Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

Aleksander Johnsen Solberg

Musical Source Separation using Diffusion Models

Master's thesis in Applied Physics and Mathematics
Supervisor: Erlend Aune
Co-supervisor: Daesoo Lee
June 2023

NTNU
Norwegian University of
Science and Technology

Aleksander Johnsen Solberg

Musical Source Separation using Diffusion Models

Master's thesis in Applied Physics and Mathematics
Supervisor: Erlend Aune
Co-supervisor: Daesoo Lee
June 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

NTNU
Norwegian University of
Science and Technology

Abstract

Musical Source Separation (MSS) is a challenging task in audio signal processing that aims to extract individual sources from a musical mixture, such as separating vocals from background music. In recent years, diffusion models have emerged as a powerful class of generative models that have shown remarkable performance in various domains. This thesis explores the application of diffusion models to MSS and investigates their advantages and limitations compared to previous models.

The primary objective of this study is to develop and evaluate diffusion models for MSS using three different approaches. The first approach involves applying a standard diffusion process to generate the spectrograms of the separated sources. The second approach is similar to the first, but it employs a latent diffusion process. The third approach uses a latent diffusion process to generate a Time-Frequency (TF) mask which is then applied to the mixture audio. These models are compared to a non-diffusion-based model for evaluation. Evaluation is done using objective performance measures in addition to a subjective evaluation.

The results show that the trained diffusion models were not able to surpass the simpler model in this task in terms of audio quality. However, the advantage of directly generating the spectrograms of the separated sources rather than relying on mask-based approaches is evident. The direct approach outperforms the mask-based approach in terms of suppressing the other sources in the estimated audio and could therefore hold promise for future advancements in MSS.

Sammendrag

Musikalsk kildeseparering er en utfordrende oppgave innen lydsignalbehandling som har som mål å ekstrahere individuelle kilder fra en musikalsk blanding, for eksempel å separere vokal fra bakgrunnsmusikk. I de siste årene har diffusjonsmodeller dukket opp som en kraftig klasse av generative modeller med bemerkelsesverdig ytelse på ulike områder. Denne oppgaven utforsker anvendelsen av diffusjonsmodeller til separering av musikalske kilder og undersøker deres fordeler og svakheter i forhold til tidligere modeller.

Det primære målet med denne studien er å utvikle og evaluere diffusjonsmodeller for musikalsk kildesepararering ved hjelp av tre ulike tilnærminger. Den første tilnærmingen benytter en standard diffusjonsprosess for å generere spektrogrammene til de separate kildene. Den andre tilnærmingen er lignende den første, men bruker en latent diffusjonsprosess. Den tredje tilnærmingen bruker en latent diffusjonsprosess for å generere en tid-frekvens maske som deretter påføres den musikalske blandingen. Disse modellene sammenlignes med en modell som ikke er basert på diffusjon for evaluering. Evalueringen gjennomføres ved bruk av objektive målinger i tillegg til en subjektiv vurdering.

Resultatene viser at de trente diffusjonsmodellene ikke klarte å overgå den enklere modellen i denne oppgaven når det gjelder lydkvalitet. Imidlertid er fordelen med direkte generering av spektrogrammene til de separerte kildene fremfor å basere seg på maskeringsmetoder tydelig. Den direkte tilnærmingen viser bedre eliminering av de andre kildene i den estimerte lyden og kan derfor være lovende for fremtidige fremskritt innen musikalsk kildeseparering.

Preface

This thesis marks the conclusion of my seven-year-long journey as a student at the Norwegian University of Science and Technology (NTNU). During this time, I have completed a bachelor's degree in Music Technology, and with the completion of this thesis will also conclude my master's degree in Industrial Mathematics.

I would like to thank my supervisor Erlend Aune and my co-supervisor Daesoo Lee for their valuable guidance during the writing of this thesis. I also want to thank my family for their help and emotional support throughout the past thirteen months. Lastly, thanks to all the friends I made along the way and to everyone who has made the past seven years so enjoyable.

Aleksander Johnsen Solberg
Trondheim, 28th June, 2023

Contents

List of Figures

List of Tables

Acronyms

AoAAN absence of additional artificial noise. viii, 15, 31, 32

CNN Convolutional Neural Network. iv, 1, 2, 7, 8, 13, 17, 18

COLA Constant Overlap-Add. 7, 34

DDPM Denoising Diffusion Probabilistic Model. 9, 12, 18

DNN Deep Neural Network. v, 1, 2, 17, 18

EMA Exponential Moving Average. 25, 26

FFNN Feedforward Fully-connected Neural Network. 7, 17

GC global quality. viii, 15, 31, 32

GELU Gaussian Error Linear Unit. 23, 24

ISTFT Inverse Short-Time Fourier Transform. vi, 6, 26

MIDI Musical Instrument Digital Interface. 21

MIR Music Information Retrieval. 1

ML Machine Learning. 1, 25

MLDM Mask Latent Diffusion Model. v, 25–28, 30–33

MSS Musical Source Separation. i, v, vi, 1–3, 5, 7–9, 14, 17–19, 21, 33, 35, 36

PoTS perservation of target source. viii, 15, 31, 32

RNN Recurrent Neural Network. 1, 7, 17

RQ Research Question. 2, 35

SI-SAR Scale-Invariant Source-To-Artefacts Ratio. 14, 15, 27, 30

SI-SDR Scale-Invariant Source-To-Distortion Ratio. 14, 15, 27, 30

SI-SIR Scale-Invariant Source-To-Interferences Ratio. 14, 15, 27–31, 33

SLDM Spectrogram Latent Diffusion Model. v, 25–27, 30, 31, 33

SNR Source-To-Noise Ratio. 14, 27, 30

SoOS suppression of other sources. viii, 15, 31–33

SSDM Spectrogram Standard Diffusion Model. v, 24, 26–31, 33

STFT Short-Time Fourier Transform. vi, 6, 22, 27, 33, 34, 36

TF Time-Frequency. i, 5–7, 17, 19, 22, 35

VAE Variational Autoencoder. 24, 25

Chapter 1

Introduction

1.1 Motivation

Musical Source Separation (MSS), the task of extracting individual audio sources from a musical mixture, remains a challenging problem in the field of audio signal processing. This technique has numerous applications, such as removing background noise from a musical recording, creating karaoke tracks, or remixing a song by isolating individual parts of the recording. Furthermore, MSS can be utilized as a pre-processing step in several Music Information Retrieval (MIR) tasks such as automatic musical transcription (Demirel et al., 2020). However, the task's inherent complexity arises from the overlapping and entangled nature of the audio sources in the time-frequency domain. This is further complicated by the fact that real-life musical recordings can have different instrumentation, recording conditions and mixing styles, making generalization extremely challenging. Nevertheless, musical audio sources possess inherent structures and patterns that can be of use in this task, such as harmonic structures, frequency contours and temporal patterns (Cano et al., 2019).

In recent years, developments in Machine Learning (ML) technology have led to a large boost in the performance of MSS models, utilizing Deep Neural Network (DNN) architectures such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to learn a mapping between the audio mixture and the separated sources (Rafii et al., 2018). Although these approaches have produced impressive results, the research within the field of MSS is still progressing to achieve even better performance and improved generalization in real-world applications. Thus, there is a need for advanced techniques that can overcome these challenges and improve the performance of MSS models, which is the motivation for this thesis.

1.2 Problem Statement

The problem addressed in this research is to develop and evaluate how diffusion models may be used for Musical Source Separation. Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) are a class of generative models that have gained attention in various domains over the last few years. They have been utilized for a wide variety of tasks, including image generation, natural language processing and audio generation (Yang et al., 2023). This research aims to apply this powerful new method to the task of MSS and explore its advantages and limitations. We also compare the diffusion model's performance with that of the existing methods in the field of MSS, specifically, a mask-based UNet model.

To concretize the objectives of this thesis, we formulate a set of Research Questions (RQs) as follows.

RQ1 *To what extent can diffusion models surpass the performance of existing models such as CNNs and DNNs in the domain of Musical Source Separation?*

Previous models used for MSS, such as CNNs and DNNs, have shown excellent results. This research question seeks to explore whether diffusion models can improve the performance of these previous models. By comparing the results of diffusion models with the existing models, we can assess their potential as a more effective and precise solution for MSS. Having seen the impressive results that diffusion models have shown in many fields, the hope is that they can also lead to advancements in MSS. Specifically, the hope is that by using this more complex technology, the diffusion models are able to capture the more fine-grained information of the audio and use it to obtain more accurate estimates of the separated sources.

RQ2 *What are the advantages and limitations of directly generating the separated sources compared to filtering the mixture?*

Traditionally, source separation methods involve generating a mask that is then applied to the mixture, filtering the audio to obtain the separated sources (see section 2.1.2). As diffusion models are generative, the question arises of whether it is possible to directly generate the separated source directly, rather than generating a mask, and whether this leads to better performance compared to the masking approach. The mask-based approach has its inherent limitations in that it often is unable to suppress the unwanted sources completely. The ambition is that by using the powerful generative abilities of diffusion models, we can eliminate the need for this intermediate step and thus eliminate interference.

RQ3 *What are the advantages and limitations of using latent diffusion models compared to standard diffusion models for Musical Source Separation?*

Latent diffusion models (Rombach et al., 2022) introduce a lower-dimensional latent space in which the diffusion process occurs (see section 2.3.5). This research question aims to compare the performance and characteristics of latent diffusion models with standard diffusion models for MSS. This comparison can shed light on the trade-offs between computational complexity, training requirements, and source separation performance between these two approaches. The hope in regard to this is that both approaches are able to complete the task and that the latent approach can provide a less computationally expensive alternative to the standard diffusion process without sacrificing significant performance.

These research questions will be revisited in chapter 6, summarizing the findings of the research in terms of how these questions have been answered.

1.3 Outline

All code used in this project, the produced models, as well as a selection of audio examples, can be found on GitHub[1]. The remainder of this thesis is organized as follows:

- Chapter 2 presents the background information and theory relevant to the research. It discusses the fundamental concepts and techniques in MSS and introduces the theory behind diffusion models. The section also presents the methods used for evaluating the performance of the models.
- Chapter 3 reviews the existing literature and research work related to MSS and explores existing research on the application of diffusion models to tasks similar to MSS.
- Chapter 4 describes the experimental setup employed in this research, including details about the dataset used for training, as well as the implementation of the diffusion models and their training setup.
- Chapter 5 presents the experimental results, analyzes the findings, and discusses the performance of diffusion models. It also compares the diffusion models to each other and to the non-diffusion-based model.
- Chapter 6 summarizes the key findings of this thesis. It reflects on the research questions posed in the beginning and discusses how they have been addressed through the experimental evaluations. It also suggests future research directions.

[1] `https://github.com/aleksolberg/MSS-diffusion`

Chapter 2

Background

In this chapter, we introduce the concepts underlying our approach of using diffusion models for MSS. We begin by discussing the representation of audio signals, with a focus on the Time-Frequency (TF) representation and its benefits for source separation tasks. Next, we delve into deep learning techniques commonly employed in MSS. Subsequently, we introduce diffusion models and explain the mathematical foundations as well as how they can be used for MSS through conditioning on the mixture audio. We also go into Latent Diffusion Models where the diffusion process is done in a latent space in order to increase efficiency. Finally, we introduce the evaluation metrics used to assess the performance of the trained models.

2.1 Representation of Audio Signals

Digital audio signals are typically represented as waveforms, which are sequences of discrete samples capturing the audio signal's amplitude at regular time steps. A visualization of a waveform can be seen in figure 2.1. The rate at which these samples are captured is called the *sampling rate* of the signal and is for most recordings at 44.1 kHz. The sampling rate largely determines the quality of the recording, limiting the range of frequencies the recording is able to represent. A waveform cannot represent frequencies higher than the Nyquist frequency, which is half of its sampling rate (Landau, 1967). Thus, a recording with a sampling rate of 44.1 kHz is able to represent frequencies up to 22.05 kHz, which is higher than what is considered audible for humans.

While the waveform representation provides a direct and intuitive representation of audio signals, it may not be optimal for certain audio processing tasks, such as source separation. One can easily interpret the changes in the volume of the audio by studying the waveform but will have difficulties identifying the spectral and harmonic information of the audio, which is an important component of source separation tasks. This motivates the use of an alternative representation: the TF representation.

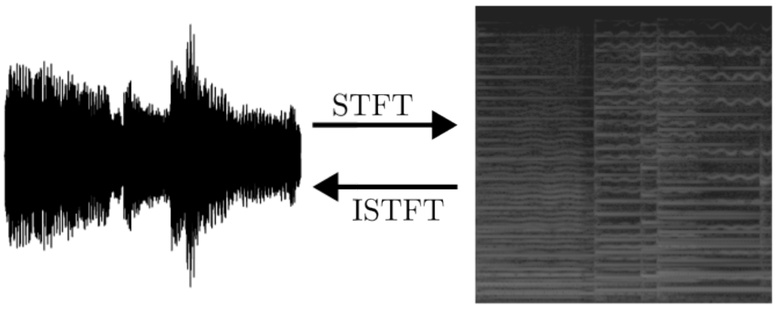


Figure 2.1: A visualization of the two representations of the same audio, a waveform on the left, and a magnitude spectrogram on the right. The waveform represents the audio as the amplitude of the soundwave at each sample, while the spectrogram represents it as the amplitude of each frequency at each time, with the lower frequencies at the bottom and the higher frequencies at the top. The lighter parts of the spectrogram indicate higher energy and the darker parts indicate lower energy. The two representations are converted to the other by an STFT and an ISTFT respectively.

2.1.1 Time-Frequency representation

A Time-Frequency (TF) representation is a matrix that represents the audio in the time-frequency domain. The most commonly used representation is the magnitude spectrogram, as illustrated in figure 2.1. Magnitude spectrograms are usually visualized as a heatmap showing the amplitude of each frequency at each time in the waveform (Rafii et al., 2018). They are produced by taking a Short-Time Fourier Transform (STFT) of the waveform, which is done by dividing the waveform into small, overlapping windows and taking a discrete Fourier transform of each window. This results in a complex-valued matrix Y, which can be interpreted as a spectrogram with time τ along its x-axis and frequency f along its y-axis.

We denote the amplitude of the waveform at timestep n as $x[n]$ and the window of length M as $w[n]$. Using a hopsize H determining the shift of the window in the time dimension, the STFT is calculated as

$$Y(f,\tau) = \mathrm{STFT}\{x[n]\}(f,\tau) = \sum_{n=0}^{M-1} x[n+\tau H]w[n]e^{-\frac{2\pi i f n}{M}}. \tag{2.1}$$

Since the values of the STFT are complex, we obtain the magnitude spectrogram by taking the absolute value of each point in the matrix, eliminating the phase information of the audio. The size of the produced spectrogram, $D_\tau \times D_f$, and thus the resolution of the representation in the time and frequency dimensions, is determined by the window length M and the hopsize H, where the number of time bins $D_\tau = \frac{N-M}{H}$ and the number of frequency bins $D_f = \lfloor \frac{M}{2} \rfloor + 1$. We can also convert a magnitude spectrogram back to a waveform representation by adding the phase back in and performing an Inverse Short-Time Fourier Transform (ISTFT). The ISTFT is defined as

$$x[n] = \mathrm{STFT}^{-1}\{Y(f,\tau)\}[n] = \frac{1}{Mw[n+\tau H]} \sum_{f=0}^{M-1} Y(f,\tau)e^{\frac{2\pi i f n}{M}}. \tag{2.2}$$

The parameters of the STFT, such as the shape of the window $w[n]$, the window length M and the hopsize H, will have an effect on whether the process of converting a waveform into a spectrogram and back again will reproduce the original waveform perfectly. Parameters that mathematically allow for this are called Constant Overlap-Add (COLA) parameters (Zhivomirov et al., 2019).

2.1.2 Spectrogram Masking

A common approach to source separation using the TF representation is filtering the mixed audio's spectrogram to eliminate the unwanted audio source. In most cases, this involves creating a TF mask, which is a matrix of the same shape as the spectrogram. Masks can be either binary, meaning they only contain values of 0 or 1, or soft, where they can take any take any values between 0 and 1. The mask $M \in \mathbb{R}^{D_\tau \times D_f}$ is applied to the mixture spectrogram $Y \in \mathbb{R}^{D_\tau \times D_f}$ by element-wise multiplication, resulting in a new spectrogram S representing the estimated source audio:

$$S = Y \odot M.$$

Thus, the parts of the mix where the mask contains low values will be attenuated, while the parts with high values will be kept intact. This operation is visualized in figure 2.2. In MSS, the aim is then to create such masks where only the values originating from the target source are kept. An ideal mask, representing the best possible performance for a mask-based approach, can be calculated by element-wise division of the source spectrogram by the mixture spectrogram.

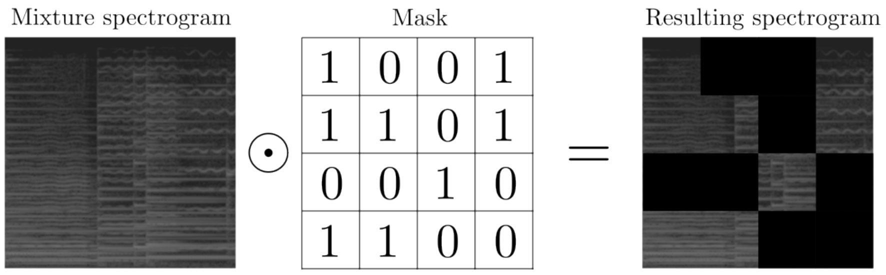


Figure 2.2: A visualization of the masking operation, using a simplified example for demonstration purposes. The mask in the example has large blocks of zeroes and ones, eliminating large portions of the mixture spectrogram, while completely retaining the others. In a real case MSS scenario, the mask would be much more fine-grained and contain values between 0 and 1.

2.2 Deep Learning in Source Separation

Conventionally, source separation models have exploited specific knowledge about the sources in order to achieve separation, for example, the assumption of harmonicity or repetitiveness (Rafii et al., 2018). In recent years, deep learning techniques have revolutionized the field of source separation. These techniques rather take advantage of large datasets of songs containing both the mixture and the separated sources and let the model be learned from these through supervised learning. Many different architectures have been proposed, such as Feedforward Fully-connected Neural Networks (FFNNs), Recurrent Neural Networks (RNNs), or Convolutional Neural Networks (CNNs) (Rafii et al., 2018).

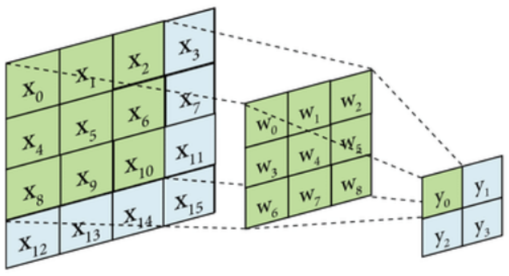


Figure 2.3: An example of one step in the convolution operation. The output is produced by taking the dot product of the input matrix and the kernel at each position in the input. The weights of the kernel w_i are learned through training of the neural network. The figure is taken from Mosser et al. (2017)

2.2.1 Convolutional Neural Networks

In this project, a CNN architecture will be utilized in the implementation of diffusion models. The same architecture will also be used in the simpler model, allowing for a comparison between the diffusion-based approach and previously applied methods. CNNs have gained widespread adoption in the field of musical source separation due to their ability to capture intricate spectral and temporal patterns within the spectrogram. These networks leverage convolutional layers, where a convolution operation is performed by sliding a kernel or filter over the input matrix and computing the dot product between the kernel and the input at each position. One such step is shown in figure 2.3. Typically, the kernel size is relatively small, such as 3x3 or 5x5 points. The size of the output of this operation depends on its parameters, such as whether the input matrix is padded at the borders, the size of the kernel, and the strides the kernel takes at each step. For example, a convolutional operation with strides of 2, meaning the kernel moves two positions for each step will produce a matrix half the size of its input. The opposite operation, where the output is larger than the input, is called a transposed convolution. This is done by inserting zeros in between the points in the input and performing a convolutional operation in order to produce a larger output.

By utilizing multiple convolutional layers, CNNs progressively extract and discern intricate patterns and features from the input data. This behaviour proves advantageous in our context as it allows the network to capture the harmonic structure of distinct sources in the spectrograms. These structures can then be recognized as the different sources in the mixture spectrogram and allow for the separation of the sources.

A specific type of CNN is the UNet, originally designed for use in biomedical image segmentation (Ronneberger et al., 2015). The overall structure of the UNet from the original paper can be seen in figure 2.4. UNets have been successfully used in MSS research and have become an often used approach to the problem (Jansson et al., 2017; Stoller et al., 2018; Kadandale et al., 2020). The architecture is composed of an encoder and a decoder part, forming a U-shape. In the encoder, several convolutional layers are applied to the input, downsampling it into a lower-dimensional representation. At each layer, the dimensions of the representation are halved, and the number of channels is increased, leading to a small representation of the input at the centre. The decoder part then applies

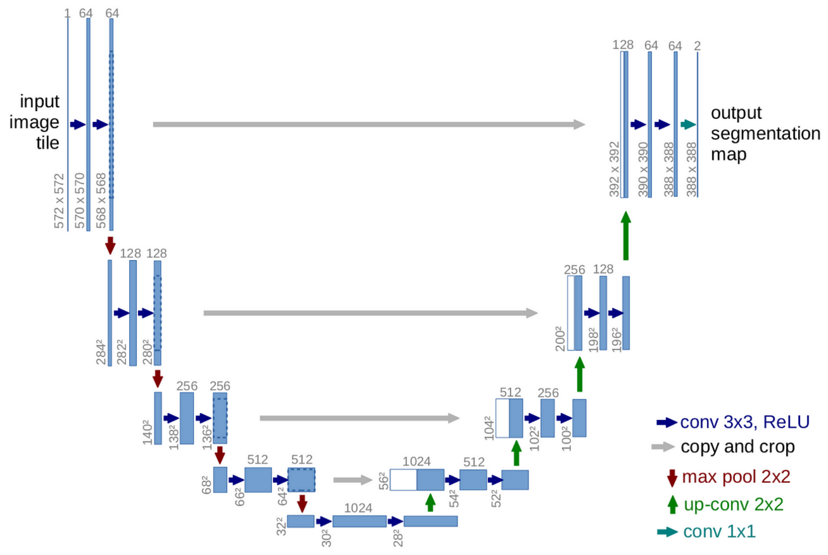
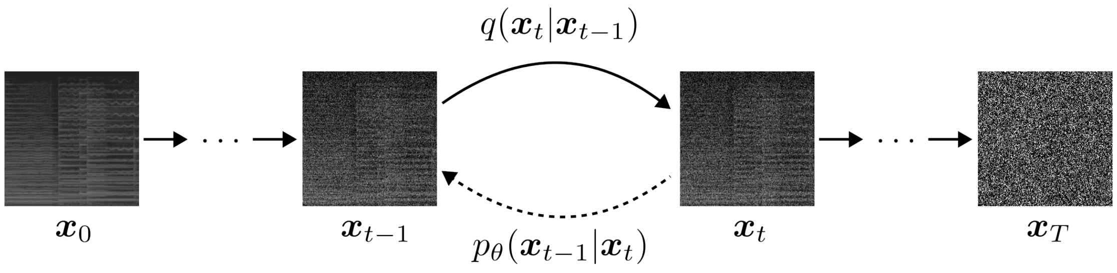


Figure 2.4: The UNet architecture used in the original paper. The architecture uses an encoder-decoder structure using several layers of convolutional operations to form a U-shape. The figure is taken from Ronneberger et al. (2015).

transposed convolution operations, eventually restoring the original dimensionality. Here, each layer applies transposed convolutional layers to increase the size of the representation and decrease the number of channels. Additionally, the architecture uses skip connections between the corresponding layers, where the output of one layer in the encoder is concatenated to the input of the corresponding layer in the decoder. This facilitates the fusion of low-level and high-level feature extraction. The U-shaped architecture enables the UNet to capture both local and global dependencies in the input spectrogram, making it suitable for source separation tasks.

2.3 Diffusion Models

Inspired by non-equilibrium thermodynamics, diffusion models are a type of generative model that aims to generate new data from noise. This type of model was originally proposed for use in high-quality image synthesis by Sohl-Dickstein et al. (2015) and was further developed by Ho et al. (2020) as Denoising Diffusion Probabilistic Models (DDPMs). Diffusion models have shown impressive results within fields such as image synthesis, computer vision, natural language processing and temporal data modelling, to name a few (Yang et al., 2023). They are characterized by a forward process in which data is systematically destroyed by slowly adding Gaussian noise, and a backward process in which the noise is removed to restore the data structure. In this way, new data can be generated by starting from pure Gaussian noise and applying the backward process. In this section, we formulate the theoretical background of these processes and introduce how we can utilize them for MSS by using conditional diffusion models. We base our formulation of the backward and forward process on the original DDPM paper by Ho et al. (2020).



$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$
$\boldsymbol{x}_0$
$\boldsymbol{x}_{t-1}$
$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$
$\boldsymbol{x}_t$
$\boldsymbol{x}_T$

Figure 2.5: A visualization of the backward and forward diffusion process. In the forward process, the original spectrogram $\boldsymbol{x}_0$ is iteratively destroyed through the process q adding noise at each step. In the backward process, p_θ attempts to remove noise from pure noise $\boldsymbol{x}_T$ to obtain the original spectrogram.

2.3.1 Forward Process

The forward process can be formulated as a Markov chain of T time steps, meaning each step only depends on the previous one. At each time step t, we add Gaussian noise with variance β_t to the data. Starting from a data point sampled from the real data distribution $\boldsymbol{x}_0 \sim q(x)$, i.e. a spectrogram, the forward process produces a sequence of increasingly noisy samples $\boldsymbol{x}_1, ..., \boldsymbol{x}_T$. This is done by sampling from the distribution

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}; \beta_t \boldsymbol{I}) \tag{2.3}$$

Where $\mathcal{N}(\dots)$ is the Gaussian distribution. Repeating this process T times is formulated as

$$q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) = \prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) \tag{2.4}$$

The variance β_t can be either fixed or defined as a schedule over the T timesteps. This noise schedule determines the amount of noise added in each step (Sohl-Dickstein et al., 2015). Common choices for the noise schedule include the linear schedule, where β_t increases linearly with t, as well as options such as using a cosine or sigmoid function. Figure 2.5 shows a visualization of the process, in which the data sample gradually loses its distinguishable features as t becomes larger. Eventually, after T time steps, the sample has been transformed into pure Gaussian noise.

In order to be able to sample any $\boldsymbol{x}_t$ without having to sample all its predecessors, we use the *reparametrization trick* (Kingma and Welling, 2013). In this way, we can compute $\boldsymbol{x}_t$ for any t by simply sampling from standard Gaussian noise once.

Let $\alpha_t = 1 - \beta_t$, $\overline{\alpha}_t = \prod_{i=1}^{t} \alpha_i$ and $\boldsymbol{\epsilon}_0, ..., \boldsymbol{\epsilon}_{t-2}, \boldsymbol{\epsilon}_{\boldsymbol{t-1}} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Then, using the addition property of the Gaussian distribution, we have

$$\begin{aligned}
\boldsymbol{x}_t &= \sqrt{\alpha_t}\boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{\alpha_t \alpha_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{1-\alpha_t \alpha_{t-1}}\boldsymbol{\epsilon}_{t-2} \\
&= ... \\
&= \sqrt{\overline{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}_0 \\
q(\boldsymbol{x}_t|\boldsymbol{x}_0) &= \mathcal{N}(\boldsymbol{x}_t; \sqrt{\overline{\alpha}_t}\boldsymbol{x}_0, (1-\overline{\alpha}_t)\boldsymbol{I})
\end{aligned} \tag{2.5}$$

Note that for sufficiently small β_t, $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ will be approximately Gaussian.

2.3.2 Backward Process

In the backwards process, we want to do the reverse of the forward process, sampling from pure noise $\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ and ending up with the original data. However, $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ can not be estimated and instead needs to be approximated with a parameterized model p_θ. Since $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ will also be approximately Gaussian for small β_t, p_θ is also chosen to be Gaussian.

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t)) \tag{2.6}$$

Applying this for all timesteps, $p_\theta(\boldsymbol{x}_{0:T})$, we can go from $\boldsymbol{x}_T$ to the original data distribution.

$$p_\theta(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T)\prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) \tag{2.7}$$

Although $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ is intractable, it becomes tractable when we also condition on the original data point $\boldsymbol{x}_0$, as shown by Sohl-Dickstein et al. (2015).

$$\begin{aligned} q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) &= \mathcal{N}(\boldsymbol{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_0, \boldsymbol{x}_t), \tilde{\beta}_t \boldsymbol{I}) \\ \tilde{\beta}_t &= \frac{1-\overline{\alpha}_{t-1}}{1-\overline{\alpha}_t}\cdot \beta_t \\ \tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) &= \frac{\sqrt{\overline{\alpha}_{t-1}}\beta_t}{1-\overline{\alpha}_t}\boldsymbol{x}_0 + \frac{\sqrt{\alpha_t}(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}\boldsymbol{x}_t \end{aligned} \tag{2.8}$$

Rewriting the term for $\boldsymbol{x}_t$ obtained in equation 2.5, we obtain $\boldsymbol{x}_0 = \frac{1}{\sqrt{\overline{\alpha}_t}}(\boldsymbol{x}_t - \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}_t)$. Using this in equation 2.8, we obtain an expression for $\tilde{\boldsymbol{\mu}}_t$ that is only dependent on $\boldsymbol{x}_t$:

$$\tilde{\boldsymbol{\mu}}_t = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\boldsymbol{\epsilon}_t\right) \tag{2.9}$$

Thus, we use a neural network $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$ to approximate the noise $\boldsymbol{\epsilon}_t$ and consequently $\tilde{\boldsymbol{\mu}}_t$. The variance is kept fixed following the chosen noise schedule. Experimentally, it has been found by Ho et al. (2020) that using $\tilde{\beta}_t$ and β_t produced similar results.

2.3.3 Conditional Diffusion Model

In order to use diffusion models for the task of source separation and we need to train a model to generate spectrograms from Gaussian noise using the backward process. However, we do not want the model to generate just any spectrogram that imitates the training data. Instead, we need the model to be able to generate a *specific spectrogram*, namely one containing only the target source audio given an input mixture spectrogram. Therefore, we need to condition the removal of noise in the backward process on this input $\boldsymbol{y}$. We take inspiration in the formulation of the conditional model from Saharia, Chan, et al. (2022). The backward process in the conditional model becomes

$$p_\theta(\boldsymbol{x}_{0:T}|\boldsymbol{y}) = p(\boldsymbol{x}_T)\prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{y}). \tag{2.10}$$

This means the noise $\boldsymbol{\epsilon}_t$ removed at each time step is also conditioned on $\boldsymbol{y}$, and thus we need to include this in our approximation $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \boldsymbol{y}, t)$ The way in which this conditioning is done can vary, but the method used in this project will be simply concatenating the input spectrogram $\boldsymbol{y}$ with the current noisy sample $\boldsymbol{x}_t$. Using the simple loss function found by (Ho et al., 2020), we end up with the training loop in algorithm 1.

Algorithm 1 The training loop for the conditional diffusion model. l depends on the loss function used for optimization.

```
repeat
    $\boldsymbol{x}_0, \boldsymbol{y} \sim q(\boldsymbol{x}_0, \boldsymbol{y})$
    $t \sim \text{Uniform}(\{1, \ldots, T\})$
    $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
    $\boldsymbol{x}_t \leftarrow \sqrt{\overline{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\overline{\alpha}_t}\boldsymbol{\epsilon}$
    $\hat{\boldsymbol{\epsilon}}_t \leftarrow \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \boldsymbol{y}, t)$
    Take gradient descent on $\nabla_\theta||\hat{\boldsymbol{\epsilon}}_t - \boldsymbol{\epsilon}||^l$
until converged
```

2.3.4 Sampling

During inference, the trained neural network is used to create the magnitude spectrogram of the separated source by gradually denoising an input containing pure Gaussian noise. This process is conditioned on the mixture spectrogram of the audio we want to separate the sources of. The conditioning is done in the same way as was done during training. Although there are alternative ways of designing the sampling process, such as the accelerated generation process used by Song et al. (2022), we choose to use the original sampling process from the original DDPM paper for simplicity (Ho et al., 2020). The sampling process is described in algorithm 2.

Algorithm 2 The sampling process for the conditional diffusion model. For our purpose, the condition will be the mixture spectrogram.

```
$\boldsymbol{y} \leftarrow$ Condition
$\boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
for $t = T, \ldots, 1$ do
    $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ if $t > 1$, else $\boldsymbol{z} \leftarrow \boldsymbol{0}$
    $\boldsymbol{x}_{t-1} \leftarrow \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \boldsymbol{y}, t)\right) + \sqrt{\beta_t}\boldsymbol{z}$
end for
return $\boldsymbol{x}_0$
```

2.3.5 Latent Diffusion

Both the training and sampling of diffusion models are very computationally expensive. The number of diffusion timesteps T is most commonly 1000, so it takes a long time for the model to learn how much noise to remove at each time step. Training usually takes several days, and even the sampling process can take minutes due to the need to repeat the denoising process 1000 times.

To improve efficiency, a latent diffusion model can be utilized, in which the diffusion process is done in a smaller latent space rather than the full spectrogram space (Rombach et al., 2022). Given an input $\boldsymbol{x} \in \mathbb{R}^{D_\tau \times D_f \times C}$, we use an encoder $\mathcal{E}$ that encodes $\boldsymbol{x}$ into a latent representation $\boldsymbol{z} = \mathcal{E}(\boldsymbol{x})$ where $\boldsymbol{z} \in \mathbb{R}^{d_\tau \times d_f \times c}$, *downsampling* the input by a factor $2^k = D_\tau/d_\tau = D_f/d_f$, where $k \in \mathbb{N}$. The diffusion process is then done in this latent space. Following this, a decoder $\mathcal{D}$ can be used to reconstruct the data from the latent space $\tilde{\boldsymbol{x}} = \mathcal{D}(\boldsymbol{z})$, obtaining the original dimensionality. When conditioning on the mixture spectrogram in the diffusion process, both the noisy sample $\boldsymbol{x}_t$ and the mixture spectrogram $\boldsymbol{y}$ are encoded into this alternative representation before concatenation and input to the neural network.

The autoencoder is created by training a separate CNN using an encoder-decoder architecture. This creates a bottleneck for data that ensures only the main structured part of the information can go through and be reconstructed. The objective of this neural network is simply to regenerate the original input data. The closer the output of the autoencoder is to the input, the better it is able to capture the most important features of the input and represent them in the smaller latent space.

2.4 Evaluation Metrics

In order to evaluate the performance, we need a set of metrics that compares the output of the model to the target audio. In this section, we introduce the metrics used in this project, separated into objective evaulation and subjective evaluation.

2.4.1 Objective Evaluation

The performance of the separation models is evaluated using a set of scale-invariant measures proposed by Le Roux et al. (2019) for use in source separation. These measures use the waveform representation of the estimated source audio $\hat{s}$ and of the target source audio s. The measures are based on a decomposition of the estimated source into a target term and a residual term, $\hat{s} = e_{\text{target}} + e_{\text{res}}$. This is done by ensuring that the residual is orthogonal to the target by scaling the target source s.

$$e_{\text{target}} = \alpha s, \quad \alpha = \frac{\hat{s}^\top s}{||s||^2}$$
$$e_{\text{res}} = \hat{s} - e_{\text{target}}$$

The residual term can be further decomposed to an interferences term and an artefacts term, $e_{\text{res}} = e_{\text{interf}} + e_{\text{artef}}$. Here, e_{interf} is defined as the orthogonal projection of e_{res} onto the subspace spanned by both the target source s and the other sources. The interferences term refers to the amount of sound from other sources that are "leaked" into the estimated signal and the artefacts term refers to forbidden distortions of the sources or "burbling" artefacts (Vincent et al., 2006). These terms are used to define the objective performance measures used to evaluate the performance of the MSS models, where each measure is defined using some ratio between two of these terms.

Scale-Invariant Source-To-Distortion Ratio (SI-SDR) SI-SDR is a measure comparing the amount of the target signal to the amount of any other audio in the estimated signal. This is the measure that is the most frequently used in literature to evaluate performance in MSS.

$$\text{SI-SDR} := 10 \log_{10} \frac{||e_{\text{target}}||^2}{||e_{\text{res}}||^2}$$

Scale-Invariant Source-To-Interferences Ratio (SI-SIR) SI-SIR measures how much of the other sources are left in the estimated signal. This is often referred to as "leakage" or "bleed".

$$\text{SI-SIR} := 10 \log_{10} \frac{||e_{\text{target}}||^2}{||e_{\text{interf}}||^2}$$

Scale-Invariant Source-To-Artefacts Ratio (SI-SAR) SI-SAR can be interpreted as the amount of unwanted artefacts produced by the separation process.

$$\text{SI-SAR} := 10 \log_{10} \frac{||e_{\text{target}}||^2}{||e_{\text{artef}}||^2}$$

Source-To-Noise Ratio (SNR) In addition to the scale-invariant measures, we provide the simple Source-To-Noise Ratio. SNR is a measure comparing the amount of the target source to the amount of other audio in the estimated audio.

$$\text{SNR} := 10 \log_{10} \frac{||s||^2}{||s - \hat{s}||^2}$$

2.4.2 Subjective Evaluation

While mathematical measures are a good indicator of a model's performance, human evaluation is what ultimately matters in evaluating how well the sources are separated. This is hard to do in practice, however, as high-quality tests using human subjects are often expensive and time-consuming. A high-quality subjective evaluation is therefore outside of the scope of this thesis. Even so, some measure of how the generated audio sounds compared to the true source audio is beneficial. In this project, a subjective evaluation of the results will be done by the author, providing an indicator of the models' performance evaluated by human perception.

The evaluation will be done using the multi-criteria subjective test protocol proposed by Emiya et al. (2011). This protocol consists of four separate listening tests, each aiming to assess the perceived quality with respect to different kinds of distortion in the produced audio. The four tasks are the following:

1. Rate the global quality (GC) compared to the reference for each song.
2. Rate the quality in terms of perservation of target source (PoTS) in each song.
3. Rate the quality in terms of suppression of other sources (SoOS) in each song.
4. Rate the quality in terms of absence of additional artificial noise (AoAAN) in each song.

The rating will be done using a scale from 1 to 5, where 1 indicates the lowest quality and 5 indicates the highest quality. Each of these tests will be performed in the above order, going through all of the songs before moving on to the next test. Both the original mixture audio and the target source audio will be available for comparison during the tests.

The four subjective evaluation tests relate to the objective evaluation metrics, in that the SI-SDR measure should give a good indicator of the performance on the GC test. Similarly, the SI-SIR measure relates to the SoOS test, and the SI-SAR measure relates to the AoAAN test. In this way, we can obtain a good idea of the model's quality with respect to the different kinds of distortion, both from an objective standpoint and a subjective one.

Chapter 3

Related Work

In this section, we explore the existing literature and research related to our project. Although the amount of research done within these fields is immense, we highlight some of the work that is the most relevant to ours and that puts this thesis into perspective. By surveying the existing work, we gain insights into the advancements, methodologies, and challenges in these areas, which inform our own research and contribute to the development of our approach.

We begin by discussing the use of DNNs for musical source separation, highlighting relevant studies that have leveraged the power of deep learning techniques to tackle this task. Next, we look into the literature on diffusion models and their applications in tasks similar to MSS, such as image segmentation and audio generation. We then specifically explore the use of diffusion models for MSS. We investigate studies that have employed diffusion models to separate mixed audio sources and examine the methodologies, architectures, and performance achieved in these works.

3.1 Deep Neural Networks for Musical Source Separation

A thorough overview of approaches in MSS was done by Rafii et al. (2018). Although there have been methodologies proposed using audio in its waveform representation, this section will focus on applications using DNNs on audio in its TF representation.

Early efforts by Huang et al. used Recurrent Neural Networks for creating TF masks for both speech and singing voice separation (Huang et al., 2014a; Huang et al., 2014b; Huang et al., 2015). They showed that this methodology achieved improved results when compared to previous non-DNN methods. Uhlich et al. (2015) used a FFNN to directly approximate the separated instruments rather than generating a TF mask. Instead of using the full magnitude spectrogram as input, they used a few non-overlapping consecutive frames of the spectrogram. However, this approach struggled when compared to non-DNN masking methodologies.

Simpson et al. (2015) introduced the use of CNNs on the task of MSS by generating binary masks, while Chandna et al. (2017) and Grais and Plumbley (2017) used different setups of convolutional encoder-decoder architectures to create a soft mask. The use of CNNs was in this research shown to be able to extract local features in the spectrogram rather than relying on global features across the entire frequency spectrum. Jansson et al. (2017) was the first to utilize the UNet architecture for MSS. Their approach utilized the architecture for soft mask estimation, achieving state-of-the-art performance, and outperforming the previous DNN models. Building on this, Brocal and Peeters (2019) modified the UNet architecture to be conditioned on which instrument to be separated, eliminating the need for training a separate model for each instrument. Kadandale et al. (2020) used a multi-channel UNet in order to separate several sources using the same model, and found that it performs better than using dedicated UNets for each source, also in a 2-source setting.

3.2 Diffusion Models for Similar Tasks

While Diffusion Models were originally used for image generation, they have been utilized for a variety of other tasks, many of which are related to our problem of MSS. The amount of existing research in this field is large, so this section will only highlight a selection of the literature relating to our problem. A comprehensive collection of papers on diffusion models and their many uses can be found on the `Awesome-Diffusion-Models` GitHub repository[1].

One application of diffusion models related to MSS is that of image segmentation. Image segmentation is the task of assigning a label to each pixel in an image in order to identify whether it belongs to a specific class or not. This is related to using the binary mask approach in MSS, in which we would use a spectrogram as the image and the classification of whether to eliminate or keep each pixel. Amit et al. (2022) used Diffusion Models for this task to obtain state-of-the-art segmentation results. In their implementation, the removal of noise was conditioned on an input image by summation of the noisy image and the condition image at each time step. Image segmentation using diffusion models is also very applicable to medical image segmentation, demonstrated by the work of Wolleb, Sandkühler, Bieder, Valmaggia, et al. (2022), where they conditioned the network on several images by concatenation to the noisy image.

Another related task is that of image-to-image translation, in which the goal is learning a mapping between images from a source domain and images from a target domain. The work of Sasaki et al. (2021) was the first demonstrating the use of DDPMs for this task but has been quickly followed by several projects such as Choi et al. (2021), Batzolis et al. (2021), Wolleb, Sandkühler, Bieder, and Cattin (2022), and Saharia, Chan, et al. (2022). The methodologies of these are all very similar to the one applied in this thesis but generally train on images with smaller dimensions than our spectrograms.

Latent diffusion models were introduced by Rombach et al. (2022), who in their paper also demonstrate their use in image-to-image translation tasks. This has since also been used in later research such as the work of Wu and Torre (2022), Avrahami et al. (2023), and Parmar et al. (2023).

Diffusion models have also become a popular technique used in audio generation. Chen et al. (2020) and Kong et al. (2021) both train a conditional diffusion

[1]`https://github.com/heejkoo/Awesome-Diffusion-Models`

model for audio synthesis, though they use waveforms rather than spectrograms. Within the field of audio generation, text-to-speech tasks have been heavily researched in projects such as Jeong et al. (2021) and Popov et al. (2021), where they operate in the TF domain, but condition the diffusion model on text rather than audio. J. Liu et al. (2022) uses diffusion models to generate spectrograms of singing voice audio conditioned on a symbolic representation of the target audio.

While all the methodologies utilized in the mentioned research are very similar to the ones used in this thesis, one major difference is that many of the tasks that diffusion models are applied to are artistic in nature. Diffusion models excel in generating and transforming images and audio, where the output is meant to be a new, creative sample that is similar to the data it was trained on. This gives room for some inconsistency and improvisation that will inevitably occur as a result of the inherent random nature of the models. MSS, in contrast, requires somewhat consistent, conservative outputs that are as close to the ground truth as possible, especially if we give an emphasis to objective evaluation.

3.3 Diffusion Models for Musical Source Separation

Some efforts have been made to apply diffusion models to the source separation problem, though research is still in its early phases. In addition, they have mostly focused on the separation of speech sources. Scheibler et al. (2022) used a diffusion-based model where the forward process involved gradually mixing the two speech sources in addition to adding noise. However, their model did not achieve better results than its previous models. A pre-trained diffusion model trained on male speakers was in the works of Lutati et al. (2023) used as one component in a larger model used for separation of multiple speech sources. Hirano et al. (2023) also uses a Diffusion Model to refine the output of a preceding speech separation model. Finally, Mariani et al. (2023) uses a conditional diffusion model for both music generation and separation, although working in the waveform domain.

As is evident, there is much still to explore within the use of diffusion models in MSS. This thesis aims to cover ground by applying methodologies inspired by those of similar tasks such as image-to-image translation, in addition to working with musical instruments rather than speech.

Chapter 4

Experimental Setup

In this section, we present the experimental setup for our study of using diffusion models for MSS. We describe the dataset used for training and evaluation and the preprocessing done before input to the models. This is followed by the description of the neural network used in our models, as well as the two types of diffusion models implemented: the standard diffusion model and the latent diffusion model. Within the latent diffusion model, we explore two different approaches for generating the separated sources: direct spectrogram generation and mask generation. Finally, we discuss the training procedure employed to train the diffusion models, and how sampling is done to produce the separated source audio. All code is written in Python using the `PyTorch`[1] library. The complete code for the project, as well as the produced models, are available in a GitHub repository[2].

4.1 Dataset

A dataset is needed for the training and evaluation of the diffusion models. To keep the complexity of the task relatively low for this project, a simple dataset was generated containing only audio from two sources: piano and violin. This gives us full control over the data used for training, providing a controlled context in which we can conduct the experiments. The dataset was created by generating random MIDI notes using the `mido`[3] python library. A range of suitable MIDI notes was selected covering the desired pitch range for both the violin and the piano, and varying the length and velocity of each note to keep the audio somewhat true to real life. The MIDI files are then synthesized into audio using the `midi2audio`[4] library and converted into mono audio. To reduce the computational cost of training, we use a sampling frequency of 11025hz. This sacrifices some of the audio quality, lowering the Nyquist frequency to around 5500Hz; however, the instruments are still clearly recognizable. For each song, files containing each source individually, as well as a mixture of the two, were created. The dataset was split into a training set of 500 songs, a validation set of 50 songs, and a test set of 50 songs.

[1]`https://pytorch.org/`
[2]`https://github.com/aleksolberg/MSS-diffusion`
[3]`https://github.com/mido/mido/`
[4]`https://github.com/bzamecnik/midi2audio`

4.2 Preprocessing

In this section, we discuss the preprocessing steps applied to the audio data before training the models. First, the audio is converted from its waveform representation to its TF representation by doing a STFT. This is done with a window length of $M = 511$ samples and a hop length of $H = 128$ samples. This configuration was chosen so that the number of frequency bins is $D_f = 256$. This, in addition to truncating the audio to contain $D_\tau = 256$ time bins, results in spectrograms of shape $D_\tau \times D_f = 256 \times 256$. This is beneficial to us as it will enable us to use a pre-trained autoencoder when doing latent diffusion, which requires dimension sizes that are a power of two.

The absolute value of the complex-valued spectrogram is taken to obtain the magnitude spectrogram, and the magnitudes are then converted to decibels (dB) and normalized to values between 0 and 1. The phase information is stored for later to be used when converting the audio back to its waveform representation. In the experiments with mask generation, an ideal soft mask for obtaining the violin source audio from the mixture spectrogram was also created.

4.3 UNet implementation

In this section, we present the implementation of the UNet architecture which is used to predict the noise in the diffusion models. In addition, this architecture will also be used to create a simpler mask-based source separation model to compare to the diffusion models. The implementation of the UNet architecture is taken from the `Cold-Diffusion-Models`[5] GitHub repository and follows a modified UNet structure. It comprises an encoder-decoder structure with an additional middle block between the two parts. Figure 4.1 shows a visualization of the overall structure of the UNet.

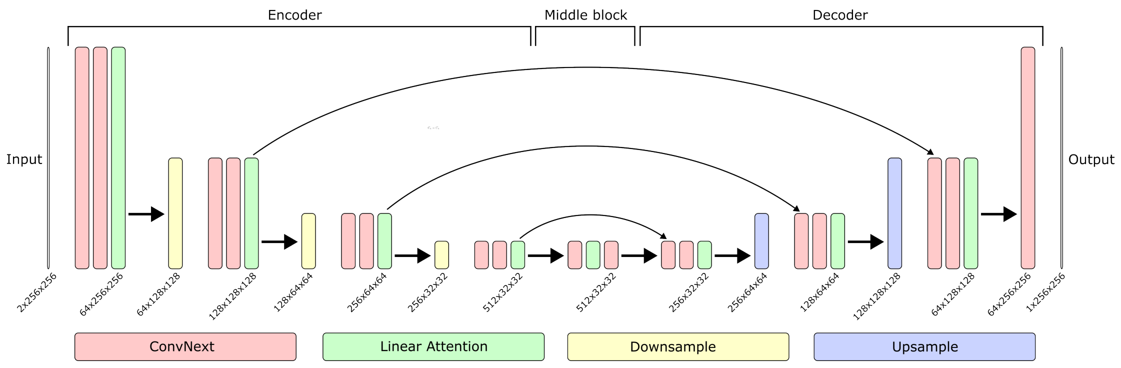


Figure 4.1: A visualization of the high-level structure of the UNet model used in the diffusion models, as well as the comparison model. In the figure, each block is colour coded indicating which type of block it is. The large arrows indicate the direction of the data, and the small overarching arrows show the skip connections between layers in the encoder and the decoder. The size of the representation at each layer is indicated below them.

Each layer consists of several blocks of four different variations: the ConvNext-block (Z. Liu et al., 2022), the Linear Attention block (Li et al., 2020) and down- and upsample blocks. The high-level architecture can be summarized as follows:

[5]`https://github.com/arpitbansal297/Cold-Diffusion-Models`

Encoder The encoder part of the UNet consists of a series of blocks downsampling the input. Each downsampling layer applies two ConvNext-blocks and one linear attention block, before downsampling to half of the input dimensions. The downsampling operation reduces the spatial dimensions while increasing the number of channels, effectively extracting meaningful representations of the input mixture. In our implementation, there are 4 layers in the encoder, with each layer reshaping the input to having channels $C_n = 64 \times 2^{(n-1)}$, where n is the layer depth.

Middle Block Following the downsampling blocks, the middle block further processes the features to capture global dependencies and refine the representations. It includes two ConvNext-blocks separated by a linear attention block, allowing the model to encode complex relationships in the data effectively.

Decoder The decoder part of the UNet consists of a series of blocks upsampling the lower dimensional representation back to its original dimensionality. These blocks leverage skip connections from the encoder to combine low-level and high-level features, done by concatenation of the output from the encoder layers to the input of the corresponding decoder layers. Each upsampling layer, similarly to the downsampling layers, applies two ConvNext-blocks and one linear attention block before an upsample block. The upsampling operation increases the spatial dimensions while reducing the feature channels, mirroring the blocks in the encoder. This enables the model to generate fine-grained predictions.

Final Convolution After the upsampling blocks, a final set of convolutional layers is applied to generate the final output of the neural network. This block consists of one ConvNext-block and a 1x1 convolutional layer to produce the output with the original dimensionality and one channel.

Low Level Architecture At the lower level, the blocks that make up each layer in the UNet architecture can be described as follows:

The **ConvNext-block** consists of an initial 7×7 convolution, which is followed by a layer normalization before a 1×1 convolution, tripling the number of channels. Following this, a GELU activation function is applied before another 1×1 convolution reduces the number of channels back to that of the input. Finally, a residual connection adds the input of the block to the output of the last convolution.

In the **linear attention block**, a layer normalization is first applied to the input, before a 1×1 convolution transforms it into three tensors known as queries, keys and values. The queries are scaled using a pre-calculated scale factor. A softmax function is applied to the keys along the last dimension to obtain attention weights, representing the importance of different spatial positions for each query. The values are then weighted by these attention weights and combined to create a context tensor that captures the relevant information from different spatial positions. There is also a residual connection between the input and output of the linear attention block, adding the two together. This process is designed to enhance the model's ability to capture long-range dependencies.

The **downsampling block** is a single convolution operation using a 4×4 kernel to increase the number of channels before the next layer. The **upsampling block** does the opposite, using a transposed convolution operation to decrease the number of channels.

4.4 Standard Diffusion Model

In this section, we describe the implementation of the standard diffusion model used for source separation by generating the spectrograms of the separated source. We refer to this model as the Spectrogram Standard Diffusion Model (SSDM). The implementation was written using the `PyTorch Lightning`[6] library, where we have used the `DiffusionFastForward`[7] GitHub repository as a starting point.

Following the suggested values of Ho et al. (2020), a Gaussian diffusion process of $T = 1000$ time steps with a linear noise schedule having values increasing from $1e^{-4}$ to $2e^{-2}$ is used. At each training step, we follow the pseudocode in Algorithm 1. First, Gaussian noise $\boldsymbol{\epsilon}_t$ for a randomly chosen time step t is added to the source spectrogram. We then concatenate this noisy spectrogram with the mixture spectrogram and input it into the neural network. In this way, the neural network is conditioned on the mixture spectrogram. Although there are more sophisticated methods of conditioning, simple concatenation has been found to yield good results (Saharia, Ho, et al., 2022), and thus this method was chosen for simplicity.

In order for the neural network to accurately predict the amount of noise to be removed throughout the process, it needs to be provided with information about the current time step of the diffusion process. Thus, the UNet model described in section 4.3 is modified to accommodate this. We use a sinusoidal positional embedding of the time step (Vaswani et al., 2017). At each of the ConvNext-blocks, this time embedding is put through a GELU and a linear layer and added to the output of the first convolution in the main part of the block. In this way, the estimation of the added noise is also conditioned on the time step of the diffusion process.

4.5 Latent Diffusion Model

This section will describe the implementation of the latent diffusion model, building on the description of the standard diffusion model described in the previous section.

Instead of using the full spectrograms as the input to the neural network, we encode them into a latent space using a pre-trained autoencoder. The autoencoder architecture used is the Variational Autoencoder (VAE) model with KL loss described by Kingma and Welling (2013). The pre-trained model used is produced by Huggingface[8]. Since this model was trained on colour images containing values in three channels (RGB), we need to convert the spectrograms to also have three channels. This is done by simply copying the spectrograms to all three channels.

[6]`https://www.pytorchlightning.ai/`

[7]`https://github.com/mikonvergence/DiffusionFastForward`

[8]`https://huggingface.co/stabilityai/sd-vae-ft-ema`

The resulting input tensors have a shape of $D_\tau \times D_f \times C = 256 \times 256 \times 3$. The VAE then encodes the tensors into a latent space of shape $d_\tau \times d_f \times c = 32 \times 32 \times 4$, reducing the dimensionality by a factor $2^k = 8$.

The diffusion process is then done in this latent space in the same way as it was done in the non-latent case, and the noise is predicted by a UNet with the same architecture. The loss is also calculated in this latent space rather than the spectrogram space. During sampling, the output from the model is decoded by the VAE back to the original shape of $256 \times 256 \times 3$. We then take the average of the three channels to obtain the produced magnitude spectrogram.

Within the latent diffusion approach, we look at two different ways of estimating the isolated source. The first is the same approach as in the standard diffusion model, where the model produces the magnitude spectrogram of the source directly. We refer to this model as the Spectrogram Latent Diffusion Model (SLDM). In the other approach, we aim to estimate a mask that when applied to the mixture spectrogram will produce the isolated source. The model trained using this approach will be referred to as the Mask Latent Diffusion Model (MLDM). The difference between the two approaches is that the first will aim to generate the separated source from scratch, while the second will start from the mixture spectrogram and aim to eliminate all audio other than the target source. In the training of the MLDM, we still condition the neural network on the mixture spectrogram, but the target is the ideal mask rather than the target source spectrogram.

4.6 Training

Optimization of the neural network is done by comparing the output $\hat{\boldsymbol{\epsilon}}$ the noise added at that time step in the forward process $\boldsymbol{\epsilon}$. This is done using the $\mathcal{L}_1$ loss function:

$$\mathcal{L}_1 = \|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}\|_1$$

This loss function was chosen due to the findings of Saharia, Chan, et al. (2022), noting that $\mathcal{L}_1$ produces more conservative outputs than other loss functions in image-to-image diffusion models. Since we want our model to generate very consistent outputs, meaning as close to the ground-truth source as possible, this behaviour is desirable for our implementation.

Training is done using the Machine Learning platform Paperspace Gradient[9], giving access to better hardware for quicker training. The models are trained on the training set of 500 songs containing violin and piano for 2000 epochs, using a batch size of 8. We also use the validation set of 50 songs to monitor the training. Once again following the training details of Saharia, Chan, et al. (2022), we use a constant learning rate of $1e^{-4}$ with a linear warmup schedule of 10000 steps. We also use a 0.9999 Exponential Moving Average (EMA). The hyperparameters of the four models is summarized in table 4.1

[9] `https://www.paperspace.com/gradient`

Hyperparameter	SSDM/UNet Only	SLDM/MLDM
Input/output shape	$256 \times 256 \times 1$	$256 \times 256 \times 3$
Condition shape	$256 \times 256 \times 1$	$256 \times 256 \times 3$
Latent shape	N/A	$32 \times 32 \times 4$
Layers	$4 + 1 + 4$	$4 + 1 + 4$
Loss function	L1	L1
Optimizer	AdamW	AdamW
Learning rate	$1e^{-4}$	$1e^{-4}$
Linear LR warm-up	10000 steps	10000 steps
EMA	0.9999	0.9999
Batch size	8	8
Epochs	2000	2000

Table 4.1: Hyperparameters for the four trained models. All models use the same UNet architecture and training parameters, with the difference being that the two latent diffusion models convert the input into a latent space before input to the UNet

4.7 Sampling

During inference, we produce a new spectrogram by computing the reverse diffusion process, starting from pure Gaussian noise. The noise is removed iteratively in 1000 time steps using the trained neural network. We condition the noise removal on the mixture spectrogram of the audio we would like to perform the separation on by concatenation at each time step. The resulting output of the SSDM and SLDM is the estimated source spectrogram. In the case of the MLDM and the simple UNet model, we do element-wise multiplication of the output with the mixture spectrogram in order to obtain the estimated source spectrogram.

Finally, the phase information from the original mixture is added to the estimated spectrogram, and an ISTFT is done in order to obtain the estimated audio waveform. To isolate the remainder of the sources in the mixture, we simply subtract the waveform of the estimated source from the mixture waveform.

Chapter 5

Results and Discussion

In this chapter, we present the trained models for the four approaches: the Spectrogram Standard Diffusion Model (SSDM), the Spectrogram Latent Diffusion Model (SLDM), the Mask Latent Diffusion Model (MLDM), and the simple UNet model. We compare the resulting performance and discuss the different advantages and limitations of each model. The objective evaluation was done by generating both separated sources for all 50 songs in the test set and all 50 songs in the validation set. The measures from section 2.4.1 were then calculated by comparing the produced waveforms to the target waveforms, and averaged over all songs and both sources. The subjective evaluation was done on 10 of the songs from the test set, randomly chosen, in identical listening environments. The produced audio from these 10 songs are also available in the GitHub Repository[1].

Initially, we present ideal values for the three diffusion approaches on the objective measures in table 5.1. These values will be used as a reference to what is the best performance we could achieve with these methods when applied to the test set. For the SSDM, the ideal values are found by comparing the true source audio waveform to the waveform having been simply converted to a spectrogram and back using the STFT process. This process is not without loss of audio quality, and will not be able to reproduce the original audio perfectly. For the SLDM the ideal values are found by also encoding and decoding the target audio spectrogram using the autoencoder. Finally, the ideal values for the MLDM are found by comparing the true source audio waveform to the audio generated by applying the ideal mask to the mixture spectrogram.

Interestingly, the ideal values for the SLDM are significantly lower than the

[1]`https://github.com/aleksolberg/MSS-diffusion`

Ideal model	SI-SDR	SI-SIR	SI-SAR	SNR
Ideal SSDM	15.59 ± 4.93	28.10 ± 8.85	16.04 ± 4.69	15.67 ± 4.64
Ideal SLDM	4.20 ± 4.28	19.50 ± 9.21	4.60 ± 4.23	3.54 ± 4.78
Ideal MLDM	15.22 ± 4.77	25.47 ± 6.89	15.78 ± 4.55	15.32 ± 4.46

Table 5.1: Objective performance measures for the theoretical **ideal models** using the three diffusion-based approaches. The values are calculated using the measures discussed in section 2.4.1 for both the violin and the piano estimations and averaged over both sources and all songs in the test set. Values are in decibels (dB), where higher is better.

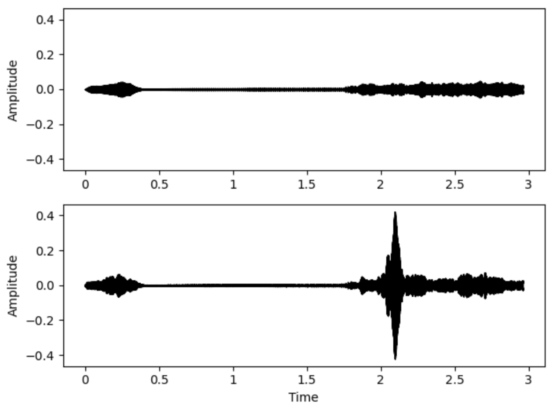


Figure 5.1: A visualization of how audio transformed into the latent space and back again can significantly change its waveform. The upper waveform is the original, and the lower is after it has been encoded and decoded by the autoencoder. The sudden jump in volume around the 2-second mark is an artefact of this process.

ideal values for the SSDM and the MLDM. This suggests that the process of encoding and decoding loses a great amount of audio quality. This can be caused by the autoencoder not being specifically trained on spectrogram data, and that it is therefore not able to create an accurate representation in the latent space. When listening to the audio after it has been encoded and decoded, there is a noticeable drop in quality compared to the original audio, as there are a few added artefacts. As seen in figure 5.1, it also appears to amplify parts of the audio randomly, which would significantly decrease its performance on the objective measures. However, the audio is still easily recognizable as being the original audio, and would not have scored as comparatively low in the subjective evaluation metrics as it does in the objective measures. This suggests that the objective measures are perhaps a little too strict in this case.

The ideal values for the SI-SIR metric especially highlights the theoretical advantage that a spectrogram generative approach could have over a mask-based approach. Since the latter directly uses the mixture spectrogram in its final estimation, any mistakes in the applied mask will allow for interferences. In fact, the ideal values for the SI-SIR metric suggest that even the ideal masks used to train the mask-based models are not able to eliminate all of the other sources from the mixture spectrogram. In contrast, mistakes in direct spectrogram generation will likely result in added noise or distortion. Thus, our hope is that using the generative powers of diffusion models, one might be able to consistently generate audio that is without any interference from the rest of the mixture.

Figure 5.2 shows the training and validation losses for the models during training. We can see that for all models, both the training loss and the validation loss decrease throughout the training, indicating that the model is able to learn the task. For some of the models, the loss is still decreasing at the end of the training, suggesting that there might still be potential for improvement with more training. The training loss and the validation loss of each model follow each other closely, showing that there are no signs of overfitting to the training set.

We note that for all three diffusion models, the loss is very unstable, fluctuating significantly. This is likely because of the nature of the task being predicting random noise, which will inevitably lead to a lot of randomness in the evaluation

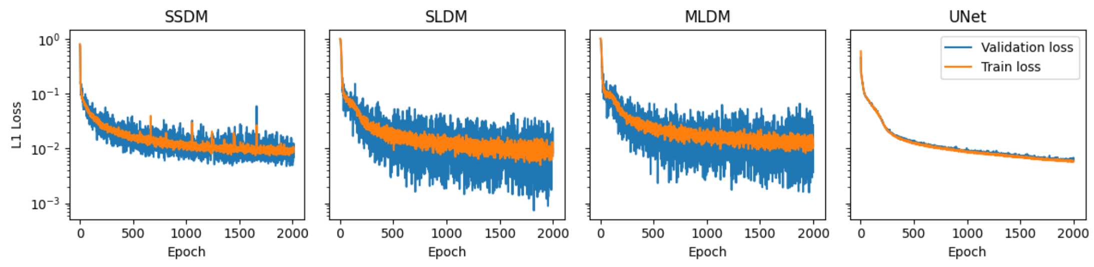


Figure 5.2: Graphs showing the training and validation losses for each of the three diffusion models and the simple UNet model during training. The values are evaluated using the L1 loss function and averaged over each training step in the epoch.

of the loss function as well. The presence of more variance in the validation loss than the training loss can be explained by the fact that the training set is 10 times larger than the validation loss, likely causing the average loss for each epoch to be more stable. We also see that the loss for the SSDM is not as varied as for the two latent diffusion models, which can be explained by the fact that the tensors being compared in the standard model are much larger, and will cause a more stable loss function. Finally, we note that for both latent diffusion models, the training seems to hit a plateau after around 50 epochs, before starting to decrease again after around 100 epochs. This might be caused by the model initially finding a local minimum at a simple solution, before learning a more complex one after some more training. It could also be due to the fact that we use a linear warm-up on the learning rate, causing slower learning in the beginning.

The advantage of doing the diffusion process in a smaller latent space is significantly noticeable in the time it takes for the different model types to finish training. Each iteration of the training loop of the latent models took about half the time of one iteration in the standard model. This difference is even more noticeable during inference, where sampling takes around four times longer for non-latent diffusion.

In the following sections, we present the results from each of the models, both in the form of objective measures and subjective evaluations. We also compare the three models to each other, and to the simple UNet model without the diffusion process. A selection of spectrograms of the produced audio for the four models can be seen in figure 5.3. The objective evaluation can be seen in table 5.2, and the subjective evaluation can be seen in table 5.3.

5.1 Spectrogram Standard Diffusion Model

As is visible in figure 5.3, the results from the standard diffusion model are very varied. In some cases, it produces seemingly excellent spectrograms, as with the leftmost example. Most of them, however, contain a large amount of noise and are visibly brighter than the target spectrogram suggesting it is significantly louder in volume. Even so, it is possible to identify the traces of the target source audio.

This poor performance is also seen in the objective measures seen in table 5.2. In fact, the value is negative for all measures except the SI-SIR, indicating that the produced audio contains less of the target source than any other audio, such as distortion, artefacts and noise. The positive value of the SI-SIR measure could

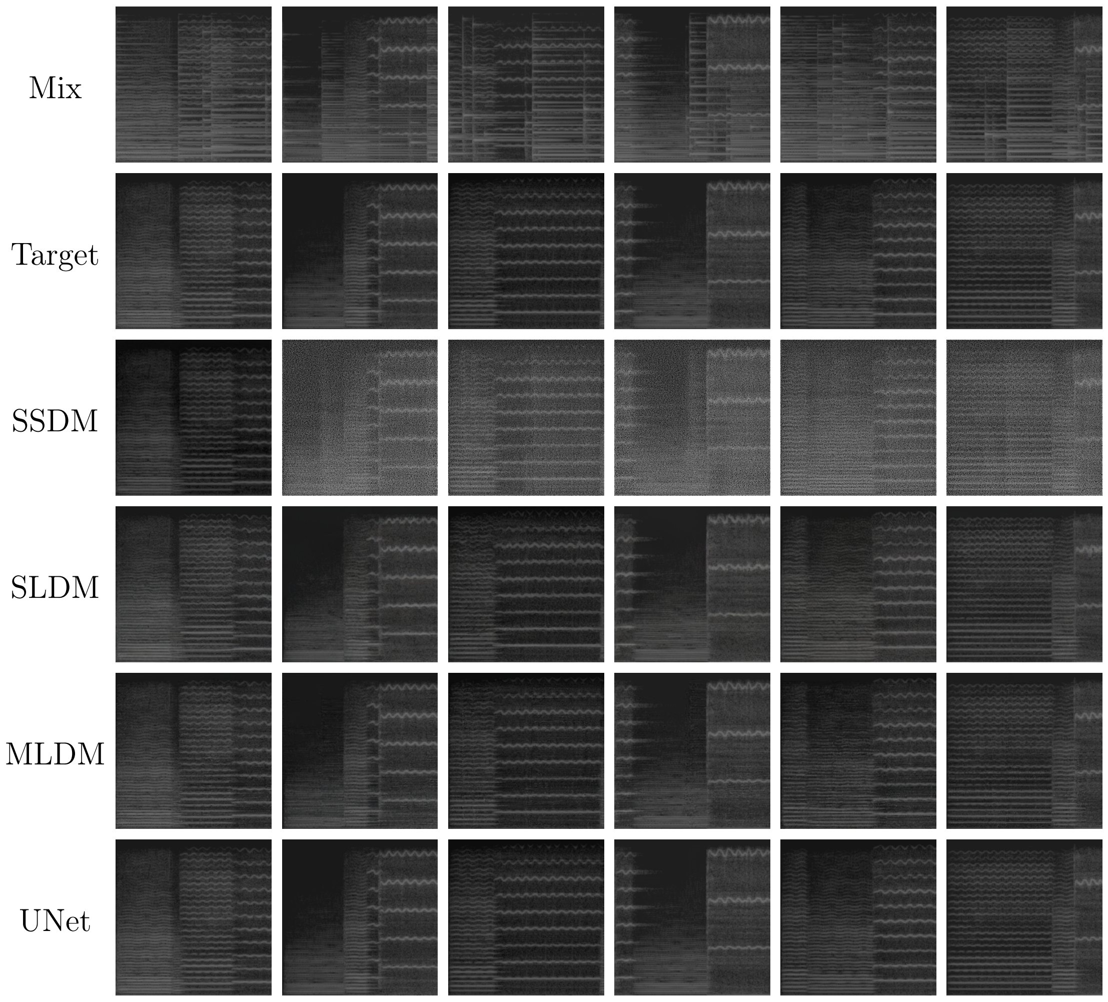


Figure 5.3: Spectrograms of the mixture audio, the target source audio, as well as the audio produced by the three diffusion models and the simple UNet model. The six songs are taken from the test set.

Model	Set	SI-SDR	SI-SIR	SI-SAR	SNR
SSDM	Validation	-8.47 ± 11.98	2.82 ± 16.13	-2.99 ± 20.29	-28.29 ± 20.32
	Test	-6.64 ± 12.50	6.75 ± 17.69	-1.13 ± 20.78	-28.60 ± 22.14
SLDM	Validation	5.15 ± 4.15	20.75 ± 9.21	5.62 ± 4.29	5.44 ± 4.35
	Test	4.61 ± 4.30	20.86 ± 9.91	5.08 ± 4.49	4.72 ± 4.80
MLDM	Validation	12.64 ± 4.39	21.80 ± 5.82	13.33 ± 4.20	12.86 ± 4.05
	Test	12.50 ± 4.45	21.73 ± 6.24	13.19 ± 4.24	12.71 ± 4.07
UNet	Validation	15.21 ± 4.49	25.48 ± 6.50	15.76 ± 4.28	15.30 ± 4.24
	Test	15.22 ± 4.77	25.47 ± 6.89	15.78 ± 4.55	15.32 ± 4.46

Table 5.2: Objective performance measures for the three diffusion models and the comparison UNet model. The values are calculated using the measures discussed in section 2.4.1 for both the violin and the piano estimations and averaged over both sources and all songs in each of the sets. Values are in decibels (dB), where higher is better.

Model	GC	PoTS	SoOS	AoAAN
SSDM	1.5	1.9	5.0	1.8
SLDM	2.7	3.0	4.8	2.7
MLDM	3.7	4.3	4.0	3.6
UNet	3.9	4.6	4.5	4.1

Table 5.3: The average values for the subjective evaluation tasks presented in 2.4.2 on each of the three diffusion models and the comparsion UNet model. The four tasks are **global quality (GC)**, **perservation of target source (PoTS)**, **suppression of other sources (SoOS)** and **absence of additional artificial noise (AoAAN)**. The evaluation was done on ten songs from the test dataset using a scale from 1 to 5, where higher is better.

indicate that the model at least does not produce much of the remaining sources in its estimation. We also observe that there is a large variance in these values, which validates our observation that the model produces extremely varied results.

Our findings in the objective performance measures are also backed up by the subjective evaluation, which can be seen in table 5.3, as it receives low scores for the GC, PoTS and AoAAN tasks. In eight of the ten songs, the audio produced is horribly distorted and painful to listen to. It is possible to hear the pitch of the notes in the audio, and even possible to make out that it is a violin playing the notes, but it is overwhelmingly infused by the noise and distortion.

The good news is that, perhaps also due to the noise, it is impossible to identify any of the piano notes from the original mixture audio in the estimated violin audio, leading to the model scoring high on the SoOS task. Furthermore, in two of the produced audio, there is no distortion or noise, and the violin sounds very much like it does in the target source. However, it is much quieter than in the original audio, and so does not achieve full points in the PoTS test even on these songs.

The poor performance of this model could be attributed to the complexity of generating data of large dimensions using standard diffusion models, as has been pointed out by Hoogeboom et al. (2023). As this model is working with 256×256 points of data, it might not be able to learn the denoising process effectively and thus is unable to remove the right amount of noise at each time step.

5.2 Spectrogram Latent Diffusion Model

The spectrograms generated by the SLDM initially look to be very similar to the target spectrograms, as seen in figure 5.3. Using the target spectrograms as a reference, the spectral lines of the violin are clearly visible, and the piano notes are also seemingly not present. Looking closely, however, we do see a bit of noise, and the lines of the harmonics of the violin are not as sharp as they are in the target spectrograms, indicating that there may be some loss in audio quality.

Table 5.2 shows the values of the objective evaluation metrics, which confirms our suspicion of low audio quality. The latent diffusion model does very well in the SI-SIR metric indicating that it does a good job of not containing any of the piano audio. However, the values for the other three metrics are not very impressive, only achieving around a 5dB gain on these measures. Even so, the model actually performs a little better than the ideal values shown in table 5.1 on all measures. This can be explained by the autoencoder not initially being trained to encode spectrograms, and that it becomes more finetuned to representing them in the latent space during the training of the diffusion model. This shows that the performance of this method could likely be significantly improved by initially finetuning the autoencoder on spectrograms before the training of the latent diffusion model.

The low audio quality is also audible when doing the subjective evaluation. In general, the audio unquestionably sounds like a violin, but the notes are somewhat distorted and unclear. In fact, the audio quality of the produced audio is perceived as significantly worse than that of the ideal latent case, where the target audio is encoded to the latent space and back again. This is likely because the diffusion process still produces a significant amount of noise and artefacts, even though the finetuning of the autoencoder has improved performance. This leads to the model achieving mediocre scores in the GC and AoAAN tests. Yet, the produced audio does mostly contain all of the violin notes from the target audio. It appears to have struggled somewhat with some of the lower-frequency notes, which are often lower in volume or more unclear than notes of higher frequency. The model achieves a high score in the SoOS task, meaning it does a good job of suppressing the piano audio, as was suggested by the objective evaluation.

5.3 Mask Latent Diffusion Model

The spectrograms produced by the mask-producing model can also be seen in figure 5.3. Since this method can only eliminate elements of the mixture spectrogram, it has the benefit of not being able to produce any audio not already in the mixture. This method therefore produces spectrograms that are clear and sharp, looking seemingly very similar to the target spectrograms. We can clearly identify the lines from the target spectrogram, and find no visible remains of the piano notes.

The model's performance on the evaluation measures, as seen in table 5.2 is also excellent. The model performs well on all four metrics, and with relatively little variance, indicating that it is consistent. In fact, in comparison to the ideal values for this approach found in table 5.1, it receives scores only around 3 or 4 dB lower on all metrics.

In the subjective evaluation, the model also does well. It has no problem preserving all violin notes, and therefore gets a relatively high score in the PoTS task. The general quality is also perceived to be relatively good, and thus it receives a good score on the GC task as well. There are several examples where the piano notes can be heard in the background, causing it to not as well in the SoOS. We note, however, that the same notes are present when applying the ideal mask to the spectrogram, and so this failure to suppress the piano audio is seemingly a consequence of the mask-based approach as a whole. Finally, there are a few audible artefacts present in the estimated sources, and the audio quality is in general lower than the target audio because of this.

5.4 Comparison

As is seen from both the objective and the subjective performance measures, the MLDM has the best performance of the three diffusion models. It achieves higher scores in all measures and tasks, except for the SoOS in the subjective evaluation. We have seen that the two models directly generating spectrograms struggle with accurately regenerating the target spectrograms, causing them to produce lower-quality audio. The SLDM performs significantly better than the SSDM, showing that the use of a latent space for the diffusion process is beneficial. The latent approach produces more stable and consistent outputs but still is unable to recreate spectrograms with a high enough precision. The superiority of the mask-based models in terms of audio quality can be attributed to the use of the mask-based approach, as any audio in the result must originate from the target audio spectrogram.

None of the trained diffusion models are able to surpass the objective performance of the simple UNet model. Based on what we have observed, it seems that the trained diffusion models are too inconsistent and generate too much noise in their predictions to be able to achieve the same quality as the non-diffusion method. This is likely due to the model being unable to correctly learn the amount of noise to be removed at each time step, causing the noise to still be present in the final output. These inconsistencies lead to a large amount of distortion, noise and artefacts in the spectrogram-generating approach, and to significantly more interference in the mask-based approach. The simple UNet model does not have this problem and thus is able to produce the estimated spectrograms with more precision. Better-trained models would likely benefit all diffusion approaches.

However, as predicted by the ideal values in table 5.1, the two spectrogram-generating models have the advantage of being able to eliminate interference completely. Both the SSDM and the SLDM scores higher than both mask-based models on their ability to suppress other sources in the subjective evaluation. Also, even though the SLDM does not surpass on the SI-SIR metric, it gets very close to that of the MLDM, and the shortcomings here might also be caused by the lower audio quality caused by the latent diffusion process. This shows that this approach does have its advantages in that it aims to generate the separated source directly. Therefore, an improved spectrogram-generative model could be superior in this aspect of MSS.

One way such a model could be improved is by using more advanced techniques in the diffusion model. The implementations used in this project are a relatively basic version of diffusion models, and many improvements have been researched that could improve its performance and consistency. For example, Hoogeboom et al. (2023) points out that when working in higher resolutions, a more complex noise schedule defined with respect to some lower reference resolution should be beneficial. The same paper also suggests utilizing drop-out, where values are randomly set to zero during training, or implementing a modified version of the UNet architecture.

We also note that there are faults in our use of the magnitude spectrogram representation in general. In fact, many of the added additional artefacts in the audio produced by all three models are also present in the audio produced by the ideal direct approach, meaning that these artefacts are produced by the STFT process. This may be due to the parameters used in the STFT, which were chosen to produce spectrograms of preferable shape for use in the autoencoder. Using a window size of $M = 511$ samples causes the parameters not to be COLA as discussed in section 2.1.1 If we had used $M = 512$, this would not have been the case (Zhivomirov et al., 2019). Tweaking the parameters of the STFT to ensure that they are COLA while still being able to encode it to a latent space should therefore be beneficial.

Chapter 6

Conclusion and Future Work

In this thesis, we have presented three methodologies of applying diffusion models to the task of separating musical sources. The first approach used a standard diffusion process to estimate the target source by direct estimation of the spectrogram. The second approach also used direct spectrogram estimation but did so through a latent diffusion process. The third approach also used a latent diffusion model to generate TF masks for application to the mixture spectrogram in order to obtain the estimated source. The three produced models were compared to a non-diffusion UNet model.

We now summarize the achievements of this thesis in reference to the three Research Questions formulated in section 1.2. RQ1 asks to what extent diffusion models can surpass the performance of existing models such as CNNs and DNNs in the domain of musical source separation. RQ2 asks about the advantages and limitations of directly generating the separated sources compared to using a mask-based approach. RQ3 asks about the advantages and limitations of using latent diffusion models compared to standard diffusion models for MSS.

While the three diffusion models were able to complete the task in varying degrees, the overall quality of their estimation did not surpass that of the UNet. As such, in regards to **RQ1**, it can be concluded that the diffusion models presented in this thesis were not able to surpass the performance of existing models in the task of MSS. However, that does not conclude that diffusion models are unfit for this application, as it is possible that other methodologies, more training, or more optimized architectures could lead to improved performance.

Furthermore, relating to **RQ2**, we have shown that the use of TF masks in MSS does have its limitations in terms of its ability to suppress unwanted sources. This is a problem that the approach of directly generating the separated source does not struggle with to the same degree, as it does not need to eliminate any other sources already present in the spectrogram. The limitation of this approach, however, is the complexity of accurately generating the audio data, which leads to much more distortion, noise and added artefacts than in the mask-based approach. If this problem were to be completely eliminated, the direct approach would likely be the superior one.

Finally, **RQ3** concerns the comparison of latent diffusion models to standard diffusion models for use in MSS. It has been shown that the standard diffusion model is both slower and less accurate than the latent diffusion model in this application. The standard diffusion model generated spectrograms containing large amounts of noise and distortion, reducing the quality considerably. On the other hand, both the latent diffusion models were able to generate spectrograms of higher quality, though they still produced inconsistencies that had negative effects on the final results.

In terms of future work, there is still much to be explored in the utilization of diffusion models in MSS. Further research could focus on achieving more consistency in the output of the diffusion models, and thereby look to achieve more accurate estimations in both the direct spectrogram approach and the mask-based approach. The diffusion models implemented in this project were fairly simplistic, and there exists a number of different methodologies and techniques for diffusion models that were not employed in this project that could have produced an improved result. For example, one might achieve better performance from the standard diffusion model by utilizing some of the methods that have been shown to be beneficial when working in high resolutions. These include using a different noise schedule, adding drop-out in the neural network, or using a different configuration for the UNet, as suggested by Hoogeboom et al. (2023). Better consistency might also be achieved by lowering the dimensions of the input spectrograms, for example by dividing the audio into smaller parts.

In the latent diffusion approach, using an autoencoder specifically trained on musical spectrograms may be beneficial and should therefore be explored. This will likely provide a more accurate representation of the audio in the latent space, and should therefore eliminate most of the error associated with the encoding operation. Thus, one could better utilize the faster and more stable properties of the latent diffusion model compared to the standard diffusion model. We also suggest attempting to eliminate the artefacts produced by the STFT by using a different set of parameters for this process.

References

Amit, Tomer et al. (2022). *SegDiff: Image Segmentation with Diffusion Probabilistic Models*. arXiv: `2112.00390 [cs.CV]`.

Avrahami, Omri, Ohad Fried, and Dani Lischinski (2023). *Blended Latent Diffusion*. arXiv: `2206.02779 [cs.CV]`.

Batzolis, Georgios et al. (2021). *Conditional Image Generation with Score-Based Diffusion Models*. arXiv: `2111.13606 [cs.LG]`.

Brocal, Gabriel Meseguer and Geoffroy Peeters (2019). "Conditioned-U-Net: Introducing a Control Mechanism in the U-Net for Multiple Source Separations". In: DOI: `10.5281/ZENODO.3527766`. URL: `https://zenodo.org/record/3527766`.

Cano, Estefania et al. (2019). "Musical Source Separation: An Introduction". In: *IEEE Signal Processing Magazine* 36.1, pp. 31–40. DOI: `10.1109/MSP.2018.2874719`.

Chandna, Pritish et al. (2017). "Monoaural audio source separation using deep convolutional neural networks". In: *Latent Variable Analysis and Signal Separation: 13th International Conference, LVA/ICA 2017, Grenoble, France, February 21-23, 2017, Proceedings 13*. Springer, pp. 258–266.

Chen, Nanxin et al. (2020). *WaveGrad: Estimating Gradients for Waveform Generation*. arXiv: `2009.00713 [eess.AS]`.

Choi, Jooyoung et al. (2021). *ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models*. arXiv: `2108.02938 [cs.CV]`.

Demirel, Emir, Sven Ahlback, and Simon Dixon (2020). *Automatic Lyrics Transcription using Dilated Convolutional Neural Networks with Self-Attention*. arXiv: `2007.06486 [eess.AS]`.

Emiya, Valentin et al. (2011). "Subjective and objective quality assessment of audio source separation". In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.7, pp. 2046–2057.

Grais, Emad M. and Mark D. Plumbley (2017). *Single Channel Audio Source Separation using Convolutional Denoising Autoencoders*. arXiv: `1703.08019 [cs.SD]`.

Hirano, Masato et al. (2023). *Diffusion-based Signal Refiner for Speech Separation*. arXiv: `2305.05857 [eess.AS]`.

Ho, Jonathan, Ajay Jain, and Pieter Abbeel (2020). "Denoising diffusion probabilistic models". In: *Advances in Neural Information Processing Systems* 33, pp. 6840–6851.

Hoogeboom, Emiel, Jonathan Heek, and Tim Salimans (2023). *simple diffusion: End-to-end diffusion for high resolution images*. arXiv: `2301.11093 [cs.CV]`.

Huang, Po-Sen et al. (2014a). "Deep learning for monaural speech separation". In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 1562–1566.

— (2014b). "Singing-Voice Separation from Monaural Recordings using Deep Recurrent Neural Networks." In: *ISMIR*, pp. 477–482.

— (2015). "Joint optimization of masks and deep recurrent neural networks for monaural source separation". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23.12, pp. 2136–2147.

Jansson, Andreas et al. (2017). "Singing Voice Separation with Deep U-Net Convolutional Networks". In: *International Society for Music Information Retrieval Conference*.

Jeong, Myeonghun et al. (2021). *Diff-TTS: A Denoising Diffusion Model for Text-to-Speech*. arXiv: `2104.01409 [eess.AS]`.

Kadandale, Venkatesh S. et al. (2020). *Multi-channel U-Net for Music Source Separation*. arXiv: `2003.10414 [cs.SD]`.

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

Kong, Zhifeng et al. (2021). *DiffWave: A Versatile Diffusion Model for Audio Synthesis*. arXiv: `2009.09761 [eess.AS]`.

Landau, H.J. (1967). "Sampling, data transmission, and the Nyquist rate". In: *Proceedings of the IEEE* 55.10, pp. 1701–1706. DOI: `10.1109/PROC.1967.5962`.

Le Roux, Jonathan et al. (2019). "SDR–half-baked or well done?" In: *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 626–630.

Li, Rui et al. (2020). *Linear Attention Mechanism: An Efficient Attention for Semantic Segmentation*. arXiv: `2007.14902 [cs.CV]`.

Liu, Jinglin et al. (2022). *DiffSinger: Singing Voice Synthesis via Shallow Diffusion Mechanism*. arXiv: `2105.02446 [eess.AS]`.

Liu, Zhuang et al. (2022). *A ConvNet for the 2020s*. arXiv: `2201.03545 [cs.CV]`.

Lutati, Shahar, Eliya Nachmani, and Lior Wolf (2023). *Separate And Diffuse: Using a Pretrained Diffusion Model for Improving Source Separation*. arXiv: `2301.10752 [eess.AS]`.

Mariani, Giorgio et al. (2023). *Multi-Source Diffusion Models for Simultaneous Music Generation and Separation*. DOI: `10.48550/ARXIV.2302.02257`. URL: `https://arxiv.org/abs/2302.02257`.

Mosser, Lukas, Olivier Dubrule, and Martin J. Blunt (2017). *Stochastic reconstruction of an oolitic limestone by generative adversarial networks*. arXiv: `1712.02854 [cs.CV]`.

Parmar, Gaurav et al. (2023). *Zero-shot Image-to-Image Translation*. arXiv: `2302.03027 [cs.CV]`.

Popov, Vadim et al. (2021). *Grad-TTS: A Diffusion Probabilistic Model for Text-to-Speech*. arXiv: `2105.06337 [cs.LG]`.

Rafii, Zafar et al. (2018). "An Overview of Lead and Accompaniment Separation in Music". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26.8, pp. 1307–1335. DOI: `10.1109/TASLP.2018.2825440`.

Rombach, Robin et al. (2022). "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). *U-Net: Convolutional Networks for Biomedical Image Segmentation.* DOI: `10.48550/ARXIV.1505.04597`. URL: `https://arxiv.org/abs/1505.04597`.

Saharia, Chitwan, William Chan, et al. (2022). "Palette: Image-to-Image Diffusion Models". In: *ACM SIGGRAPH 2022 Conference Proceedings.* SIGGRAPH '22. Vancouver, BC, Canada: Association for Computing Machinery. ISBN: 9781450393379. DOI: `10.1145/3528233.3530757`. URL: `https://doi.org/10.1145/3528233.3530757`.

Saharia, Chitwan, Jonathan Ho, et al. (2022). "Image super-resolution via iterative refinement". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Sasaki, Hiroshi, Chris G. Willcocks, and Toby P. Breckon (2021). *UNIT-DDPM: UNpaired Image Translation with Denoising Diffusion Probabilistic Models.* arXiv: `2104.05358 [cs.CV]`.

Scheibler, Robin et al. (2022). *Diffusion-based Generative Speech Source Separation.* DOI: `10.48550/ARXIV.2210.17327`. URL: `https://arxiv.org/abs/2210.17327`.

Simpson, Andrew JR, Gerard Roma, and Mark D Plumbley (2015). "Deep karaoke: Extracting vocals from musical mixtures using a convolutional deep neural network". In: *Latent Variable Analysis and Signal Separation: 12th International Conference, LVA/ICA 2015, Liberec, Czech Republic, August 25-28, 2015, Proceedings 12.* Springer, pp. 429–436.

Sohl-Dickstein, Jascha et al. (2015). "Deep unsupervised learning using nonequilibrium thermodynamics". In: *International Conference on Machine Learning.* PMLR, pp. 2256–2265.

Song, Jiaming, Chenlin Meng, and Stefano Ermon (2022). *Denoising Diffusion Implicit Models.* arXiv: `2010.02502 [cs.LG]`.

Stoller, Daniel, Sebastian Ewert, and Simon Dixon (2018). *Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation.* arXiv: `1806.03185 [cs.SD]`.

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)*, pp. 2135–2139. DOI: `10.1109/ICASSP.2015.7178348`.

Vaswani, Ashish et al. (2017). *Attention Is All You Need.* arXiv: `1706.03762 [cs.CL]`.

Vincent, E., R. Gribonval, and C. Fevotte (2006). "Performance measurement in blind audio source separation". In: *IEEE Transactions on Audio, Speech, and Language Processing* 14.4, pp. 1462–1469. DOI: `10.1109/TSA.2005.858005`.

Wolleb, Julia, Robin Sandkühler, Florentin Bieder, and Philippe C. Cattin (2022). *The Swiss Army Knife for Image-to-Image Translation: Multi-Task Diffusion Models.* arXiv: `2204.02641 [cs.CV]`.

Wolleb, Julia, Robin Sandkühler, Florentin Bieder, Philippe Valmaggia, et al. (2022). "Diffusion models for implicit image segmentation ensembles". In: *Inter-*

national Conference on Medical Imaging with Deep Learning. PMLR, pp. 1336–1348.

Wu, Chen Henry and Fernando De la Torre (2022). *Unifying Diffusion Models' Latent Space, with Applications to CycleDiffusion and Guidance*. arXiv: `2210.05559 [cs.CV]`.

Yang, Ling et al. (2023). *Diffusion Models: A Comprehensive Survey of Methods and Applications*. arXiv: `2209.00796 [cs.LG]`.

Zhivomirov, Hristo et al. (2019). "On the development of STFT-analysis and ISTFT-synthesis routines and their practical implementation". In: *TEM Journal* 8.1, pp. 56–64.

NTNU
Norwegian University of
Science and Technology