Multi-Modal Deep Generative Audio Source Separation

Max Jappert
max.jappert.23@ucl.ac.uk
S23096767

Supervised by Dr Brooks Paige

A thesis submitted for the degree of
Master of Science in Machine Learning

University College London
Faculty of Engineering
Department of Computer Science

September 8, 2024

Acknowledgements

I would like to thank Dr Brooks Paige, the supervisor of this thesis, for the excellent guidance. Thank you also to Johanna Flach and my parents for the support along the way. This would not have been possible without you.

Declaration

I, Max Jappert, declare that this thesis has been composed by myself and that the work has not been submitted for any other degree or professional qualification. I confirm that the work submitted is my own. The report may be freely copied and distributed provided the source is explicitly acknowledged.

Abstract

Isolating sound sources in an auditory environment is a skill humans use daily. Developing computational systems to replicate this ability could drive enhancements in hearing aids, music production and forensic audio. This thesis investigates the application of deep generative models for this purpose, culminating in a modified Bayesian Annealed Signal Source (BASIS) separation approach being introduced. This method leverages variational autoencoders to compute deep generative priors used in noise-annealed Langevin dynamics to sample from the posterior over the sources given a mixed signal, enabling effective separation. A multi-modal method for incorporating visual information into Modified BASIS is also proposed, but fails to improve performance. Experiments using pop and chamber music show promising results, but suggest a sensitivity to in-class variability. This highlights both the potential of deep generative models for audio source separation, but also the challenges future work must address.[1]

[1]The code repository for this thesis can be found at `https://github.com/maxjappert/mmdgass` (accessed September 7, 2024).

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

(C)ASA (Computational) Auditory Scene Analysis

(I)STFT (Inverse) Short-Time Fourier Transform

(V)AE (Variational) Autoencoder

ADC Analog-to-Digital Converter

BASIS Bayesian Annealed Signal Source

BSS Blind Source Separation

CNN Convolutional Neural Network

CPP Cocktail Party Problem

DAC Digital-to-Analog Converter

dB Decibel

Hz Hertz

ISR Image-to-Spatial-Distortion Ratio

MSE Mean Squared Error

MUSDB Music Separation Database

NMF Non-Negative Matrix Factorisation

RNN Recurrent Neural Network

SAR Signal-to-Artefact Ratio

SDR Signal-to-Distortion Ratio

SIR Analog-to-Interference Ratio

URMP University of Rochester Music Performance

Chapter 1

Introduction

1.1 Problem Statement

Audio signals arriving at the ear usually consist of a mixture of sources, such as the sound originating from social environments, music recordings or noisy voicemails. Separating the sources of mixed signals is intuitively easy for humans, as it is a task essential for survival and is performed on a daily basis. A computational system to rival human performance in this domain is yet to be developed. Developing and deploying such a system could have a significant impact on many fields:

- Audio-to-text systems could transcribe only the relevant voice and ignore interference.
- Cochlear implants could similarly separate background noise from relevant signals.
- Individual instruments could be isolated from a mixed recording when producing music. Bleed could be minimised.
- Forensic audio could be enhanced, extracting only the relevant signal.

There have been many attempts to use deep learning for audio source separation. This has been the case especially since the AI boom in the late 2010s, where increasingly deep and powerful models were made possible by revolutionary architectures like AlexNet (Krizhevsky, Sutskever, and G. E. Hinton, 2012), GANs (Goodfellow et al., 2014), ResNet (K. He et al., 2016) and transformers (Vaswani, 2017), technical innovation (Nvidia, CUDA, PyTorch), as well as financial incentive. There is an ongoing research effort to find new algorithms, architectures and training regimes to improve the state-of-the-art in computational audio source

separation. The goal is to develop a system that can reliably and efficiently separate audio sources.

1.2 Objectives

Generative artificial intelligence effectively models complex high dimensional data. Since audio falls into this category, it stands to reason that deep generative models can be leveraged for this task.

This thesis focuses on identifying potential deep generative approaches, comparing them to both traditional methods and state-of-the-art architectures. Based on these evaluations, this thesis will suggest a novel method. This best-performing approach will be analysed and evaluated, with suggestions provided for future work. A secondary objective is to explore the integration of visual information to enhance audio source separation, emphasising a shift from uni-modal to multi-modal processing.

A helpful heuristic for designing AI systems and selecting features is to consider how humans — or other biological systems shaped by evolution — would approach the problem and which features they would intuitively use. As discussed in Sections 2.1.3 and 2.1.4, audio-visual integration plays a crucial role in human auditory scene analysis, to the point where the brain has evolved a separate region to integrate multiple stimuli. This thesis attempts to leverage such insights from human perception in order to inform the development of a novel audio source separation approach.

This thesis will evaluate the described approaches using baselines and experiments described in the following sections. As this is an ambitious task limited to a three-month Master's thesis, the focus lies on a proof of concept for novel methods.

1.3 Thesis Structure

In Section 2 provides a review of existing literature, covering previous work on the topics addressed in this thesis. Section 2.1 provides a brief introduction into audio signal processing, in order to make the thesis self-contained. Section 2.2.3 focuses on the quantitative metrics proposed by Vincent, Gribonval, and Févotte (2006) to assess source separation performance. In Section 2.2, seminal audio source separation techniques are discussed alongside more advanced deep learning approaches. Section 2.3 reviews prior work on multi-modal integration. As the visual information is provided by video data, Section 2.4 outlines various methods for extracting video features.

The algorithms and trained models used in the experiments outlined in Section 4 are described in Section 3. Section 3.1 introduces the datasets and how they are preprocessed. Section 3.2 introduces the models, their architecture, hyperparameters and the training process.

Section 4.1 describes the experiments and 4.2 presents their results, which are discussed in Section 5. The thesis concludes with Section 6, summarising what has been written, offering suggestions for future work and describing the derived implications.

1.4 Notation

A variable typed in **boldface** is either a vector $\boldsymbol{v} \in \mathbb{R}^n$ or a matrix $\boldsymbol{M} \in \mathbb{R}^{m \times n}$. $\boldsymbol{I}$ denotes the identity matrix and i describes the imaginary unit satisfying $i^2 = -1$. A complex number $z \in \mathbb{C}$ is defined as

$$z = a + ib.$$

for $a, b \in \mathbb{R}$. Its magnitude is defined as

$$|z| = \sqrt{a^2 + b^2}.$$

All other lower-case, standard-face variables describe scalars. A variable with a hat $\hat{x}$ describes the approximation or reconstruction of a corresponding ground truth x. Square brackets denote the index or indices of a vector or matrix, e.g. $\boldsymbol{v}[n] = v_n \in \mathbb{R}$ such that

$$\boldsymbol{v} = \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix}$$

and $\boldsymbol{X}[m, n] = x_{mn}$ such that

$$\boldsymbol{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix}.$$

The operator $\oplus$ denotes vector concatenation

$$\boldsymbol{v} \oplus \boldsymbol{w} = \begin{pmatrix} v_1 \\ \vdots \\ v_n \\ w_1 \\ \vdots \\ w_n \end{pmatrix}.$$

Chapter 2

Literature Review

2.1 Background

2.1.1 An Introduction to Sound

Sound consists of oscillating pressure changes in the air. These pressure changes can be detected by a membrane, such as the ear drum or a microphone diaphragm, and can then be converted to electrical signals which can be processed further. We can represent the oscillating pressure changes as waves. Sound, i.e., the oscillating change in air pressure, is generated by vibrations, which can be induced by anything from a string on a violin to a membrane in a loudspeaker or vocal cords. Air particles vibrate around an equilibrium position, at which their velocity is maximised. The velocity is minimised at the position of maximum displacement (Berg, 2024).

Audio can be mathematically represented in two spaces:

- In **real space**, where the x-axis represents time (as consecutive samples in the discrete case) and the y-axis represents amplitude as oscillating displacement from the equilibrium position.

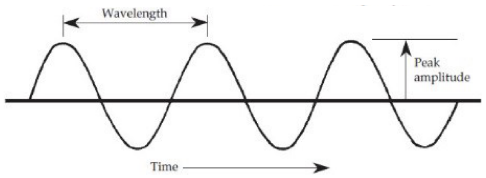


Figure 2.1: Visualisation of a sine-shaped sound wave in real space. Reproduced from Everest (2022).

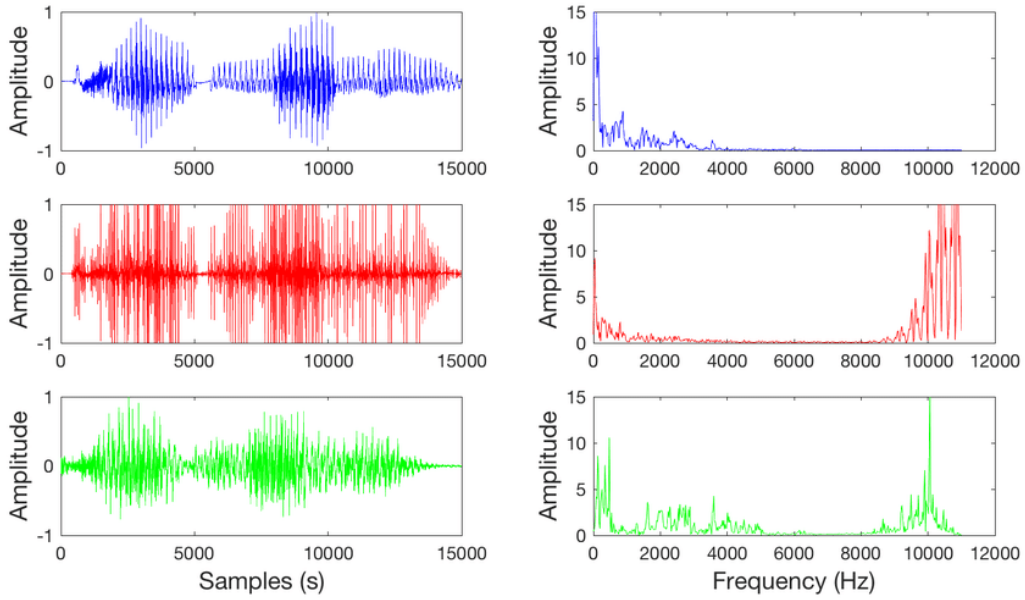


Figure 2.2: A visualisation of three audio signals represented in real space on the left and frequency space on the right. Reproduced from Abdullah et al. (2019).

- In **frequency space**, where the x-axis represents the frequencies in Hertz (Hz) and the y-axis represents amplitude in decibel (dB).

An example of the two spaces is provided in Figure 2.2. Hz measures frequency as the number of oscillations per second. dB measures the relative loudness of a sound on a logarithmic scale based on sound pressure level P as

$$dB(P) = 10 \log_{10} \left(\frac{P}{P_0} \right) \tag{2.1}$$

where $P_0 = 20 \mu Pa$ is the reference sound pressure representing the threshold of human hearing (Berg, 2024).

When recording with a microphone, the signal is captured in real space as oscillating voltage, representing sound as a function of time. To visualise the constituent frequency components of a sound during a given time interval, it can be transformed into frequency space, where it is represented as a function of frequency. This transformation is achieved through a Fourier transform. The Fourier transform is computed as in Equation 2.2 in the continuous case and 2.3 in the discrete case.

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-i2\pi ft} \, dt \tag{2.2}$$

$$X[k] = \sum_{n=0}^{N-1} \boldsymbol{x}[n] e^{-i\frac{2\pi}{N}kn} \tag{2.3}$$

Thereby $f \in \mathbb{R}_{\geq 0}$ are the continuous frequencies and $k \in \mathbb{N}$ are the frequency bin indices in the discrete case. When working with digital audio, which is inherently discrete, the Fast Fourier Transform (FFT) algorithm, introduced in a seminal paper by Cooley and Tukey (1965), is commonly used. FFT significantly reduces the $O(N^2)$ time complexity of computing Equation 2.3 to a more efficient $O(N \log N)$ for a sequence of length N. It achieves this through a divide-and-conquer approach, breaking down the computation into smaller parts while exploiting the symmetry and periodicity of the complex exponential functions in Equation 2.3.

Algorithm 1 Fast Fourier Transform (FFT) (Cooley and Tukey, 1965)

Input: Array of complex numbers a of length n (where n is a power of 2)
Output: Array y of length n representing the discrete Fourier transform of a
$n \leftarrow \text{length}(a)$
if $n = 1$ **then**
    return a
end if
$\omega_n \leftarrow \exp\left(-\frac{2\pi i}{n}\right)$
$\omega \leftarrow 1$
$a_0 \leftarrow (a_0, a_2, \ldots, a_{n-2})$
$a_1 \leftarrow (a_1, a_3, \ldots, a_{n-1})$
$y_0 \leftarrow \text{FFT}(a_0)$
$y_1 \leftarrow \text{FFT}(a_1)$
for $k = 0$ to $n/2 - 1$ **do**
    $y[k] \leftarrow y_0[k] + \omega \cdot y_1[k]$
    $y[k + n/2] \leftarrow y_0[k] - \omega \cdot y_1[k]$
    $\omega \leftarrow \omega \cdot \omega_n$
end for
return y

A sine wave, as shown in Figure 2.1, contains only a single frequency and is referred to as a pure-tone. A real-space representation is given in Equation 2.4

$$x(t) = A \sin(2\pi f t + \phi) \tag{2.4}$$

where $A \in \mathbb{R}_{>0}$ is the amplitude and thus determines the loudness, $f \in \mathbb{R}_{>0}$ is the frequency in Hz, $t \in \mathbb{R}_{\geq 0}$ is the time (or the sample index $t \in \mathbb{N}$ in the

Figure 2.3: Visualisation of how sound waves are converted to a continuous electrical signal. Reproduced from My New Microphone (2019).

discrete case) and $\phi \in \mathbb{R}$ is the phase, i.e., the offset along the x-axis.

Digital Signal Processing for Audio

Sound, as continuous vibrations in the air, must be transformed into a discrete format for digital processing. Microphones relay the sound they record using a diaphragm, a thin, flexible membrane which vibrates when struck by sound waves. These vibrations are then converted into a continuous electrical signal by a transducer, which can be directly amplified. Figure 2.3 provides a visualisation of this process.

Analog-to-digital and digital-to-analog converters (ADC and DAC respectively) convert to and from a digital representation of a signal. ADCs work by taking samples of the continuous signal at a given sampling rate. Nyquist (1928) shows that higher sampling rates improve reconstruction quality up to a limit. The Nyquist sampling theorem defines the Nyquist rate, which is the minimum sampling rate required to perfectly reconstruct a continuous signal from its discrete samples, as shown in Figure 2.4. The Nyquist rate is defined as twice the maximum frequency of the signal. As such, the sampling rate f_s must satisfy Equation 2.5 for perfect reconstruction.

$$f_s \geq 2f_{max} \tag{2.5}$$

It follows that the necessary sampling rate is dependent on the maximum frequency of the signal. Thus, for a signal with frequencies $f \in [a, b]$, the sampling rate $f_s = 2b$ will results in a reconstruction perceptually identical to any higher rate $f_s' > 2b$. The Federal Standard 1037C (Ingram and Gray, 1998), an authoritative telecommunications standard, defines the maximum talking frequency of the human voice at 3400 Hz. Thus, a sampling rate of 8000 Hz suffices for telephone communication. For more complex signals covering the entire range of human hearing, approximately 20 – 20'000 kHz, a higher sampling rate of $f_s \geq 40$ kHz is required. Compact disks (CDs) operate on a sampling rate of 44.1 kHz, which is

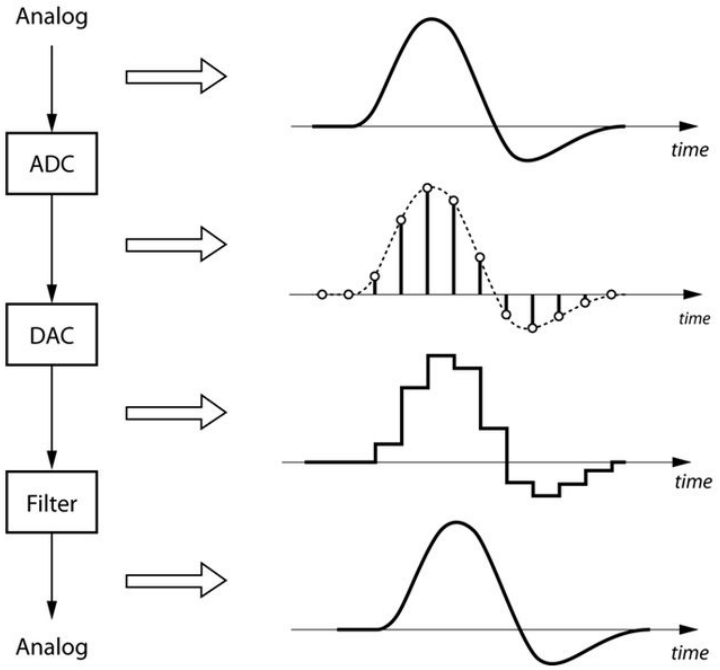


Figure 2.4: Visualisation of an analog signal, i.e., the output in Figure 2.3, being converted to a digital signal using an ADC and back into an analog signal using a DAC. The lines represent the sampling operation. The smoothing filter operation is a low-pass filter with a cutoff frequency $f_c \leq \frac{f_s}{2}$ trivially derived from Equation 2.5. Reproduced from Embedded Robotics (2020).

sufficient to perfectly reconstruct a signal within the human hearing range and additionally decomposes into the product of the squares of the first four prime numbers as

$$44100 = 2^2 \cdot 3^2 \cdot 5^2 \cdot 7^2. \tag{2.6}$$

2.1.2 Spectrograms

Spectrograms $\boldsymbol{X} \in \mathbb{C}^{T \times F}$ are a complex sound representation capturing time, frequency, amplitude and phase information. The real part, called the magnitude spectrogram $|\boldsymbol{X}|$, can be plotted as a 2-dimensional figure where the x-axis is time $t_i \in \mathbb{R}_{\geq 0}$, the y-axis denotes the frequencies $f_j \in \mathbb{R}_{>0}$ and the magnitude of each complex scalar $|\boldsymbol{X}[i, j]|$ is the amplitude of the frequency f_j at time t_i in dB. Thereby $i \in \{1, \ldots, T\}$ and $j \in \{1, \ldots, F\}$ are the indices. Figure 2.5

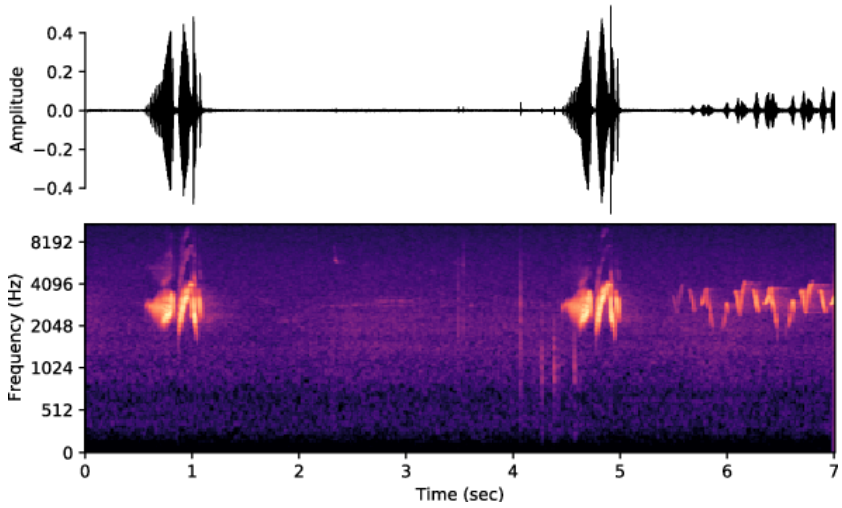


Figure 2.5: A signal $\boldsymbol{x}$ in real space on the top with the corresponding magnitude spectrogram $|\boldsymbol{X}^T|$ on the bottom. Reproduced from Gupta et al. (2021).

provides an example of a magnitude spectrogram $|\boldsymbol{X}^T|$ with its corresponding wave form in real space. The complex part of the spectrogram is used to infer the phase $\boldsymbol{\Phi}$ as its angle.

Spectrograms are useful as they show how the frequency components change over time. Additionally, they are useful for deep learning approaches to audio as they can be fed into convolutional neural networks (CNNs), as described by Hershey et al. (2017) and Palanisamy, Singhania, and Yao (2020).

Converting from Real to Time-Frequency Space

The Fourier transform provides frequency information over a given time interval. A spectrogram, on the other hand, is an analysis of frequency over time. To obtain the spectrogram, one must use a Short-Time Fourier Transform (STFT), which slides a window along the time axis of a real-space signal, capturing its frequency information at different overlapping time intervals.

In the discrete case, this involves computing

$$\boldsymbol{X}[m,k] = \sum_{n=0}^{N-1} \boldsymbol{x}[n] w(n - mL) \exp(-i\frac{2\pi kn}{N}) \tag{2.7}$$

$$= |\boldsymbol{X}[m,k]| \exp(i\boldsymbol{\Phi}[m,k]) \tag{2.8}$$

whereby $\boldsymbol{x}$ is a vector representing the discrete signal in real space, m is the window index, k is the frequency bin index, L is the step size between the starting points of consecutive windows (called the *hop size*), and N is the window

size. $w(n - mL)$ is the window function isolating the required segment for analysis. Note that spectrograms are conventionally plotted with time on the x-axis, whereby $\boldsymbol{X}$ computed with STFT in Equation 2.7 provides the transpose of the common visualisation.

$|\boldsymbol{X}|$ determines the magnitude spectrogram and $\boldsymbol{\Phi}[m, k] = \arg(\boldsymbol{X}[m, k])$ the phase of the frequency bin k of the original signal at time m. The function

$$\arg(z) = \tan^{-1}(\frac{b}{a}) = \theta \tag{2.9}$$

maps a complex number $z = a + ib$ with real component $a \in \mathbb{R}$ and complex component $b \in \mathbb{R}$ to its angle $\theta \in (-\pi, \pi]$. Using the angle, it is possible to represent z in polar form as $z = r(\cos(\theta) + i\sin(\theta))$ with radius r, providing a trigonometric interpretation. Using this, Oppenheim (1999) shows that

$$\arg(\boldsymbol{X}[m, k]) = \boldsymbol{\Phi}[m, k] \tag{2.10}$$

holds, i.e., that the angle of the spectrogram is equal to the phase.

Window Functions

The most trivial window function is the boxcar or Dirichlet window, simply a constant value $c \in \mathbb{R}$ across the window length N.

$$w_{\text{boxcar}}(n) = \begin{cases} c & \text{if } 0 \leq n < N \\ 0 & \text{else} \end{cases} \tag{2.11}$$

Ideally, we want a very small window size to capture the maximal time resolution and a very large number of spectral bins to capture the maximal spectral resolution. The Heisenberg Uncertainty Principle (HUP), interpreted for signal processing, states that

$$\Delta t \cdot \Delta f \geq \frac{1}{4\pi} \tag{2.12}$$

holds for window size Δt and spectral bin size Δf (Cohen, 1995). This implies that their product is lower bounded by a constant $1/4\pi$. Thus, STFT is subject to a time-frequency trade-off.

The boxcar window function from Equation 2.11 has a hard cut-off, which leads to significant spectral leakage and thus lower spectral resolution. Spectral leakage occurs when discontinuities at the edges of a window function cause energy from different frequency components to spread across the spectrum, reducing the accuracy of the frequency representation. To achieve higher spectral resolution, it is common to use window functions with tails on each end, as they

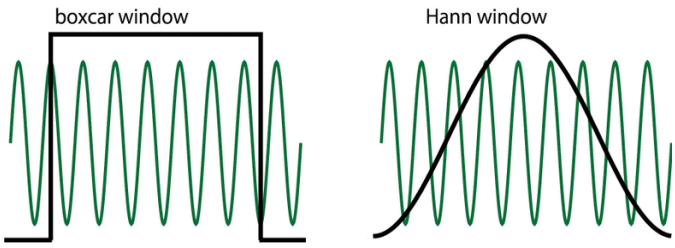


Figure 2.6: Visualisation of the boxcar and Hann window functions in black with the green signal in real space. Reproduced from Sawa, Yamada, and Obata (2022).

reduce leakage, despite leading to a higher Δt. Throughout this thesis, the `scipy` STFT implementation is used in conjunction with the Hann window function (Virtanen et al., 2020). The Hann window function is defined in Equation 2.13. Figure 2.6 visualises the difference between the boxcar and Hann window functions.

$$w_{\text{Hann}}(n) = 0.5 \left(1 - \cos(\frac{2\pi n}{N-1})\right) \tag{2.13}$$

Converting from Time-Frequency to Real Space

If we have access to phase information, we can transform a spectrogram from time-frequency space into real space using the Inverse STFT (ISTFT). In the discrete case, the ISTFT is computed as

$$\boldsymbol{x}[n] = \sum_{m,k} w(n - mL)\boldsymbol{X}[m,k] \exp(i\frac{2\pi kn}{N}) \tag{2.14}$$

with the same variables as in Equation 2.7. Often, we will only have access to the magnitude spectrogram $|\boldsymbol{X}|$ and will thus not have access to any phase information. Plugging Equation 2.8 into Equation 2.14, we get

$$\boldsymbol{x}[n] = \sum_{m,k} |\boldsymbol{X}[m,k]| \exp(\boldsymbol{\Phi}[m,k]) w(n - mL) \exp(i\frac{2\pi kn}{N}). \tag{2.15}$$

It follows that we need the phase $\boldsymbol{\Phi}[m,k]$ to reconstruct the audio signal using ISTFT. If we only have access to the magnitude spectrogram, we can use the Griffin-Lim algorithm (Griffin and Lim, 1984) to iteratively approximate the phase. The idea behind this algorithm is similar to the expectation-maximisation (EM) algorithm (Moon, 1996), whereby in each iteration t a real-space signal $\boldsymbol{y}^{(t)}$ is approximated using a current phase estimate $\boldsymbol{\Phi}^{(t)}$ (equivalent to the E-step),

which is then used to approximate a new phase estimate $\boldsymbol{\Phi}^{(t+1)}$ which better matches the magnitude spectrogram $|\boldsymbol{X}|$ (equivalent to the M-step). This thesis uses the `librosa` (McFee et al., 2015) implementation of Algorithm 2.

Algorithm 2 Griffin-Lim Algorithm (Griffin and Lim, 1984)

Require: Magnitude spectrogram $|\boldsymbol{X}[m,k]|$, number of iterations N_{iters}, window length N, hop size L

1: Initialize $\boldsymbol{\Phi}[m,k]^{(1)}$ randomly
2: $\boldsymbol{Y}[m,k]^{(1)} \leftarrow |\boldsymbol{X}[m,k]|e^{i\boldsymbol{\Phi}[m,k]^{(1)}}$
3: **for** $t = 1$ to N_{iters} **do**
4: $\quad \boldsymbol{y}^{(t)}[n] \leftarrow \text{ISTFT}(\boldsymbol{Y}[m,k]^{(t)})$
5: $\quad \boldsymbol{Y}[m,k]^{(t+1)} \leftarrow \text{STFT}(\boldsymbol{y}^{(t)}[n])$
6: $\quad \boldsymbol{Y}[m,k]^{(t+1)} \leftarrow |\boldsymbol{X}[m,k]| \frac{\boldsymbol{Y}[m,k]^{(t+1)}}{|\boldsymbol{Y}[m,k]^{(t+1)}|}$
7: **end for**
8: $\boldsymbol{y}_{\text{reconstructed}}[n] \leftarrow \text{ISTFT}(\boldsymbol{Y}[m,k]^{(N_{\text{iters}})})$

Ensure: Reconstructed time-domain signal $\boldsymbol{y}_{\text{reconstructed}}[n]$

2.1.3 The Cocktail Party Effect

Auditory environments typically consist of a mixture of sound sources. A common example is a supermarket, where the auditory environment includes background music, the clatter of shopping carts, the beeping of tills, and people talking, among other sounds. Mathematically, we can define this mixed audio signal $\boldsymbol{m} \in \mathbb{R}^n$ as a mixture of sources $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \in \mathbb{R}^n$ with a mixing function $f(\cdot)$ such that

$$\boldsymbol{m} = f(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k) \tag{2.16}$$

Often, we can work under the assumption that the mixed signal $\boldsymbol{m}$ is a linear combination of source signals with added Gaussian noise, such that

$$f(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k) \approx \sum_{i=1}^{n} \alpha_i \boldsymbol{s}_i + \boldsymbol{\epsilon} \tag{2.17}$$

where $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ are scalar mixing coefficients and $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$ is additive noise.

Humans are remarkably good at extracting individual sound sources from mixed signals, a skill that offers clear evolutionary advantages for both survival and social interaction (McDermott, 2009). This ability is known as the Cocktail Party Problem (CPP), a term coined by British cognitive scientist Colin Cherry

(1953). The term refers to cocktail parties, where multiple conversations occur simultaneously, requiring individuals to focus on a single voice from a mixture of voices.

Auditory Scene Analysis

The task of organising sounds into perceptually meaningful elements is known as Auditory Scene Analysis (ASA). Audio sources arrive at the brain after being converted into electrical signals in the cochlea, a coiled structure receiving vibrations from the ear drum via the ossicles. The brain then processes these electrical signals into perceived discrete audio streams, sub-consciously approximating a disentanglement (Bregman, 1994). The field of psychoacoustics studies ASA and describes different cues the brain uses to perform this task.

Monaural and Binaural Cues

The brain uses monaural and binaural cues for sound localisation. Binaural cues, such as the interaural time and level differences, are used for horizontal localisation. Monaural cues, such as the Head-Related Transfer Function, an idiosyncratic spectral filter given by the body shaping the signal before it arrives at the eardrums, are useful for vertical localisation (Musicant and Butler, 1985). Sound source localisation helps the brain group stimuli into discrete streams, as described by the Gestalt principles.

The Gestalt Principles

The brain's primary organisational ability is described by the Gestalt principles (Köhler, 1967). These principles are a set of rules that explain how humans perceive stimuli as organised patterns, such as discrete audio streams. They include, but are not limited to:

- **Proximity:** Objects that are localised as being physically close to each other tend to be perceived as a group.
- **Similarity:** Objects with a similar appearance are perceived as a group.
- **Closure:** The mind automatically fills gaps when a stimulus is interrupted.

It is plausible that the Gestalt principles could be learned by a universal function approximator, such as a deep neural network, to perform Computational Auditory Scene Analysis (CASA).

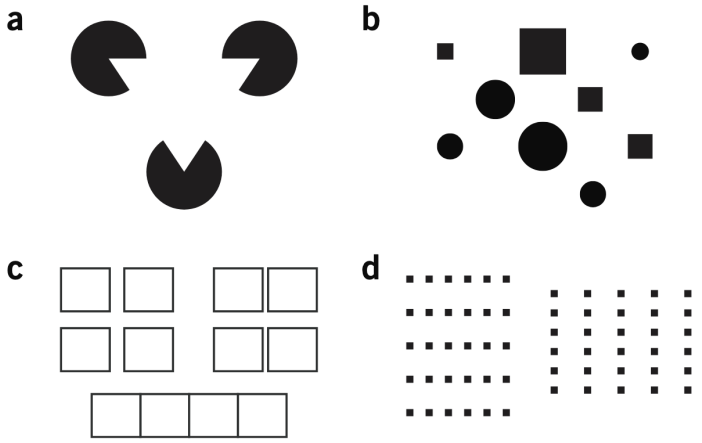


Figure 2.7: Visualisation of the Gestalt principles, which work analogously in the auditory domain. The *closure* principle can be observed in subfigures (a) and (d), *similarity* in (b) and *proximity* in (c) and (d). Reproduced from Wong (2010).

2.1.4 Audio-Visual Perception

Humans are able to separate audio sources more easily when visual information provides prior knowledge about the source decomposition. Neuroscientists, such as Sams et al. (1991), have shown that the presence of multiple sensory inputs neurologically influences audio perception. McGurk and MacDonald (1976) coined the *McGurk effect*, describing a phenomenon which occurs when the auditory component of one speech sound is paired with visual component of another. In such cases, the visual component can change what is understood given identical audio clips.[1] Rouger et al. (2007) argue that the presence of visual information significantly enhances speech perception in patients with hearing impairments. The human ability to integrate multiple modes and combine this information into more than the sum of its parts also plays an important role in designing immersive and satisfying user interfaces (Slater and Wilbur, 1997).

Van Wassenhove, Grant, and Poeppel (2007) argue that temporal synchrony, the simultaneous occurrence of auditory and visual signals, increases the likelihood of these signals being perceived as originating from the same source. The integration of multi-modal data is believed to occur in the *superior temporal sulcus*, a brain region specifically responsible for processing multisensory stimuli (Beauchamp, Nath, and Pasalar, 2010).

In sum, research from many different fields concludes that visual information is an important factor when making sense of our auditory environment.

[1] A compelling example of the McGurk effect can be found here: `https://www.youtube.com/watch?v=2k8fHR9jKVM` (accessed August 29, 2024).

2.2 Computational Audio Source Separation

ASA describes how the human auditory system separates sound sources and CASA attempts to mimic this. This definition differs from the task of audio source separation, which does not necessarily attempt to imitate the human biological process, but rather uses any mathematical approach to estimate a maximally accurate separation of the sources comprising a mixed signal with respect to some metric.

2.2.1 Independent Component Analysis

Independent Component Analysis (ICA) is an early solution to the CPP. Introduced by Jutten and Herault (1991) and expanded on by Comon (1994), it is a blind source separation (BSS) approach, separating the signals without prior knowledge on the decomposition. ICA relies on statistical assumptions to separate the signal into individual components. These assumptions are

- **Statistical Independence:** The source signals are independent of each other.
- **Non-Gaussianity:** At most one of the source signals can be normally distributed.
- **Linearity:** The observed mixed signals are linear combinations of the same set of source signals, such as when recording one auditory scene with an array of m microphones.

The ICA model for k source signals, n samples per signal and m observed signals can be expressed as $\boldsymbol{X} = \boldsymbol{A}\boldsymbol{S}$, where $\boldsymbol{X} \in \mathbb{R}^{m \times n}$ is a matrix whose rows are the observed mixed signals, $\boldsymbol{A} \in \mathbb{R}^{m \times k}$ is a mixing matrix and $\boldsymbol{S} \in \mathbb{R}^{k \times n}$ is a matrix whose rows are the approximated true sources which comprise the mixed signals in $\boldsymbol{X}$. ICA thus operates under the assumption that all m signals in $\boldsymbol{X}$ are comprised of the same k sources, i.e., each observed mixed signal is a linear combination of the same components.

The objective of ICA is to find an unmixing matrix $\boldsymbol{W} \in \mathbb{R}^{m \times m}$ such that $\boldsymbol{W} \approx \boldsymbol{A}^{-1}$. Once $\boldsymbol{W}$ is computed, we can approximate the sources as $\boldsymbol{S} \approx \boldsymbol{W}\boldsymbol{X}$.

Methods for Solving ICA

$\boldsymbol{W}$ can be found using infomax ICA, which was proposed by Bell and Sejnowski (1995) and is based on the linear infomax principle first suggested by Linsker (1988). Infomax ICA involves training a neural network to maximise the mutual

information $I(X_{\text{in}}; X_{\text{out}})$ between the input layer X_{in} and the output layer X_{out}. By the definition of mutual information as

$$I(X_{\text{in}}; X_{\text{out}}) = H(X_{\text{out}}) - H(X_{\text{out}} \mid X_{\text{in}}) \tag{2.18}$$

where $H(\cdot)$ denotes the entropy, we can maximise the mutual information $I(X_{\text{in}}; X_{\text{out}})$ by maximising the entropy of the output $H(X_{\text{out}})$, as $H(X_{\text{out}} \mid X_{\text{in}})$ is assumed to be constant.

Infomax ICA achieves this with a single-layer neural network maximising the independence of the signals and relying on the principle that the output signals are maximally informative if they are maximally independent. The network is trained using gradient ascent.

The FastICA algorithm, first proposed by Hyvärinen and Oja (1997), provides a faster approach. FastICA maximises the non-Gaussianity of the estimated sources by maximising kurtosis and negentropy using a fixed-point iteration scheme.

2.2.2 Non-Negative Matrix Factorisation

Non-Negative Matrix Factorisation (NMF) is another approach to solving the CPP. It was introduced by D. D. Lee and Seung (1999) and conceived for disentangling image mixtures. NMF approximates the decomposition of a non-negative matrix while enforcing a non-negativity constraint on the decomposition. This is beneficial for mixed signals with an additive nature, such as images and sounds. Smaragdis (2004) was the first to apply NMF to the domain of audio source separation.

NMF aims to approximate a given non-negative matrix $\boldsymbol{X} \in \mathbb{R}_{\geq 0}^{m \times n}$ as the product of two non-negative matrices $\boldsymbol{W} \in \mathbb{R}_{\geq 0}^{m \times r}$ and $\boldsymbol{H} \in \mathbb{R}_{\geq 0}^{r \times n}$ as $\boldsymbol{X} \approx \boldsymbol{W}\boldsymbol{H}$. Thereby $r \ll m$ and $r \ll n$ should hold, such that $\boldsymbol{W}$ and $\boldsymbol{H}$ have significantly lower rank than $\boldsymbol{X}$. r thereby denotes the number of spectral patterns and thus sources $\boldsymbol{X}$ is decomposed into.

Algorithm 3 shows the procedure for solving the NMF according to D. Lee and Seung (2000) with a general loss function introduced by Cichocki, Cruces, and Amari (2011), whereby the $\circ$ operator denotes element-wise multiplication (the Hademard product) and powers of matrices are element-wise. Initial estimates $\boldsymbol{W}_{\text{init}} \in \mathbb{R}_{\geq 0}^{m \times r}$ and $\boldsymbol{H}_{\text{init}} \in \mathbb{R}_{\geq 0}^{r \times n}$ can be initialised as random non-negative matrices. Boutsidis and Gallopoulos (2008) argue that faster convergence is possible by initialising these matrices using a non-negative double singular value decomposition (NNDSVD).

Algorithm 3 involves using multiplicative updates in the loop, as suggested in the original paper. Zou, Hastie, and Tibshirani (2006) later proposed using co-

ordinate descent as an alternative optimisation technique, leading to faster convergence. The update rule is governed by the β-loss, which is originally proposed to be the Frobenius norm

$$D_{\beta=2}(\boldsymbol{X}\|\boldsymbol{W}\boldsymbol{H}) = \|\boldsymbol{X} - \boldsymbol{W}\boldsymbol{H}\|_F^2 = \sum_{i,j} \left(\boldsymbol{X}[i,j] - (\boldsymbol{W}\boldsymbol{H})[i,j]\right)^2 . \tag{2.19}$$

Alternatives include the KL divergence (Kullback and Leibler, 1951)

$$D_{\beta=1}(\boldsymbol{X}\|\boldsymbol{W}\boldsymbol{H}) = \sum_{i,j} \left(\boldsymbol{X}[i,j] \log \frac{\boldsymbol{X}[i,j]}{(\boldsymbol{W}\boldsymbol{H})[i,j]} - \boldsymbol{X}[i,j] + (\boldsymbol{W}\boldsymbol{H})[i,j] \right) \tag{2.20}$$

as used by Yang et al. (2011) and the Itakura-Saito divergence (Itakura, 1968)

$$D_{\beta=0}(\boldsymbol{X}\|\boldsymbol{W}\boldsymbol{H}) = \sum_{i,j} \left(\frac{\boldsymbol{X}[i,j]}{(\boldsymbol{W}\boldsymbol{H})[i,j]} - \log \frac{\boldsymbol{X}[i,j]}{(\boldsymbol{W}\boldsymbol{H})[i,j]} - 1 \right) \tag{2.21}$$

as used by Févotte, Bertin, and Durrieu (2009). They argue that the Itakura-Saito divergence correctly captures the semantics of audio and is thus best suited for this use-case.

Algorithm 3 Non-Negative Matrix Factorization (NMF) using multiplicative updates. Original algorithm adapted from D. Lee and Seung (2000) with general β-loss adapted from Cichocki, Cruces, and Amari (2011).

1: **Input:** Non-negative matrix $\boldsymbol{X} \in \mathbb{R}_{\geq 0}^{m \times n}$, number of sources $r \in \mathbb{N}$, initial estimates $\boldsymbol{W}_{\text{init}} \in \mathbb{R}_{\geq 0}^{m \times r}$ and $\boldsymbol{H}_{\text{init}} \in \mathbb{R}_{\geq 0}^{r \times n}$, $\beta \in \{0, 1, 2\}$
2: **Output:** Non-negative matrices $\boldsymbol{W} \in \mathbb{R}_{\geq 0}^{m \times r}$ and $\boldsymbol{H} \in \mathbb{R}_{\geq 0}^{r \times n}$
3: $\boldsymbol{H} \leftarrow \boldsymbol{H}_{\text{init}}$
4: $\boldsymbol{W} \leftarrow \boldsymbol{W}_{\text{init}}$
5: **repeat**
6: $\quad \boldsymbol{H} \leftarrow \boldsymbol{H} \circ \left(\frac{\boldsymbol{W}^T \left(\boldsymbol{X} \circ (\boldsymbol{W}\boldsymbol{H})^{\beta-2} \right)}{\boldsymbol{W}^T \boldsymbol{W}\boldsymbol{H}} \right)$
7: $\quad \boldsymbol{W} \leftarrow \boldsymbol{W} \circ \left(\frac{\left(\boldsymbol{X} \circ (\boldsymbol{W}\boldsymbol{H})^{\beta-2} \right) \boldsymbol{H}^T}{\boldsymbol{W}\boldsymbol{H}\boldsymbol{H}^T} \right)$
8: **until** convergence

NMF can be used for separating audio sources by first converting the real space signal into time-frequency space to obtain spectrogram $\boldsymbol{X}$ using STFT, then performing the separation on the magnitude spectrogram $|\boldsymbol{X}|$. NMF is used

to obtain $\boldsymbol{W}$ and $\boldsymbol{H}$ and the original mixed signal is approximated as $|\boldsymbol{X}| \approx \boldsymbol{W}\boldsymbol{H}$. Thereby $\boldsymbol{W}$ is the basis matrix with column vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_r \in \mathbb{R}^m_{\geq 0}$ representing spectral bases, such as percussive patterns, chords or other distinct sounds that comprise the original signal $|\boldsymbol{X}|$. $\boldsymbol{H}$ is the coefficient matrix with row vectors $\boldsymbol{h}_1^T, \ldots, \boldsymbol{h}_r^T \in \mathbb{R}^n_{\geq 0}$ acting as weights to $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_r$ (D. D. Lee and Seung, 1999). Thus, the sources $\hat{\boldsymbol{S}}_1, \ldots, \hat{\boldsymbol{S}}_r \in \mathbb{R}^{m \times n}_{\geq 0}$ are approximated as the outer product

$$\hat{\boldsymbol{S}}_i \approx \boldsymbol{w}_i \boldsymbol{h}_i^T. \tag{2.22}$$

This provides a blind source separation under the assumption that $|\boldsymbol{X}|$ is a linear combination of r spectral bases.

2.2.3 Evaluation

Signal Decomposition

The most commonly used metrics for measuring audio source separation are described by Vincent, Gribonval, and Févotte (2006). They show that a reconstructed source signal $\hat{\boldsymbol{s}}$ in real space can be decomposed into

$$\hat{\boldsymbol{s}} = \boldsymbol{s}_{\text{target}} + \boldsymbol{e}_{\text{interf}} + \boldsymbol{e}_{\text{noise}} + \boldsymbol{e}_{\text{artif}} \tag{2.23}$$

where $\boldsymbol{s}_{\text{target}}$ is best possible separated source, $\boldsymbol{e}_{\text{interf}}$ are interferences from other signals, and $\boldsymbol{e}_{\text{noise}}$, $\boldsymbol{e}_{\text{artif}}$ are noise and artefacts respectively, which are introduced during the separation process. This implies that the reconstructed source signal $\hat{\boldsymbol{s}}$ consists entirely of the sum of the target, the interference from other signals, random noise and artefacts. This allows for computing metrics measuring specific aspects of the reconstruction. The terms in Equation 2.23 can be computed using projection operators on $\hat{\boldsymbol{s}}$. Note that $\boldsymbol{s} \neq \boldsymbol{s}_{\text{target}}$, as $\boldsymbol{s}_{\text{target}}$ includes a permitted minimal degree of distortion.

Metrics

The Signal-to-Distortion Ratio (SDR) measures the difference between the distortion present in the approximated source $\hat{\boldsymbol{s}}$. It is computed as

$$SDR := 10 \log_{10} \frac{\|\boldsymbol{s}_{\text{target}}\|^2}{\|\boldsymbol{e}_{\text{interf}} + \boldsymbol{e}_{\text{noise}} + \boldsymbol{e}_{\text{artif}}\|^2} \tag{2.24}$$

where the $\|\boldsymbol{s}\|^2$ denotes the squared L2-norm of the signal in real space and is called the *power* of a signal. The SDR thus provides a solid and widely-used general measure of separation ability, taking into account all three error sources.

The other three note-worthy ratios only take exactly one of the error sources into account. They are the Signal-to-Interference Ratio (SIR)

$$SIR := 10 \log_{10} \frac{\|\boldsymbol{s}_{\text{target}}\|^2}{\|\boldsymbol{e}_{\text{interf}}\|^2} \tag{2.25}$$

the Signal-to-Artefact Ratio (SAR)

$$SAR := 10 \log_{10} \frac{\|\boldsymbol{s}_{\text{target}} + \boldsymbol{e}_{\text{interf}} + \boldsymbol{e}_{\text{noise}}\|^2}{\|\boldsymbol{e}_{\text{artif}}\|^2} \tag{2.26}$$

and finally the Signal-to-Noise Ratio (SNR)

$$SNR := 10 \log_{10} \frac{\|\boldsymbol{s}_{\text{target}} + \boldsymbol{e}_{\text{interf}}\|^2}{\|\boldsymbol{e}_{\text{noise}}\|^2}. \tag{2.27}$$

The metrics are measured in dB and defined on $[-\infty, L]$ for upper limit $L \in \mathbb{R}_{>0}$ imposed by the physical properties of sound signals. Higher values indicate better performance and a metric at 0 dB implies that the signal power equals the power of the measured interference. The code for this thesis uses the `mir_eval` library to compute the SDR, SIR and SAR metrics (Raffel et al., 2014). This library additionally computes the Image-to-Spatial-Distortion Ratio (ISR), which measures the preservation of the target source.

2.2.4 Deep Learning Approaches

There have been numerous attempts since since the early 2010s to use deep learning for audio source separation. Huang et al. (2014) use a recurrent neural network (RNN) to predict a soft separation mask, achieving a mean SDR of 2.3 dB on a singing voice dataset and thus outperforming both NMF and ICA. Uhlich, Giron, and Mitsufuji (2015) achieve a slightly higher SDR using a fully connected neural network to extract individual instruments from an orchestral recording, training on a specific instrument only and thus incorporating prior information. Nugraha, Liutkus, and Vincent (2016) use a pipeline consisting of deep neural networks and the EM algorithm on multi-channel audio files, where each channel corresponds to a recording from one microphone. This approach leverages spatial information, much as humans do using the Gestalt principles. They report an SDR of 7.72 dB using the EM approach with NMF and 13.25 dB with the EM-DNN combination on the six-channel CHiME-3 voice dataset (Barker et al., 2015). Similar results are reported by Durrieu et al. (2009), who utilise a NMF-based stochastic model which is able to exploit the spatial nature of their stereophonic dataset. Chandna et al. (2017) use CNNs on monaural spectrograms to construct a low-latency model for extracting instruments. The proposed model requires

less than 200 ms to extract different stems with a mean SDR of 2 dB. They note significant variations in separation performance depending on the instrument.

The AI boom triggered rapid advancements in the field with the sudden feasibility of very deep neural networks in the late 2010s. This coincided with the introduction of the MUSDB18 dataset by Rafii et al. (2017), becoming the benchmark for many uni-modal source separation architectures.

A hybrid approach combining real and time-frequency representations is presented by Défossez (2021). They employ two symmetric convolutional U-Net architectures (Ronneberger, Fischer, and Brox, 2015), one for the spectrogram and one for for the real-space representation. The bottlenecks and the output of both models are concatenated to obtain the final separation. Kim et al. (2021) propose KUIELab-MDX-Net, which seeks to balance performance and computational cost by training one small model per source rather than a single large model. They report a significant decrease in computational demands with only a negligible decrease in SDR when compared to other, more computationally demanding models. CWS-PResUNet (H. Liu, Kong, and J. Liu, 2021) utilises a channel-wise subband (CWS) approach to reduce the required computational resources, enabling deeper networks. Kong et al. (2021) point out that all state-of-the-art approaches to music source separation operate solely on magnitude spectrograms. This necessitates phase approximation using Algorithm 2 during reconstruction, which can degrade performance. They propose ResUNetDecouple+, predicting complex segmentation masks using a very deep residual U-Net architecture.

The most successful model in terms of SDR on the MUSDB18 dataset is the BSRNN architecture proposed by Luo and J. Yu (2023). BSRNN splits a signal into multiple frequency bands, thus exploiting intrinsic musical characteristics. The performances of the reported models are discussed and compared in Section 5.2.

2.2.5 Generative Models for Audio

Introduced by Van Den Oord et al. (2016) at DeepMind, WaveNet is the seminal architecture for audio generation. It is designed for text-to-speech applications and significantly outperformed the state-of-the-art models at the time of publication. It takes raw audio as input, employing convolutional layers and finally a softmax layer to output a categorical distribution over the next audio sample value. Temporal order is enforced by conditioning sample $\hat{\boldsymbol{x}}[n]$ only on past samples $\hat{\boldsymbol{x}}[0 \leq n' < n]$. WaveNet uses dilated convolutions to increase the receptive field without substantially increasing the number of parameters. Dilation involves skipping samples in both directions with a step size determined by a dilation parameter $d(l) \in \mathbb{N}$ for a given layer l.

Inspired by WaveNet operating in real space, Stoller, Ewert, and Dixon (2018) propose the Wave-U-Net, a one-dimensional adaptation of the U-Net (Ronneberger,

Fischer, and Brox, 2015). The U-Net is a CNN originally proposed for pixel-wise medical image segmentation, but has been shown to perform exceptionally well in many domains (Williams et al., 2024). It derives its name from its symmetric encoder-bottleneck-decoder structure featuring copy-and-crop skip connections from each encoder block to the corresponding decoder block. Each block consists of convolutional and max-pooling layers. By encoding and then decoding an input signal, the U-Net captures context and features at multiple scales, allowing for powerful representations. The Wave-U-Net is specifically taylored to audio source separation, taking an audio mixture in real space as input and outputting all k sources in real space directly.

Non-Audio Signal Separation

Webster and J. Lee (2023) propose a self-supervised approach to BSS using an autoencoder with multiple encoders, a concatenated latent space and a shared decoder. The model is trained using a linear combination of a reconstruction loss and additional regularisation losses, defined as follows:

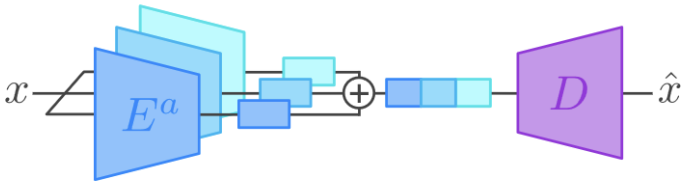
$$\ell_{\text{total}} = \ell_{\text{recon.}} + \lambda_1 \ell_{\text{mixing}} + \lambda_2 \ell_{\text{zero recon.}} + \lambda_3 \ell_z \tag{2.28}$$

where $\ell_{\text{recon.}}$ is a reconstruction loss (such as cross-entropy or mean squared error) and ℓ_{mixing} is the separation or sparse mixing loss that encourages the mixing weights in the decoder to be sparse, ensuring that each encoder specialises on a distinct feature subspace. The term $\ell_{\text{zero recon.}}$ encourages a zero-input to be decoded as a zero-output and ℓ_z is a simple L2 regularisation that penalises large encoder outputs. The parameters λ_i are the respective coefficients used to weight the influence of each term.

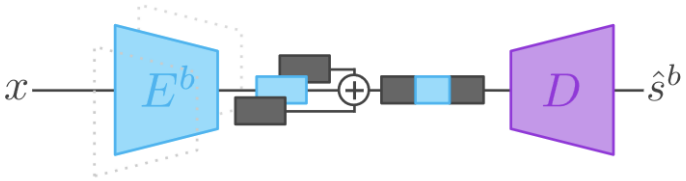
The sparse mixing loss is given by

$$\ell_{\text{mixing}} = \sum_{i \neq j} \alpha_{i,j} \|\boldsymbol{\mathcal{W}}_{i,j}\|_1 \tag{2.29}$$

where $\boldsymbol{\mathcal{W}}_{i,j}$ is the block of the weight matrix $\boldsymbol{W}$ that corresponds to the interaction between the latent spaces $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$ of different encoders (i.e., $i \neq j$). The term $\| \cdot \|_1$ denotes the L1 norm and $\alpha_{i,j}$ are scaling factors normalising the contribution of each block $\boldsymbol{\mathcal{W}}_{i,j}$ based on its size. This loss function encourages the decoder to maintain non-zero weights primarily in the sections of the weight matrix $\boldsymbol{W}$ that do not mix the latent spaces from different encoders. The authors report promising results extracting respiratory signals from electrocardiogram (ECG) and photoplethysmography (PPG) signals, as well as separating overlapping shapes in a toy dataset. The toy dataset and the separation performance is visualised in Figure 2.9.



(a) Blind source separation training procedure.



(b) Blind source separation inference procedure.

Figure 2.8: Visualisation of the self-supervised approach suggested by Webster and J. Lee (2023). Training occurs by concatenating the latent spaces of each encoder, while inference works by masking out all but one encoder. Reproduced from Webster and J. Lee (2023).

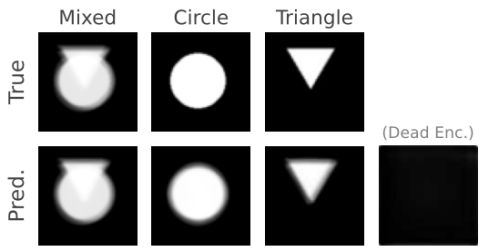


Figure 2.9: Toy problem of separating a circle and a triangle provided by Webster and J. Lee (2023). The dead encoder is automatically recognised as being superfluous and outputs a fully black signal. Reproduced from Webster and J. Lee (2023).

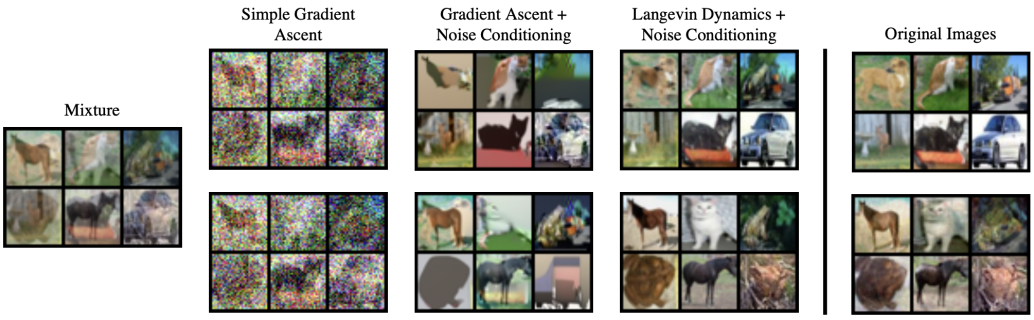


Figure 2.10: BASIS mixture separation performance on CIFAR-10 (Krizhevsky and G. Hinton, 2009). BASIS separation is denoted as *Langevin Dynamics + Noise Conditioning*. Reproduced from Jayaram and Thickstun (2020).

Jayaram and Thickstun (2020) propose the Bayesian Annealed Signal Source separation (BASIS) algorithm for separating mixed image sources. Rather than directly learning the separation function $f^{-1}(\cdot)$ from Equation 2.16, the authors use noise-annealed Langevin dynamics with deep generative priors to sample from the distribution of sources given the observed mixture $p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \mid \boldsymbol{m})$. Here, $\boldsymbol{m}$ is the mixed signal and $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k$ are the k sources.

Computing the posterior over sources directly is challenges because, according to Bayes' rule, it requires computing the partition function $p(\boldsymbol{m})$. Instead, the authors use noise-annealed Langevin dynamics to sample from the posterior $p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \mid \boldsymbol{m})$. Langevin dynamics (Welling and Teh, 2011) is a technique that allows for sampling from a posterior distribution

$$p(\boldsymbol{\theta} \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) \propto p(\boldsymbol{\theta}) \prod_{i=1}^{n} p(\boldsymbol{x}_i \mid \boldsymbol{\theta}). \tag{2.30}$$

over model parameters $\boldsymbol{\theta}$ given data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$. We can do this by constructing a Markov chain using

$$\Delta\boldsymbol{\theta}^{(t)} = \frac{\epsilon_t}{2} \nabla \log p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) + \sqrt{\epsilon_t}\eta_t \tag{2.31}$$

$$= \frac{\epsilon_t}{2} \left(\nabla \log p(\boldsymbol{\theta}^{(t)}) + \sum_{i=1}^{n} \nabla \log p(\boldsymbol{x}_i \mid \boldsymbol{\theta}^{(t)}) \right) + \sqrt{\epsilon_t}\eta_t \tag{2.32}$$

where $\eta_t \sim \mathcal{N}(0, \epsilon_t)$ is Gaussian noise with exploration hyperparameter ϵ_t. This introduces the stochasticity necessary for escaping local optima. The update rule is given by

$$\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} + \Delta\boldsymbol{\theta}^{(t)}. \tag{2.33}$$

By assuming a Gaussian likelihood function

$$p_\gamma(\boldsymbol{m} \mid \boldsymbol{s}_1, \ldots, \boldsymbol{s}_k) = \mathcal{N}(\boldsymbol{m} \mid f(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k), \gamma^2 \boldsymbol{I}) \tag{2.34}$$

with $f(\cdot)$ as in Equation 2.17, it is possible to use this approach to generate samples from $p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \mid \boldsymbol{m})$. To do this, one must plug Equation 2.31 into Equation 2.33 and plug in the values from the source separation, where $\mathcal{S} = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k\} \equiv \boldsymbol{\theta}$ denotes the parameters and $\boldsymbol{m}$ the observed data, to construct a Markov chain with the update rule

$$\begin{aligned}
\mathcal{S}^{(t+1)} &\equiv \mathcal{S}^{(t)} + \frac{\epsilon_t}{2} \nabla_{\mathcal{S}} \log p_\gamma(\mathcal{S}^{(t)} \mid \boldsymbol{m}) + \sqrt{\epsilon_t}\eta_t && (2.35)\\
&= \mathcal{S}^{(t)} + \frac{\epsilon_t}{2} \nabla_{\mathcal{S}} \left(\log p(\mathcal{S}^{(t)}) + \log p_\gamma(\boldsymbol{m} \mid \mathcal{S}^{(t)}) \right) + \sqrt{\epsilon_t}\eta_t && (2.36)\\
&= \mathcal{S}^{(t)} + \frac{\epsilon_t}{2} \nabla_{\mathcal{S}} \left(\log p(\mathcal{S}^{(t)}) + \frac{1}{2\gamma^2} \|\boldsymbol{m} - f(\mathcal{S}^{(t)})\|^2 \right) + \sqrt{\epsilon_t}\eta_t. && (2.37)
\end{aligned}$$

The prior $p(\mathcal{S})$ can be modelled as a deep generative neural network, such as a variational autoencoder (VAE) (Diederik P Kingma and Welling, 2013) or Glow (Durk P Kingma and Dhariwal, 2018). The gradient of the prior $\nabla_{\mathcal{S}} p(\mathcal{S})$ is thereby easily computed using `autodiff` (Paszke et al., 2017).

Expressing the source separation problem within a Bayesian framework separates the tasks of source separation and source generation, allowing for the use of existing generative models without modifying their architecture. Jayaram and Thickstun (2020) achieved state-of-the-art performance on the MNIST dataset (LeCun et al., 1998) and strong performance on CIFAR-10 (Krizhevsky and G. Hinton, 2009) and LSUN (F. Yu et al., 2015). A visual example of the performance on the CIFAR-10 dataset can be found in Figure 2.10.

Directly influenced by this work, Frank and Ilse (2020) attempt a similar approach for audio source separation. They also use Langevin dynamics with deep generative priors to sample from the source posteriors in the time domain directly using a WaveNet. The authors conclude that this approach performs well in a toy scenario, but struggles with real music data due to high in-class variability and complexity. Additionally, the deep generative priors struggle with out-of-class data, making real-world deployment infeasible.

2.3 Audio-Visual Integration

2.3.1 Multi-modal Feature Extraction

Ngiam et al. (2011) argue that uni-modal feature extraction using deep neural networks can be improved by introducing an adjacent modality during training.

They also argue that shared representations across different modalities can be learned, which can be evaluated by training a classifier on one modality (e.g., audio) and evaluating it on a different modality (e.g., video).

To address the question of how deep learning models should integrate multiple modalities, Shi, Paige, Torr, et al. (2019) draw insights from the way humans integrate information from multiple sensory perceptions, similar to the heuristic discussed in Section 1.2. They argue that a successful multi-modal generative model should possess the following four abilities:

- **Latent Factorisation:** The model should implicitly factor the latent space into a subspace for entirely private features and an entirely separate subspace for shared features.
- **Coherent Joint Generation:** When sampling from the generative model across modalities, the samples should be semantically consistent amongst each other.
- **Coherent Cross-Generation:** The model should be able to condition samples of one modality on another modality, thereby preserving the underlying commonality between them.
- **Synergy:** The model should facilitate learning for individual modalities through the integration of multi-modal observations.

They propose a multi-modal VAE (MMVAE), which uses a mixture-of-experts (MoE) approach to combine the latent representations from different modalities. Each expert is a VAE trained on a different modality and the expert for modality m learns a posterior $q_{\phi_m}(\boldsymbol{z} \mid \boldsymbol{x}_m)$. The MoE approach then involves computing the posterior over all modalities as

$$q_\Phi(\boldsymbol{z} \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_M) = \sum_{m=1}^{M} \alpha_m q_{\phi_m}(\boldsymbol{z} \mid \boldsymbol{x}_m) \tag{2.38}$$

with mixing weights α_m. This approach allows for sampling from the posterior $q_\Phi(\boldsymbol{z} \mid \boldsymbol{x}_1, \ldots, \boldsymbol{x}_M)$ without requiring access to all modalities. Consequently, it enables the generation of samples in one modality that are consistent with the parameters of the other modalities.

2.3.2 Audio-Visual Source Separation

Having discussed source separation and multi-modal integration, this section explores the literature at the intersection between both fields. Zhao, Gan, Rouditchenko, et al. (2018) introduce PixelPlayer, a system using self-supervised

learning to perform pixel-wise segmentation of each frame based on each pixel's likelihood of generating sound. They combine this approach with a U-Net to obtain an average SDR of 6.05 dB on the MUSIC dataset introduced by them.

Gao and Grauman (2019) propose a method for learning object-level sounds from unlabeled performance videos, leveraging video data to disentangle sounds from a mixture. They concatenate the learned features of the detected sound source in a video, extracted by a ResNet, with the bottleneck of deep CNN on the spectrogram in order to obtain a separation mask isolating the given sound source in the mixed spectrogram.

To address the complex interaction between visual and auditory information, Chatterjee et al. (2021) introduce an Audio Visual Scene Graph Segmenter (AVSGS). AVSGS uses a deep neural network to segment the visual structure of a scene into a graph, the embedding of which is combined with the latent space of a spectrogram encoder and input into a decoder.

Islam et al. (2024) use a meta-consistency driven time-test adaptation scheme for video-aided music source separation. This approach enables a pre-trained model to quickly adapt to unseen data. The authors suggest that this method is robust for visually guided audio source separation, as it does not rely on fixed weights and allows for efficient parameter updates through a meta-learning approach. They achieve an SDR of 12.81 dB on the MUSIC dataset, outperforming all previously discussed approaches. As of August 2024, this seems to be the best reported result on any audio-visual dataset. The other results are reported on and discussed in Section 5.2.

2.4 Video Feature Extraction

This thesis aims to identify suitable methods for extracting rich and meaningful features from video. These features are required for integrating video into a separation algorithm. Therefore, this section is dedicated to evaluating potential methods for video feature extraction.

The introduction of the RNN with long short-term memory (LSTM) by Hochreiter and Schmidhuber (1997) enabled the capture of long-term dependencies in sequences, such as videos. Many models have since surpassed this approach in terms of action recognition. Simonyan and Zisserman (2014) propose a two-stream architecture using two 2-dimensional CNNs, one capturing the spatial information, the other capturing temporal information in order to predict optical flow. Optical flow describes the per-pixel movement within a video.

Tran et al. (2015) propose using a 3-dimensional CNN for feature extraction, arguing that a $3 \times 3 \times 3$ kernel, combined with a linear classifier, achieves state-of-the-art video classification results with a conceptually simpler approach and

faster inference. Carreira and Zisserman (2017) emphasise the importance of pre-training models on a large dataset before fine-tuning them for specific tasks. Additionally, they suggest a 3-dimensional two-stream architecture.

A newer approach to optical flow estimation is outlined by Teed and Deng (2020). They propose Recurrent All-Pairs Field Transforms (RAFT), combining a feature encoder that produces a per-pixel feature map for each frame with a 4D correlation volume that stores the correlation of all pixel pairs across two frames, along with a convolutional gated recurrent unit (GRU) to iteratively refine an initial flow estimate. The authors claim that RAFT achieves superior results on several noteworthy optical flow estimation benchmarks, the details of which exceed the scope of this thesis.

2.5 Synthesis

This literature review provides an overview of relevant developments in audio source separation, focusing on both traditional and state-of-the-art approaches. Early seminal methodologies, such as ICA and NMF, are discussed. The review also highlights more recent advancements using deep neural networks and generative models, which have expanded the possibilities in source separation. Nonetheless, none of these approaches have managed to achieve an SDR of more than 15 dB on real-world data. Consequently, no approach manages to achieve a source separation that could be considered to be true separation. Multi-modal integration is also discussed, particularly the incorporation of visual data to enhance audio separation, reflecting a growing trend in the field.

The main insights from the literature review provide the foundation for the subsequent chapters. NMF will be used as a baseline method. The self-supervised method using a multi-encoder AE suggested by Webster and J. Lee (2023) will be one of the evaluated methods, as it offers an elegant solution to a complicated problem. BASIS separation (Jayaram and Thickstun, 2020) will be another, aligning with the original goal of using deep generative models. BASIS separation allows for using any generative model that can compute the probability of data and shows promise in the 2D image domain. Furthermore, both RAFT (Teed and Deng, 2020) and the use of a 3D ResNet will be assessed for video feature extraction.

Chapter 3

Methodology

3.1 Datasets and Pre-Processing

All datasets are subject to a 70/20/10 split, where 70% of the data is allocated to the training set, 20% to the validation set for hyperparameter tuning, and the remaining 10% is reserved as the test set for the final evaluation.

All audio data is preprocessed by saving the monaural magnitude spectrograms $|\boldsymbol{X}^T|$ as `png` and the phase $\boldsymbol{\Phi}^T$ as a `numpy` array in a `npy` file. The MUSDB18 and URMP datasets provide binaural audio with two separate audio streams per track. The monaural audio stream is constructed by taking the mean of both streams. The magnitude spectrograms $|\boldsymbol{X}^T|$ are stored in full scale but downscaled to 64×64 pixels at runtime to reduce the computational load. All data is preprocessed and saved as 5-second chunks.

3.1.1 Toy

For establishing a proof of concept, a toy dataset is utilised, consisting of sine, triangle, pulse and sawtooth waves. The waves are combined to form mixtures by simply adding the waveforms, weighted by a coefficient $\alpha = 1/k$ for k sources. Only one parameter, the frequency $f \in [200, 1500]$, is used to create each data point of a given wave. The waves have a sampling rate of 16 kHz. The `scipy.signal` library is used to generate the toy data. The toy dataset is visualised in Figure 3.1.

3.1.2 MUSDB18

The Music Database (MUSDB18) dataset (Rafii et al., 2017) is utilised for the unimodal experiments on real-life data. This dataset consists of pop songs split into the four categories *vocals*, *bass*, *drums* and *other*. The original sampling rate of

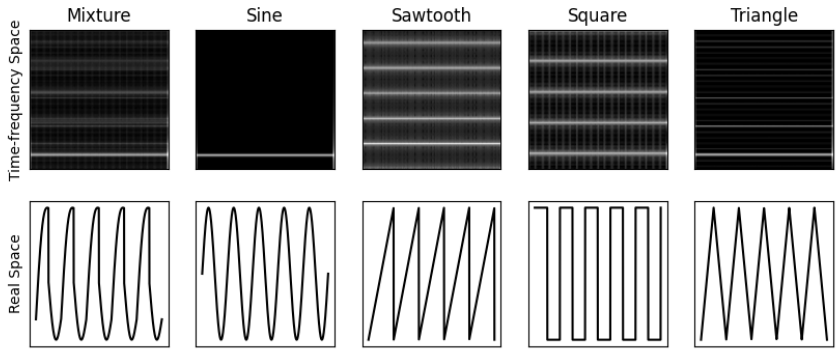


Figure 3.1: Downscaled sample from the toy dataset consisting of sine, triangle, pulse and sawtooth waves. The saved images, which can be converted back into real space audio, are of size 1025×128 and are re-scaled to 64×64.

Figure 3.2: Downscaled sample from the MUSDB18 dataset (Rafii et al., 2017). The saved images are of size 1025×384 and are re-scaled to 64×64.

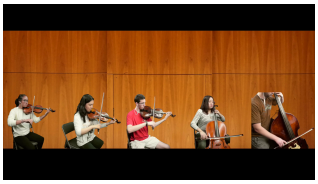
44100 Hz is reduced to 22050 Hz in order to further reduce the computational load. This new sampling rate is lower than the Nyquist rate and the signal is therefore not perfectly reconstructed. Nonetheless, it suffices for being obviously recognisable. A sample from the preprocessed MUSDB18 dataset can be seen in Figure 3.2.

3.1.3 URMP

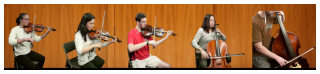
The University of Rochester Multitrack Classical Music Performance (URMP) dataset (Li et al., 2018) is used for audio-visual integration. This dataset contains 44 data points, each consisting of a chamber music performance, which includes the video of all musicians and the audio generated by each individual. The represented instruments are violin, violoncello, trumpet, saxophone, trombone, flute, oboe, cello, horn and viola. As with the MUSDB18 dataset, the sampling rate is reduced from 44100 Hz to 22050 Hz.

The videos are of size 1080×1920 and have a frame rate of 30 frames per second (fps). The data is preprocessed by first slicing the video and corresponding audio files into five second chunks. A data point consisting of k sources is then converted into $\binom{k}{2}$ pairs, each using the same video, effectively re-framing the dataset to as $k = 2$. The frame rate is reduced to 15 fps, which suffices for recognising movement.

Finally, the height of each video is cropped to only retain pixels from index 500 to 900 in order to avoid processing unnecessary (i.e., stationary) elements. The video is then converted to a tensor $\boldsymbol{V} \in \mathbb{R}^{c \times h \times w \times T}$ with $c = 3$ channels, height h, width w and T frames. Finally, the frames are re-shaped to 128×128 pixels.



(a) The original 1080×1920 frame.



(b) The non-moving parts of the image are removed.

(c) The pruned frame is reshaped to 128×128 pixels.

Figure 3.3: The video frame pre-processing pipeline from left to right.

3.1.4 MUSIC

The Multimodal Sources of Information for Contextual Understanding (MUSIC) dataset (Zhao, Gan, Rouditchenko, et al., 2018) is the most widely used dataset

for audio-visual source separation. It consists of a collection of video clips featuring individuals playing various musical instruments. The data is extracted from YouTube.

This dataset was not used for the experiments in this thesis. Using this dataset would require additional pre-processing, appropriate for future work, as described in Section 5.3.

3.2 Proposed Methods

In this section, the different approaches that will later be evaluated in Section 4 are described. For simplicity, all models are limited to performing experiments on $k = 2$ sources by re-mixing the stems within each dataset.

3.2.1 Modified BASIS

This thesis proposes a modified BASIS approach, hereafter referred to as Modified BASIS, which is adapted from the BASIS separation method described by Jayaram and Thickstun (2020). Let $\mathcal{X}$ represent the space of all vectors $\boldsymbol{x} \in \mathbb{R}^{n \cdot m}$ such that $x_i \in [0, 1]$ for all $i \in 0, \ldots, n \cdot m$. This defines the space of all flattened magnitude spectrograms $|\boldsymbol{X}^T| \in \mathbb{R}^{n \times m}$. Our BASIS approach involves using Langevin dynamics to sample from the posterior $p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \mid \boldsymbol{m})$ over the flattened sources in time-frequency space $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \in \mathcal{X}$ given a flattened mixed signal in time-frequency space $\boldsymbol{m} \in \mathcal{X}$.

Thus, the audio source separation problem is treated as an image separation problem on spectrograms. Instead of using Glow (Durk P Kingma and Dhariwal, 2018) or NCSN (Song, Garg, et al., 2019) as in the original paper, Modified BASIS uses VAEs (Diederik P Kingma and Welling, 2013) as generative models for computing the priors. This is a novel approach and potentially offers a significantly reduced computational cost when compared to the more expensive models used by Jayaram and Thickstun (2020). Our modified BASIS approach is summarised in Algorithm 11. The notation in this algorithm involves concatenating the sources and the latent vectors into one long vector $\boldsymbol{x}$ with which a Markov chain is constructed. The mixture function $g(\boldsymbol{x})$ is thereby overloaded as

$$g(\boldsymbol{x}) = \frac{1}{k} \sum_{i}^{k} \boldsymbol{s}_i \quad \text{s. t. } \boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \leftarrow \text{extract}(\boldsymbol{x}). \tag{3.1}$$

Algorithm 4 Modified BASIS Separation, adapted from Jayaram and Thickstun (2020)

Require: $\boldsymbol{m} \in \mathcal{X}$, $\{\sigma_i\}_{i=1}^{L}$, $\delta \in \mathbb{R}$, $T \in \mathbb{N}$
Sample $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \sim \text{Uniform}(\mathcal{X})$
Sample $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_k \sim \mathcal{N}(0, 1)$
Let $\boldsymbol{x}^{(1)} \leftarrow \boldsymbol{s}_1 \oplus \boldsymbol{z}_1 \oplus \cdots \oplus \boldsymbol{s}_k \oplus \boldsymbol{z}_k$
for $i = 1$ to L **do**
    $\eta_i \leftarrow \delta \cdot \sigma_i^2 / \sigma_L^2$
    for $t = 1$ to T **do**
        Sample $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
        $\boldsymbol{u}^{(t)} \leftarrow \boldsymbol{x}^{(t)} + \eta_i \nabla_x \log p_{\sigma_i}(\boldsymbol{x}^{(t)}) + \sqrt{2\eta_i}\boldsymbol{\epsilon}_t$
        $\boldsymbol{x}^{(t+1)} \leftarrow \boldsymbol{u}^{(t)} - \frac{\eta_i}{\sigma_i^2}(\boldsymbol{m} - g(\boldsymbol{x}^{(t)}))$
    end for
end for

The original BASIS involves using a deep generative model to compute the gradient of $\log p(\boldsymbol{x})$. The VAEs provide an approximate posterior

$$q_\phi(\boldsymbol{z}_i \mid \boldsymbol{s}_i) = \mathcal{N}(\boldsymbol{z}_i \mid \boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}^2)) \tag{3.2}$$

with

$$f^{\text{enc}}(\boldsymbol{s}) = \phi = \langle \boldsymbol{\mu}, \log \boldsymbol{\sigma}^2 \rangle \tag{3.3}$$

as well as a likelihood

$$p_\theta(\boldsymbol{s} \mid \boldsymbol{z}) \tag{3.4}$$

which we can sample from using the decoder f^{dec}. We also have a standard normal Gaussian prior over the latent space

$$p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z} \mid \mathbf{0}, \boldsymbol{I}). \tag{3.5}$$

A possible approach to approximating $\log p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k)$ with a VAE involves using the evidence lower bound (ELBO) and setting

$$\boldsymbol{x} = \boldsymbol{s}_1 \oplus \cdots \oplus \boldsymbol{s}_k. \tag{3.6}$$

By definition of the KL diversion (Kullback and Leibler, 1951) between the approximate posterior $q_\phi(\boldsymbol{z} \mid \boldsymbol{x})$ and the true posterior $p(\boldsymbol{z} \mid \boldsymbol{s})$, we can derive

$$
\begin{aligned}
D_{KL}(q_\phi(\boldsymbol{z} \mid \boldsymbol{s}) \| p(\boldsymbol{z} \mid \boldsymbol{s})) &= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{s})}\left[\log \frac{q_\phi(\boldsymbol{z} \mid \boldsymbol{s})}{p(\boldsymbol{z} \mid \boldsymbol{s})}\right] && (3.7)\\
&= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{s})}\left[\log \frac{q_\phi(\boldsymbol{z} \mid \boldsymbol{s}) p(\boldsymbol{s})}{p(\boldsymbol{z}, \boldsymbol{s})}\right] && (3.8)\\
&= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{s})}\left[\log p(\boldsymbol{s})\right] + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{s})}\left[\log \frac{q_\phi(\boldsymbol{z} \mid \boldsymbol{s})}{p(\boldsymbol{z}, \boldsymbol{s})}\right] && (3.9)\\
&= \log p(\boldsymbol{s}) - \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{s})}\left[\log \frac{p(\boldsymbol{z}, \boldsymbol{s})}{q_\phi(\boldsymbol{z} \mid \boldsymbol{s})}\right] && (3.10)\\
&= \log p(\boldsymbol{s}) - \mathcal{L}(\phi, \boldsymbol{s}) && (3.11)
\end{aligned}
$$

It follows that

$$
\mathcal{L}(\phi, \boldsymbol{s}) = \log p(\boldsymbol{s}) - D_{KL}(q_\phi(\boldsymbol{z} \mid \boldsymbol{s}) \| p(\boldsymbol{z} \mid \boldsymbol{s})) \tag{3.12}
$$

and we have thus derived the ELBO $\mathcal{L}(\phi, \boldsymbol{s})$. As $D_{KL} \geq 0$ always holds, we know that

$$
\log p(\boldsymbol{s}) \geq \mathcal{L}(\phi, \boldsymbol{s}) \tag{3.13}
$$

must hold. While the ELBO is useful for training a VAE, using it as an approximation for $\log p(\boldsymbol{s})$ in Algorithm 11 yields overwhelmingly noisy results and is thus not further considered.

An alternative approach involves sampling from both the sources $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k$ as well as a corresponding latent vector for each source $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_k$. Thereby the log prior becomes a $\log p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k, \boldsymbol{z}_1, \ldots, \boldsymbol{z}_k)$. This can be decomposed as

$$
\log p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k, \boldsymbol{z}_1, \ldots \boldsymbol{z}_k) = \log p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \mid \boldsymbol{z}_1, \ldots \boldsymbol{z}_k) + \log p(\boldsymbol{z}_1, \ldots \boldsymbol{z}_k) \tag{3.14}
$$

We can then approximate $p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \mid \boldsymbol{z}_1, \ldots \boldsymbol{z}_k)$ by first sampling

$$
\hat{\boldsymbol{s}}_j \sim f_j^{\text{dec}}(\boldsymbol{z}_j) \tag{3.15}
$$

then defining

$$
p_{\sigma_i}(\boldsymbol{s} \mid \boldsymbol{z}_j) = \mathcal{N}(\boldsymbol{s} \mid \hat{\boldsymbol{s}}_j, \sigma_i^2 \boldsymbol{I}) \tag{3.16}
$$

and finally computing

$$
\log p_{\sigma_i}(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \mid \boldsymbol{z}_1, \ldots \boldsymbol{z}_k) = \sum_{j=1}^{k} \log p_{\sigma_i}(\boldsymbol{s}_j \mid \boldsymbol{z}_j). \tag{3.17}
$$

We can then compute $\log p(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_k)$ as

$$\log p(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_k) = \sum_{j=1}^{k} \log p(\boldsymbol{z}_j) \tag{3.18}$$

$$= \sum_{j=1}^{k} \log \mathcal{N}(\boldsymbol{z}_j \mid \mathbf{0}, \boldsymbol{I}). \tag{3.19}$$

The gradient

$$\nabla \log p_{\sigma_i}(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k, \boldsymbol{z}_1, \ldots \boldsymbol{z}_k) \tag{3.20}$$

can then be computed using `autograd`. In Algorithm 11, this approach involves setting

$$\boldsymbol{x} = \boldsymbol{s}_1 \oplus \boldsymbol{z}_1 \oplus \cdots \oplus \boldsymbol{s}_k \oplus \boldsymbol{z}_k. \tag{3.21}$$

The experiments are performed with hyperparameters set to $\delta = 2 \times 10^{-5}$, $L = 10$, $T = 100$, $\sigma_1 = 0.1$ and $\sigma_L = 1$ whereby $\sigma_1, \ldots, \sigma_L$ are logarithmically spaced with base 10. These hyperparameters are identical to the ones used by Jayaram and Thickstun (2020), as the TPE fails to find superior hyperparameters on the new domain. The VAEs $f_1, \ldots, f_n$ used in the equations above are trained on only one data class each, i.e., one type of stem. E.g., operating on the toy data, VAE f_1 is trained on sine, f_2 on sawtooth, f_3 on square and f_4 on triangle waves. This approach of training a model per class rather than a single, large model, is inspired by Kim et al. (2021).

Noise Conditioning

The models $p_{\sigma_i}(\boldsymbol{s} \mid \boldsymbol{z})$ describe the approximated probability distribution over sources $\boldsymbol{s}$ perturbed by noise level σ_i. The original paper finetunes the model f_j for each noise level σ_i by utilising a transfer learning approach with data for each noise level being perturbed as $\boldsymbol{s}' = \boldsymbol{s} + \boldsymbol{\epsilon}$ with noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_i^2 \boldsymbol{I})$. In Section 4, two BASIS versions will be tested:

1. *BASIS*, using a single model for all $\sigma_1, \ldots, \sigma_L$.
2. *BASIS Finetuned*, using a different model f_{σ_i} per σ_i, where each model f_{σ_i} is finetuned on perturbed data with noise given by $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_i^2 \boldsymbol{I})$.

VAE Architecture

A Tree-structured Parzen Estimator (TPE), as proposed by Bergstra et al. (2011), was used as a guideline to select model architectures and hyperparameters. A TPE works by iteratively fitting one Gaussian Mixture Model (GMM) $l(\boldsymbol{\theta})$ to the set of hyperparameters that yield the best 20% of objective values (i.e., the greatest mean SDR on the validation set) and a second GMM $g(\boldsymbol{\theta})$ to the set of hyperparameter-sets which lead to the objectively worse 80%. For each subsequent iteration, the TPE selects hyperparameters

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \frac{l(\boldsymbol{\theta})}{g(\boldsymbol{\theta})} \tag{3.22}$$

Thus, the hyperparameters are iteratively selected in order to be similar to the previously successful samples while remaining dissimilar to those that were previously unsuccessful. The `optuna` library (Akiba et al., 2019) provides the implementation used to compute the TPE within the scope of this thesis.

The VAE fitted to the toy data consists of a symmetric encoder and decoder architecture, featuring three convolutional layers and two fully connected layers in the encoder (one for $\boldsymbol{\mu}$, one for $\log \boldsymbol{\sigma}^2$), with one fully connected and three transposed convolutional layers on the decoder. The number of learned features increases in each subsequent layer of the encoder, with channels progressing from 1 in the input layer to 4, 8 and then 16. This is reversed in the decoder. A kernel of size 3 for all layers and a stride of 1 ensures sufficiently detailed feature extraction. MSE loss is used as the reconstruction loss.

The architecture of the MUSDB18 model is similar with two differences. Firstly, it features an additional convolutional layer on the encoder and transposed convolutional layer on the decoder with 32 learned features to accommodate the increased data complexity. Secondly, its latent space is larger. The latent space sizes are $\boldsymbol{z} \in \mathbb{R}^8$ for the toy data and $\boldsymbol{z} \in \mathbb{R}^{32}$ for MUSDB18.

3.2.2 AE-BSS

This approach is adapted directly from Webster and J. Lee (2023). *AE-BSS* involves k' concurrent encoder networks, each encoding the same input signal into a different deterministic latent representation $\boldsymbol{z}_i$. These latent representations are then concatenated as $\boldsymbol{z} = \boldsymbol{z}_1 \oplus \cdots \oplus \boldsymbol{z}_{k'}$ and fed into a single decoder network in order to obtain the reconstructed signal $\hat{\boldsymbol{x}} = f^{\text{dec}}(\boldsymbol{z})$. This architecture is visualised in Figure 2.8. The training and inference occur as described in Section 2.2.5. Webster and J. Lee (2023) argue that this method will only learn to utilise as many encoders as there are sources, leading to superfluous dead encoders. This thesis does not test this and sets $k = k'$.

The evaluation uses a kernel size of 7, a concatenated latent space of size 196, symmetrical convolutional encoders and decoder of depth 5 with 24, 48, 96, 144 and 196 channels. The models were trained on a batch size of 64. Interestingly, the results in Figure 2.8, published by Webster and J. Lee (2023), are only reproducible with this specific batch size.

The hyperparameters include the use of a group norm, L1 loss as the separation norm, weight mixing terms $\lambda_{\text{mixing}} = 0.5$, $\lambda_{\text{zero recon.}} = 0.01$ and $\lambda_z = 0.01$. A weight decay of 5×10^{-5} is applied and $\boldsymbol{z} \in \mathbb{R}^{196}$. Training was conducted with a learning rate of $\eta = 0.001$. These hyperparameters were adapted directly from Webster and J. Lee (2023), as the TPE failed to find superior hyperparameters.

3.2.3 AE-BSS Linear

An alternative AE-BSS approach is evaluated, in which instead of concatenating $\boldsymbol{z} = \boldsymbol{z}_1 \oplus \cdots \oplus \boldsymbol{z}_k$ and using a single decoder, a separate decoder per encoder is used. This means that the concatenation of the latent space does not occur, and the sparse mixing loss ℓ_{mixing} from Equation 2.29 is ignored. Outputs $\hat{\boldsymbol{x}}_1, \ldots, \hat{\boldsymbol{x}}_k$ are obtained and then used to reconstruct

$$\hat{\boldsymbol{x}} = \sum_{i=1}^{k} \alpha_i \hat{\boldsymbol{x}}_i \tag{3.23}$$

with mixing coefficients $\alpha_i = 1/k$. This approach operates under a linear assumption that holds in the used data. This method will be referred to as *AE-BSS Linear*.

3.2.4 Modified BASIS with Video

The intuition behind incorporating visual information lies in evaluating the feasibility of a given separation and using this to guide the separator in a more profitable direction. The proposed method involves training a binary classifier to predict the Bernoulli-distributed random variable χ, which determines the joint probability of the sources being generated by events visible in the video $\boldsymbol{V}$.

$$p(\chi = 1 \mid \boldsymbol{s}_1, \ldots, \boldsymbol{s}_k) = p(\boldsymbol{V}, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_k) \tag{3.24}$$

Let $\mathcal{S} = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k\}$ and $\mathcal{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_k\}$ for simpler notation. If we can compute χ, we can, instead of sampling from Equation

$$p(\mathcal{S}, \mathcal{Z}) = p(\mathcal{S} \mid \mathcal{Z}) p(\mathcal{Z}) \tag{3.25}$$

as in Algorithm 11, sample from

$$p(\mathcal{S}, \mathcal{Z}, \chi = 1) = p(\mathcal{S} \mid \mathcal{Z}) p(\mathcal{Z}) p(\chi = 1 \mid \mathcal{S}). \tag{3.26}$$

Thus, incorporating video into the equation simply involves evaluating this additional term using a classifer network and setting

$$\boldsymbol{x} = \boldsymbol{s}_1 \oplus \boldsymbol{z}_1 \oplus \cdots \oplus \boldsymbol{s}_k \oplus \boldsymbol{z}_k \oplus \chi \tag{3.27}$$

in Algorithm 11.

Video Gradient Weighting

The gradient of the logarithm of Equation 3.26 is computed as

$$\nabla_{\boldsymbol{x}} \log p(\mathcal{S}, \mathcal{Z}, \chi = 1) = \nabla_{\boldsymbol{x}} \log p(\mathcal{S} \mid \mathcal{Z}) + \nabla \log p(\mathcal{Z}) + \nabla_{\boldsymbol{x}} \log p(\chi = 1 \mid \mathcal{S}) \tag{3.28}$$

$$= \sum_{j=1}^{k} \nabla_{\boldsymbol{x}} \log \mathcal{N}(\boldsymbol{s}_j \mid \hat{\boldsymbol{s}}_j, \sigma_i^2 \boldsymbol{I}) + \nabla_{\boldsymbol{x}} \log \mathcal{N}(\boldsymbol{z}_j \mid \boldsymbol{0}, \boldsymbol{I}) + \nabla_{\boldsymbol{x}} p(\chi = 1 \mid \mathcal{S}) \tag{3.29}$$

$$= \sum_{j=1}^{k} \sum_{l} \nabla_{\boldsymbol{x}} \log \mathcal{N}(\boldsymbol{s}_j[l] \mid \hat{\boldsymbol{s}}_j[l], \sigma_i^2) + \sum_{m} \nabla_{\boldsymbol{x}} \log \mathcal{N}(\boldsymbol{z}_j[m] \mid 0, 1) + \nabla_{\boldsymbol{x}} p(\chi = 1 \mid \mathcal{S}). \tag{3.30}$$

As can be seen in Equation 3.29, the first two terms in Equation 3.28 consist of the sum over the terms' dimensionality, which can be substantial. It follows that the first two terms tend to be of larger absolute value (as computations occur in log space) and thus have more influence on the gradient by orders of magnitude. This can be balanced out by introducing a weight $\beta \in \mathbb{R}_{>0}$ and then replacing the gradient of the actual score function with

$$\nabla_{\boldsymbol{x}} \log p(\mathcal{S} \mid \mathcal{Z}) + \nabla_{\boldsymbol{x}} \log p(\mathcal{Z}) + \beta \nabla_{\boldsymbol{x}} \log p(\chi = 1 \mid \mathcal{S}). \tag{3.31}$$

Model Architecture for Predicting χ

Let $\mathcal{V}$ be the space of all RGB (i.e., 3-channel) video tensors $\boldsymbol{V} \in \mathbb{R}^{3 \times h \times w \times T}$. The model consists of three sub-models:

- A 2-dimensional convolutional encoder $f_{2D}(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k) = \langle \boldsymbol{\mu}_1, \log \boldsymbol{\sigma}_1^2 \rangle$.
- A 3-dimensional convolutional encoder $f_{3D}(\boldsymbol{T}) = \langle \boldsymbol{\mu}_2, \log \boldsymbol{\sigma}_2^2 \rangle$ for some tensor $\boldsymbol{T} \in \mathbb{R}^{c \times h \times w \times T}$ with c channels and T frames.

- A fully connected network $f_{FC}(\boldsymbol{z}) = \hat{\chi}$.

Inference occurs by first passing our data through both encoders and then sampling

$$\boldsymbol{z}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \text{diag}(\boldsymbol{\sigma}_1^2)) \tag{3.32}$$
$$\boldsymbol{z}_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, \text{diag}(\boldsymbol{\sigma}_2^2)) \tag{3.33}$$

Thereafter, the latent space is concatenated as

$$\boldsymbol{z} = \boldsymbol{z}_1 \oplus \boldsymbol{z}_2 \tag{3.34}$$

and finally

$$\hat{\chi} = f_{FC}(\boldsymbol{z}). \tag{3.35}$$

is obtained. A loss function for training is used consisting of

$$\ell(\hat{\chi}, \chi) = D_{KL}\left(\mathcal{N}(\boldsymbol{\mu}_1, \text{diag}(\boldsymbol{\sigma}_1^2)) \| \mathcal{N}(\mathbf{0}, \boldsymbol{I})\right) \tag{3.36}$$
$$+ D_{KL}\left(\mathcal{N}(\boldsymbol{\mu}_2, \text{diag}(\boldsymbol{\sigma}_2^2)) \| \mathcal{N}(\mathbf{0}, \boldsymbol{I})\right) \tag{3.37}$$
$$+ \ell_{BCE}(\hat{\chi}, \chi) \tag{3.38}$$

The stochastic latent representation performed better in the preliminary tests without any noticeable performance costs. This improvement may be attributed to a regularised latent space and the incentive for $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ to be within a similar range. Additionally, the distribution can be used for uncertainty quantification, thereby enhancing the interpretability of the model. This is not explored within the scope of this thesis, but may be relevant for future work.

Tested Configurations

The video is important as the movement between frames provides valuable insights. Therefore, the optical flow between the frames of the video is the primary point of interest. In Section 2.4, it is mentioned that a 3-dimensional CNN is capable of extracting optical flow. It is also established that RAFT (Teed and Deng, 2020) can lead to superior results. Additionally, it stands to reason that a ResNet (K. He et al., 2016), outperforming many other architectures in tasks ranging from image classification (Sarwinda et al., 2021) (T. He et al., 2019) to semantic segmentation (Xia, Yin, and Zhang, 2019) and transfer learning (Rezende et al., 2017), is capable of extracting meaningful features from a spectrogram.

From these premises, four feasible model architectures emerge. These models are trained and compared based on their validation accuracy during training, to determine which is best suited for use in Algorithm 11. These configurations are as follows:

1. f_{2D} is the same model architecture used for the deep generative priors in Modified BASIS, as described in Section 3.2.1. f_{3D} takes RGB video $\boldsymbol{V} \in \mathcal{V}$ as input.

2. f_{2D} is identical and $f_{RAFT}(\boldsymbol{F}_t, \boldsymbol{F}_{t+1}) = \boldsymbol{U}_t \in \mathbb{R}^{2\times h\times w}$ is defined for video frames $\boldsymbol{F}_t$. The stacked optical flows $\boldsymbol{U}_1 \oplus \cdots \oplus \boldsymbol{U}_{T-1} = \boldsymbol{U} \in \mathbb{R}^{2\times h\times w\times T-1}$ are then used as an input to $f_{3D}(\cdot)$, instead of the video $\boldsymbol{V}$.

3. f_{2D} is a ResNet18 and f_{RAFT} is used to extract optical flow.

4. f_{2D} is a ResNet18 and f_{RAFT} is not used.

All configurations have $\boldsymbol{z}_1, \boldsymbol{z}_2 \in \mathbb{R}^{64}$. The RAFT implementation used is the pre-trained RAFT-Small model provided by `torchvision`. This model is not finetuned during training.

The function f_{3D} is implemented as 3D ResNet18 (Tran et al., 2015) and f_{FC} has one hidden layer $\boldsymbol{h} \in \mathbb{R}^{64}$ with a ReLU activation function (Agarap, 2018). Training occurs with a learning rate $\eta = 10^{-4}$ and a batch size of 3 for the RAFT + ResNet18 architecture and 4 for the other models. These small batch sizes were the largest possible given the available VRAM.

Chapter 4

Experiments and Results

4.1 Experimental Setup

4.1.1 Tested Methods

All proposed methods from Section 3.2 are tested in the experiments. The results are compared to three baselines. The idea behind this is that if an approach cannot outperform the baselines, it shall not be further considered. The first baseline is NMF, as described in Section 2.2.2. For this, the implementation provided by the `scikit-learn` library (Pedregosa et al., 2011) is used. This involves coordinate descent as the optimisation algorithm, the Frobenius norm as a β-loss (i.e., $\beta = 2$ in Algorithm 3, as, despite the suggestion by Févotte, Bertin, and Durrieu (2009), the Itakura-Saito divergence did not outperform the Frobenius norm), random initialisation, no regularisation and a maximum of 200 iterations. The second baseline consists of random samples from the respective VAEs $\hat{\boldsymbol{x}}_j = f_j^{\text{dec}}(\boldsymbol{z})$ with $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. These images are generated with prior knowledge on the structure of the data but no information on the mixed signal. The third baseline consists of noise images $\tilde{\boldsymbol{x}} \sim \text{Uniform}(\mathcal{X})$.

4.1.2 Hardware

All experiments were conducted on a computer with 32 GB RAM, a 2 TB SSD, an Nvidia RTX 4080 GPU with 16 GB VRAM and an Intel i7 13700k processor, running Debian[1] 12. The code was written in Python version 3.11.2 and can be found at `https://github.com/maxjappert/mmdgass`.

[1] `https://www.debian.org/` (accessed August 29, 2024).

4.1.3 Statistical Measures

The mean of $\boldsymbol{x} \in \mathbb{R}^N$ is computed as

$$\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i. \tag{4.1}$$

The standard deviation measures the spread of individual data points, estimating how much the data deviates from the mean. The test validation sets, on which the results are evaluated, are samples from the entire dataset. Thus, Bessel's correction (So, 2008) must be used to compute an unbiased estimator of the variance. We therefore compute the estimated standard deviation as

$$\hat{\sigma}_{\boldsymbol{x}} = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (x_i - \bar{x})^2}. \tag{4.2}$$

The standard error, on the other hand, measures the precision of the sample mean as an estimate of the true mean. It reflects the variability of the sample means if the experiment were to be repeated multiple times. It is computed as

$$\hat{\sigma}_{\bar{x}} = \frac{\hat{\sigma}_{\boldsymbol{x}}}{\sqrt{N}}. \tag{4.3}$$

Equations 4.1, 4.2 and 4.3 are adapted from Wasserman (2013) and computed using the `numpy` library.

4.1.4 Description of Experiments

A seed `s = 42` is set to ensure the reproducibility of the experiments. The `mir_eval` package (Raffel et al., 2014) provides the functions for computing the evaluation metrics during the experiments, as described in Section 2.2.3. The experiments are conducted with random mixtures of all source types with $k = 2$. The following experiments are conducted:

1. **Uni-Modal Separation Experiment:** 450 samples are uniformly drawn from the toy and MUSDB18 test datasets and then separated by all methods described in Section 3.2. The separation attempts are evaluated based on SDR, ISR, SIR and SAR and the results are displayed both as *mean* $\pm$ *standard error* in Tables 4.1 and 4.2 and as violin plots in Figures 4.1 and 4.2.

2. **Visual Samples:** Following a similar premise as Experiment 1, four samples each are uniformly drawn from the toy and MUSDB18 test datasets

and then separated using all methods described in Section 3.2. These separations are visualised in Figures 4.3 and 4.4, permitting qualitative analysis.

3. **ELBO Confusion Matrices:** Figure 4.5 presents a confusion matrix for both the toy and MUSDB18 datasets, illustrating every stem type under the ELBO $\mathcal{L}(\phi, \boldsymbol{s}) \leq \log p(\boldsymbol{s})$ computed by each VAE type. The rows denote the models and the columns denote the data classes. Each row is normalised by a softmax layer resulting in a valid probability distribution over all stem types per model, with a sample space Ω with cardinality $|\Omega| = 4$. An ideal scenario would yield a diagonal matrix, indicating that the models assign high probability to in-class data and low probability to out-of-class data. The data is drawn from the test datasets.

4. **Audio-Visual Matching Classifier Performance:** Figure 4.11 shows the validation accuracy over the training epochs for all four video model configurations discussed in Section 3.2.4.

5. β**-Evaluation:** For each video weight $\beta \in \{2^i \mid 0 \leq i \leq 7, i \in \mathbb{N}_0\}$ 30 samples are drawn from the URMP validation set and the mean and standard error of the four metrics is computed. These values are recorded in Table 4.3 and plotted in Figure 4.6.

6. **Video Separation Experiment:** Following a similar premise as Experiment 1, 50 samples are drawn from the URMP test set and separated by Modified BASIS both with and without the inclusion of video with $\beta = 128$, as explained in Section 3.2.4. The comparison as *mean* $\pm$ *standard error* can be found in Table 4.4 and as a violin plot in Figure 4.10.

7. **Additional BASIS Visual Samples:** Three samples are uniformly drawn from the toy, MUSDB18 and URMP test datasets. These are then separated using Modified BASIS and the separated spectrograms are printed in Figures 4.8, 4.9 and 4.7. The latter includes both the separation with and without incorporating video and $\beta = 128$.

4.2 Results

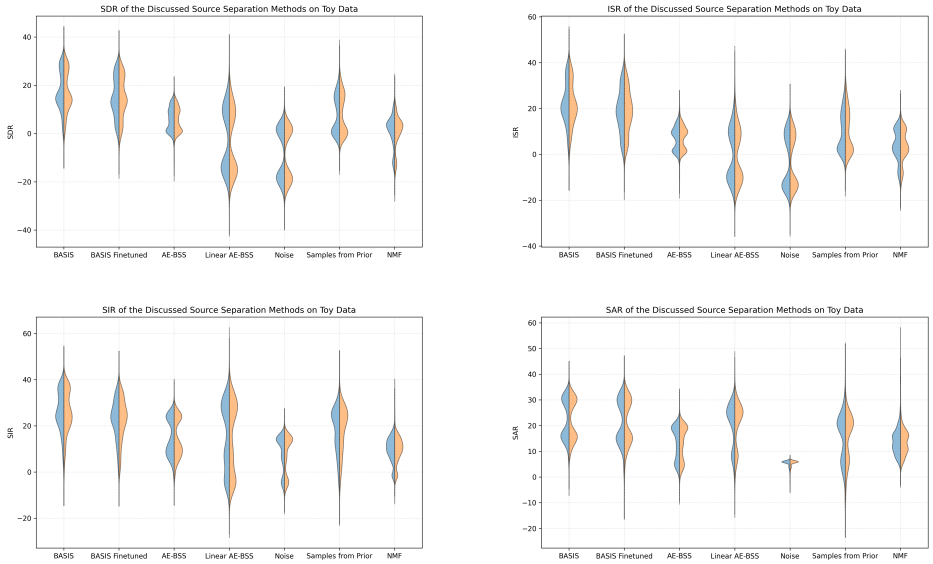


Figure 4.1: Uni-Modal Separation Experiment on the toy data. Evaluated metrics in dB over 450 samples from the test data. The violins are asymmetric, the blue side describing Source 1 and the orange side Source 2.

	BASIS	BASIS F.	AE-BSS	L. AE-BSS	Noise	P. Samples	NMF
SDR	**17.4 ± 0.3**	14.8 ± 0.3	5.7 ± 0.2	-2.6 ± 0.5	-8.7 ± 0.4	7.9 ± 0.3	0.8 ± 0.2
ISR	**22.0 ± 0.4**	18.7 ± 0.4	7.0 ± 0.2	-0.2 ± 0.4	-2.8 ± 0.4	9.9 ± 0.3	3.7 ± 0.3
SIR	**26.1 ± 0.4**	22.7 ± 0.3	14.6 ± 0.2	15.2 ± 0.4	7.8 ± 0.3	18.2 ± 0.3	9.5 ± 0.2
SAR	**22.1 ± 0.3**	21.2 ± 0.3	13.2 ± 0.2	20.5 ± 0.4	5.2 ± 0.1	15.6 ± 0.4	13.7 ± 0.2

Table 4.1: Results of Uni-Modal Separation Experiment on the toy dataset in dB. Evaluated metrics over 450 samples from the test set. The entries are *mean* ± *standard error*.

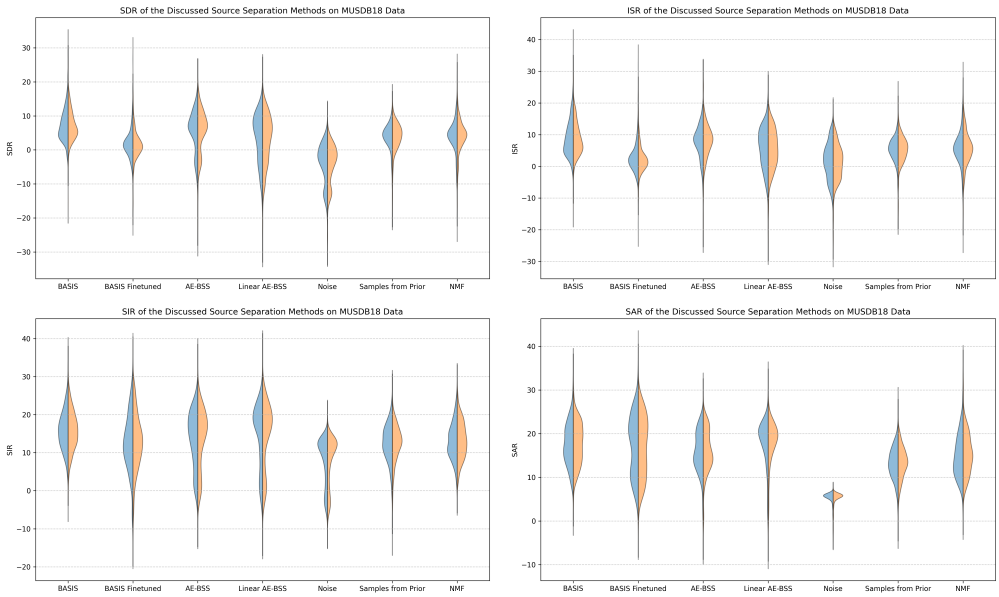
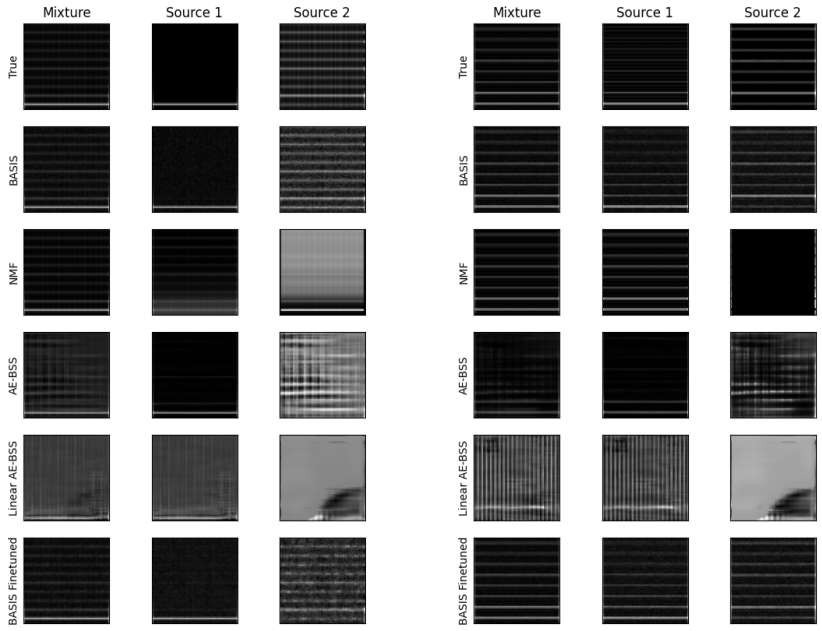


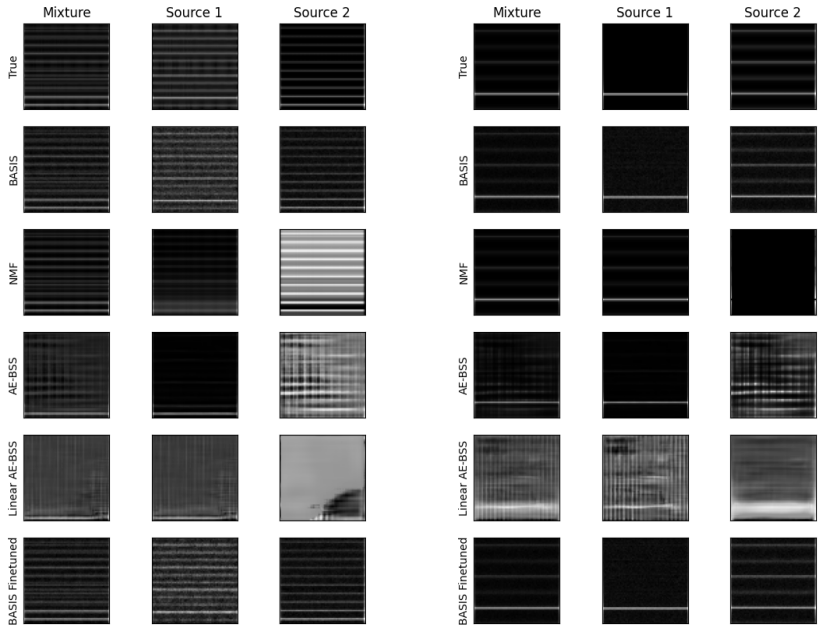
Figure 4.2: Results of Uni-Modal Separation Experiment on the MUSDB18 data. Evaluated metrics in dB over 450 samples from the test dataset. The blue side describes Source 1 and the orange side Source 2.

	BASIS	BASIS F.	AE-BSS	L. AE-BSS	Noise	P. Samples	NMF
SDR	**7.2 ± 0.2**	1.7 ± 0.2	5.4 ± 0.2	3.2 ± 0.2	-4.1 ± 0.2	3.8 ± 0.1	4.0 ± 0.2
ISR	**8.8 ± 0.2**	2.5 ± 0.2	7.8 ± 0.2	5.5 ± 0.2	1.3 ± 0.2	5.3 ± 0.1	5.7 ± 0.2
SIR	**15.8 ± 0.2**	12.8 ± 0.2	13.6 ± 0.2	14.5 ± 0.3	7.7 ± 0.2	12.5 ± 0.2	13.2 ± 0.2
SAR	**17.6 ± 0.2**	17.1 ± 0.2	16.0 ± 0.2	18.0 ± 0.2	5.2 ± 0.1	13.3 ± 0.1	15.2 ± 0.2

Table 4.2: Results of Uni-Modal Separation Experiment on the MUSDB18 data in dB. Evaluated metrics for 400 samples from the test dataset. The entries are *mean* ± *standard error.*



(a) (b)



(c) (d)

Figure 4.3: Visual samples and their separations from the toy test set using the discussed approaches.

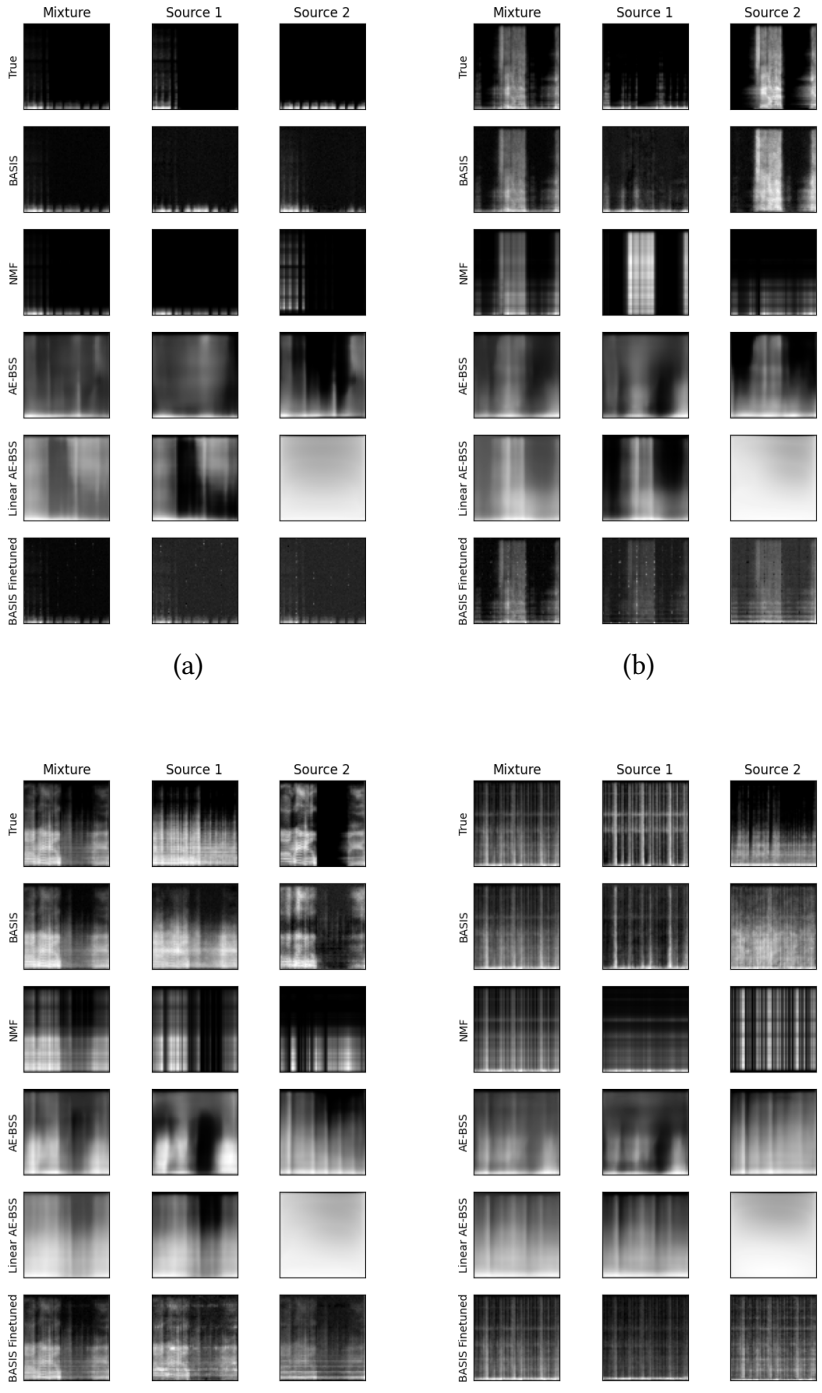
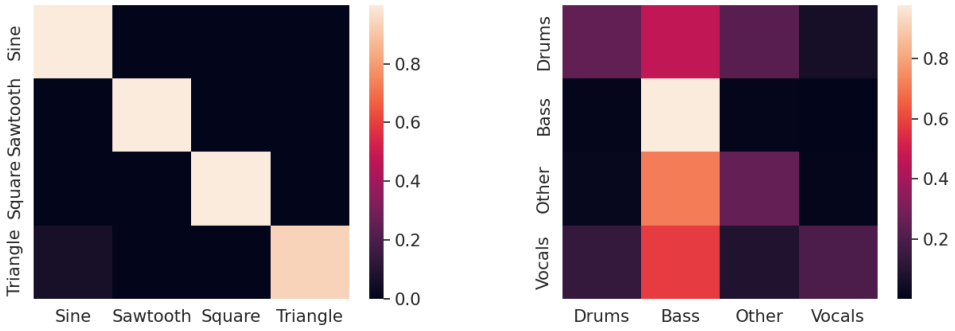


Figure 4.4: Visual samples and their separations from the MUSDB18 test set using the discussed approaches.



(a) Toy

(b) MUSDB18

Figure 4.5: Confusion matrices showing the approximated prior probabilities $p(\boldsymbol{s})$ as the exponentiated ELBOs $\exp[\mathcal{L}(\phi, \boldsymbol{s})] \leq p(\boldsymbol{s})$ of each source on the x-axis under each VAE on the y-axis normalised by a softmax layer for the MUSDB18 and toy datasets. The data is drawn from the test dataset. For MUSDB18 in Figure 4.5b, a temperature scaling parameter $\tau = 1/8$ is used.

β	Mean $\pm$ Standard Error
1	9.75 $\pm$ 0.39
2	10.04 $\pm$ 0.31
4	10.10 $\pm$ 0.43
8	**10.42 $\pm$ 0.31**
16	9.63 $\pm$ 0.43
32	10.08 $\pm$ 0.43
64	9.99 $\pm$ 0.28
128	10.07 $\pm$ 0.31

Table 4.3: Mean and standard error for different video weights β in dB. The same values can be found in Figure 4.6. These values were evaluated using 30 samples from the URMP validation set.

	No Video	Video with $\beta = 128$
SDR	10.03 $\pm$ 0.23	**10.07 $\pm$ 0.22**
ISR	11.19 $\pm$ 0.26	**11.24 $\pm$ 0.25**
SIR	**17.55 $\pm$ 0.27**	17.55 $\pm$ 0.25
SAR	22.22 $\pm$ 0.27	**22.25 $\pm$ 0.26**

Table 4.4: Results of Video Separation Experiment in dB, comparing the performance on the URMP test set when incorporating video vs. when not incorporating video. The data is printed as *mean* $\pm$ *standard error*.

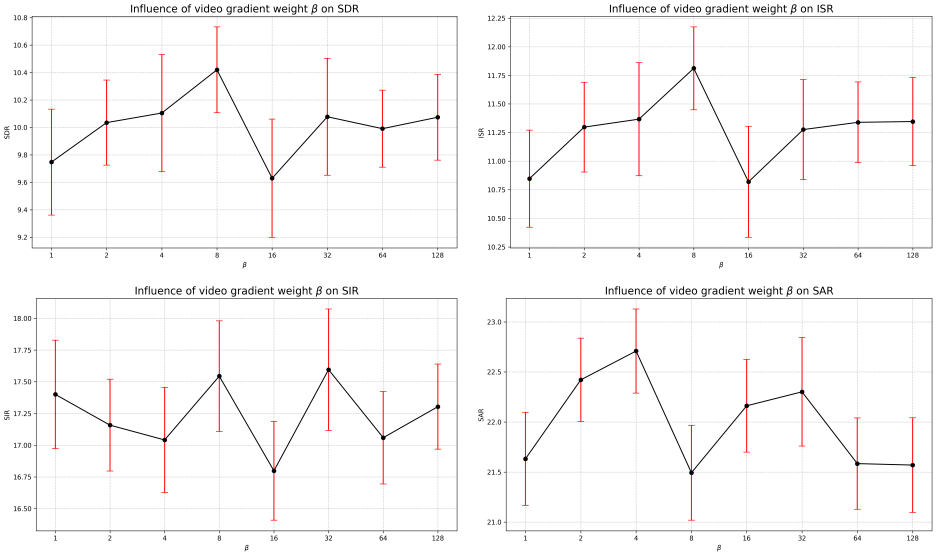


Figure 4.6: Mean and standard error for different video weights β. Each β performance was evaluated using the same 30 samples from the URMP validation set. The same values can be found in Table 4.3.

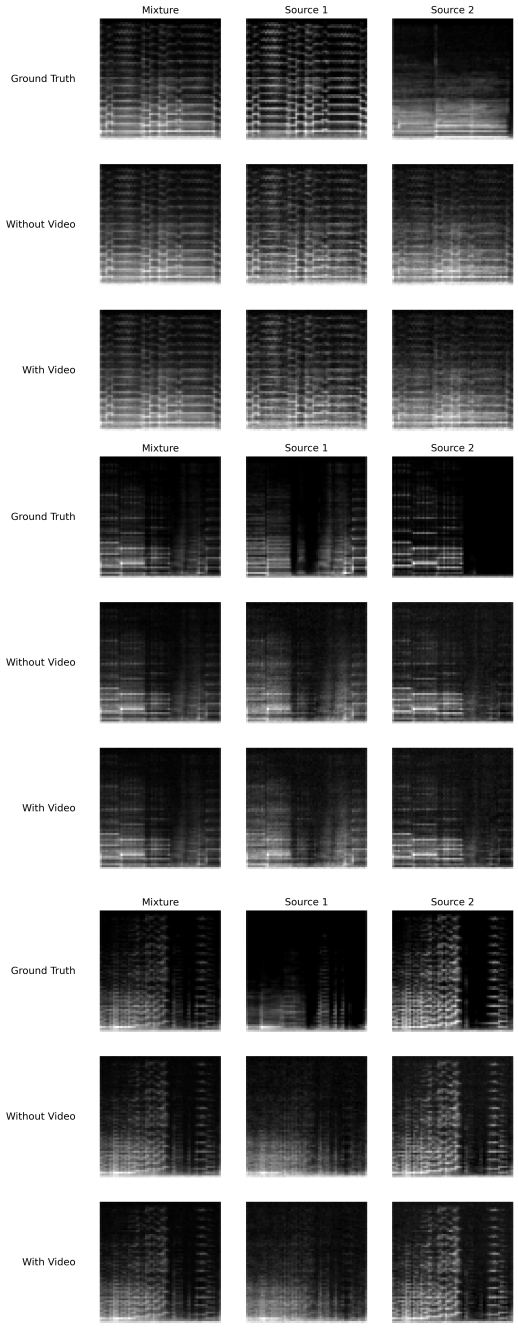


Figure 4.7: BASIS separation attempt of three random samples from the URMP test set.

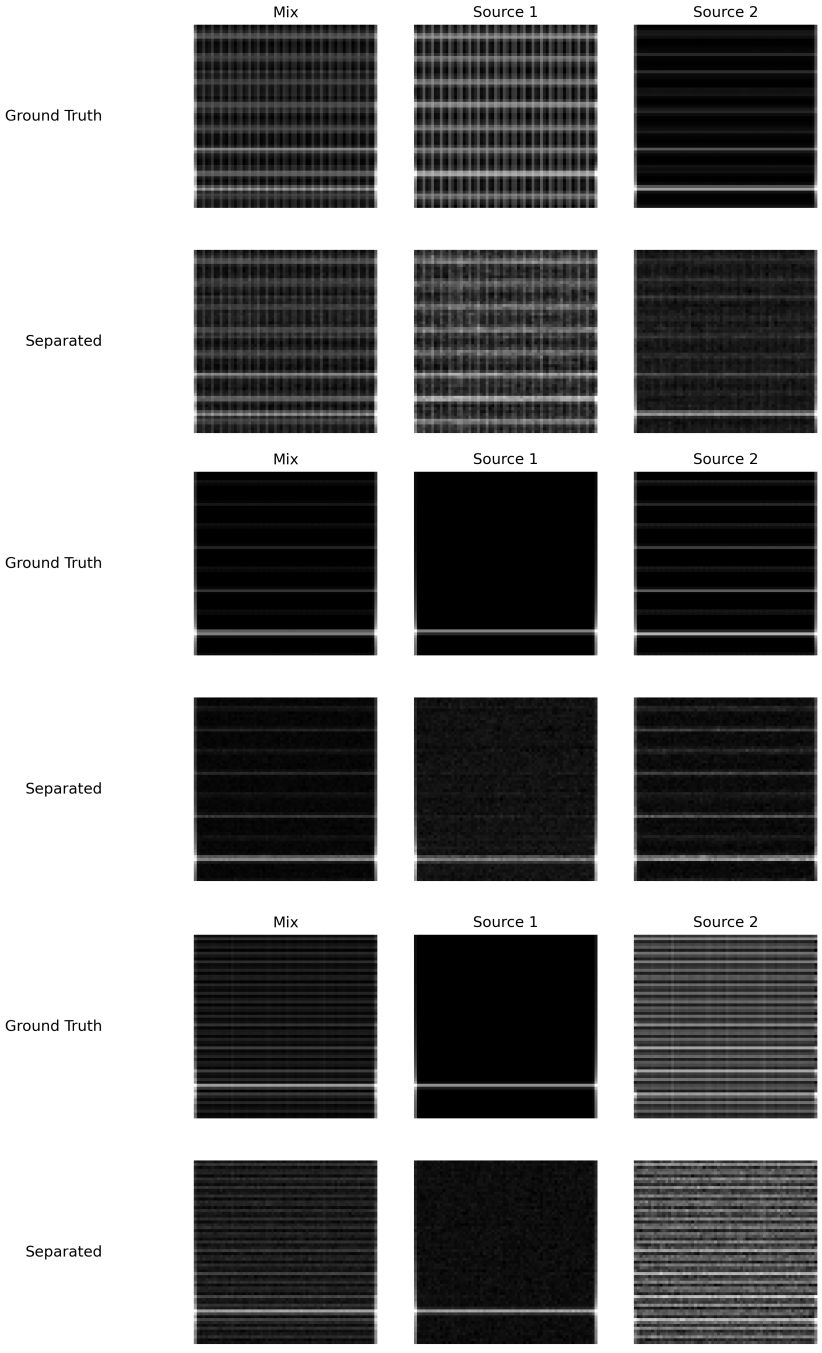


Figure 4.8: BASIS separation attempt of three random samples from toy test set.

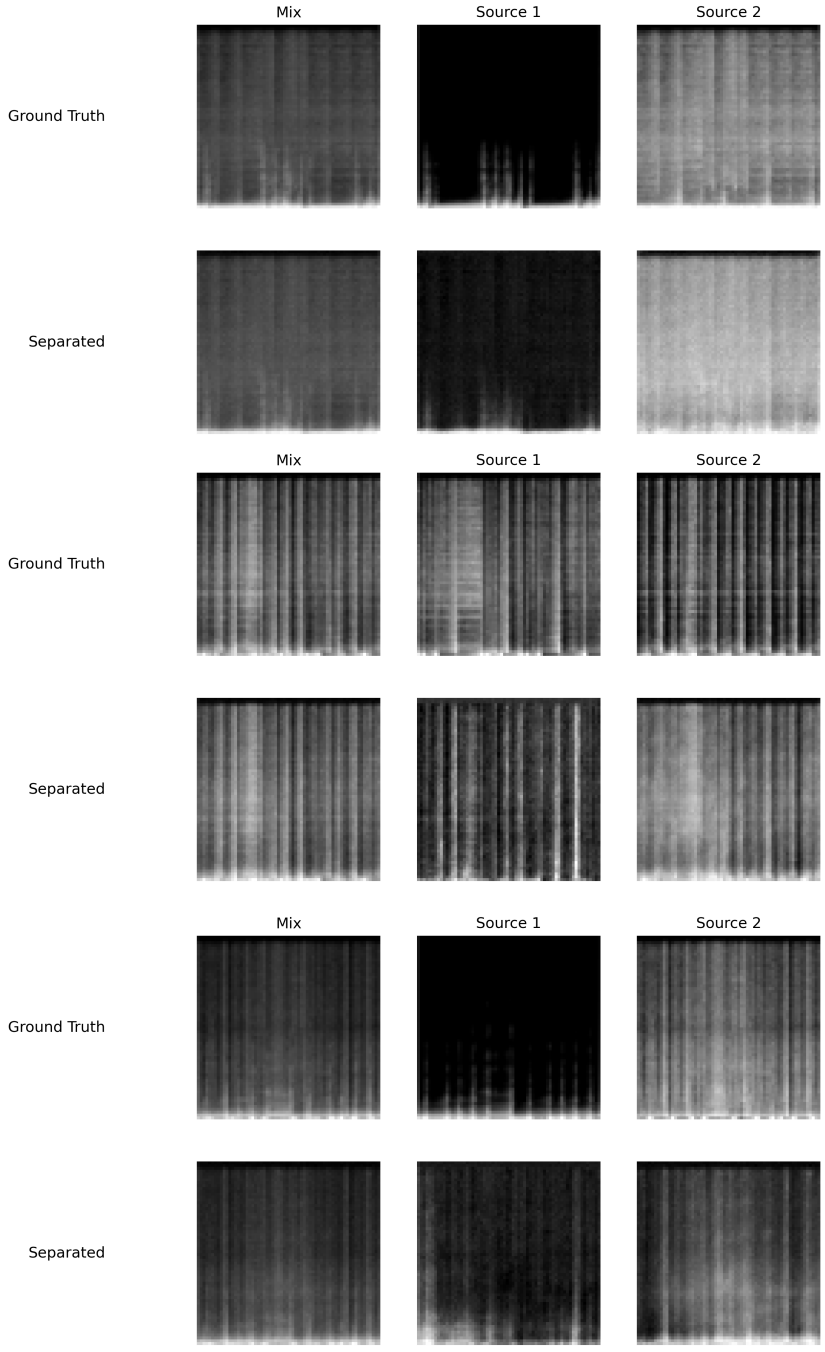


Figure 4.9: BASIS separation attempt of three random samples from MUSDB18 test set.

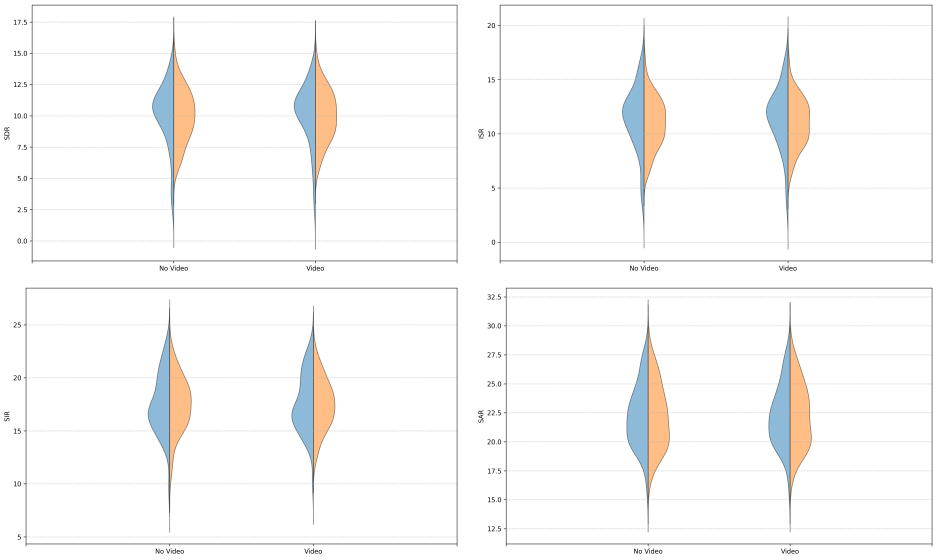
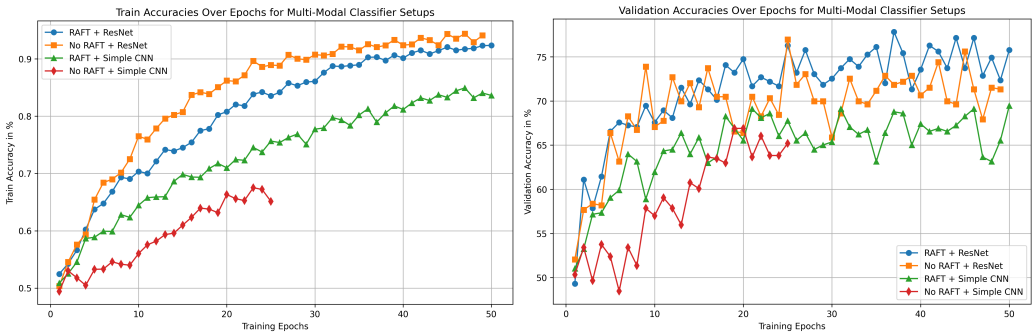


Figure 4.10: Results of Video Separation Experiment, comparing the performance of incorporating video to not incorporating video the URMP test dataset, as described in Section 3.2.4. The blue side denotes Source 1, the orange side Source 2. The means and standard errors can be found in Table 4.4.



(a) Training set.

(b) Validation set.

Figure 4.11: Accuracy of the four tested video configurations during training. Plot (a) shows the accuracy on the train set and (b) shows the accuracy on the validation set.

Chapter 5

Discussion

5.1 Summary of Findings

5.1.1 Uni-Modal Separation Experiment

The results of the Uni-Modal Separation Experiment can be found in Tables 4.1 and 4.2 and as violin plots in Figures Figures 4.1 and 4.2. These results suggest that BASIS performs best with respect to all four metrics on both non-video datasets. AE-BSS achieves a mean SDR of 5.7 dB on the toy data and a slightly lower 5.4 dB on the MUSDB18, thereby coming close to BASIS on the latter and outperforming BASIS Finetuned.

BASIS achieves a significantly higher mean SDR of 17.4 dB on the toy test set compared to 7.2 dB on the MUSDB18 test set. This is to be expected, as the toy dataset is significantly more predictable than MUSDB18 due to the simple rules used to generate it. BASIS with the VAE-priors finetuned on noise-perturbed data (denoted as BASIS Finetuned in the experiments) achieves 14.8 dB and 1.7 dB respectively and is thus consistently outperformed by BASIS using a single model for all noise levels. On the MUSDB18 data, it is additionally outperformed by the baseline consisting of random samples at 3.8 dB. A reason for the sub-par performance with the finetuned models could be that the VAEs may struggle to balance noise information and the actual distribution, thus becoming less well calibrated during finetuning. This could lead to a loss of generality when compared to a single model handling all noise levels. The fact that the mean SDR on the more predictable toy data falls off drastically on the less predictable MUSDB18 data suggests overfitting.

AE-BSS achieves a mean SDR of 5.7 dB on the toy data and a slightly lower 5.4 dB on MUSDB18, thereby coming close to the 7.2 dB achieved by BASIS on the latter and outperforming BASIS Finetuned. Nonetheless, it is also outperformed by the mean SDR of 7.9 dB achieved by the baseline consisting of random samples

from the VAE priors on the toy data. As it can not consistently outperform all baselines, AE-BSS is not suggested to be a competitive architecture for audio source separation. Linear AE-BSS does not manage to achieve a positive average SDR on the toy dataset and thus the same conclusion is drawn.

Figure 4.1 reveals that all methods overlap for all metrics, with the exception of the SAR on the noise baseline. This is even more pronounced in Figure 4.2. This figure also reveals that despite BASIS Finetuned being outperformed by the other methods with respect to the mean SDR, its best attempts extend up to > 30 SDR, a similar maximum to the best-performing BASIS.

5.1.2 Visual Samples

Toy Data

Figure 4.3 permits qualitative analysis of the different approaches on the toy data. BASIS manages to cleanly separate all four samples, albeit with the introduction of slight noise. The blind NMF reveals its limitations and generates an empty source in samples (b) and (d). In samples (a) and (c), it separates the mixture into two intuitively obvious sources, which are nonetheless inaccurate. This reveals the limitation of BSS approaches such as NMF.

AE-BSS assigns a signal with a low fundamental with few overtones to Source 1 and variations of a grid pattern to Source 2 on all four samples. This implies a lack of flexibility, possibly due to the fact that one model was trained on all data instead of training one model per combination of stem types. Linear AE-BSS fails to reconstruct an accurate mixed signal on all four samples, suggesting a fundamentally misguided approach. BASIS Finetuned performs very similar separations to BASIS, but with more noise.

MUSDB18 data

Figure 4.4 provides the same for the MUSDB18 data. The BASIS separations here are less accurate and have more artefacts when compared to the toy data. Interestingly, the NMF separations look more convincing than they do for the toy data. This aligns with the results of the previous experiment. The sources are sometimes switched, as NMF has no information on which source is which. The reconstructed mixture consists of a grid-like texture, which is to be expected given that the reconstruction is a linear combination of k spectral bases.

AE-BSS provides very blurry separations and thus a very blurry reconstruction. The generated signals are obviously not similar enough to the ground truth to be considered a convincing separation. Linear AE-BSS provides an even blurrier separation with a more-or-less entirely white image as Source 2 for all sam-

ples. BASIS Finetuned again provides a similar, but noisier version of the BASIS separation. Despite the additional noise, the separation is overly sharp, especially in samples (a) and (b).

5.1.3 ELBO Confusion Matrices

The confusion matrices in Figure 4.5 provide an estimate of the VAEs' ability to function as priors for the sources. On the toy data, all VAEs assign the highest probability to their in-class data, as expected; that is, for each row, the corresponding column has a higher probability than the others. However, this does not extend to the MUSDB18 data, where the highest probability is assigned to the *bass* data for all models and thus only the *bass* model assigns the highest probability to in-class data. This observation reinforces the conclusion that deep generative priors struggle with high data variability, thus aligning with the findings of Frank and Ilse (2020).

5.1.4 Audio-Visual Matching Classifier Performance

Figure 4.11 indicates that both approaches utilising a ResNet outperform those that do not, on both training and validation data. When using RAFT to pre-process the video and a ResNet18 for the spectrograms, the model achieves a validation accuracy of 77%, narrowly outperforming the architecture using a less sophisticated VAE on the validation data, which achieves 74% on the validation set. Interestingly, the opposite is the case for the training data. This suggests that the architecture not using RAFT — directly taking the video as an input to the 3D ResNet — fits the data more closely but does not generalise as well to unseen data compared to the model which pre-processes the video using RAFT to estimate the optical flow between frames. Additionally, the model using RAFT performs inference quicker, as the input to f_{3D} has 2 instead of 3 channels and a depth of $T - 1$ instead of T. Consequently, all subsequent video experiments are performed using RAFT with a 2D ResNet18 for the spectrograms.

5.1.5 β-Evaluation

Figure 4.6 and Table 4.3 indicate no trend in the SDR for increasing β. Modified BASIS results in `NaN`-values for $\beta \geq 256$.

5.1.6 Video Separation Experiment

Table 4.4 shows that incorporating video into Modified BASIS with $\beta = 128$ leads to a slightly higher mean SDR of 10.07 dB compared to 10.03 dB without video.

This falls within the standard error margin and it is thereby not large enough to definitively conclude a causal relationship between incorporating video into Modified BASIS and better separation. The same narrow margin can be observed for ISR, SAR and SIR. Figure 4.10 paints a more nuanced picture of the SDR distribution of the experiment, featuring asymmetric violin plots. These plot reveal only subtle differences between the two versions.

Without using video, Modified BASIS achieves a higher SDR on the URMP dataset than on the MUSDB18 dataset. This is likely due to the spectral qualities of the featured chamber music instruments having lower in-class variability than the stems in the MUSDB18 pop music.

5.1.7 Additional BASIS Visual Samples

Figure 4.7 shows the Modified BASIS separation of three samples from the URMP test set with and without including video with $\beta = 128$. The separation performed with the inclusion of video seems slightly more accurate for all three samples. Figure 4.9 shows the same for the MUSDB18 dataset, revealing inferior separation performance when compared to the URMP or the toy dataset in Figure 4.8.

5.2 Comparison with Existing Work

5.2.1 Uni-Modal

It has already been established that Modified BASIS outperforms NMF (D. D. Lee and Seung, 1999). Comparing these results to cutting-edge approaches is hard due to the downscaled nature of the data used in this thesis (see Section 3.1). This must be taken into consideration in Table 5.1.

Model	Mean SDR in dB
ResUNetDecouple+ (Kong et al., 2021)	6.73
CWS-PResUNet (H. Liu, Kong, and J. Liu, 2021)	6.77
Modified BASIS (us)	7.2
KUIELab-MDX-Net (Kim et al., 2021)	7.47
Hybrid Demucs (Défossez, 2021)	7.68
BSRNN Luo and J. Yu (2023)	**8.42**

Table 5.1: Comparison of mean SDR values on the MUSDB18 dataset with the Modified BASIS results of the Uni-Modal Separation Experiment from Table 4.2.

This comparison suggests that the mean SDR of 7.2 achieved by Modified

BASIS in the Uni-Modal Separation Experiment is comparable to state-of-the-art approaches. It is likely that further work, as suggested in Section 5.3, would allow for achieving comparable results to BSRNN. Future work is required to evaluate whether these results hold up on full-scale data.

5.2.2 Bi-Modal

Comparing the results of Modified BASIS on the URMP dataset to the results in Table 5.2 requires similar assumptions as the ones outlined in Section 5.2.1. Additionally, these results are achieved on the MUSIC dataset, while our results were achieved on the more obscure URMP dataset.

Model	Mean SDR in dB
Minus-Plus Net (Xu, Dai, and Lin, 2019)	7.0
Sound of Pixels (Zhao, Gan, Rouditchenko, et al., 2018)	7.26
Co-Separation (Gao and Grauman, 2019)	7.64
Sound of Motion (Zhao, Gan, Ma, et al., 2019)	8.2
AVSGS (Chatterjee et al., 2021)	11.4
VGMCL (Islam et al., 2024)	**12.81**

Table 5.2: Comparison of SDR values on the audio-visual MUSIC dataset.

The Modified BASIS mean SDR of 10.07 obtained in the Video Separation Experiment, to be found in Table 4.4, is not directly comparable due to the differing datasets and is thus left out of Table 5.2.

5.3 Limitations and Future Work

Modified BASIS offers numerous opportunities for improvement. Firstly, the scaled-down spectrograms do not allow for transforming the signal back into real space, limiting this thesis to a toy scenario. It acts as a proof of concept for the Modified BASIS approach to audio source separation in time-frequency space.

Due to the downscaled nature of the data, a qualitative analysis is not feasible. In order to allow for such an analysis, as performed by Défossez (2021), one must operate on full-size spectrograms which can be converted to real space using either Algorithm 2 or Equation 2.14 using the stored phase $\mathbf{\Phi}^T$. Operating on full-size spectrograms could potentially exceed the capabilities of a VAE, thus failing to provide sufficiently detailed reconstructions. This issue is easily circumvented, as BASIS allows for using any generative model which can compute the probability of data. Thus, a more capable generative model such as Glow

(Durk P Kingma and Dhariwal, 2018) or NCSN (Song and Ermon, 2019) could be used.

The audio-visual separation approach using Modified BASIS is performed on the relatively obscure URMP dataset (Li et al., 2018). In contrast, the state-of-the-art papers use the MUSIC dataset (Zhao, Gan, Rouditchenko, et al., 2018), which contains more data. Training the χ-model on the MUSIC dataset, in conjunction with the previously mentioned aspect of training on full-size spectrograms, would facilitate a direct comparison with the state-of-the-art approaches.

The fact that the contributions of this thesis utilise monaural signals also presents room for improvement. Both the MUSDB18 and the URMP datasets provide binaural signals, whereby both channels are preprocessed to create a new monoraul signal. Considering the argument presented in Section 1.2 regarding the human approach being a useful heuristic for designing AI systems, it could be interesting to evaluate if a binaural approach to the BASIS algorithm could improve results. Given that CNNs are used on spectrograms, the dataset could easily be preprocessed as a stereo signal, using two input channels instead of one.

The video models were trained on RGB data, as explained in Section 3.1, although single-channel monochrome frames would have probably sufficed. This would have allowed for higher batch sizes and faster computation, potentially without sacrificing performance.

Given the genetically evolved human ability to separate sound sources, it may be interesting to evaluate the performance of a genetic algorithm (GA) on this problem. A possible approach could involve training a VAE on spectrogram data, then learning the weights and architecture of a separate neural network using a GA to convert the latent representation of a mixed source signal into k latent representations of possible mixtures. While the VAE trained on spectrograms could be based on this thesis, the GA-based neural network could be trained using the recently proposed CoDeepNEAT algorithm (Miikkulainen et al., 2024).

Chapter 6

Conclusion

6.1 Summary

In this thesis, deep generative models are utilised to separate the constituent sources from mixed audio signals. Various approaches are explored, leading to the conclusion that the BASIS algorithm, originally proposed by Jayaram and Thickstun (2020), is a viable method for audio source separation by framing the problem as an image separation task. This thesis introduces Modified BASIS, which incorporates video into the separation process in order to potentially benefit from multiple modalities, much as humans do. Modified BASIS performs separation by sampling from the posterior $p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \mid \boldsymbol{m})$ over source signals, given a mixed signal in time-frequency space, using noise-annealed Langevin dynamics.

The obtained results suggest that Modified BASIS can compete with state-of-the-art approaches while utilising computationally inexpensive VAEs. Furthermore, the findings suggest that including video into the separation process could increase the separation performance. To do this, a method is presented using a classifier predicting if a given source separation matches a video, using RAFT (Teed and Deng, 2020) to extract the optical flow from the video and using a concatenated latent space to achieve a classification accuracy of 77% on the validation set. The output is used to compute the gradient of the Bernoulli-distributed log probability of a given separation matching the video, guiding the sampler in a more profitable direction.

This thesis compares Modified BASIS to the blind source separation approach using multi-encoder AEs suggested by Webster and J. Lee (2023) and NMF (D. D. Lee and Seung, 1999), both of which are outperformed with respect to the SDR metric. Despite this, a key observation includes that performance of the Modified BASIS is noticeably dependent on in-class variability.

6.2 Implications

The results of this thesis demonstrate that the integration of deep generative priors provided by VAEs and noise-annealed Langevin dynamics can effectively separate audio sources from mixed signals in time-frequency space. Nonetheless, the performance thereof is highly dependent on the in-class variability of a dataset. Thus, more dynamic approaches must be considered for systems which can effectively be deployed for varied and stable real-world use. An example of a robust model that is potentially more capable to generalise across different data types and conditions is provided by Islam et al. (2024), as described in Section 2.3.2. Nonetheless, it is likely that future work on Modified BASIS could lead to state-of-the-art results.

The incorporation of visual information into the audio source separation process highlights a novel method for integrating multi-modal data into separation tasks. This approach aligns with human auditory scene analysis, which often relies on multiple sensory inputs. The implications extend beyond audio processing, suggesting that multi-modal techniques could be applied to a broader range of tasks where multiple data types are available.

Bibliography

Abdullah, Hadi et al. (2019). "Practical hidden voice attacks against speech and speaker recognition systems". In: *arXiv preprint arXiv:1904.05734*.

Agarap, AF (2018). "Deep Learning Using Rectified Linear Units (ReLU)". In: *arXiv preprint arXiv:1803.08375*.

Akiba, Takuya et al. (2019). "Optuna: A next-generation hyperparameter optimization framework". In: *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631.

Barker, Jon et al. (2015). "The third 'CHiME'speech separation and recognition challenge: Dataset, task and baselines". In: *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, pp. 504–511.

Beauchamp, Michael S, Audrey R Nath, and Siavash Pasalar (2010). "fMRI-guided transcranial magnetic stimulation reveals that the superior temporal sulcus is a cortical locus of the McGurk effect". In: *Journal of Neuroscience* 30.7, pp. 2414–2417.

Bell, Anthony J and Terrence J Sejnowski (1995). "An information-maximization approach to blind separation and blind deconvolution". In: *Neural computation* 7.6, pp. 1129–1159.

Berg, R. E. (May 20, 2024). *sound*. Encyclopedia Britannica. URL: `https://www.britannica.com/science/sound-physics` (visited on 06/19/2024).

Bergstra, James et al. (2011). "Algorithms for hyper-parameter optimization". In: *Advances in neural information processing systems* 24.

Boutsidis, Christos and Efstratios Gallopoulos (2008). "SVD based initialization: A head start for nonnegative matrix factorization". In: *Pattern recognition* 41.4, pp. 1350–1362.

Bregman, Albert S (1994). *Auditory scene analysis: The perceptual organization of sound*. MIT press.

Carreira, Joao and Andrew Zisserman (2017). "Quo vadis, action recognition? a new model and the kinetics dataset". In: *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308.

Chandna, Pritish et al. (2017). "Monoaural audio source separation using deep convolutional neural networks". In: *Latent Variable Analysis and Signal Sep-*

aration: 13th International Conference, LVA/ICA 2017, Grenoble, France, February 21-23, 2017, Proceedings 13. Springer, pp. 258–266.

Chatterjee, Moitreya et al. (2021). "Visual scene graphs for audio source separation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1204–1213.

Cherry, E Colin (1953). "Some experiments on the recognition of speech, with one and with two ears". In: *The Journal of the acoustical society of America* 25.5, pp. 975–979.

Cichocki, Andrzej, Sergio Cruces, and Shun-ichi Amari (2011). "Generalized alpha-beta divergences and their application to robust nonnegative matrix factorization". In: *Entropy* 13.1, pp. 134–170.

Cohen, Leon (1995). *Time-frequency analysis*. Prentice Hall PTR New Jersey.

Comon, Pierre (1994). "Independent component analysis, a new concept?" In: *Signal processing* 36.3, pp. 287–314.

Cooley, James W and John W Tukey (1965). "An algorithm for the machine calculation of complex Fourier series". In: *Mathematics of computation* 19.90, pp. 297–301.

Défossez, Alexandre (2021). "Hybrid spectrogram and waveform source separation". In: *arXiv preprint arXiv:2111.03600*.

Durrieu, Jean-Louis et al. (2009). "Main instrument separation from stereophonic audio signals using a source/filter model". In: *2009 17th European Signal Processing Conference*. IEEE, pp. 15–19.

Embedded Robotics (2020). *Microcontroller Basics*. Accessed: 2024-06-17. URL: https://www.embedded-robotics.com/microcontroller-basics/.

Everest, F Alton (2022). *Master handbook of acoustics*.

Févotte, Cédric, Nancy Bertin, and Jean-Louis Durrieu (2009). "Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis". In: *Neural computation* 21.3, pp. 793–830.

Frank, Maurice and Maximilian Ilse (2020). "Problems using deep generative models for probabilistic audio source separation". In.

Gao, Ruohan and Kristen Grauman (2019). "Co-separating sounds of visual objects". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3879–3888.

Goodfellow, Ian et al. (2014). "Generative adversarial nets". In: *Advances in neural information processing systems* 27.

Griffin, Daniel and Jae Lim (1984). "Signal estimation from modified short-time Fourier transform". In: *IEEE Transactions on acoustics, speech, and signal processing* 32.2, pp. 236–243.

Gupta, Gaurav et al. (2021). "Comparing recurrent convolutional neural networks for large scale bird species classification". In: *Scientific reports* 11.1, p. 17085.

He, Kaiming et al. (2016). “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

He, Tong et al. (2019). “Bag of tricks for image classification with convolutional neural networks”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 558–567.

Hershey, Shawn et al. (2017). “CNN architectures for large-scale audio classification”. In: *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, pp. 131–135.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). “Long short-term memory”. In: *Neural computation* 9.8, pp. 1735–1780.

Huang, Po-Sen et al. (2014). “Singing-Voice Separation from Monaural Recordings using Deep Recurrent Neural Networks.” In: *ISMIR*, pp. 477–482.

Hyvärinen, Aapo and Erkki Oja (1997). “A fast fixed-point algorithm for independent component analysis”. In: *Neural computation* 9.7, pp. 1483–1492.

Ingram, William and Evie Gray (1998). *A Federal Standard on electronic media*. Citeseer.

Islam, Md Amirul et al. (2024). “Visually Guided Audio Source Separation with Meta Consistency Learning”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3014–3023.

Itakura, Fumitada (1968). “Analysis synthesis telephony based on the maximum likelihood method”. In: *Reports of the 6ˆ¡ th¿ Int. Cong. Acoust., 1968*.

Jayaram, Vivek and John Thickstun (2020). “Source separation with deep generative priors”. In: *International Conference on Machine Learning*. PMLR, pp. 4724–4735.

Jutten, Christian and Jeanny Herault (1991). “Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture”. In: *Signal processing* 24.1, pp. 1–10.

Kim, Minseok et al. (2021). “KUIELab-MDX-Net: A two-stream neural network for music demixing”. In: *arXiv preprint arXiv:2111.12203*.

Kingma, Diederik P and Max Welling (2013). “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114*.

Kingma, Durk P and Prafulla Dhariwal (2018). “Glow: Generative flow with invertible 1x1 convolutions”. In: *Advances in neural information processing systems* 31.

Köhler, Wolfgang (1967). “Gestalt psychology”. In: *Psychologische forschung* 31.1, pp. XVIII–XXX.

Kong, Qiuqiang et al. (2021). “Decoupling magnitude and phase estimation with deep resunet for music source separation”. In: *arXiv preprint arXiv:2109.05418*.

Krizhevsky, Alex and Geoffrey Hinton (2009). *Learning Multiple Layers of Features from Tiny Images*. Technical Report TR-2009. University of Toronto.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25.

Kullback, Solomon and Richard A Leibler (1951). "On information and sufficiency". In: *The annals of mathematical statistics* 22.1, pp. 79–86.

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

Lee, Daniel and H Sebastian Seung (2000). "Algorithms for non-negative matrix factorization". In: *Advances in neural information processing systems* 13.

Lee, Daniel D and H Sebastian Seung (1999). "Learning the parts of objects by non-negative matrix factorization". In: *nature* 401.6755, pp. 788–791.

Li, Bochen et al. (2018). "Creating a multitrack classical music performance dataset for multimodal music analysis: Challenges, insights, and applications". In: *IEEE Transactions on Multimedia* 21.2, pp. 522–535.

Linsker, Ralph (1988). "Self-organization in a perceptual network". In: *Computer* 21.3, pp. 105–117.

Liu, Haohe, Qiuqiang Kong, and Jiafeng Liu (2021). "CWS-PResUNet: Music source separation with channel-wise subband phase-aware resunet". In: *arXiv preprint arXiv:2112.04685*.

Luo, Yi and Jianwei Yu (2023). "Music source separation with band-split RNN". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 31, pp. 1893–1901.

McDermott, Josh H (2009). "The cocktail party problem". In: *Current Biology* 19.22, R1024–R1027.

McFee, Brian et al. (2015). "librosa: Audio and music signal analysis in python." In: *SciPy*, pp. 18–24.

McGurk, Harry and John MacDonald (1976). "Hearing lips and seeing voices". In: *Nature* 264.5588, pp. 746–748.

Miikkulainen, Risto et al. (2024). "Evolving deep neural networks". In: *Artificial intelligence in the age of neural networks and brain computing*. Elsevier, pp. 269–287.

Moon, Todd K (1996). "The expectation-maximization algorithm". In: *IEEE Signal processing magazine* 13.6, pp. 47–60.

Musicant, Alan D and Robert A Butler (1985). "Influence of monaural spectral cues on binaural localization". In: *The Journal of the Acoustical Society of America* 77.1, pp. 202–208.

My New Microphone (2019). *How Do Microphones Work? A Helpful Illustrated Guide*. Accessed: 2024-06-17. URL: https://mynewmicrophone.com/how-do-microphones-work-a-helpful-illustrated-guide/.

Ngiam, Jiquan et al. (2011). "Multimodal deep learning". In: *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 689–696.

Nugraha, Aditya Arie, Antoine Liutkus, and Emmanuel Vincent (2016). "Multichannel audio source separation with deep neural networks". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24.9, pp. 1652–1664.

Nyquist, Harry (1928). "Certain topics in telegraph transmission theory". In: *Transactions of the American Institute of Electrical Engineers* 47.2, pp. 617–644.

Oppenheim, Alan V (1999). *Discrete-time signal processing*. Pearson Education India.

Palanisamy, Kamalesh, Dipika Singhania, and Angela Yao (2020). "Rethinking CNN models for audio classification". In: *arXiv preprint arXiv:2007.11154*.

Paszke, Adam et al. (2017). *Automatic differentiation in pytorch*.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830.

Raffel, Colin et al. (2014). "MIR_EVAL: A Transparent Implementation of Common MIR Metrics." In: *ISMIR*. Vol. 10, p. 2014.

Rafii, Zafar et al. (Dec. 2017). *The MUSDB18 corpus for music separation*. DOI: 10.5281/zenodo.1117372. URL: https://doi.org/10.5281/zenodo.1117372.

Rezende, Edmar et al. (2017). "Malicious software classification using transfer learning of resnet-50 deep neural network". In: *2017 16th IEEE international conference on machine learning and applications (ICMLA)*. IEEE, pp. 1011–1014.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-net: Convolutional networks for biomedical image segmentation". In: *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, pp. 234–241.

Rouger, Julien et al. (2007). "Evidence that cochlear-implanted deaf patients are better multisensory integrators". In: *Proceedings of the National Academy of Sciences* 104.17, pp. 7295–7300.

Sams, Mikko et al. (1991). "Seeing speech: visual information from lip movements modifies activity in the human auditory cortex". In: *Neuroscience letters* 127.1, pp. 141–145.

Sarwinda, Devvi et al. (2021). "Deep learning in image classification using residual network (ResNet) variants for detection of colorectal cancer". In: *Procedia Computer Science* 179, pp. 423–431.

Sawa, Teiji, Tomomi Yamada, and Yurie Obata (2022). "Power spectrum and spectrogram of EEG analysis during general anesthesia: Python-based computer programming analysis". In: *Journal of clinical monitoring and computing*, pp. 1–13.

Shi, Yuge, Brooks Paige, Philip Torr, et al. (2019). "Variational mixture-of-experts autoencoders for multi-modal deep generative models". In: *Advances in neural information processing systems* 32.

Simonyan, Karen and Andrew Zisserman (2014). "Two-stream convolutional networks for action recognition in videos". In: *Advances in neural information processing systems* 27.

Slater, Mel and Sylvia Wilbur (1997). "A framework for immersive virtual environments (FIVE): Speculations on the role of presence in virtual environments". In: *Presence: Teleoperators & Virtual Environments* 6.6, pp. 603–616.

Smaragdis, Paris (2004). "Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs". In: *Independent Component Analysis and Blind Signal Separation: Fifth International Conference, ICA 2004, Granada, Spain, September 22-24, 2004. Proceedings 5*. Springer, pp. 494–499.

So, Stephen (2008). "Why is the sample variance a biased estimator". In: *Griffith University, Tech. Rep.* 9.

Song, Yang and Stefano Ermon (2019). "Generative modeling by estimating gradients of the data distribution". In: *Advances in neural information processing systems* 32.

Song, Yang, Sahaj Garg, et al. (2019). "Sliced Score Matching: A Scalable Approach to Density and Score Estimation". In: *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence, UAI 2019, Tel Aviv, Israel, July 22-25, 2019*, p. 204. URL: http://auai.org/uai2019/proceedings/papers/204.pdf.

Stoller, Daniel, Sebastian Ewert, and Simon Dixon (2018). "Wave-u-net: A multi-scale neural network for end-to-end audio source separation". In: *arXiv preprint arXiv:1806.03185*.

Teed, Zachary and Jia Deng (2020). "Raft: Recurrent all-pairs field transforms for optical flow". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, pp. 402–419.

Tran, Du et al. (2015). "Learning spatiotemporal features with 3d convolutional networks". In: *Proceedings of the IEEE international conference on computer vision*, pp. 4489–4497.

Uhlich, Stefan, Franck Giron, and Yuki Mitsufuji (2015). "Deep neural network based instrument extraction from music". In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 2135–2139.

Van Den Oord, Aaron et al. (2016). "Wavenet: A generative model for raw audio". In: *arXiv preprint arXiv:1609.03499* 12.

Van Wassenhove, Virginie, Ken W Grant, and David Poeppel (2007). "Temporal window of integration in auditory-visual speech perception". In: *Neuropsychologia* 45.3, pp. 598–607.

Vaswani, A (2017). "Attention is all you need". In: *Advances in Neural Information Processing Systems*.

Vincent, Emmanuel, Rémi Gribonval, and Cédric Févotte (2006). "Performance measurement in blind audio source separation". In: *IEEE transactions on audio, speech, and language processing* 14.4, pp. 1462–1469.

Virtanen, Pauli et al. (2020). "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17, pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

Wasserman, L (2013). *All of Statistics: a Concise Course in Statistical Inference*.

Webster, Matthew B and Joonnyong Lee (2023). "Self-Supervised Blind Source Separation via Multi-Encoder Autoencoders". In: *arXiv preprint arXiv:2309.07138*.

Welling, Max and Yee W Teh (2011). "Bayesian learning via stochastic gradient Langevin dynamics". In: *Proceedings of the 28th international conference on machine learning (ICML-11)*. Citeseer, pp. 681–688.

Williams, Christopher et al. (2024). "A unified framework for U-Net design and analysis". In: *Advances in Neural Information Processing Systems* 36.

Wong, Bang (2010). "Points of view: Gestalt principles (Part 1)". In: *nature methods* 7.11, p. 863.

Xia, Kai-jian, Hong-sheng Yin, and Yu-dong Zhang (2019). "Deep semantic segmentation of kidney and space-occupying lesion area based on SCNN and ResNet models combined with SIFT-flow algorithm". In: *Journal of medical systems* 43.1, p. 2.

Xu, Xudong, Bo Dai, and Dahua Lin (2019). "Recursive visual sound separation using minus-plus net". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 882–891.

Yang, Zhirong et al. (2011). "Kullback-Leibler divergence for nonnegative matrix factorization". In: *International Conference on Artificial Neural Networks*. Springer, pp. 250–257.

Yu, Fisher et al. (2015). "LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop". In: *arXiv preprint arXiv:1506.03365*.

Zhao, Hang, Chuang Gan, Wei-Chiu Ma, et al. (2019). "The sound of motions". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1735–1744.

Zhao, Hang, Chuang Gan, Andrew Rouditchenko, et al. (2018). "The sound of pixels". In: *Proceedings of the European conference on computer vision (ECCV)*, pp. 570–586.

Zou, Hui, Trevor Hastie, and Robert Tibshirani (2006). "Sparse principal component analysis". In: *Journal of computational and graphical statistics* 15.2, pp. 265–286.