2010). Dremel: Interactive Analysis of Web-Scale Datasets. *PVLDB*, *3*(1), 330–339.

Meseguer-Brocal, G., & Peeters, G. (2019). Conditioned-U-Net: Introducing a Control Mechanism in the U-Net for Multiple Source Separations. In *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, Delft, The Netherlands, November 4-8, 2019*.

Mimilakis, S. I., Drossos, K., Virtanen, T., & Schuller, G. (2017). A Recurrent Encoder-Decoder Approach with Skip-Filtering Connections for Monaural Singing Voice Separation. In N. Ueda, S. Watanabe, T. Matsui, J.-T. Chien, & J. Larsen (Eds.), *MLSP* (pp. 1–6). IEEE.

MIREX 2014:Singing Voice Separation. (2014). Retrieved January 15, 2019, from https://www.music-ir.org/mirex/wiki/2014:Singing_Voice_Separation

MIREX 2016:Singing Voice Separation Results. (2016). Retrieved January 15, 2019, from https://www.music-ir.org/mirex/wiki/2016:Singing_Voice_Separation_Results

Miron, M., Janer, J., & Gómez, E. (2017). Generating Data to Train Convolutional Neural Networks for Classical Music Source Separation. In *Proceedings of the 14th Sound and Music Computing Conference* (pp. 227–233). Aalto, Finland.

Moore, B. C. J. (2004). *An Introduction to the Psychology of Hearing* (5th ed.). Academic Press.

Muth, J., Uhlich, S., Perraudin, N., Kemp, T., Cardinaux, F., & Mitsufuji, Y. (2018). Improving DNN-based Music Source Separation using Phase Features. *CoRR*, *abs/1807.02710*.

Nah, S., Kim, T. H., & Lee, K. M. (2017). Deep Multi-scale Convolutional Neural Network for Dynamic Scene Deblurring. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017* (pp. 257–265). IEEE Computer Society.

Narayanan, A., & Wang, D. (2013). Ideal Ratio Mask Estimation using Deep Neural Networks for Robust Speech Recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013* (pp. 7092–7096). IEEE.

Noh, H., Hong, S., & Han, B. (2015). Learning Deconvolution Network for Semantic Segmentation. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1520–1528).

Nugraha, A. A., Liutkus, A., & Vincent, E. (2018). Deep Neural Network Based Multichannel Audio Source Separation. *Signals and Communication Technology*, 157–185.

Oppenheim, A. V., & Lim, J. S. (1981). The Importance of Phase in Signals. *Proceedings of the IEEE*, *69*(5), 529–541.

Palangi, H., Deng, L., & Ward, R. K. (2014). Recurrent Deep-stacking Networks for Sequence Classification. In *2014 IEEE China Summit & International Conference on Signal and Information Processing (ChinaSIP)* (pp. 510–514). IEEE.

Paliwal, K. K., Wójcicki, K. K., & Shannon, B. J. (2011). The Importance of Phase in Speech Enhancement. *Speech Communication*, *53*(4), 465–494.

Pandey, A., & Wang, D. (2018). On Adversarial Training and Loss Functions for Speech Enhancement. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018* (pp. 5414–5418). IEEE.

Park, S., Kim, T., Lee, K., & Kwak, N. (2018). Music Source Separation Using Stacked Hourglass Networks. In *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23-27, 2018*.

Pascual, S., Bonafonte, A., & Serrà, J. (2017). SEGAN: Speech Enhancement Generative Adversarial Network. *CoRR*, *abs/1703.09452*.

Pathak, D., Krähenbühl, P., Donahue, J., Darrell, T., & Efros, A. A. (2016). Context Encoders: Feature Learning by Inpainting. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016* (pp. 2536–2544). IEEE Computer Society.

Peng, J., Estradab, G., Pedersoli, M., & Desrosiers, C. (2019). Deep Co-Training for Semi-Supervised Image Segmentation. *CoRR*, *abs/1903.11233*.

Qiao, S., Chen, L., & Yuille, A. L. (2020). DetectoRS: Detecting Objects with Recursive Feature Pyramid and Switchable Atrous Convolution. *arXiv preprint arXiv:2006.02334*.

Raffel, C., McFee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., & Ellis, D. P. W. (2014). MIR_EVAL: A Transparent Implementation of Common MIR Metrics. In *Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR 2014, Taipei, Taiwan, October 27-31, 2014* (pp. 367–372).

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., ... Liu, P. J. (2019). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *arXiv preprint arXiv:1910.10683v2*.

Rafii, Z., & Pardo, B. (2013). REpeating Pattern Extraction Technique (REPET): A Simple Method for Music/Voice Separation. *IEEE Trans. Audio, Speech & Language Processing*, *21*(1), 71–82.

Rethage, D., Pons, J., & Serra, X. (2018). A Wavenet for Speech Denoising. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018* (pp. 5069–5073). IEEE.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (pp. 234–241). Springer.

Rosenblatt, F. (1958). The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological review*, *65*(6), 386.

Ruder, S. (2017). An Overview of Multi-Task Learning in Deep Neural Networks. *CoRR*, *abs/1706.05098*.

Salamon, J., Gómez, E., Ellis, D. P., & Richard, G. (2014). Melody Extraction from Polyphonic Music Signals: Approaches, Applications, and Challenges. *IEEE Signal Processing Magazine*, *31*(2), 118–134.

Schmidhuber, J. (2015). Deep Learning in Neural Networks: An Overview. *Neural Networks*, *61*, 85–117.

Schönberg, A. (1922). *Harmonielehre* (third). Vienna: Universal Edition.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., & chun Woo, W. (2015). Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. *CoRR*, *abs/1506.04214*.

Simpson, A. J. R. (2015). Deep Transform: Cocktail Party Source Separation via Complex Convolution in a Deep Neural Network. *CoRR*, *abs/1504.02945*.

Simpson, A. J. R., Roma, G., & Plumbley, M. D. (2015). Deep Karaoke: Extracting Vocals from Musical Mixtures Using a Convolutional Deep Neural Network. *Lecture Notes in Computer Science*, 429–436.

Smaragdis, P., & Brown, J. (2003). Non-Negative Matrix Factorization for Polyphonic Music Transcription. *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No.03TH8684)*.

Smule. (2017). The Stanford Digital Archive of Mobile Performances. Retrieved January 15, 2019, from https://ccrma.stanford.edu/damp/

Snir, M., Otto, S. W., Huss-Lederman, S., Walker, D. W., & Dongarra, J. (1996). *MPI: The complete reference*. Cambridge, MA: MIT Press.

Spoorthi, G. E., Gorthi, S., & Gorthi, R. K. S. S. (2019). PhaseNet: A Deep Convolutional Neural Network for Two-Dimensional Phase Unwrapping. *IEEE Signal Processing Letters*, *26*(1), 54–58.

Srivastava, R. K., Greff, K., & Schmidhuber, J. (2015). Highway Networks. *arXiv preprint arXiv:1505.00387v2*.

Stoller, D., Durand, S., & Ewert, S. (2019). End-to-end Lyrics Alignment for Polyphonic Music using an Audio-to-character Recognition Model. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019* (pp. 181–185). IEEE.

Stoller, D., Ewert, S., & Dixon, S. (2018). Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation. In *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018, Paris, France, September 23-27, 2018* (pp. 334–340).

Stöter, F.-R., Liutkus, A., & Ito, N. (2018). The 2018 Signal Separation Evaluation Campaign. In Y. Deville, S. Gannot, R. Mason, M. D. Plumbley, & D. Ward (Eds.), *Latent Variable Analysis and Signal Separation - 14th International Conference, LVA/ICA 2018, Guildford, UK,*

July 2-5, 2018, Proceedings (Vol. 10891, pp. 293–305). Lecture Notes in Computer Science. Springer.

Takahashi, N., & Mitsufuji, Y. (2017). Multi-scale Multi-band DenseNets for Audio Source Separation. *CoRR, abs/1706.09588*.

Tamura, S., & Waibel, A. (1988). Noise Reduction using Connectionist Models. *ICASSP-88., International Conference on Acoustics, Speech, and Signal Processing*.

Tao, A., Sapra, K., & Catanzaro, B. (2020). Hierarchical Multi-Scale Attention for Semantic Segmentation. *arXiv preprint arXiv:2005.10821*.

TechDirt. (2010). Interview With Will Page, Music Industry Economist. Retrieved December 27, 2018, from https://www.techdirt.com/articles/20100429/0116199232.shtml

Thiede, T., Treurniet, W. C., Bitto, R., Schmidmer, C., Sporer, T., Beerends, J. G., & Colomes, C. (2000). PEAQ-The ITU Standard for Objective Measurement of Perceived Audio Quality. *Journal of the Audio Engineering Society*, *48*(1/2), 3–29.

Thusoo, A., Sarma, J. S., Jain, N., Shao, Z., Chakka, P., Zhang, N., ... Murthy, R. (2010). Hive - A Petabyte Scale Data Warehouse using Hadoop. In F. Li, M. M. Moro, S. Ghandeharizadeh, J. R. Haritsa, G. Weikum, M. J. Carey, ... V. J. Tsotras (Eds.), *Proceedings of the 26th International Conference on Data Engineering, ICDE 2010, March 1-6, 2010, Long Beach, California, USA* (pp. 996–1005). IEEE Computer Society.

Uhlich, S., Porcu, M., Giron, F., Enenkl, M., Kemp, T., Takahashi, N., & Mitsufuji, Y. (2017). Improving Music Source Separation Based on Deep Neural Networks through Data Augmentation and Network Blending. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017* (pp. 261–265). IEEE.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., ... Kavukcuoglu, K. (2016). WaveNet: A Generative Model for Raw Audio. *arXiv preprint arXiv:1609.03499*.

Van Hulle, M. (1999). Clustering Approach to Square and Non-square Blind Source Separation. *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No.98TH8468)*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention Is All You Need. In *Advances in Neural Information Processing Systems* (pp. 5998–6008).

Venkataramani, S., Casebeer, J., & Smaragdis, P. (2018). End-to-end Source Separation with Adaptive Front-ends. In *2018 52nd Asilomar Conference on Signals, Systems, and Computers* (pp. 684–688). IEEE.

Vincent, E. (2012). Improved Perceptual Metrics for the Evaluation of Audio Source Separation. In F. J. Theis, A. Cichocki, A. Yeredor, & M. Zibulevsky (Eds.), *Latent Variable Analysis and Signal Separation - 10th International Conference, LVA/ICA 2012, Tel Aviv, Israel, March 12-15, 2012. Proceedings* (Vol. 7191, pp. 430–437). Lecture Notes in Computer Science. Springer.

Vincent, E., Araki, S., & Bofill, P. (2009). The 2008 Signal Separation Evaluation Campaign: A Community-Based Approach to Large-Scale Evaluation. In T. Adali, C. Jutten, J. M. T. Romano, & A. K. Barros (Eds.), *Independent Component Analysis and Signal Separation, 8th International Conference, ICA 2009, Paraty, Brazil, March 15-18, 2009. Proceedings* (Vol. 5441, pp. 734–741). Lecture Notes in Computer Science. Springer.

Vincent, E., Gribonval, R., & Févotte, C. (2006). Performance Measurement in Blind Audio Source Separation. *IEEE Transactions on Audio, Speech, and Language Processing*, *14*(4), 1462–1469.

Vincent, E., & Rodet, X. (2004). Underdetermined Source Separation with Structured Source Priors. In C. G. Puntonet & A. Prieto (Eds.), *Independent Component Analysis and Blind Signal Separation, Fifth International Conference, ICA 2004, Granada, Spain, September 22-24, 2004, Proceedings* (Vol. 3195, pp. 327–334). Lecture Notes in Computer Science. Springer.

Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A. S., Yeo, M., ... Tsing, R. (2017). StarCraft II: A New Challenge for Reinforcement Learning. *arXiv preprint arXiv:1708.04782*.

Virtanen, T., Mesaros, A., & Ryynänen, M. (2008). Combining Pitch-based Inference and Non-negative Spectrogram Factorization in Separating Vocals from Polyphonic Music. In *INTERSPEECH 2008, 9th Annual Conference of the International Speech Communication Association, Brisbane, Australia, September 22-26, 2008*.

Wagemans, J., Elder, J. H., Kubovy, M., Palmer, S. E., Peterson, M. A., Singh, M., & von der Heydt, R. (2012). A Century of Gestalt Psychol-

ogy in Visual Perception: I. Perceptual Grouping and Figure–Ground Organization. *Psychological bulletin*, *138*(6), 1172.

Wang, B., Mary, Q., Plumbley, M. D., & Mary, Q. (2005). Musical Audio Stream Separation by Non-negative Matrix Factorization. In *in Proc. UK Digital Music Research Network (DMRN) Summer Conf.*

Wang, D. (2005). On Ideal Binary Mask as the Computational Goal of Auditory Scene Analysis. In *Speech separation by humans and machines* (pp. 181–197). Springer.

Wang, R., Li, B., Hu, S., Du, W., & Zhang, M. (2020). Knowledge Graph Embedding via Graph Attenuated Attention Networks. *IEEE Access*, *8*, 5212–5224.

Wang, X., Girshick, R. B., Gupta, A., & He, K. (2018). Non-Local Neural Networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018* (pp. 7794–7803). IEEE Computer Society.

Wang, Z.-Q., & Wang, D. (2017). Recurrent Deep Stacking Networks for Supervised Speech Separation. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017* (pp. 71–75). IEEE.

Weninger, F., Hershey, J. R., Roux, J. L., & Schuller, B. W. (2014). Discriminatively Trained Recurrent Neural Networks for Single-channel Speech Separation. In *2014 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2014, Atlanta, GA, USA, December 3-5, 2014* (pp. 577–581). IEEE.

White, T. (2010). *Hadoop: the Definitive Guide* (Second). Sebastopol: O'Reilly Media, Inc.

Williamson, D. S., Wang, Y., & Wang, D. (2016). Complex Ratio Masking for Monaural Speech Separation. *IEEE/ACM Trans. Audio, Speech & Language Processing*, *24*(3), 483–492.

Wisdom, S., Hershey, J., Le Roux, J., & Watanabe, S. (2016). Deep Unfolding for Multichannel Source Separation. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, China, March 20-25, 2016* (pp. 121–125).

Xia, X., & Kulis, B. (2017). W-Net: A Deep Model for Fully Unsupervised Image Segmentation. *CoRR*, *abs/1711.08506*.

Xiao, H., Wei, Y., Liu, Y., Zhang, M., & Feng, J. (2018). Transferable Semi-Supervised Semantic Segmentation. In S. A. McIlraith & K. Q. Weinberger (Eds.), *Proceedings of the Thirty-Second AAAI Conference on*

Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018 (pp. 7420–7427). AAAI Press.

Yu, T., Li, Z., Zhang, Z., Zhang, R., & Radev, D. R. (2018). TypeSQL: Knowledge-Based Type-Aware Neural Text-to-SQL Generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*.

Yun, S., Han, D., Chun, S., Oh, S. J., Yoo, Y., & Choe, J. (2019). CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*.

Zhang, X., Zhang, H., Nie, S., Gao, G., & Liu, W. (2016). A Pairwise Algorithm Using the Deep Stacking Network for Speech Separation and Pitch Estimation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *24*(6), 1066–1078.

Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., & Fu, Y. (2018). Image Super-Resolution Using Very Deep Residual Channel Attention Networks. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*.

Zhao, H., Zhang, Y., Liu, S., Shi, J., Loy, C. C., Lin, D., & Jia, J. (2018). PSANet: Point-wise Spatial Attention Network for Scene Parsing. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *ECCV (9)* (Vol. 11213, pp. 270–286). Lecture Notes in Computer Science. Springer.

Zhao, Y., Xu, B., Giri, R., & Zhang, T. (2018). Perceptually Guided Speech Enhancement using Deep Neural Networks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, Calgary, AB, Canada, April 15-20, 2018* (pp. 5074–5078). IEEE.