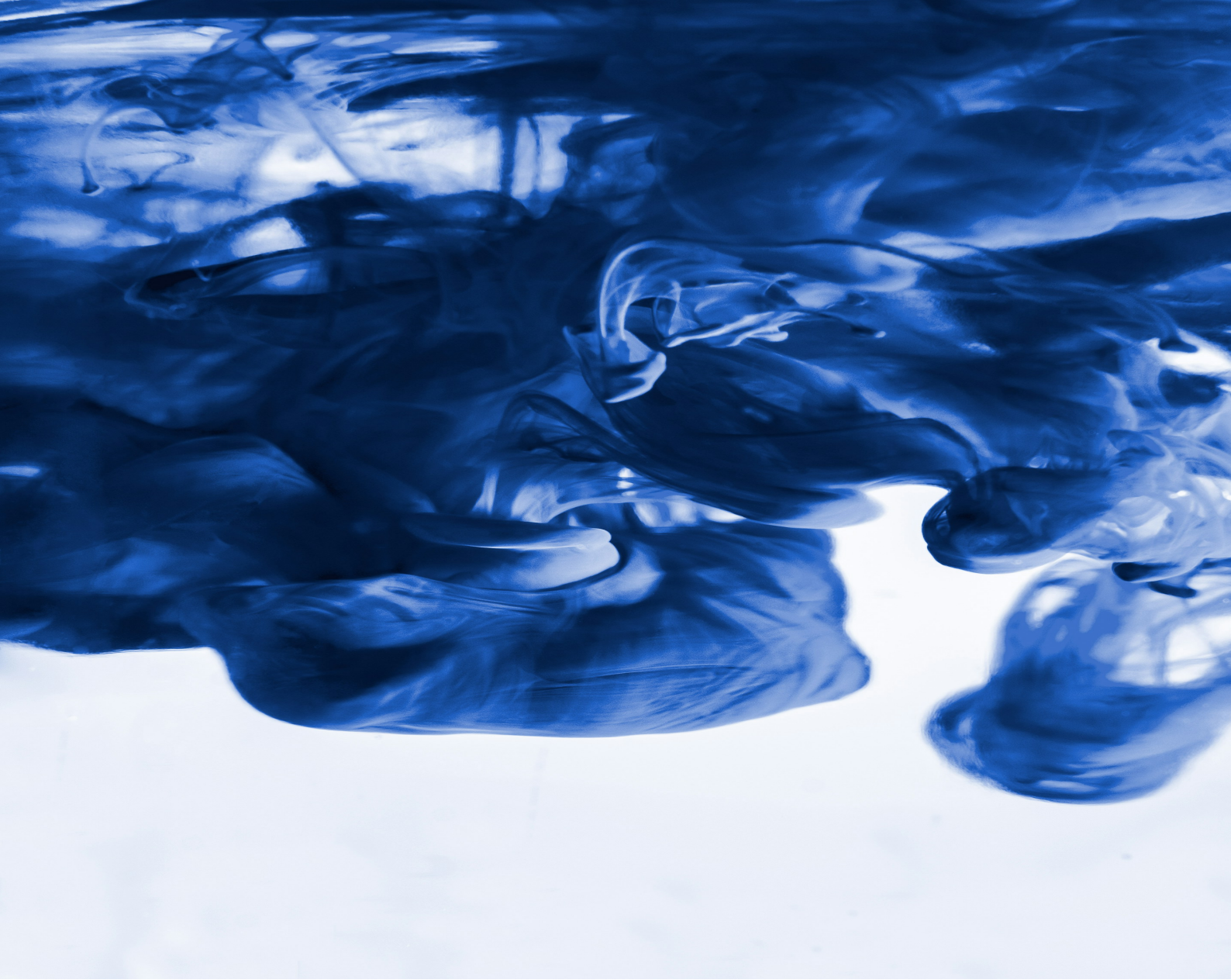


The effect of different time-embeddings on an end-to-end diffusion speech enhancement model

Magnus Munk Jensen

Electronic Systems 2024-06

Master's Thesis

This page is blank on purpose

Electronic systems
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
The effect of different time-embeddings on an end-to-end diffusion speech enhancement model

Theme:
ES10 Master Thesis

Project Period:
Spring 2024

Project Group:
1029c

Participant(s):
Magnus Munk Jensen

Supervisor(s):
Jesper Rindom Jensen
Tobias Piechowiak
Diego Caviedes-Nozal

Page Numbers: 84

Date of Completion:
31st of May 2024

Abstract:

This research proposes an end-to-end diffusion model for speech enhancement, trained directly on raw audio waveforms. While aiming to achieve performance comparable to existing methods that rely on Short-Time Fourier Transform (STFT) representations, the model utilizes a U-Net structure with a time step embedding. Here, the embedding leverages an existing technique but applies it in a novel way for speech enhancement within a diffusion model framework. This embedding facilitates the model's awareness of its position within the diffusion process, potentially improving performance.
The results demonstrate that incorporating the time step embedding is a key factor, significantly enhancing the model's capabilities. However, the model's performance remains below current state-of-the-art methods like SGMSE and Facebook Demucs for speech enhancement.

Summary

Noise is an unavoidable part of recording an acoustic signal - including recording speech for amplification in hearing assistive devices such as hearing aids. It is though known, that noise vastly increase the listening effort for the user of such hearing aids. Therefore, speech enhancement is a crucial part of a good experience with hearing aids.
Speech enhancement is a well-known concept, explored both in statistical signal processing, with solutions such as Wiener filtering, but is also a highly researched topic within neural networks. Speech enhancement is possible with classical deep neural networks, such as recurrent neural networks, where speech is mapped from noisy, to clean. Neural networks can also be used in cooperation with classical signal processing, example by estimating wheter there is speech present or not, in order to only process the signal when necessary.
Speech enhancement is though also possible with a newer era of neural networks, namely generative models. Generative models does not map noisy speech to clean speech, but instead learns how to generate new speech. Multiple such models exist, eg. Generative Adversarial Networks, and Autoencoders.
This research is looking into how a third type, generative diffusion model, can be developed as an end to end model. It tries to achieve this, by utilizing existing neural network architectures, not initially designed to work within a diffusion context. The architecture is then optimized, by making it diffusion-aware, in order to improve upon performance.
Evaluating a generative model is not as simple, as looking at the Signal to Noise ratio of the clean speech compared to the enhanced speech. This is due to the fact, that a generative model generates the speech. This means, that it might sound different, than the true clean speech - but even if it do, it might give the user a better experience, as there are reduced levels of noise. This scenario will though not necessarily perform well if only judged by Signal to Noise ratio. Therefore, the model is evaluated using both specifically chosen objective measures, and a subjective listening experiment to ensure that it is evaluated appropriately.
From the objective metrics it is shown, how SGMSE and Facebook Demucs outperform the proposed end-to-end model. If the end-to-end models enhanced speech is evaluated separately as speech and background signal, it is evident from evaluation with DNSmos that the quality of the speech is higher from the end-to-end model than what Facebook Demucs achieve, being on par with SGMSE.
This is also evident in the subjective listening test, where on average the proposed end-to-end model is close to performing as well as Facebook Demucs, even outperforming it at specific input SnR scenarios.
From the results, an apparent problems has been identified - the model was not able remove all of the white noise. It is proposed, that this can be solved eg. by continuing training, as the proposed model training was stopped early. From this one can evaluate if training can reduce any of the last white noise. In order to improve

further on the model, a study into how the scheduler affects it is also proposed, and finally, a new loss function is proposed, where the loss is determined in the frequency domain.
It can also be evaluated if an U-net architecture was the right choice, even though valid arguments was given for this. Other models, that also aims to develop and end-to-end model uses different architectures, but they differ from the proposed solution as they condition on STFT representations.

1 | Preface

This project report was typeset with the online LaTeX editor Overleaf.
The front page illustration is by Adrian on Unsplash illustrating liquid diffusion.
This is a 10th semester master thesis project by Magnus Munk Jensen, at Aalborg University (AAU) Electronic Systems. It is made in the period from February 2024 to May 2024.

Reading instructions

The reader is expected to have general knowledge of machine learning, including, different layer types, loss functions, and optimization methods.

The report makes use of references according to the Harvard method. The references will occur in the text in the following manner: [Surname/publisher, year (possible page number)], where the end of the report is a comprehensive list of literature.

Reference numbers are made as hyperlinks in the digital copy. Figures and tables are listed as the chapter number, followed by the figure/table/equation number, as an example Figure 7.2 is the second figure in Chapter 7. Figure and table explanatory text can be found below the figure and above tables. Furthermore, section references are made using the section numbering. The unit system used in the report is the SI system with a dot as the decimal separator. All figures without references are created by the author using the python package Matplotlib and the online flowchart and diagram maker draw.io.

Acknowledgment

I would like to express my sincere gratitude to GN Audio, especially to Tobias Piechowiak and Diego Caviedes Nozal.
Diego Caviedes Nozal built the framework, that made this study possible, and have through the project been ready to assist at all times, when i had problems.
Furthermore, both Tobias and Diego have offered their expertise, and guidance throughout the project.

Nomenclature

Symbols

$+$	Elemt-wise Addition
$\&$	Concatenation
$\cdot$	Elemt-wise multiplication
$\leftrightarrow$	Broadcasting

Abbreviations

CI	Cochlear Implant
GAN	Generative Adversarial Network
HA	Hearing Aid
HASQI	Hearing-Aid Speech Quality Index
LSTM	Long Short-Term Memory
MOS	Mean Opinion Score
MUSHRA	MUlti Stimulus test with Hidden Reference and Anchor
NCSN++	Noise Conditional Score Network
NN	Neural Network
PESQ	Perceptual Evaluation of Speech Quality
SDR	Signal-to-Distortion Ratio
SE	Speech enhancement
SI-SDR	Scale-Invariant Signal-to-Distortion Ratio
SnR	Signal-to-Noise ratio
STFT	Short-time Fourier transform
STOI	Short-Time Objective Intelligibility
VAE	Variational Autoencoders
WARPQ	Weighted Average of Relative Phone Quality

Contents

2 | Introduction

In a world, where noise is an apparent part of life, it also needs an apparent focus. In applications where a microphone captures a signal, it often captures unwanted noise or reverberations alongside the intended sound. This interference can stem from various sources, ranging from passing cars to background conversations. Aside from the noise picked up by the microphone, the speech signal may also suffer degradation from compression, clipping, and downsampling.
Speech enhancement techniques aim to mitigate this interference, enhancing recorded signals and ensuring clear communication in real-world settings.

In the hearing assistive device (HA) industry speech enhancement is widely adopted. People invalidated by hearing loss can get access to hearing aids, that enable them to hear - but studies have shown, that to reduce the listening effort, noise reduction algorithms must be used [Desjardins og Doherty, 2014; Sarampalis et al., 2009; Picou et al., 2013]. This probably stems from habit, meaning the HA user probably has not been used to background noises in the period where they have had degraded hearing. Therefore it is overwhelming to suddenly hear everything around you. Furthermore, it is shown that the presence of babble noise significantly diminishes the intelligibility of human speech for HA users [Park og Lee, 2016].

Consequently, emphasizing speech enhancement for users of hearing aids emerges as a prudent strategy. Besides being relevant in the hearing aid industry, it finds extensive application across various sectors including mobile phones, teleconferencing systems, and speech recognition.

Speech enhancement poses a significant challenge for two primary reasons [Benesty et al., 2005]. Firstly, the nature of the signal can undergo variations over time and across different scenarios. These variations may be due to factors such as fluctuating background noise levels, differences in microphone characteristics, and changes in recording conditions. Secondly, the performance criteria for speech enhancement can vary significantly. For instance, the emphasis might be on maximizing speech clarity and minimizing background noise. Therefore, developing speech enhancement techniques that can meet the diverse requirements of different scenarios, such as when used with hearing assistive devices, poses a significant challenge for researchers and engineers.

Several methods for speech enhancement already exist and are widely adopted. Such methods include Wiener filtering, spectral subtraction, statistical model-based approaches, and subspace algorithms. Wiener filtering applies a linear filter to minimize the Mean Squared Error between the estimated clean signal and the observed noisy signal [Haykin, 2014; Benesty et al., 2005]. Spectral subtraction aims to estimate the noise spectrum and subtract it from the observed signal in the frequency domain [Berouti et al., 1979]. Statistical model-based methods utilize probabilistic models

to distinguish between signal and noise components[Ephraim, 1992]. Subspace algorithms exploit the subspace structure of the signal and noise to separate them effectively [Ephraim og Van Trees, 1995].

One limitation of these methods is that, even though they handle noise reduction rather well, they do not directly address reverberation [Benesty et al., 2005]. The methods might improve upon this indirectly by enhancing the overall signal-to-noise ratio (SnR), but their primary focus remains on noise reduction. However, it is crucial to focus on reverberation as well, particularly because it significantly impacts speech intelligibility for hearing-impaired individuals using assistive devices [Kokkinakis et al., 2011]. Therefore, there is a pressing need to develop algorithms specifically designed to handle reverberation. Fortunately, such algorithms already exist, like the single-channel blind dereverberation method, based on the harmonicity of speech signals [Benesty et al., 2005]. This method approximates an inverse room transfer function, resulting in high-quality dereverberation [Benesty et al., 2005].

In order to further improve speech enhancement, Neural Networks (NN) are utilized. The idea of using NN is not new, dating back to the 1980s, when researchers started to use NN for noise reduction. In 1988 Waibel and Tamura published an article where they used a four-layered feed-forward network to make a model that maps a noisy signal to a noise-free signal [Tamura og Waibel, 1988]. They showed, that they were able to make a NN model that worked on both new signals, and noise that differed from the noise present at training.

In recent times, there has been increased interest in using generative models. Deep Neural Networks (DNNs), Recurrent Neural Networks, and similar models have long been at the forefront of speech enhancement models. Despite the rise in popularity of generative models, these traditional architectures remain widely utilized, particularly in edge applications where their efficiency and effectiveness shine. Classical models like DNNs has also been used in combination with classical signal processing, where the DNN estimate something, eg. if speech is present, in order to optimize the signal processing [Tao et al., 2023].
The approach for the aforementioned non-generative model is, that they directly map a noisy signal to a corresponding clean signal. This approach can be simpler to both train and implement compared to the generative models, but they often struggle with capturing variability and complexity in speech enhancement.
Generative models distinguish themselves from non-generative models by not just transforming a noisy signal into a clean one. Rather, they approximate the distribution of clean speech signals. Utilizing this learned distribution, they can produce novel, improved speech samples from noisy inputs. One advantage of this methodology, beyond its enhanced performance, is its capability to reconstruct lost speech, such as that caused by clipping, saturation, or bandwidth constraints. This is not possible for "classical" Deep Learning Approaches. Leading generative models for effective speech enhancement include Variational Autoencoders (VAEs) and Gener-

ative Adversarial Networks (GANs) [Fang et al., 2021]
VAEs, illustrated as a noise-aware VAE on figure 2.1, are in many ways similar to classical autoencoders. VAEs differ from an autoencoder in their latent space, which must adhere to a standard Gaussian distribution $z \sim \mathcal{N}(\mu_x, \sigma_x)$. Classical VAEs are not very suitable for speech enhancement as it is purely trained on clean speech. This results in a sensitivity to noise, such that when the SnR is low the performance is degraded [Fang et al., 2021]. Therefore more advanced VAEs have been designed, which are noise-aware [Fang et al., 2021]. Noise-aware VAEs are initially trained on clean speech. Following pre-training on clean speech, a *noise-aware encoder* is subsequently trained using both clean and noisy signals in a supervised manner. This process aims to achieve a latent space, $\hat{Z}$, that resembles the original encoders latent space, Z, as closely as possible. For inference, the noise-aware encoder is then used, but the decoder is the one trained on clean speech. These noise-aware VAEs are showing improved result compared to the regular VAE [Fang et al., 2021].

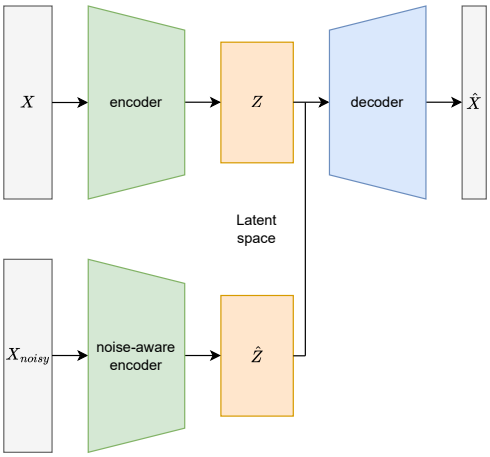


Figure 2.1: Block diagram of an noise-aware VAE

GANs are unsupervised models, that learn from a low-dimensional random latent space. GANs trained for speech enhancement works by training two NN. The process is shown in figure 2.2. One NN is called the generator, and the other the discriminator [Donahue et al., 2017; Asante et al., 2023]. The generator enhance the noisy speech, while the discriminator learns to distinguish between enhanced speech and true clean speech. This have shown to be effective, but there is also shortcomings [Donahue et al., 2017; Asante et al., 2023]. GANs can be unstable while training, risking collapse, cause the generated speech to contain a limited diversity aand therefor the potentioal for generating unrealistic speech and a risk that they do not converge [Goo, 2022].
Instead of GANs and VAEs mentioned earlier, there is another type of neural network known as Diffusion Probabilistic Models. These models have demonstrated promising outcomes in producing high-quality images [Dhariwal og Nichol, 2021a]. In this study, we'll explore the potential of utilizing diffusion networks in improving speech quality.

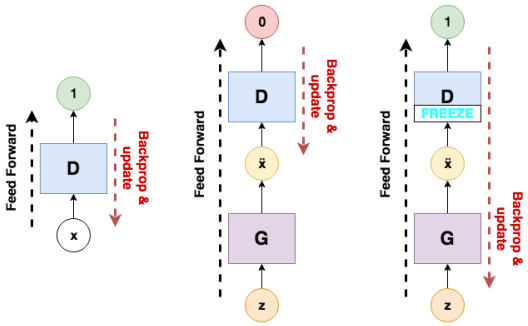


Figure 2.2: Training process of a SEGAN[Pascual et al., 2017]

2.1 Research question

Score-based Generative Diffusion Models are already shown to be better at image synthesis than GANs[Dhariwal og Nichol, 2021a], be able to denoise images[Ho et al., 2020a], and have been shown to be effective for speech enhancement as well [Welker et al., 2022a; Richter et al., 2023].
This research will look further into diffusion models, for speech enhancement. What is noticeable about current studies is, that they all work completely or partly in the Short-time Fourier transform (STFT) frequency domain, either in training and inference or in training[Richter et al., 2023; Welker et al., 2022a; Lemercier et al., 2023; Lu et al., 2021, 2022; Kong et al., 2021]. This is probably because, diffusion models are initially made for image synthesis, making it easy to adapt from images into spectrograms.
Thus, this research will focus on adapting an architecture to enable a diffusion model to directly process raw audio waveforms rather than STFT representations. The reason for this is, to simplify the pipeline when the model is deployed, as it removes the need for pre-processing to get the data into the STFT domain and removing the post-processing to get it back into the time domain. Furthermore, the research aims to simplify the training process, as this should no longer be dependent on STFT representations. By learning directly on the raw audio waveform, the model might capture more nuanced features, and temporal dependencies that otherwise might be lost in the transformation into the STFT domain.
The research will address how well a new end-to-end model scores in specific chosen evaluation metrics, and a listening experiment, compared to current SE models. The research question is:

How does incorporating diffusion-aware training techniques into non-diffusion networks in the waveform domain affect speech quality compared to diffusion models that use Short-Time Fourier Transform (STFT) representations?

3 | Diffusio-based speech Enhancement

This chapter aims to give the necessary background about diffusion-based generative models for speech enhancement. It does not aim to show all the underlying math, as this is well described in the literature [Ho et al., 2020b; Dhariwal og Nichol, 2021b; Lu et al., 2021, 2022; Welker et al., 2022a].
The objective of a diffusion model for SE is, to denoise a noisy object. In this brief background section, the object will be assumed to be a sound, but it is not limited to such, which is easily seen in the literature as diffusion is original made for image synthesis [Dhariwal og Nichol, 2021a].
The process is based on two sub processes. A forward process, called the diffusion process, and a reverse process. The explanation of the sub processes is based on the explanation by S. Welker in [Welker et al., 2022a], which is based on the work in the CDiffuSE [Lu et al., 2022]. Welker uses a score-based generative diffusion model, to enhance a noisy speech signal. For the mathematical proofs please refer to the article by S. Welker [Welker et al., 2022a].

3.1 Diffusion process

The forward process take place before training the model. It transforms a clean object, $\mathbf{x}_0$, into pure Gaussian noise by gradually adding small amount of Gaussian noise to it. This process of adding a small amount of noise is repeated for T steps, until the object has turned into Gaussian noise, $\mathbf{x}_T$ [Dhariwal og Nichol, 2021b; Kong et al., 2021; Ho et al., 2020b]. The process is a so-called Markov chain, meaning that it is a sequence of possible events, where each event is only dependent on the last event. The process is illustrated in figure 3.1:

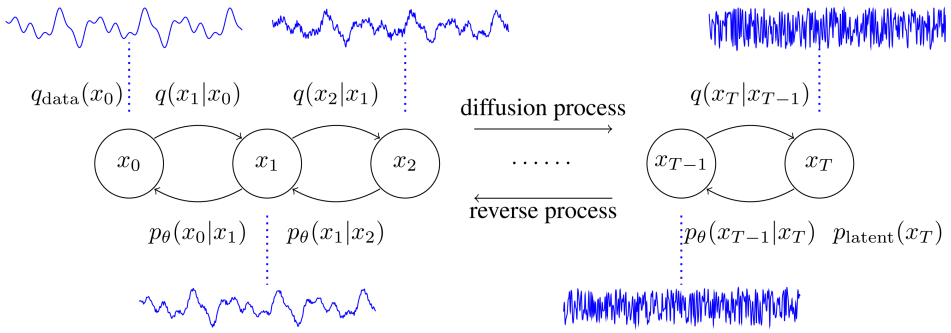


Figure 3.1: The diffusion process, shown with a audio wave [Kong et al., 2021]

To adapt the diffusion process to a use case where the goal is SE the diffusion process has been defined as in equation 3.1, a so-called stochastic differential equation (SDE) [Welker et al., 2022a]. This process turns $\mathbf{x}_0$ into $\mathbf{x}_T$ as illustrated in figure 3.1.

$$d\mathbf{x_t} = f(\mathbf{x_t}, \mathbf{y})dt + g(t)d\mathbf{w} \tag{3.1}$$

where,
$f(\mathbf{x_t}, \mathbf{y})$ a function called the drift coefficient, that determines how the process state changes over time, based on the current state x_t
t is a variable describing the process
$\mathbf{y}$ is noisy speech
$g(t)$ is a function called the diffusion coefficient, which control the amount of Gaussian white noise added at each time step
w is a standard Wiener process.

The drift coefficient is further defined as in equation 3.2, and the diffusion coefficient as in 3.3 [Welker et al., 2022a].

$$f\mathbf{x_t}, \mathbf{y}) = \gamma(\mathbf{y} - \mathbf{x_t}) \tag{3.2}$$

$$g(t) = \sigma_{\min} \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^t \sqrt{2 \log \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)} \tag{3.3}$$

where,
γ is a constant called stiffness
σ_{min} and σ_{max} defines the noise schedule in the Wiener process.

3.2 Reverse process

The reverse process is the process of trying to estimate the Gaussian noise, such that the original signal can be restored again. Like the diffusion process, it is also a Markov process, but from the latent space $\mathbf{x}_T$, to $\mathbf{x}_0$. The reverse process is defined as in equation 3.4, which is called the plug-in reverse Stochastic Differential Equation (SDE)[Welker et al., 2022a].

$$d\mathbf{x_t} = (-f\mathbf{x_t}, \mathbf{y}) + g(t)^2 \mathbf{s}_\theta(\mathbf{x_t}, \mathbf{y}, t))dt + g(t)d\bar{\mathbf{w}} \tag{3.4}$$

where,
$\mathbf{s}_\theta(\mathbf{x_t}, \mathbf{y}, t)$ is the score model, which stems from $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x_t}|\mathbf{y})$
θ is a set of parameters

3.3 Training objective

S. Welker shows, that because the SDE in equation 3.1 describes a Gaussian process, a training objective is defined as equation 3.5 [Welker et al., 2022a]. The training objective is based on the SDE and is used to estimate the score function, which is the gradient of the log probability density function. The score function helps guide the reverse diffusion process.

$$\arg\min_{\theta} E_{t,\mathbf{x_0},\mathbf{x_t}|\mathbf{x_0}} \left[\|\mathbf{s}_\theta(\mathbf{x_t}, t, \mathbf{y}) - \nabla_{\mathbf{x_t}} \log p_{0t}(\mathbf{x_t}|\mathbf{x_0}, \mathbf{y})\|_2^2 \right] \tag{3.5}$$

This training objective differs from the one proposed in CDiffuSE, [eq. (21) [Lu et al., 2022]]. CDiffuSE is another SE diffusion model, and the thing to notice is that CDiffuSE includes the term "$\mathbf{y} - \mathbf{x_0}$" which means the network is trained to remove some of the environmental noise in each step. The solution proposed by S. Welker instead simply conditions the input to the model on the noise version of the clean speech. This means that the model by S. Welker only has to estimate the artificially added Gaussian noise.

3.4 Training algorithm

In this section, the training algorithm are shown, utilizing the math discussed prior.

Algorithm 1 Diffusion Model Training

Parameters: Score model $s_\theta(\mathbf{x}_t, t, \mathbf{y})$

1: **for** $i = 1, 2, ..., N_{\text{iter}}$ **do**
2: Sample training set of pairs $(x_0, y) \sim q_{\text{data}}$
3: Sample diffusion time $t \sim \mathcal{U}(1, 2, ..., N_{\text{iter}})$
4: Sample noise signal $z \sim \mathcal{N}(0, I)$
5: Compute the gradient of the loss function:

$$\nabla_\theta L = E_{(\mathbf{x}_t,t),\mathbf{x}_0} \|s_\theta(\mathbf{x}_t, t, \mathbf{y}) - \nabla_{\mathbf{x}_t} \log p_0(\mathbf{x}_t|\mathbf{x}_0, \mathbf{y})\|_2^2$$

6: Update network parameters θ using the gradient $\nabla_\theta L$ with respect to θ, employing backpropagation
7: **end for**

4 | Training Methodology

Knowing the basic of how diffusion works, this chapter aims to discuss how the Neural Network (NN) is trained. This includes a description of the AllInOne framework from GNaudio, the clean speech dataset, the data processing to get noisy speech signals, and the motivation behind the different architectures for the end-to-end (E2E) network.
The ultimate objective is to develop a network that performs nearly as well as, or on par with, SGMSE. The code for SGMSE is publicly available on Github[Welker et al., 2022b], and is well described in the article by S. Welker [Welker et al., 2022a].

4.1 AllInOne framework

The AllInOne framework from GNaudio, is a framework built to make experiments within a diffusion context.
The AllInOne framework is built upon an existing framework called called *Hydra*, developed by Facebook Research. Hydra is a framework, which key feature is to generate hierarchical configurations dynamically [Yadan, 2019]. It does so by using hierarchical configuration files, adjustable from multiple sources. One key benefit of using Hydra is, that it is easy to make experiments without doing hard-coding.
Based on this, the AllInOne framework is constructed, incorporating a range of essential components:

- Data loader
- Diffusion model
- Diffusion Engine
- Evaluation

These features make it easy to incorporate new NN architectures, even those not originally designed for diffusion contexts, and simplify adjustments to the diffusion engine responsible for scheduling. The built-in data loader allows for different representations of the signal, and are able to create the necessary distorted signals from clean speech signals. In figure 4.1, a block diagram of the framework is depicted, showing all the processes.

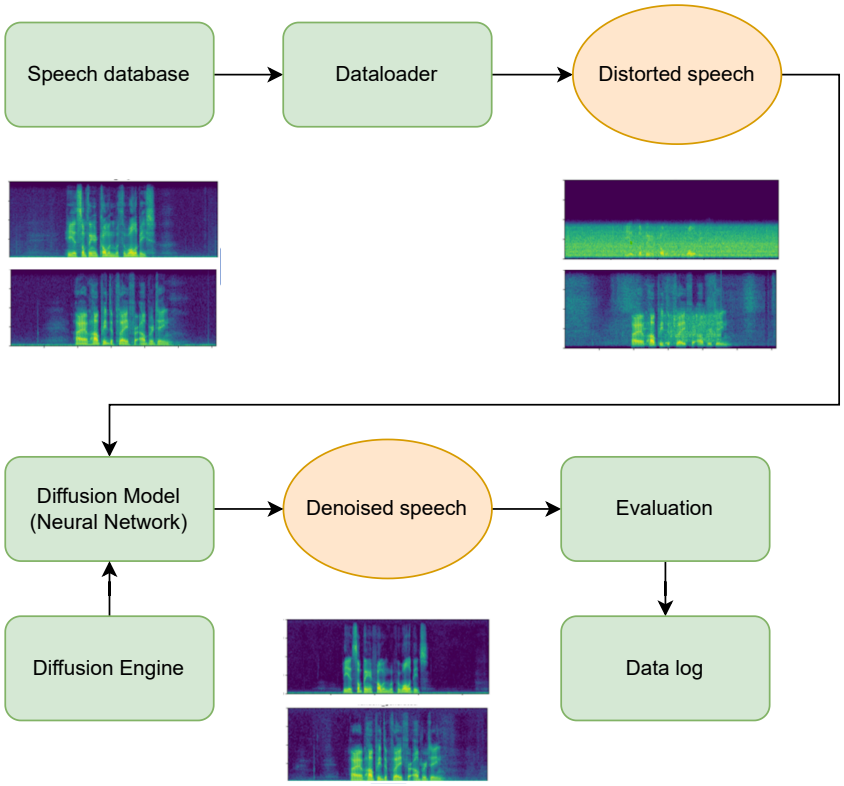


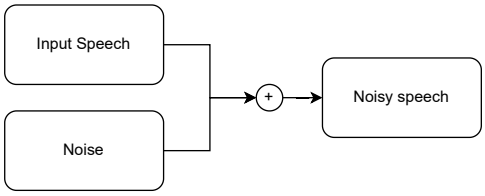
Figure 4.1: A blockdiagram showing the processes of the AllInOne framework. Block diagram based on figures by Diego Caviedes Nozal.

In the following sections, the data, data loader, and architecture will be discussed and investigated in more detail, as these will be key components in creating an end-to-end model.

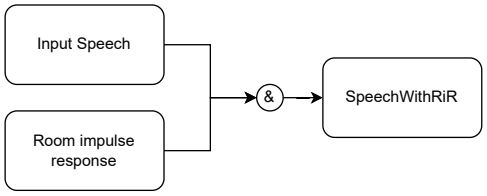
4.2 Data

The AllInOne framework has a built-in data loader, that allows for the use of only clean speech in the training, as it can distort speech on the fly. The dataset used for training the models is the open-source speech dataset from Centre for Speech Technology, called Voice Cloning Toolkit (VCTK).
The dataset is made from 109 native speakers of English, which reads around 400 sentences from a newspaper called The Herald [for Speech Technology Voice Cloning Toolkit, 2016; van den Oord et al., 2016].
Each speaker is assigned unique newspaper sentences, curated through a greedy algorithm aimed at maximizing both contextual relevance and phonetic diversity [for Speech Technology Voice Cloning Toolkit, 2016; van den Oord et al., 2016].
The dataset is exclusively a clean speech dataset, but as stated, the AllInOne framework is designed to generate distorted speech from clean speech signals. This ap-

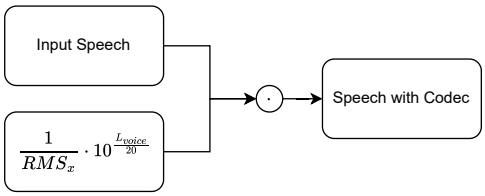
proach allows for controlled degradation of clean speech signals, making it easier to evaluate speech processing. Furthermore, it reduces the need for extensive data, as there is no requirement for both clean and distorted data to exist beforehand, only clean speech, as well as some noise signals.
The method by which the AllInOne framework distorts clean speech is illustrated in the block diagrams, depicted in figure 4.2. For all the speech distortions methods, it is true that it is user adjustable, meaning eg. for the bandwidth limited speech, the cut off frequencies can be adjusted accordingly to the use case.
It is also true, that each of the distortions can be combined, so that the generated speech both contains eg. ambient noise, and reverberation.
When combining distortions, it is important to consider the order of the non-linear distortions. It is assumed, that the order linear distortions should not matter, as it can be assumed that the rules of super positioning is true, as well as the rules for time linear systems is true. When the distortion in non-linear, like compression, it is no longer assumed that these rules are true.



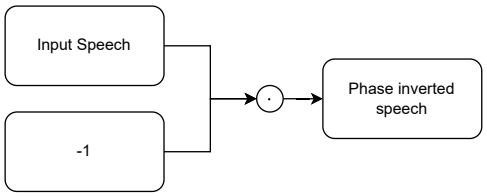
(a) Block diagram showing, how the ambient noise is added to the speech input



(b) Block diagram showing, how the room impulse response is convolved to the speech input



(c) Block diagram showing, how a codec is simulated



(d) Block diagram showing, how the speech is phase inverted

(e) Block diagram showing, how the speech can be bandwidth limited with a bandpass filter

Figure 4.2: Different on-the-fly methods of distorting the signal in the GN frameweork

The model will be trained, with ambient noise added to the clean speech. The range of Signal to Noise Ratio is adjustable within the AllInOne framework and is uniformly distributed between -5 and 50 dB skewed around lower values. The choice of training the model on ambient noise has already been indirectly discussed. It is well-known that ambient noise poses a challenge for users of hearing aid (HA), and noise cancellation helps reduce listening effort. Therefore, it is evident that removing ambient noise is a logical starting point.

4.2.1 Data representation

Data will be represented differently, in the baseline and the proposed networks. For SGMSE, the data is represented as STFT, as described in the SGMSE article [Welker et al., 2022a]. For the new network, which aims to have an end-to-end approach, the data is represented in the time domain, as an audio wave.
One of the key parts of getting the diffusion model to work for SE is, that the input signal, clean speech, must be conditioned on a noisy version of the speech [Welker et al., 2022a]. In this research, the clean speech is simply concatenated with a noisy version of the speech resulting in a 2 dimensional waveform. From prior research into conditioning, it can be argued it should not change anything, if the condition in concatenated or additioned to the input signal, but a small experiment shown in figure 4.3 shows, that if concatenation is replaced by element-wise addition the network trains significantly slower. It is not tested, if they will reach the same equilibrium at some point.

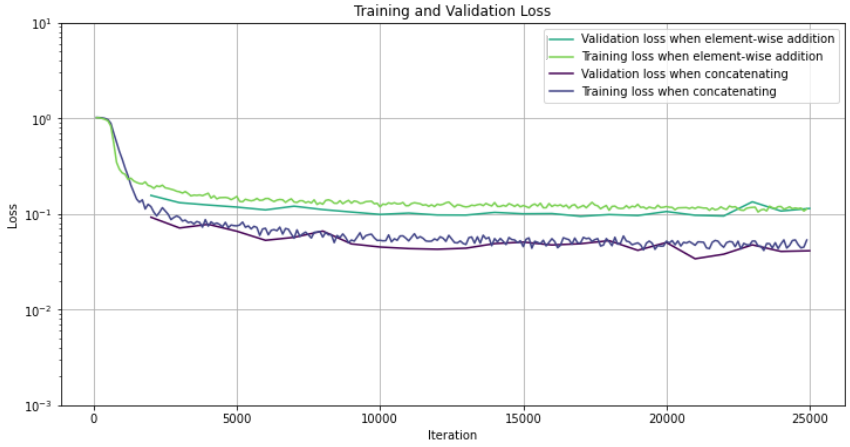


Figure 4.3: A short experiment, where the conditioning signal is either concatenated or element-wise addition to the clean speech.

4.3 Network architecture

The network has been trained with multiple different architectures, in an attempt to match the performance of SGMSE, proposed by S. Welker[Welker et al., 2022a]. SGMSE will be used as a baseline, and even though it is possible to train with the AllInOne framework, a checkpoint from the official GIT repo is used [Welker et al., 2022b].
SGMSE is based on an NCSN++ architecture depicted in figure 4.4. NCSN++ is a multi-resolution U-net structure, which has shown to be effective at generative tasks [Ronneberger et al., 2015; Welker et al., 2022a]. This is well-described in the original article, and will therefore only be briefly discussed in this research[Mittal et al., 2020].

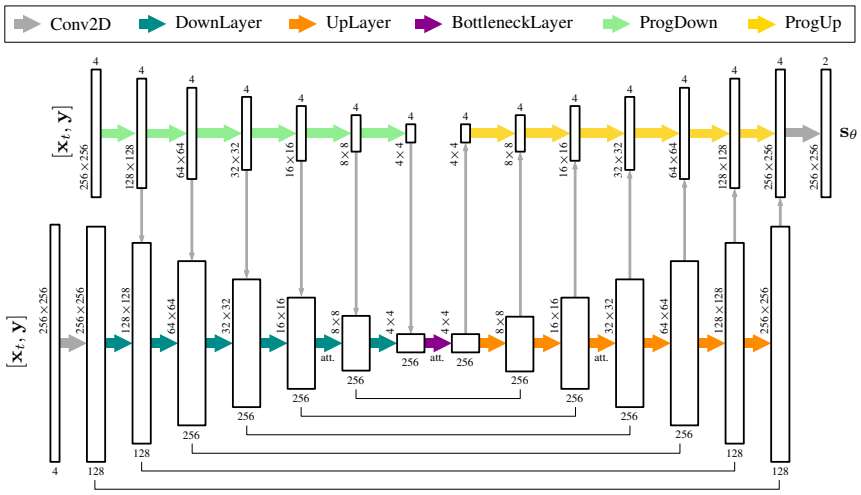


Figure 4.4: The NCSN++ architecture, as shown in [Welker et al., 2022a]

NCSN++ utilizes so-called Conv2D layers, similar to multiple other generative models for SE, like SEgan. The reason why they choose architectures that utilize the Conv2D layers is often, that the original model is made for image enhancement or image synthesis. Hence, the approach is to conduct a STFT on the speech signal, thereby transforming it from the time domain to the frequency domain. This transformation enables the creation of a spectrogram, a visual representation of the signals frequency components. It is expected, that it is easier to make a model designed for image processing work with the spectrograms. The apparent preference for image-oriented models, including those for SE, may explain their suboptimal performance when dealing directly with raw audio files in the time domain.
This research will try to obtain an end-to-end model, by using an architecture that is based on Conv1D layers. The reasoning for this is that these operate by convolving a convolution kernel with the input layer along a single spatial dimension, and not in multiple dimensions like Conv2D [Keras, 2022].
Many such architectures exists, showing good results for speech enhancement in

non-diffusion contexts. Examples of such architectures are Wave-U-Net for speech enhancement, Demucs for speech enhancement and Spiking Neural Networks [Stoller et al., 2018; Defossez et al., 2020; Riahi og Plourde, 2023]. It must be noted, that Demucs exist in multiple versions, that are also built with different architectures, that do not solely rely on Conv1D.
A unifying characteristic among these models is their foundation on the U-Net architecture, distinguished by specific modifications tailored to optimize performance for individual use cases.
It has already been described how SGMSE is using NCSN++, which is a multi-resolution U-net architecture. This means that NCSN++ has proven to perform well in a diffusion context, which is why it makes inherent sense to use a U-Net architecture, such as Wave-U-Net or Demucs for the end-to-end model.
As none of the U-Net models proposed is directly designed for a diffusion model, multiple U-Net models will be tested. The general structure of the U-net architecture is shown in figure 4.5.

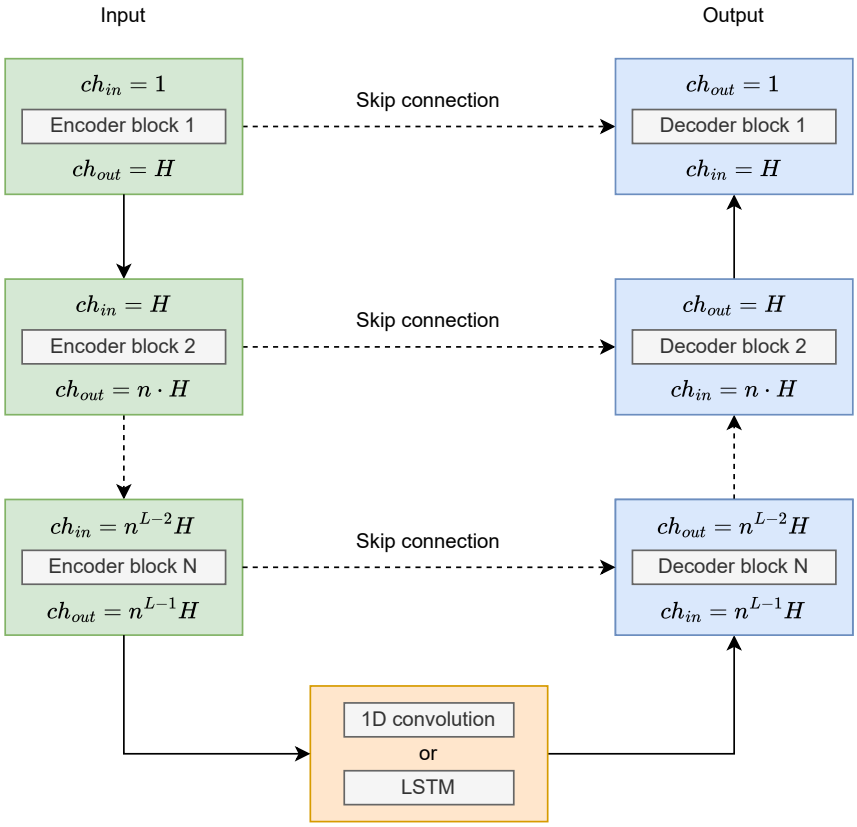


Figure 4.5: The general idea of a U-net architecture

Key points of the U-net architecture is the encoder-decoder structure, as shown in figure 4.5, where the encoder blocks is visualized with the green boxes, and the decoder blocks with the blue boxes. In this study, the U-net architecture is designed to facilitate easy modification of its details. This enables it to seamlessly integrate various established effective architectures, such as Demucs, Wave-u-net, etc.

4.3.1 Encoder and decoder

As described, the AllInOne framework, and especially the implemented U-Net architecture is made such that it is easy to test different architectures. The experimental work will be based on the architecture used in Demucs for speech enhancement, Wave-U-Net, and a combination of these. These architectures are chosen, as they are based on a U-Net structure, and Conv1D layers [Stoller et al., 2018; Defossez et al., 2020]. In the following, when "Wave-u-net" or "Demucs" is written, it solely refers to the architecture, which has been implemented in the AllInOne framework.

In-between all experiments the hyperparameters, and depth are kept constant, such that it is solely the architecture that influences the outcome of the experiment.

One key difference between Demucs and Wave-U-Net is in the bottleneck layer. Where Wave-U-Net has a regular Conv1D bottleneck layer, Demucs has a Long Short-Term Memory (LSTM) bottleneck layer. Networks with LSTM are known to capture long-range dependencies in sequential data well. The original signal is a speech signal, which is sequential data, with temporal dependencies. Due to this, it is expected that Demucs will perform better than Wave-U-Net. This is researched further in appendix A where the results from running Demucs and Wave-U-Net natively, with a simple time-embedding, are shown. From appendix A it is seen, that Demucs shows the lowest validation loss, also depicted in figure 4.6.
Based on the experiments, the choice for the encoder and decoder blocks, mirrors the implementation found in Demucs for Speech Enhancement [Defossez et al., 2020]. This architecture performs better than the Wave-U-Net architecture. Figure 4.7 displays a visual representation of the encoder and decoder from Demucs showing its structure. The bottleneck layer, as detailed in the Demucs for Speech Enhancement article, employs a LSTM [Defossez et al., 2020].
Based on this choice, a more sophisticated time-embedding will be investigated, to improve further upon the performance of the model.

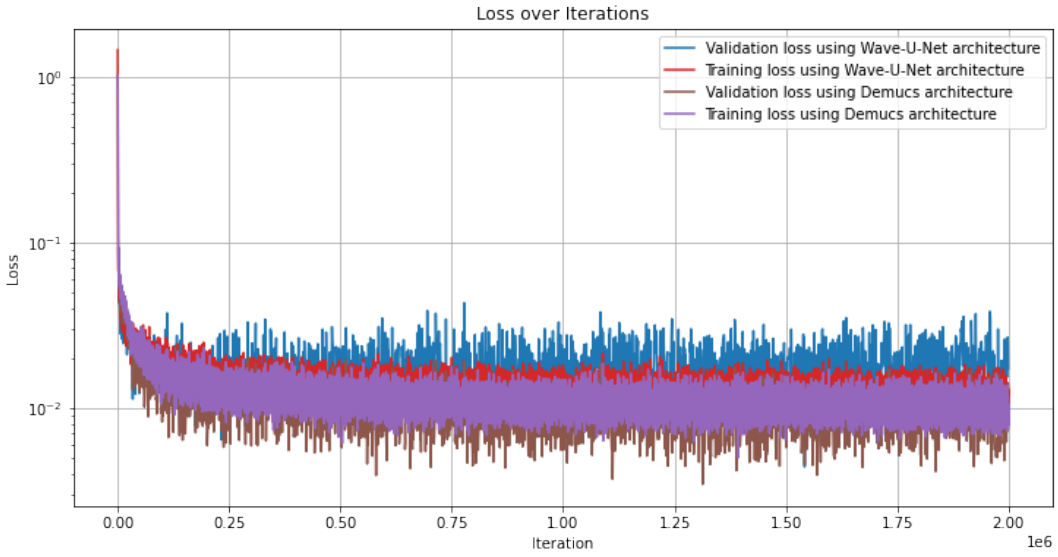


Figure 4.6: Comparison of the loss for Demucs and Wave-U-Net with simple time-embeddings.

4.3.2 Time embedding

From Diffwave, CDiffuSE and SGMSE, it is known that the diffusion-step "t" is a crucial part of the input to the model [Kong et al., 2021; Welker et al., 2022a; Lu et al., 2022]. In order to get better performance from the Demucs architecture, it will be investigated how recent embeddings can be used in a novel way, with Demucs in a diffusion context.
In SGMSE a Fourier embedding is used, transforming the timestep into an M-dimensional vector, which then can be integrated into every residual layer [Welker et al., 2022a].
In Diffwave, the time is embedded by inputting the time condition into a positional encoder and then applying three fully connected layers to the embedding. Lastly, it is broadcast over the length of the tensor and added to each input of every residual layer [Kong et al., 2021].
In the prior testing, the time step has been embedded in the simplest way possible imaginable in the beginning of the project. This embedding is shown to improve upon the Demucs architecture, shown in appendix B section B.1, where an improvement in loss is shown, by adding the time-embedding. The way, that it has been made, is by broadcasting the time condition over the length of the input tensor, from where they are concatenated together before inputting it into the network. The approach is shown in figure 4.8.
It is already shown that this simple time-embedding is able to improve upon the model when comparing loss, but more complex implementations of time-embedding are also investigated, where the time-step "t" is passed through a positional encoder before it is embedded into the NN.

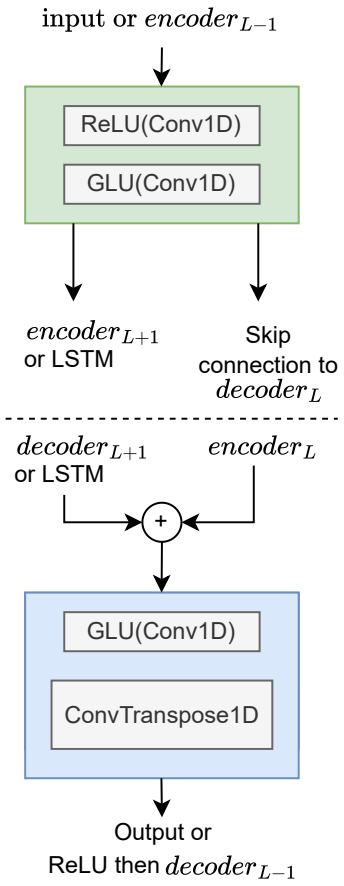


Figure 4.7: The encoder and decoder as depicted in the original Demucs paper.

The idea behind using a positional encoder is, that the time step "t" might not be able to capture the complexity of the relationship between the Gaussian noise, and the step in the diffusion process. Addutionally, when expanded into a higher dimensionality the model might generalize better.
Two different implementations are tested, both based on the positional encoder from DiffWave [Kong et al., 2021]. The encoder follows equation 4.1. In figure 4.9 an example of the result of the positional encoder is shown. This example is made with $t = 1, t = 2, t = 3$.

$$\begin{aligned} t_{\text{embedding}}(t) = [&\sin\left(10^{\frac{0\times4}{63}}t\right), \cdots, \sin\left(10^{\frac{63\times4}{63}}t\right), \\ &\cos\left(10^{\frac{0\times4}{63}}t\right), \cdots, \cos\left(10^{\frac{63\times4}{63}}t\right)] \end{aligned} \tag{4.1}$$

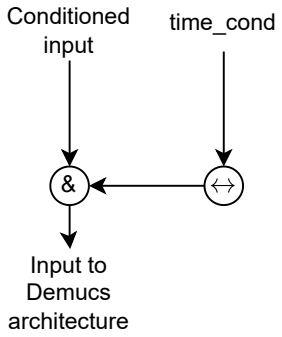


Figure 4.8: A simple embedding of time step "t"

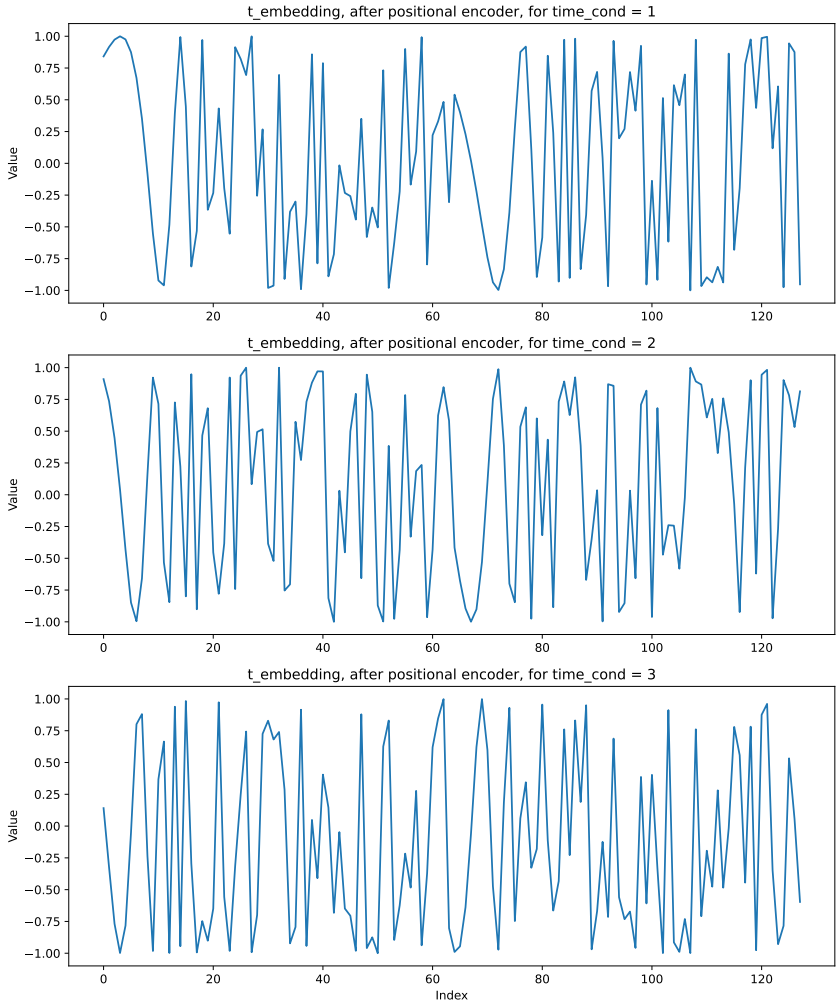


Figure 4.9: A figure showing how a simple time step "t" is transformed into higher dimension vector, where each $t_{embedding}$ is different from each other.

The embedding of $t_{embedding}$ into the network differ from DiffWave, and experiments are carried out with two different embeddings.
The method shown in figure 4.10 uses three fully connected layers, activated with ReLU, before broadcasting over the length of the input tensor and concatenating $t_{embedding}$ to the input of the network.

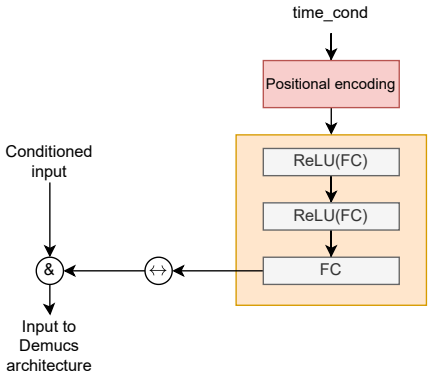


Figure 4.10: A more complex embedding of the time step "t" that utilize positional encoding

A more sophisticated embedding, depicted in figure 4.11, uses Swish instead of ReLU, and further inputs the $t_{embedding}$ into each encoder and decoder block.
The reason why multiple embeddings is tested, is that it is believed that giving the network an accurate knowledge of where in the Markov-chain it is, the better at estimating the white noise it will become.
In appendix B the results from testing the different time-embeddings are shown. It shows, that the time embedding shown in figure 4.11 is the better approach, which is also depicted in figure 4.12, where the training is stopped early in order to fine-tune the hyperparameters. As it is slower to train the final model, this was a necessity due to the time limitation in the thesis.

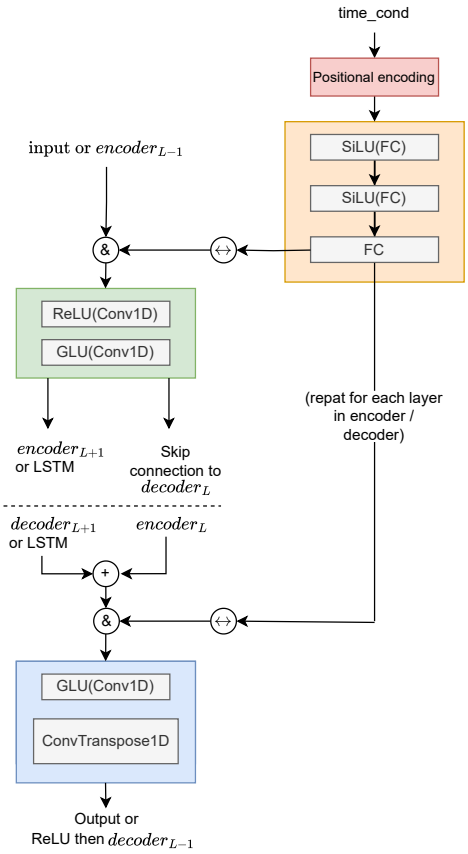


Figure 4.11: An embedding of timestep "t" that uses positional encoding, three fully connected layers, and concatenation into each encoder and decoder.

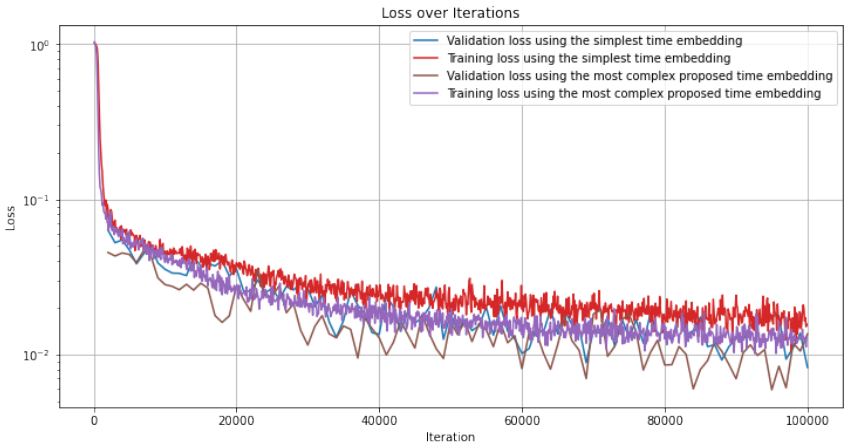


Figure 4.12: Loss curves for the simple time-embedding, and the complex time-embedding.

4.4 Training

The end-to-end model is trained on a machine with Nvidia GTX 1080TI with 11 gb of VRAM, and 64 gb of system RAM.
For all the above experiments, the following parameters, which are used in the diffusion process, have been chosen, such that this is constant between test:

- $\gamma = 1.5$
- $n_{steps} = 30$
- $\sigma_{min} = 0.05$
- $\sigma_{max} = 0.5$
- $t_\epsilon = 0.03$

All networks is trained with the AdamW optimizer, with a learning rate of 1e-4, and models that are not stopped early are trained for 2,000,000 iterations, with a fixed batch size of 8.
For the initial test with Demucs and Wave-U-Net, including the model without any time embedding, the hyperparameters are as follows:

- Initial hidden (H) size: 24
- Depth: 6
- Kernel size: 8
- Stride: 4
- Growth (n): 2

For the final end-to-end model, the hyperparameters has been tuned to the following:

- Initial hidden (H) size: 48
- Depth: 6
- Kernel size: 8
- Stride: 4
- Growth (n): 2

Please do note, that the final model also uses a GELU activation function, in the encoder and decoder as shown in figure 4.7, 4.8 and 4.11, whereas during the initial experiments, to keep things constant, all activation functions were of the ReLU type.

5 | Testing Methodology

The following chapter will describe how the models are evaluated. Evaluating a generative model requires more of a nuanced approach, than evaluating classical signal processing techniques, such as a Wiener filter.
A generative model is not bound to produce a signal equal to the input. Essentially, this implies that the resulting clean speech signal might differ from the input signal to reduce noise interference. This might not be a problem, as the speech signal can yield a better experience for the users anyway, but it creates a problem for how to evaluate it.
Classical methods, like SnR, might not give an accurate description of the speech signal, because the speech might be different from the input. In this case, the SnR might not show an improvement, but for the user, there can be an improvement.
Furthermore, it must be remembered that the speech signal is to be used by HA users. If this affects the evaluation metrics, it is also an important factor to include, and it might also mean that it is not enough with a single metric.
Therefore, a wide range of chosen metrics will be used, to ensure the optimum evaluation.
Further, an actual listening experiment will be conducted, to verify if the human test subjects align with the objective measurements, ensuring a comprehensive evaluation.

5.1 Evaluation metrics

A range of evaluation metrics will be used, to test the different models. These include Deep Noise Suppression Mean Opinion Score (DNSMOS), Hearing-Aid Speech Quality Index (HASQI), Short-Time Objective Intelligibility (STOI), and Scale-Invariant Signal-to-Distortion Ratio (SI-SDR).
DNSMOS is a rather new evaluation metric, published in 2020 [Reddy et al., 2021]. The metric is developed to tackle scalability challenges in subjective testing methodologies, and generative models. It is a non-intrusive measure, based on a neural network, emulating the perceptual judgment of human test subjects [Reddy et al., 2021]. Trying to tackle this problem is not new, and the well-known metric Perceptual Evaluation of Speech Quality (PESQ) is also developed to emulate the perceptual judgment of humans. DNSMOS is chosen instead of PESQ, due to the improved performance of DNSMOS. When evaluating a metric, a often used method is to calculate the Pearson Correlation Coefficient (PCC) between the prediction and human ground truth. In the article about DNSMOS, it is shown that DNSMOS has a PCC of 0.93, compared to PESQs lower score of 0.78 [Reddy et al., 2021].
STOI is unlike DNSMOS not a metric made for the evaluation of generative models, but it is an important metric anyway. The reason is, that a study has shown that these metrics stand out as a particularly good measure for the perceived quality of Cochlear Implant (CI) users [Falk et al., 2015]. This research does not primarily

target CI users, but if a model demonstrates exceptional performance for this use case, it should be highlighted.
What the study further showed was, that for HA with speech enhancement HASQI stood out as a good measure, and this is therefore also included in this study. For HASQI to work, an amplification must be determined. From the report prepared by "Videnscenter for Hørehandicap", it is seen that among those seeking an audiological clinic to have their hearing loss examined, the majority of those tested fall into the category called "moderate" hearing loss [Sundhedsstyrelsen, 2022]. Moderate hearing loss is determined as a hearing loss from 41 dB - 60 dB by WHO [Sundhedsstyrelsen, 2022; , WHO]. From this, an amplification can be determined. Multiple authors have proposed such standard amplification curves, but it is decided to use the one proposed by Bisgaard [Bisgaard et al., 2010]. The hearing threshold is shown in figure 5.1.

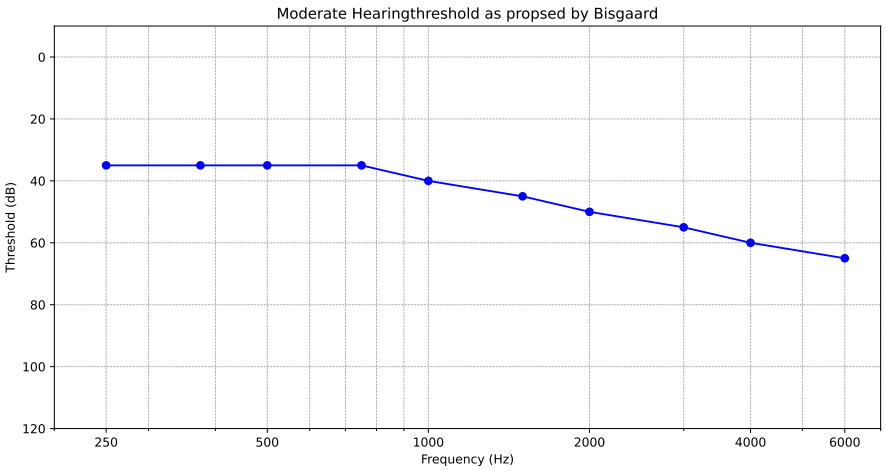


Figure 5.1: Figure showing the standard audiogram for moderate hearing loss proposed by Bisgaard [Bisgaard et al., 2010]

Lastly, SI-SDR measures the ratio of, the power of the target speech, to the power of the generated signal. It is used, as it considers both the scale and permutation ambiguities inherent in separating speech from noise. In the realm of SE it is acknowledged as a measure of how good the improved signal sounds, which is why this metric is particular interesting [Roux et al., 2018]. Unlike SnR or Signal-to-Distortion ratio (SDR), SI-SDR is an evaluation designed more specifically for SE or source separation. This is why it was chosen instead these [Roux et al., 2018].

5.2 Listening test

With the algorithmic metrics in place, a listening test has also been conducted, to verify the results from the evaluation metrics.
In all, 10 participants have been participating in the experimental listening test. 8 male, and 2 female, in the age from 20 - 29 years old. All test subjects are relatively young and have been asked if they believe they suffer from any hearing loss or are using HA. All test subjects answered no to this, this is why there is not performed pure-tone audiometry before starting.
The test method has been chosen to be a variation of the Mean Opinion Score (MOS). It can easily be argued, that this might not be the right evaluation metric, as recent studies show that speech synthesis is of such high quality that the scale does not range wide enough anymore [Le Maguer et al., 2024]. The study looked into speech synthesis, but the methods for the SE in this study are equal to those for speech synthesis in the study showing that MOS is not optimal, this is why it is assumed that the article is also true for the SE.
A better option would be to use the MUlti Stimulus test with Hidden Reference and Anchor (MUSHRA). MUSHRA has multiple advantages to MOS. It needs fewer participants to be statistically significant, and it also asks the test subjective to rate the speech in a more detailed manner, as the scale is from 0-100.
The reason why MUSHRA is not used is that, in the recommendations it clearly states that the test subjects should be experienced listeners able to listen to the sounds in a critical way [International Telecommunication Union (ITU), 2015]. It is not possible to gather such experienced listeners for this experiment. Furthermore, MUSHRA includes a reference that the test subjective always can listen to. This is believed to be undesirable, for this exact research, as the goal not necessarily is to be closest to the original signal, but to offer the best experience for the user.
This is why, a variation of MOS is used. The test system is inspired by the International Telecommunication Union's telecommunication standardization sector recommendations as described in [International Telecommunication Union (ITU), 1996, 2003]. One deliberate choice that does not cohere with the recommendations is, that instead of using an Absolute Category Scale Rating (ACR) scale from 1-5 an open-ended Visual Analog Scale (VAS) ranging from 1-100 will be used. This choice stems from the aforementioned article that shows that MOS with a fixed scale of 1-5 does not range wide enough to evaluate modern SE. The reason why it is open-ended is to avoid lumping, around the ends. To further avoid bias, there is no numbering on the scale, instead, there are verbal labels, based on the scale used in MUSHRA as proposed by ITU.
In appendix E the test setup and procedure, for the listening experiment can be found in detail.

6 | Results

In this chapter, the results will be presented. The different metrics are discussed in section 5.1, and the listening experiment in section 5.2. All results are obtained with a dataset published by the University of Edinburgh [Valentini Botinhao et al., 2016]. The SnR of the noisy speech signal is 2.5 dB, 7.5 dB, 12.5 dB or 17.5 dB.
The results from each metric are shown for SGMSE, Facebook Demucs, and then results is also presented, where the architecture from Demucs is used natively in a diffusion context (Non diffusion aware E2E model) and the final proposed model. A thing to notice is, that the final E2E model is stopped prior to the point where it stopped learning. In appendix C a visual impression of the signals are shown in both the time domain and in the frequency domain.
Lastly, in the metric matrix, and the listening test, as last experiment is included, where a post filter is added to the enhanced signal, in order to remove any last white-noise. For more details see appendix D. This result is not presented in the detailed plots, as it is no longer an end-to-end model, as the post filter works in the frequency domain.

6.1 Overall results

In this section, the mean results from the different metrics are shown. The results are to be read as: $\mu(\sigma)$, where μ is the mean result, and σ is the standard deviation. This means if a cell writes: 1(0.2) the mean is 1, and the standard deviation is 0.2. The overall result is a mean across of all the enhanced data, also meaning it does not split it into SnR of the noisy data. For all metrics and models the dataset is the same.

Table 6.1: Comparison of different models on various metrics

	Facebook Demucs	**SGMSE**	**E2E model**	**Non diffusion aware E2E model**	**E2E w. post-filter**
STOI	0.9319 (0.0930)	**0.9517 (0.0485)**	0.9148 (0.0606)	0.8923 (0.0634)	0.8505 (0.058)
SI-SDR [dB]	**18.08 (5.9302)**	17.09 (3.826)	13.47 (3.8580)	9.053 (4.564)	9.955 (2.295)
DSNMOS (OVRL)	3.083 (0.2430)	**3.190 (0.1980)**	2.8847 (0.2901)	2.770 (0.2970)	2.278 (0.3319)
DSNMOS (SIG)	3.359 (0.2205)	**3.483 (0.1643)**	3.4444 (0.1936)	3.389 (0.2585)	2.644 (0.3354)
DSNMOS (BAK)	4.0122 (0.1374)	**4.025 (0.1210)**	3.4660 (0.4776)	3.323 (0.3849)	3.461 (0.3796)
HASQI	0.9635	**0.9640**	0.94179	0.9028	0.6780

6.2 Detailed results

In this section, the results in the prior section is presented in more details, with focus on easy comparison. For each model, the three different evaluation metrics is shown, namely STOI, SI-SDR and DNSMOS. Each plot contains the results at different SnR - 2.5 dB, 7.5 dB, 12.5 dB and 17.5 dB, for all models at once.

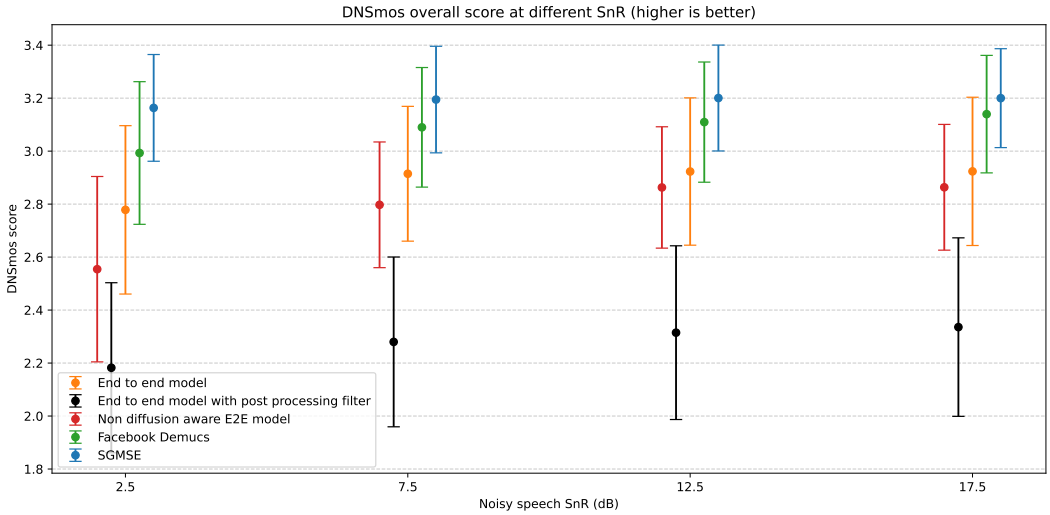


Figure 6.1: Figure showing the DNSmos overall score for all four models. The dots are the respective mean results, and the error bars shows the standard deviation.

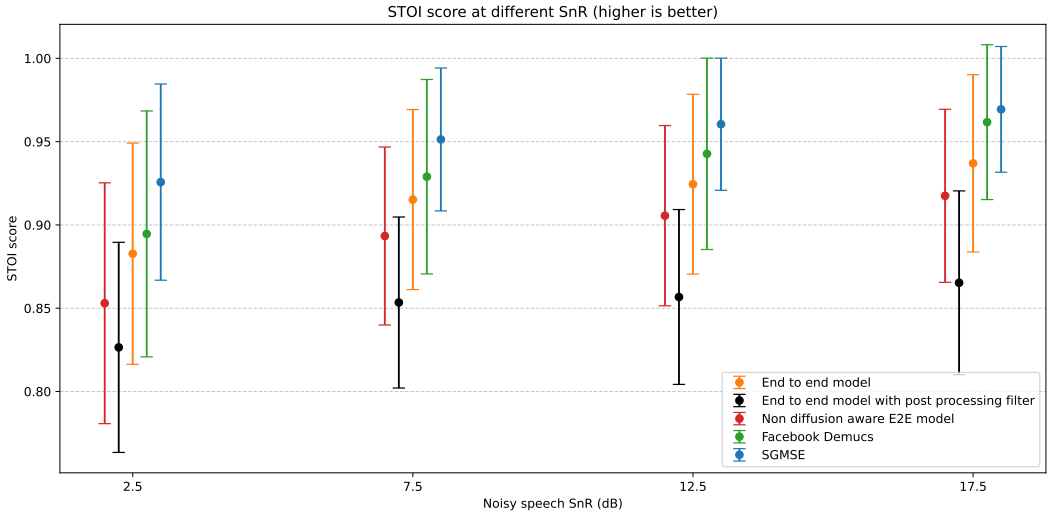


Figure 6.2: Figure showing the STOI results for all four models. The dots are the respective mean results, and the error bars shows the standard deviation.

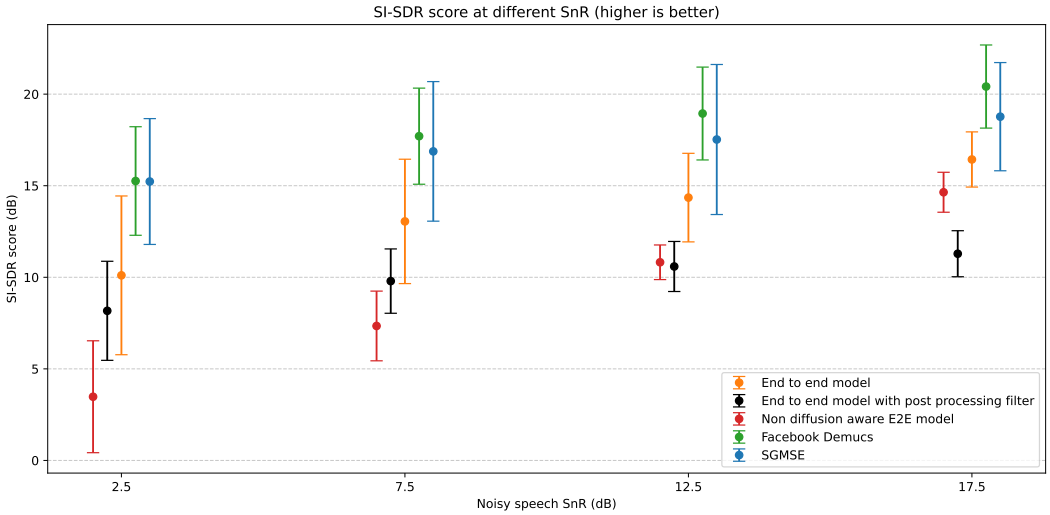


Figure 6.3: Figure showing the SI-SDR results for all four models. The dots are the respective mean results, and the error bars shows the standard deviation.

6.3 Results from subjective test

In this section, the result from the listening experiment is shown. In the results is included a model where a post filter is added, in an attempt to remove whitenoise.

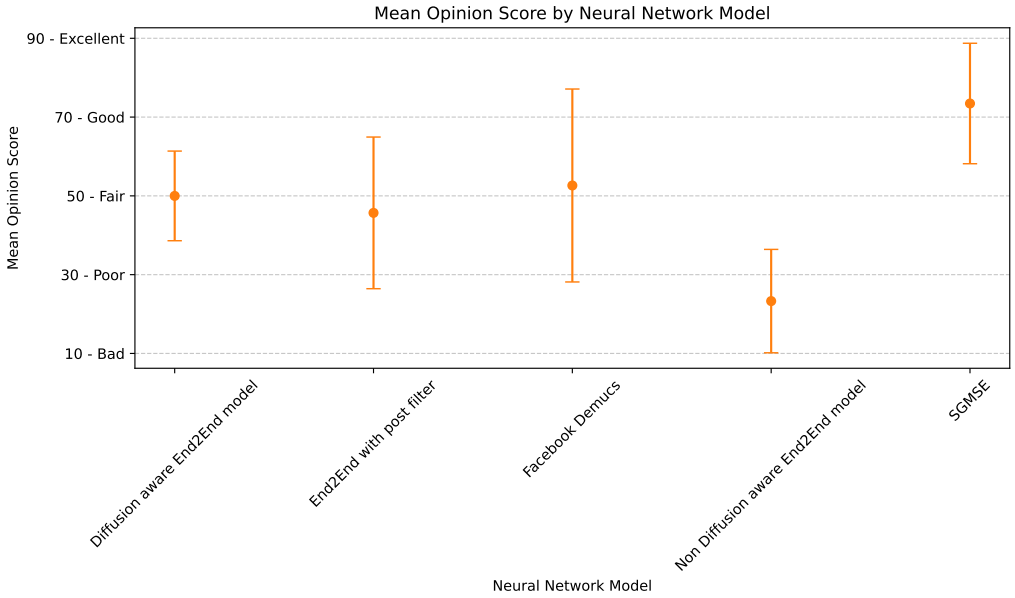


Figure 6.4: The mean results in the listening experiment. The dots are the respective mean results, and the error bars shows the standard deviation.

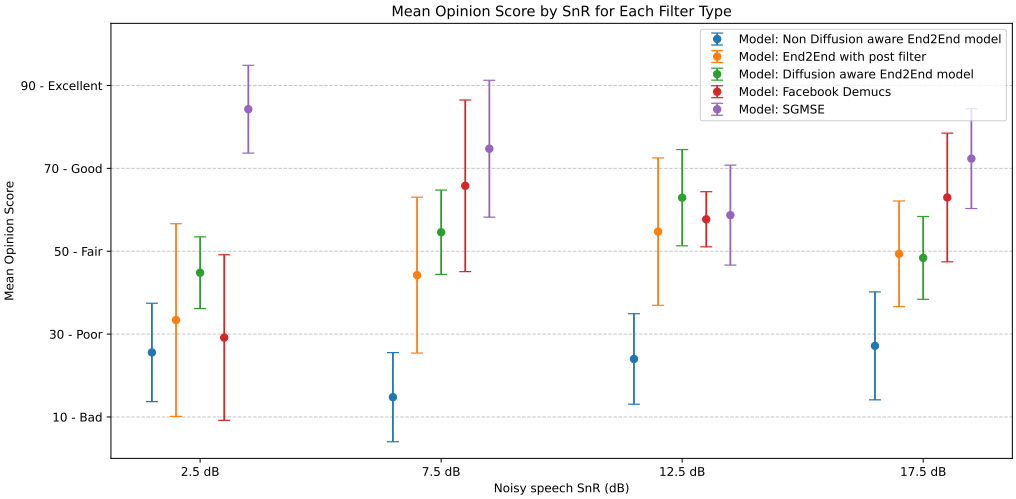


Figure 6.5: A detailed version of the results from the listening experiment, showing the result split into the SnR of the noisy signal, that has been enhanced. The dots are the respective mean results, and the error bars shows the standard deviation.

7 | Discussion

This study tried to propose an end-to-end diffusion speech enhancement model, based on an architecture not initially intended to work in a diffusion context. The study aimed to optimize the architecture with diffusion-aware training, meaning that time-step from the Markov process has been embedded into the architecture. The aim was to train a model, solely on data in the time domain, not including any data in the frequency domain.
Experimental work was carried out to see what initial architecture would perform the best. Multiple models were tested, all based on a U-Net structure, with Conv1D layers in the encoders and decoders. Reasonable arguments why this would be a good solutions was given, such as the fact that SGMSE utilize NCSN++, an architecture based on a U-Net structure, with Conv2D layers.
It was shown, that the architecture used in Demucs designed by Facebook outperformed Wave-U-Net when implemented equally in the diffusion context. To ensure, that the architecture was the only variable in performance, the hyperparameters was kept constant during the experiments, including the activation functions. It has been discussed why the Demucs architecture was best, pointing towards the LSTM layer, both due to theory, and experimental work.
From the literature, it was found that the time step "t" is an important feature for the network. In the initial experiments, the time step was therefore given in the simplest possible form, to the network, as depicted in figure 4.8.
To improve further on the network, complex embedding of time step "t" has been researched. The reason why, it has been researched, other than the statement from the literature, is that an idea arose that if the model has better knowledge of where in the Markov process the network is, it might be easier for it to learn how to remove the artificial white noise which was a problem in the simpler implementation of Demucs architecture in the diffusion context. Therefore, multiple implementations is proposed, embedding the time in different ways, with varying complexity. The different implementations is shown in section 4.3.2.
From this, multiple experiments are performed, showing that time embedding indeed matters, and the complexity of the time embedding also plays a role.
Firstly, it is shown that even the simplest embedding of time improves the loss significantly. This is shown in section 4.3.2, figure 4.12.
From this, two more complex models are made - but with focus on the one depicted in figure 4.11 one from hereon. The model utilized both a positional encoder, multiple fully connected layers, and embedded time into each decoder, and encoder block. Without changing the activation layers, it showed lower initial loss, and faster learning, shown in figure 4.12. It was stopped early, to ensure some hyperparameter tuning, within the time frame of the project.
Within the hyperparameter tuning, the depth was increased, as well as the number of hidden layers. Another change, in the final embedding of the timestep "t" is, that instead of using ReLU in the fully connected layers, the Swish function (SiLU) is

used. The reason for this, is that the author of the Swish function shows, that it performs as well as, or better than ReLU across their suit of tests[Ramachandran et al., 2017]. It is arguable that more in-depth testing should have been conducted to assess both the positive and negative effects of changing to Swish.
From the objective metrics, it is shown that SGMSE outperforms the other models, except from in SI-SDR, where Facebook Demucs had the best score. No metric stood out in either STOI or HASQI, except for SGSME, indicating that no model performed exceptionally well by chance for either HA users or CI users.
It is from the metrics also seen, that the final proposed end-to-end model, outperforms the non-diffusion aware end-to-end model, in all of the measured metrics. This shows that the time step is indeed an important feature. Additionally, when the entire enhanced speech signal is evaluated using DNSmos, the proposed end-to-end model performs worse than both SGMSE and Facebook Demucs. If the DNSmos score is split into speech signal and background noise, instead of rating the overall signal, it can be seen that the end-to-end model has a better signal quality than Facebook Demucs, being on par with SGMSE. Where it falls through is in the background noise, where one of the significant challenges identified in the end-to-end diffusion model for speech enhancement, is the high level of white noise during silent periods. It is evident in both figure C.4a and C.4c as shown in appendix C.
The listening test showed the same, namely that SGMSE beats both Facebook Demucs, and the proposed end-to-end model on average. Looking into details, it get more complicated, as the end-to-end model outperforms Facebook Demucs at 2.5 dB, and at 12.5 dB where it also outperforms SGMSE. At 7.5 dB, and 17 dB Facebook Demucs beats the proposed model, and SGMSE beats Facebook Demucs. This is even though, the objective metrics clearly shows that it should not be the case.
According to the objective metrics, aswell as the listening test, the addition of a post-filter did not improve upon the model. This can be due to various reason, but one valid guess is that the filter introduces artifacts, which the objective metrics punished, and the subjects in the listening test did not like. Instead of implementing a spectral subtraction filter, other filters might work better, eg. the General filter [Benesty et al., 2014] which could enhance a Wiener filter's performance by recovering high frequencies.
One of the considerations when evaluating generative models has been, that the evaluation metric should cohere with the nature of the model - meaning, that the model is not bound to generate the exact ground truth signal. From the objective function, it can be argued that this might not be true, as it specifically compares the ground truth, with the generated signal. But, in order to ensure comprehensive evaluation, it is still believed, that the final enhanced signal can differ, and therefore the metrics STOI, SI-SDR and DNSmos was chosen as they cover both traditional evaluation, where the ground truth is compared to the enhanced signal, aswell as a new NN approach to evaluation.
This choice align well with the results, where it is seen, that Facebook Demucs outperforms SGMSE in SI-SDR even though it does not outperform SGMSE in other

metrics, or the listening test. The higher score from Facebook Demucs compared to SGMSE, could be cause by the generative nature of SGMSE, that might introduces variability and potential artifacts, which can reduce the SI-SDR as the generated speech may not perfectly align with the ground truth.
It is also seen how, the inclusion of DNSmos give valuable insights in the models, as it can evaluate both the overall signal, the speech it self, and the background noise.
Besides the introduction of a post-filter, that can be improved further, it is known from the literature that combining the noisy speech, and the enhanced speech can result in better speech intelligibility due to recovering high frequencies [Defossez et al., 2020; Phan et al., 2020; Lu et al., 2022]. This method could be used with the Wiener filter, as this removes high-frequency signals when removing the white-noise.
Instead of believing the solution is a post-filter, another option is to let the model train longer. The reverse process in the diffusion model is a matter of estimating the white noise present, and therefore, it is plausible, that the model would improve and maybe even remove the last of the white noise if trained for a longer period than what it is trained for now. The evaluated model was stopped prior to the finishing training, in order to be able to show initial results.
If this however is not a product of the stopping training early, it is arguably an inherent problem in making diffusion work solely in the time domain with a simple U-Net architecture. It would be easy to understand why, the last white noise is hard to estimate, from looking at the time domain plots in appendix C.0.4. This is also evident in that other models that enhance end-to-end is in the training process conditioned on the clean speech in the STFT [Lu et al., 2022]. If it is truly the case, that it is an inherent problem, and one would continue the work creating, an end-to-end model trained solely in the timedomain, an optimized loss function could be a possible solution.
One simple proposal to optimize the loss function is to adapt it into the frequency domain, as shown in equation 7.1. As discussed previously, white noise can be difficult to detect when analyzing speech signals in the time domain because it blends into the background and lacks distinct patterns, making it seem insignificant. However, it might correspond to significant frequency differences, as observed in the STFT plot. Therefore, the idea is that white noise can be more easily identified and reduced in the frequency domain. By adapting the loss function into the frequency domain, it might implicitly reduce such noise by aligning the overall spectral distribution of the output and the target.

$$E_{t,\mathbf{x_0},\mathbf{x_t}|\mathbf{x_0}}\left[\|\mathcal{F}\left(\mathbf{s}_\theta(\mathbf{x_t},t,\mathbf{y})\right)-\mathcal{F}\left(\nabla_{\mathbf{x_t}}\log p_{0t}(\mathbf{x_t}|\mathbf{x_0},\mathbf{y})\right)\|_2^2\right] \tag{7.1}$$

Limitations

This study is limited to look into architectures that were not initially meant for diffusion, and how to improve upon these. The study does not look at the schedulers used in the training process. It is known, that they have an impact of performance,

and therefore a study on these might help further improve the performance seen, in the proposed end-to-end model. Different schedulers change how much training time the model uses with a given amount of white noise, and it is imaginable that this will affect the end-to-end model. Another point of improvement, of the current proposed solutions, is to adjust the hyperparameters for each architecture. In the experiments, the hyperparameters are kept constant, until the last trained model, to see differences only from the architecture. With more time, the hyperparameters could have been adjusted for each model, ensuring optimal performance. For the last model, a more thorough hyperparameter tuning might also show a further decrease in the loss.
One could also imagine, that to obtain a different end-to-end model that might perform better, than what is presented, the solution is to not use an U-Net architecture with Conv1D. Other types of architectures must be investigated, to obtain a model, that can compete with current SOA, like SGMSE or Demucs Speech Enhancement. One option is to be inspired by DiffWave or WaveNet, instead of SGMSE. DiffWave and WaveNet are not made for speech enhancement, but rather for speech synthesis in the time domain. DiffWave uses a bidirectional dilated convolution architecture, and WaveNet uses a architecture based on Casual convolutions, which both differ significantly from a U-Net structure. CDiffuSE is a diffusion model for SE that is not based on the U-Net structure either, but is conditioned on spectrograms.
A shortcoming of the research is, that it assumes the user of the model will be HA users, but most evaluation is focused around normal-hearing people. Two evaluation metrics stand out, as they are shown to work well for people with hearing assistive device, namely STOI and HASQI. STOI stood out for cochlear implant users, and HASQI is developed specifically for HA users. It is true for for HASQI, that even though it is developed for HA users, this research is limited to a single level of hearing loss, within the moderate hearing loss category. It would make sense, to broaden this to multiple categories, like mild, and severe.

8 | Conclusion

How does incorporating diffusion-aware training techniques into non-diffusion networks in the waveform domain affect speech quality compared to diffusion models that use Short-Time Fourier Transform (STFT) representations?

In this study, an end-to-end model is proposed based on a known architecture, Facebook Demucs, which is not initially designed to work within a diffusion context. It has been shown, how making the architecture diffusion aware with embedding of the time step t, from the markov process, improve upon the final result, both evaluated by loss, as well as evaluated by objective metrics and a subjective listening experiment.

It was not possible to create a model that perform as well as SGMSE, the current state of the art. SGMSE outperforms the proposed model overall, except in a single case in the subjective listening experiment, where the proposed model has a better mean result when the input before enhancement has a SnR of 12.5 dB.

It was not possible to create a model that performs as well as SGMSE, the current state-of-the-art. SGMSE outperforms the proposed model overall, except in one instance during the subjective listening experiment, where the proposed model achieves a better mean result if the SnR of the noisy speech before enhancement is 12.5 dB. When evaluated solely on signal quality using DNSmos, the proposed model's signal quality is comparable to that of SGMSE.

Based on the results and a visual inspection of the enhanced speech in the STFT domain, it can be identified that the performance issue with the end-to-end model arises from the inability to remove all the artificial white noise from the diffusion process.

It has been proposed that it can be due to a lack of training duration, or a fundamental flaw in training in the time domain. A solution on how to improve the training process, like optimizing the objective function, or looking at different architectures, has been stated. It was furthermore attempted to improve upon the model with a post-filter, which did not improve the objective measures, or the results in the listening experiment.

Bibliography

, **2022**. *Common Problems*, 2022. URL `https://developers.google.com/machine-learning/gan/problems`. Last acces: 02-16-2024.

Asante et al., **08 2023**. Bismark Asante, Clifford Broni-Bediako og Hiroki Imamura. *Exploring Multi-Stage GAN with Self-Attention for Speech Enhancement*. Applied Sciences, 13, 9217, 2023. doi: 10.3390/app13169217.

Benesty et al., **2005**. Jacob. Benesty, Shoji. Makino og Jingdong. Chen. *Speech Enhancement*. Signals and Communication Technology. Springer Berlin Heidelberg, Berlin, Heidelberg, 1st ed. 2005. edition, 2005. ISBN 1-280-33791-5.

Benesty et al., **2014**. Jacob Benesty, Jesper Rindom Jensen, Mads Graesboll Christensen og Jingdong Chen. *Speech Enhancement: A Signal Subspace Perspective*. Elsevier Science, San Diego, 1 edition, 2014. ISBN 9780128002537.

M. Berouti, Richard Schwartz og J. Makhoul, *05 1979*. M. Berouti, Richard Schwartz og J. Makhoul. Enhancement of speech corrupted by acoustic noise. volume 4, pages 208 – 211, 05 1979. doi: 10.1109/ICASSP.1979.1170788.

Bisgaard et al., **06 2010**. Nikolai Bisgaard, Marcel Vlaming og Martin Dahlquist. *Standard Audiograms for the IEC 60118-15 Measurement Procedure*. Trends in amplification, 14, 113–20, 2010. doi: 10.1177/1084713810379609.

Defossez et al., **2020**. Alexandre Defossez, Gabriel Synnaeve og Yossi Adi. *Real Time Speech Enhancement in the Waveform Domain*, 2020.

Desjardins og Doherty, **03 2014**. Jamie Desjardins og Karen Doherty. *The Effect of Hearing Aid Noise Reduction on Listening Effort in Hearing-Impaired Adults*. Ear and hearing, 35, 2014. doi: 10.1097/AUD.0000000000000028.

Dhariwal og Nichol, **05 2021a**. Prafulla Dhariwal og Alex Nichol. *Diffusion Models Beat GANs on Image Synthesis*, 2021a.

Dhariwal og Nichol, **2021b**. Prafulla Dhariwal og Alex Nichol. *Diffusion Models Beat GANs on Image Synthesis*, 2021b.

Donahue et al., **11 2017**. Chris Donahue, Bo Li og Rohit Prabhavalkar. *Exploring Speech Enhancement with Generative Adversarial Networks for Robust Speech Recognition*. 2017.

Ephraim og Van Trees, **1995**. Y. Ephraim og H.L. Van Trees. *A signal subspace approach for speech enhancement*. IEEE Transactions on Speech and Audio Processing, 3(4), 251–266, 1995. doi: 10.1109/89.397090.

Ephraim, **11 1992**. Yariv Ephraim. *Statistical-Model-Based Speech Enhancement Systems*. Proceedings of the IEEE, 80, 1526 – 1555, 1992. doi: 10.1109/5.168664.

Falk et al., **03 2015**. Tiago Falk, Vijay Parsa, João Santos, Kathryn Arehart, Oldooz Hazrati, Rainer Huber, James Kates og Susan Scollie. *Objective Quality and Intelligibility Prediction for Users of Assistive Listening Devices*. IEEE Signal Processing Magazine, 32, 114–124, 2015. doi: 10.1109/MSP.2014.2358871.

Fang et al., **02 2021**. Huajian Fang, Guillaume Carbajal, Stefan Wermter og Timo Gerkmann. *Variational Autoencoder for Speech Enhancement with a Noise-Aware Encoder*, 2021.

for Disease Control og (CDC). Centers for Disease Control og Prevention (CDC). *What Noises Cause Hearing Loss?* URL `https://www.cdc.gov/nceh/hearing_loss/what_noises_cause_hearing_loss.html`. Accessed May 13, 2024.

for Speech Technology Voice Cloning Toolkit, **2016**. Centre for Speech Technology Voice Cloning Toolkit. *VCTK*, 2016. URL `https://datashare.ed.ac.uk/handle/10283/2950`.

Haykin, **2014**. Simon Haykin. *Adaptive filter theory*. Prentice Hall, 5th edition, 2014.

Ho et al., **06 2020a**. Jonathan Ho, Ajay Jain og Pieter Abbeel. *Denoising Diffusion Probabilistic Models*, 2020a.

Ho et al., **2020b**. Jonathan Ho, Ajay Jain og Pieter Abbeel. *Denoising Diffusion Probabilistic Models*, 2020b.

International Telecommunication Union (ITU), **2015**. International Telecommunication Union (ITU). *Recommendation ITU-R BS.1534-3: Method for the subjective assessment of intermediate quality level of audio systems*, ITU-T, 2015.

International Telecommunication Union (ITU), **1996**. International Telecommunication Union (ITU). *ITU-T P.800: Methods for subjective determination of transmission quality*, ITU-T, 1996.

International Telecommunication Union (ITU), **2003**. International Telecommunication Union (ITU). *ITU-T P.835: Subjective performance assessment of speech communication quality*, ITU-T, 2003.

Keras, **2022**. Keras. *Convolution layers*, 2022. URL `https://keras.io/api/layers/convolution_layers/`. Last acces: 02-24-2024.

Kokkinakis et al., 05 2011. Kostas Kokkinakis, Oldooz Hazrati og Philipos Loizou. *A channel-selection criterion for suppressing reverberation in cochlear implants*. The Journal of the Acoustical Society of America, 129, 3221–32, 2011. doi: 10.1121/1.3559683.

Kong et al., 2021. Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao og Bryan Catanzaro. *DiffWave: A Versatile Diffusion Model for Audio Synthesis*, 2021.

Le Maguer et al., 2024. Sébastien Le Maguer, Simon King og Naomi Harte. *The limits of the Mean Opinion Score for speech synthesis evaluation*. Computer Speech & Language, 84, 101577, 2024. ISSN 0885-2308. doi: https://doi.org/10.1016/j.csl.2023.101577. URL `https://www.sciencedirect.com/science/article/pii/S0885230823000967`.

Lemercier et al., 2023. Jean-Marie Lemercier, Julius Richter, Simon Welker og Timo Gerkmann. *StoRM: A Diffusion-Based Stochastic Regeneration Model for Speech Enhancement and Dereverberation*. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 31, 27242737, 2023. ISSN 2329-9304. doi: 10.1109/taslp.2023.3294692. URL `http://dx.doi.org/10.1109/TASLP.2023.3294692`.

Lu et al., 2021. Yen-Ju Lu, Yu Tsao og Shinji Watanabe. *A Study on Speech Enhancement Based on Diffusion Probabilistic Model*, 2021.

Lu et al., 2022. Yen-Ju Lu, Zhong-Qiu Wang, Shinji Watanabe, Alexander Richard, Cheng Yu og Yu Tsao. *Conditional Diffusion Probabilistic Model for Speech Enhancement*, 2022.

Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg og Oriol Nieto, 2015. Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg og Oriol Nieto. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, volume 8, 2015.

Mittal et al., 2020. Sarthak Mittal, Alex Lamb, Anirudh Goyal, Vikram Voleti, Murray Shanahan, Guillaume Lajoie, Michael Mozer og Yoshua Bengio. *Learning to Combine Top-Down and Bottom-Up Signals in Recurrent Neural Networks with Attention over Modules*, 2020.

Park og Lee, 2016. Se Rim Park og Jinwon Lee. *A Fully Convolutional Neural Network for Speech Enhancement*, 2016.

Pascual et al., 2017. Santiago Pascual, Antonio Bonafonte og Joan Serrà. *SEGAN: Speech Enhancement Generative Adversarial Network*, 2017.

Phan et al., 2020. Huy Phan, Ian Vince McLoughlin, Lam Dang Pham, Oliver Y. Chén, Philipp Koch, Maarten De Vos og Alfred Mertins. *Improving GANs for*

Speech Enhancement. CoRR, abs/2001.05532, 2020. URL https://arxiv.org/abs/2001.05532.

Picou et al., 02 2013. Erin Picou, Todd Ricketts og Benjamin Hornsby. *How Hearing Aids, Background Noise, and Visual Cues Influence Objective Listening Effort*. Ear and hearing, 34, 2013. doi: 10.1097/AUD.0b013e31827f0431.

Ramachandran et al., 2017. Prajit Ramachandran, Barret Zoph og Quoc V. Le. *Searching for Activation Functions*. CoRR, abs/1710.05941, 2017. URL http://arxiv.org/abs/1710.05941.

Reddy et al., 2021. Chandan K A Reddy, Vishak Gopal og Ross Cutler. *DNSMOS: A Non-Intrusive Perceptual Objective Speech Quality metric to evaluate Noise Suppressors*, 2021.

Abir Riahi og Éric Plourde, 2023. Abir Riahi og Éric Plourde. Single Channel Speech Enhancement Using U-Net Spiking Neural Networks. In *2023 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 111–116, 2023. doi: 10.1109/CCECE58730.2023.10288830.

Richter et al., 2023. Julius Richter, Simon Welker, Jean-Marie Lemercier, Bunlong Lay og Timo Gerkmann. *Speech Enhancement and Dereverberation with Diffusion-based Generative Models*. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 31, 2351–2364, 2023. doi: 10.1109/TASLP.2023.3285241.

Ronneberger et al., 2015. Olaf Ronneberger, Philipp Fischer og Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*, 2015.

Roux et al., 2018. Jonathan Le Roux, Scott Wisdom, Hakan Erdogan og John R. Hershey. *SDR - half-baked or well done?*, 2018.

Sarampalis et al., 05 2009. Anastasios Sarampalis, Sridhar Kalluri, Brent Edwards og Ervin Hafter. *Objective Measures of Listening Effort: Effects of Background Noise and Noise Reduction*. Journal of speech, language, and hearing research : JSLHR, 52, 1230–40, 2009. doi: 10.1044/1092-4388(2009/08-0111).

Stoller et al., 2018. Daniel Stoller, Sebastian Ewert og Simon Dixon. *Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation*, 2018.

Sundhedsstyrelsen, 2022. Sundhedsstyrelsen. *Personer med høretab i Danmark*, 2022. URL https://www.sbst.dk/media/12157/Personer%20med%20h%C3%B8retab%20i%20Danmark.pdf. Rapport fra Sundhedsstyrelsen.

Shin'ichi Tamura og Alex Waibel, 05 1988. Shin'ichi Tamura og Alex Waibel. Noise reduction using connectionist models. pages 553 – 556 vol.1, 05 1988. doi: 10.1109/ICASSP.1988.196643.

Shuai Tao, Yang Xiang, Himavanth Reddy, Jesper Rindom Jensen og Mads Græsbøll Christensen, 2023. Shuai Tao, Yang Xiang, Himavanth Reddy, Jesper Rindom Jensen og Mads Græsbøll Christensen. Single Channel Speech Presence Probability Estimation based on Hybrid Global-Local Information. In *2023 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 1–5, 2023. doi: 10.1109/WASPAA58266.2023.10248067.

Cassia Valentini Botinhao, Xin Wang, Shinji Takaki og Junichi Yamagishi, September 2016. Cassia Valentini Botinhao, Xin Wang, Shinji Takaki og Junichi Yamagishi. Speech Enhancement for a Noise-Robust Text-to-Speech Synthesis System using Deep Recurrent Neural Networks. In *Proceedings of Interspeech 2016*, Interspeech, pages 352–356. International Speech Communication Association, September 2016. doi: 10.21437/Interspeech.2016-159. URL `http://www.interspeech2016.org/`. Interspeech 2016 ; Conference date: 08-09-2016 Through 12-09-2016.

van den Oord et al., 2016. Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior og Koray Kavukcuoglu. *WaveNet: A Generative Model for Raw Audio*, 2016.

Virtanen et al., 2020. Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt og SciPy 1.0 Contributors. *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python.* Nature Methods, 17, 261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Simon Welker, Julius Richter og Timo Gerkmann, 2022a. Simon Welker, Julius Richter og Timo Gerkmann. Speech Enhancement with Score-Based Generative Models in the Complex STFT Domain. In *Proc. Interspeech 2022*, pages 2928–2932, 2022a. doi: 10.21437/Interspeech.2022-10653.

Welker et al., 2022b. Simon Welker, Julius Richter og Timo Gerkmann. *Speech Enhancement and Dereverberation with Diffusion-based Generative Models*, 2022b. URL `https://github.com/sp-uhh/sgmse/tree/main`.

(WHO), 2021. World Health Organization (WHO). *World report on hearing 2021*, 2021. URL `https://www.who.int/health-topics/hearing-loss`. Report by the World Health Organization (WHO).

Yadan, **2019**. Omry Yadan. *Hydra - A framework for elegantly configuring complex applications*. Github, 2019. URL `https://github.com/facebookresearch/hydra`.

A | Training with different architectures

A.1 Experiments with Wave-U-Net architecture

This section shows the results from the experiments, that is based on training with a Wave-U-Net architecture, in a diffusion context.
2 different architectures, both based on Wave-U-Net is trained:

- Native Wave-U-Net architecture
- Dilated Wave-U-Net architecture

This show that an improved wave-u-net can be obtained by using dilated Conv1D layers, which improved upon the speech quality assessment metrics, even though it did not show any improvement in the loss.

A.1.1 Results with native Wave-U-Net architecture

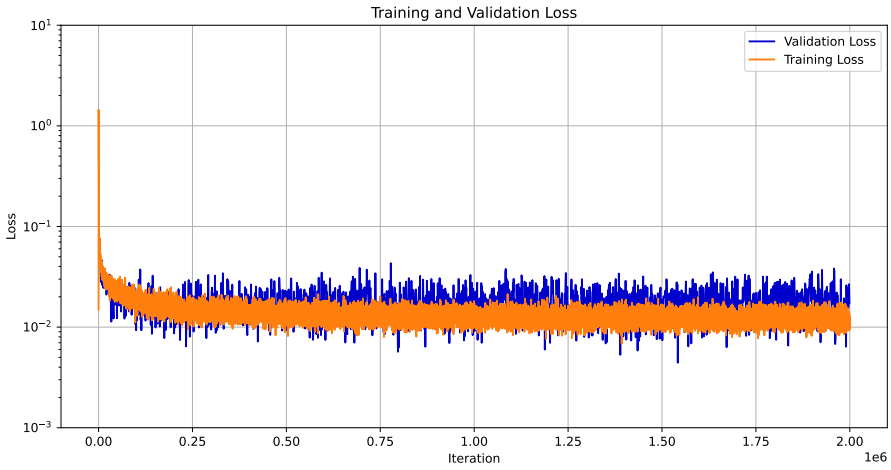


Figure A.1: native Wave-U-Net training and validation loss

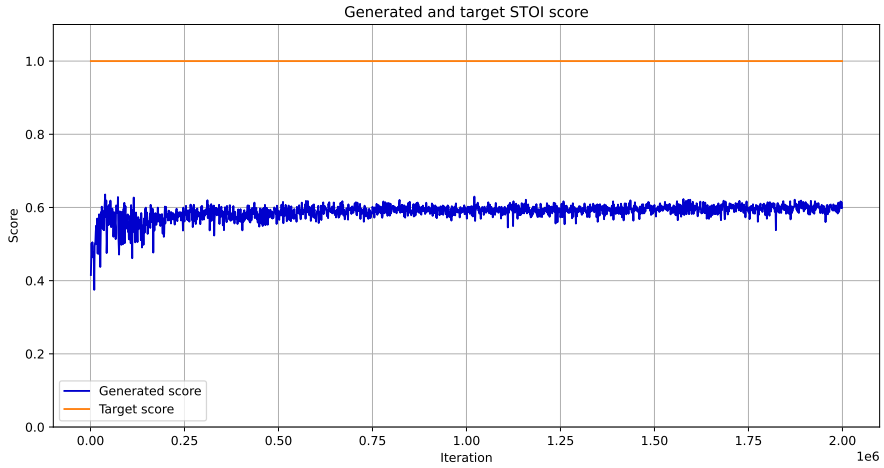


Figure A.2: native Wave-U-Net STOI result

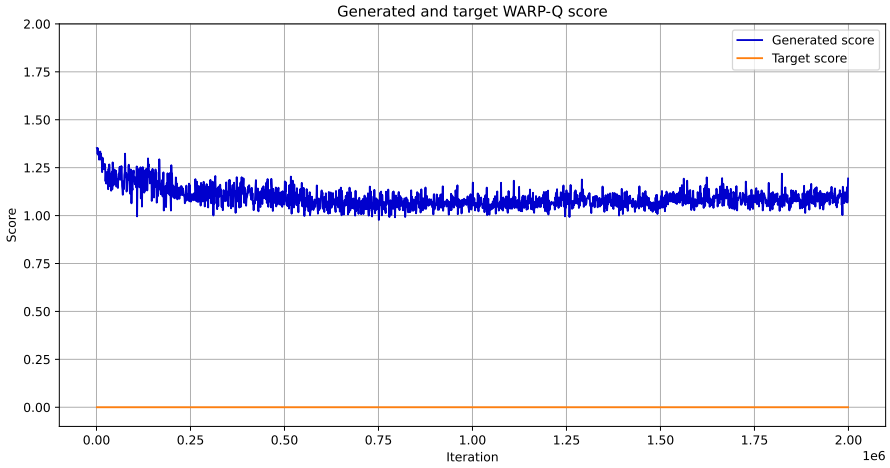


Figure A.3: native Wave-U-Net WARP-Q score

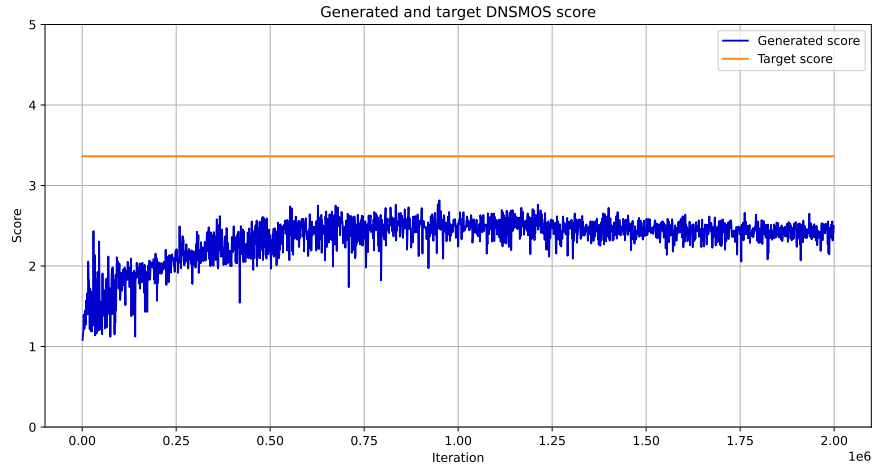


Figure A.4: native Wave-U-Net DNSMOS score

A.1.2 Results with dilated Wave-U-Net architecture

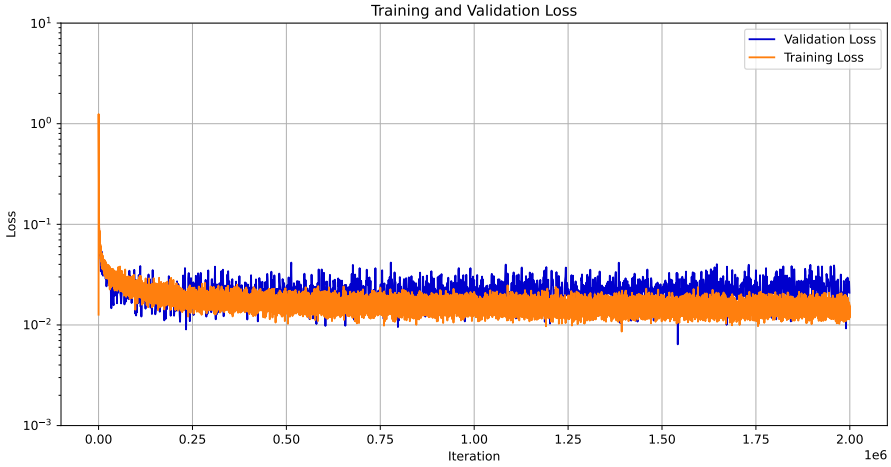


Figure A.5: Dilated Wave-U-Net training and validation loss

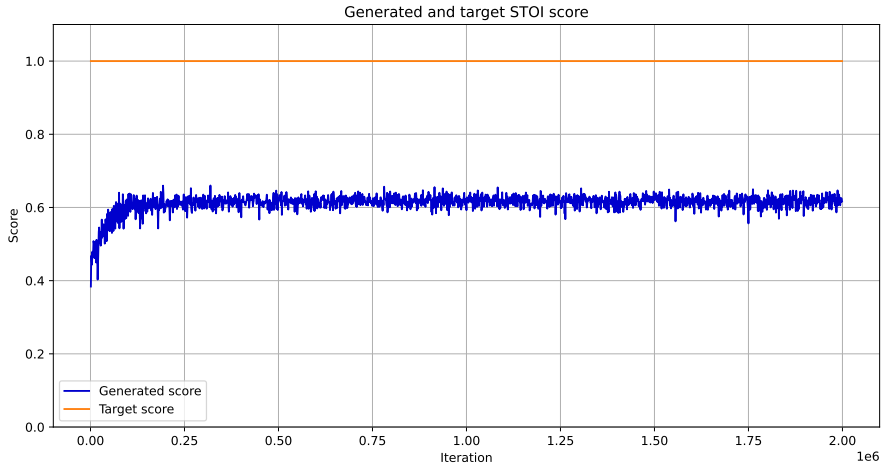


Figure A.6: Dilated Wave-U-Net STOI result

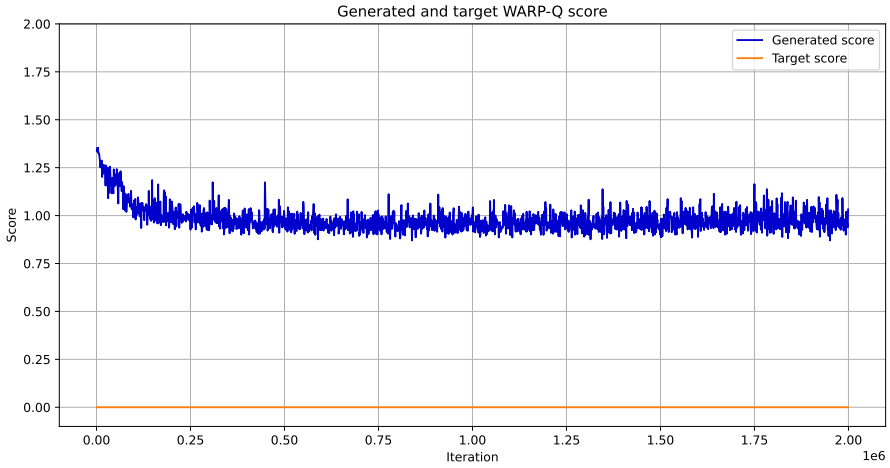


Figure A.7: Dilated Wave-U-Net WARP-Q score

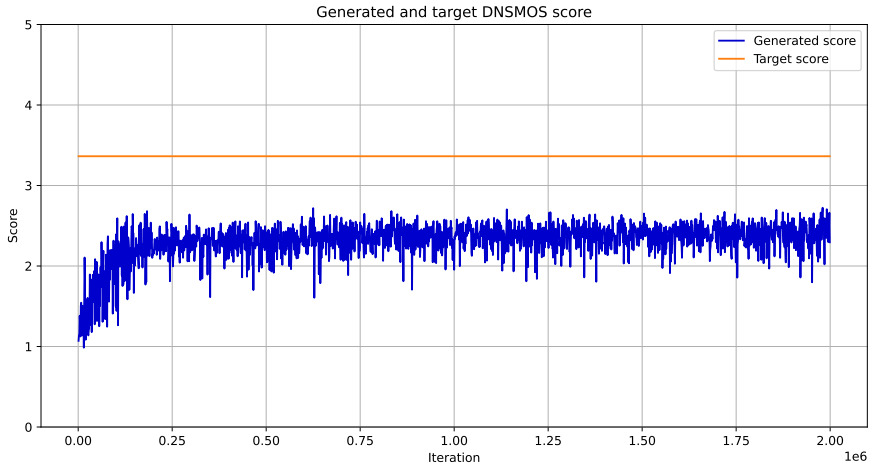


Figure A.8: Dilated Wave-U-Net DNSMOS score

A.2 Experiments with Demucs architecture

This section shows the results from the experiments, that is based on training with a Demucs architecture in a diffusion context.
Four different architectures, all based on Demucs is trained:

- Native Demucs architecture
- Dilated Demucs architecture
- Demucs with Batch Normalization
- Dilated Demucs with Batch Normalization

It was firmly believed, that Demucs could have been improved with simple measures, inspired by the experiments with Wave-U-Net. One key difference between Demucs and Wave-U-Net is the use of batch normalization in Wave-U-Net. Batch normalization is supposed to help stabilize the training process, by normalizing the input of each layer. It reduces the covariate shift in each block, which may also lead to a performance increase. This addition to the architecture is investigated, in section A.2.2, where it shows overfitting on figure A.17
Furthermore, a common way to increase the performance of models with Conv1D layers is, to use dilation. This was also shown to be the case for Wave-U-Net, but on figure A.13 it is shown how Demucs does not improve when dilation is used.
A dilated Demucs model, with batch normalization, is lastly investigated to ensure comprehensive research of the proposed improvement methods. This architecture

was not expected to work, even before the experiment was carried out, as it was already seen that adding Batch Normalization resulted in overfitting. It is though now shown, in figure A.18, that adding both dilated layers, and batchnormalization to the Demucs architecture is also prone to overfitting, why it is not been investigated further.

A.2.1 Results with native Demucs architecture

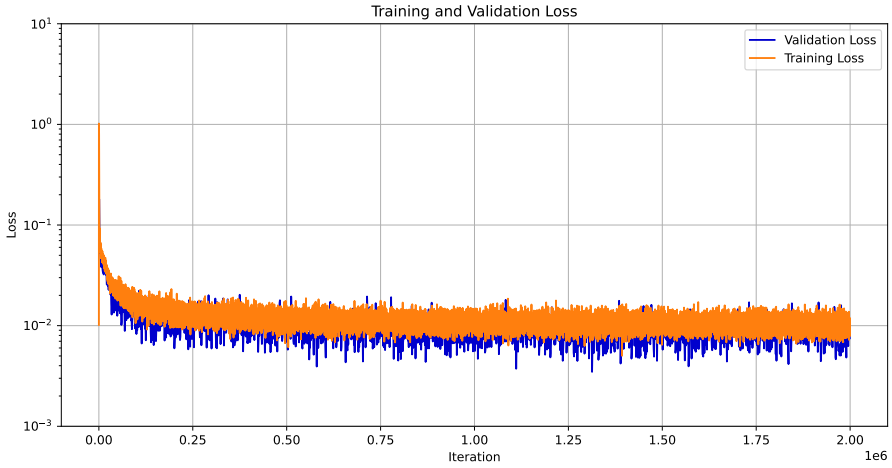


Figure A.9: native Demucs training and validation loss

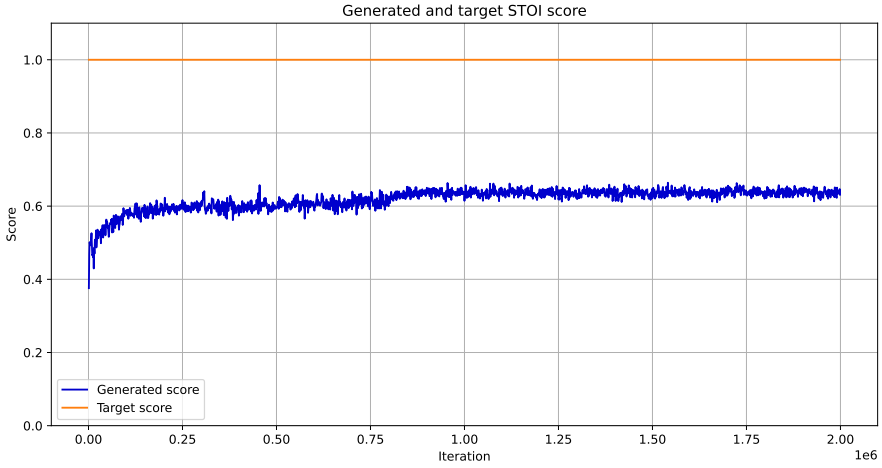


Figure A.10: native Demucs STOI result

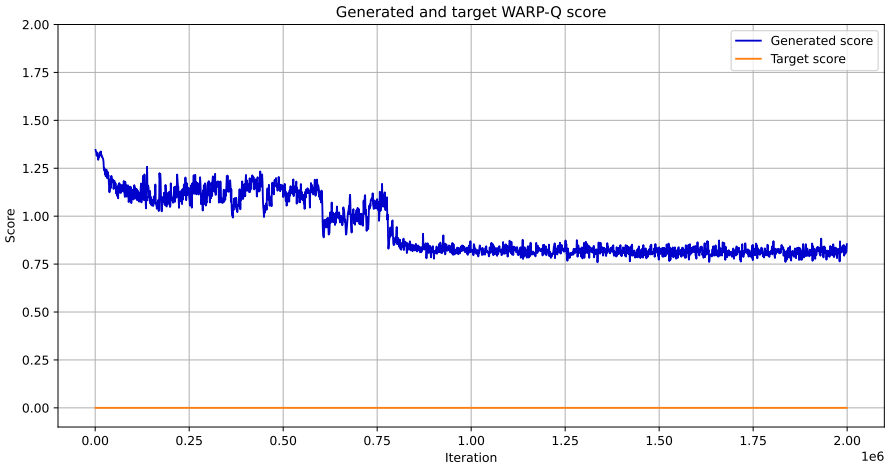


Figure A.11: native Demucs WARP-Q score

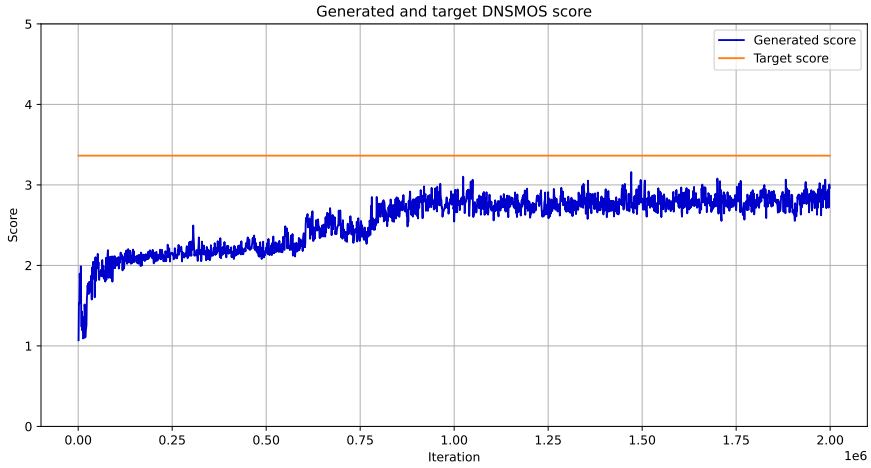


Figure A.12: native Demucs DNSMOS score

A.2.2 Results with dilated Demucs architecture

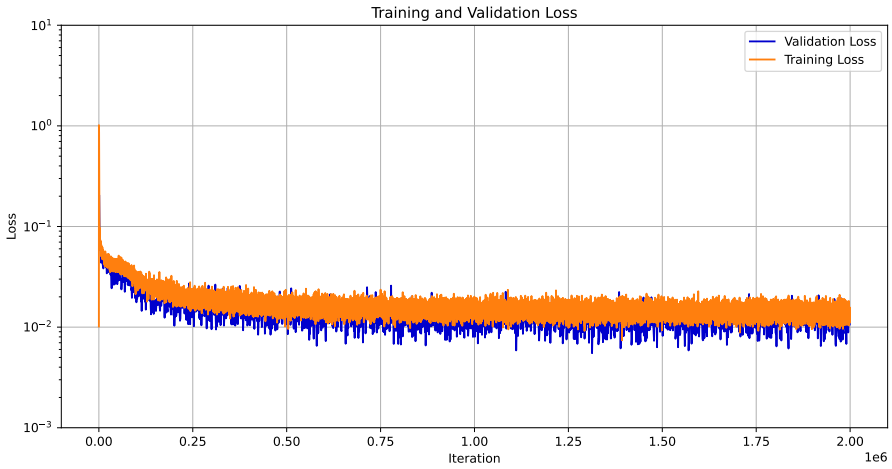


Figure A.13: Dilated Demucs training and validation loss

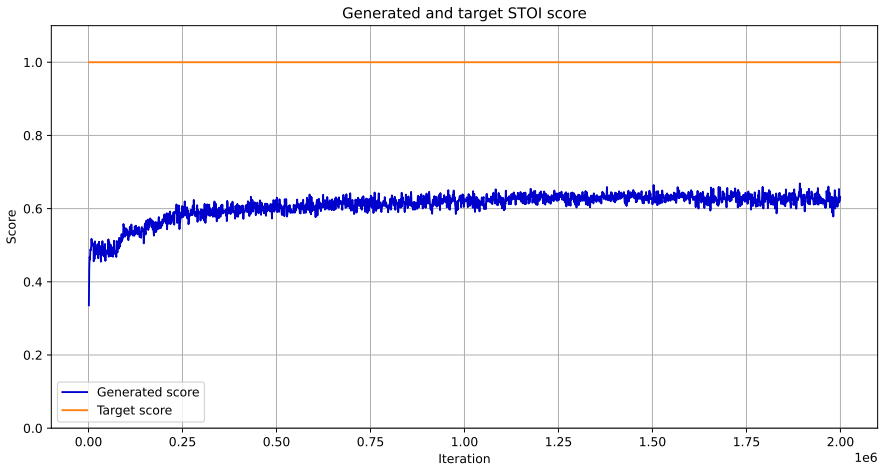


Figure A.14: Dilated Demucs STOI result

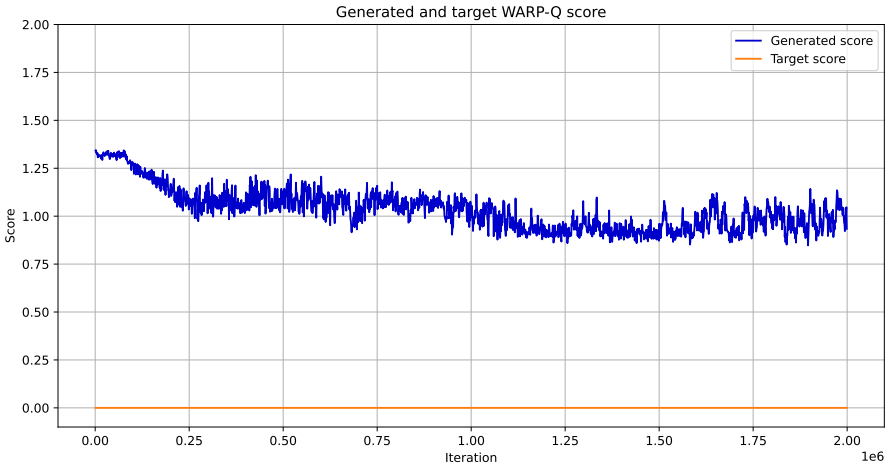


Figure A.15: Dilated Demucs WARP-Q score

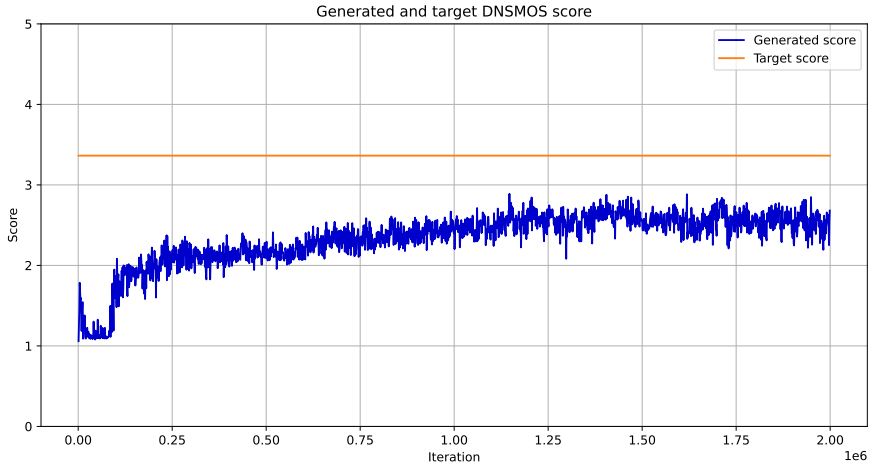


Figure A.16: Dilated Demucs DNSMOS score

A.2.3 Results for Demucs with Batch Normalization architecture

Due to not training until the end, speech quality assessment metrics are not shown. It is not trained until the end, due to overfitting.

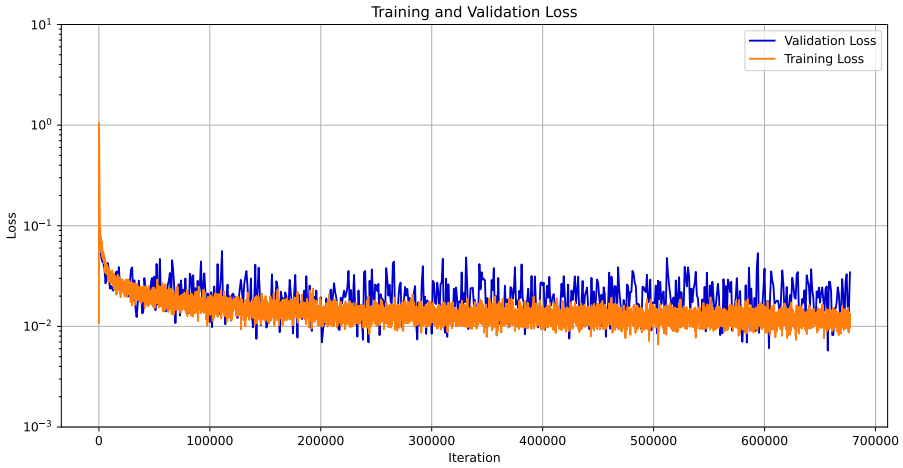


Figure A.17: Demucs with Batch Normalization training and validation loss

A.2.4 Results for Dilated Demucs with Batch Normalization architecture

Due to not training until the end, speech quality assessment metrics are not shown. It is not trained until the end, due to overfitting.

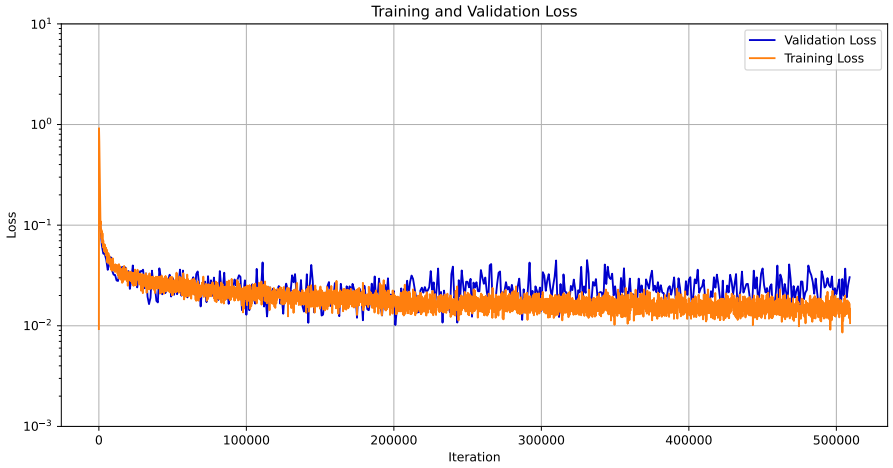


Figure A.18: Dilated Demucs with Batch Normalization training and validation loss

B | Experiment with different time embedding

This chapter is made to show, how different time embeddings perform, in the same architecture. The experiments are all carried out with the architecture from Demucs, used within the AllInOne framework.
It is shown, how even a simple time-embedding increase perform, but also how more complex embeddings can improve the results further.
All the embeddings, and the reasoning behind theese can be found in section 4.3.2

B.1 Simple time-embedding

This section aims to show the difference in training with, and without, even a simple embedding of time. The simple embedding of time, is made such that the time step is broadcast over the length of the input signal, and then concatenated. A block-diagram of this is shown in figure B.3.

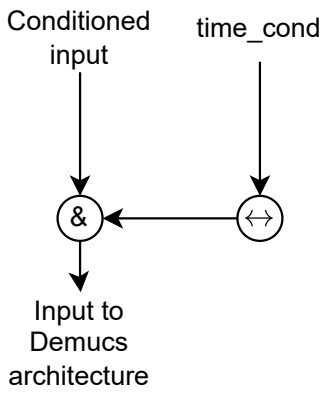


Figure B.1: Block-diagram showting how the simple embeddig of time is made.

In figure B.2 the training and validation loss is shown, when there is no embedding of time. in figure B.3 the training and validaiton loss is shown, when there is time embedding as described.

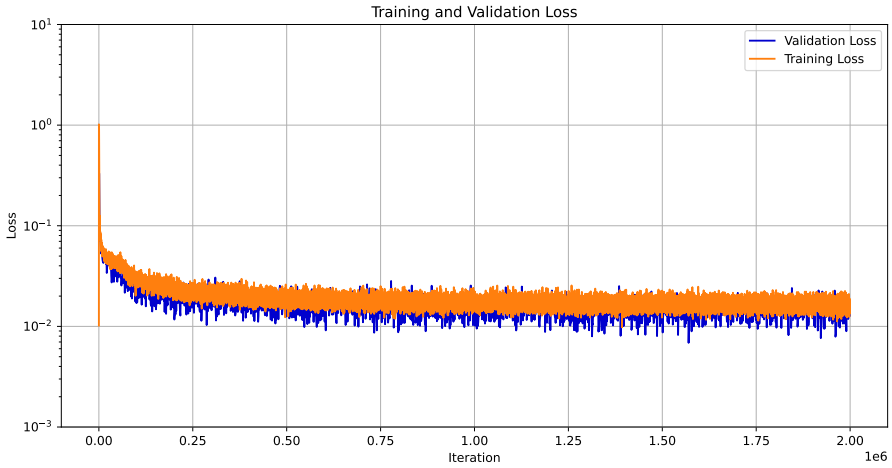


Figure B.2: Native Demucs architecture, without any time embedding, training and validation loss

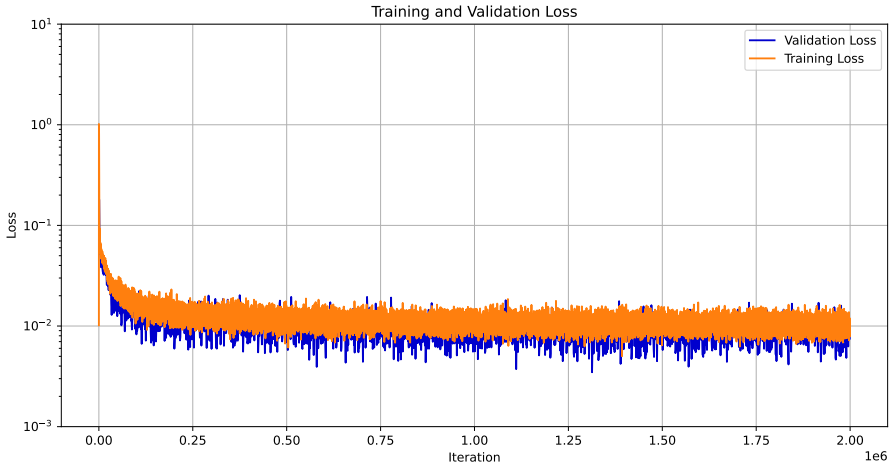


Figure B.3: Native Demucs architecture, with a simple embedding of time, training and validation loss

B.2 Time-embedding with positional encoder

This section shows the results from increasing the complexity of the time-embedding, by using a so-called positional encoder. The idea is show in figure B.4. The results from these embeddings are shown in figure B.5 - B.8.

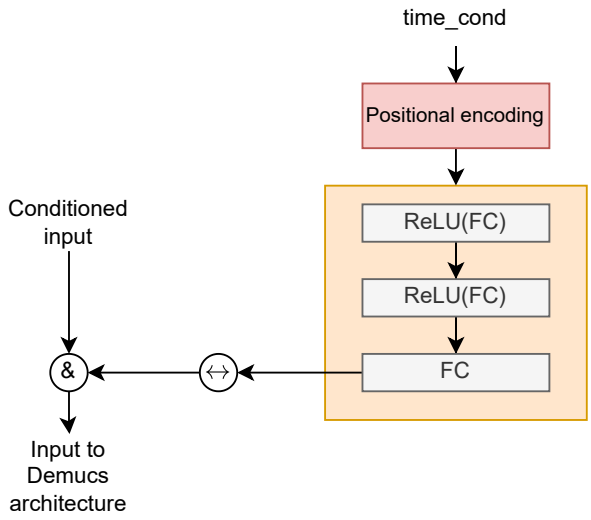


Figure B.4: A more complex embedding of the time step "t" that utilize positional encoding

Figure B.5: training and validation loss for Demucs with time-embedding as in figure B.4

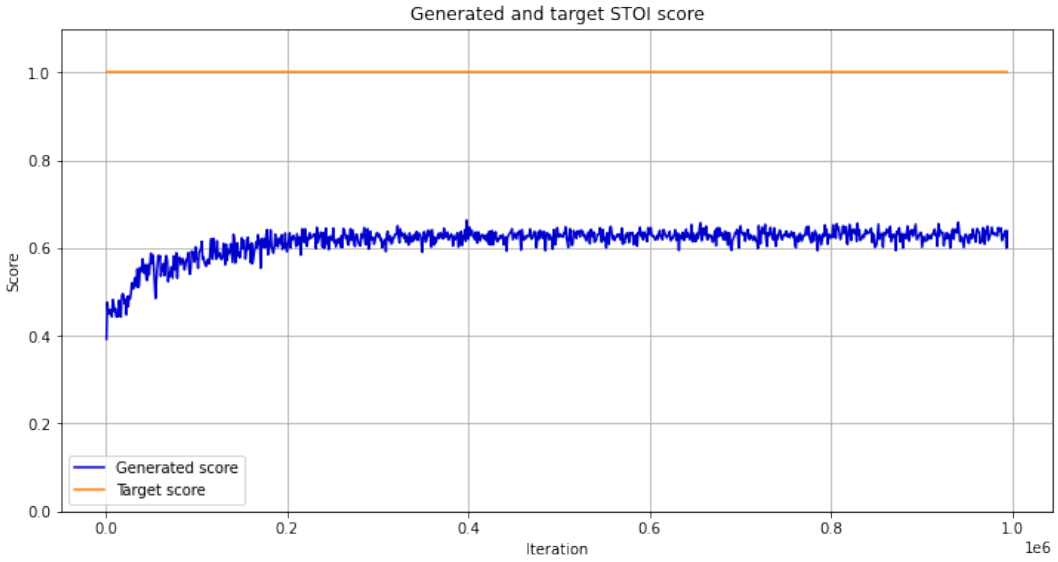


Figure B.6: STOI score for Demucs with time-embedding as in figure B.4

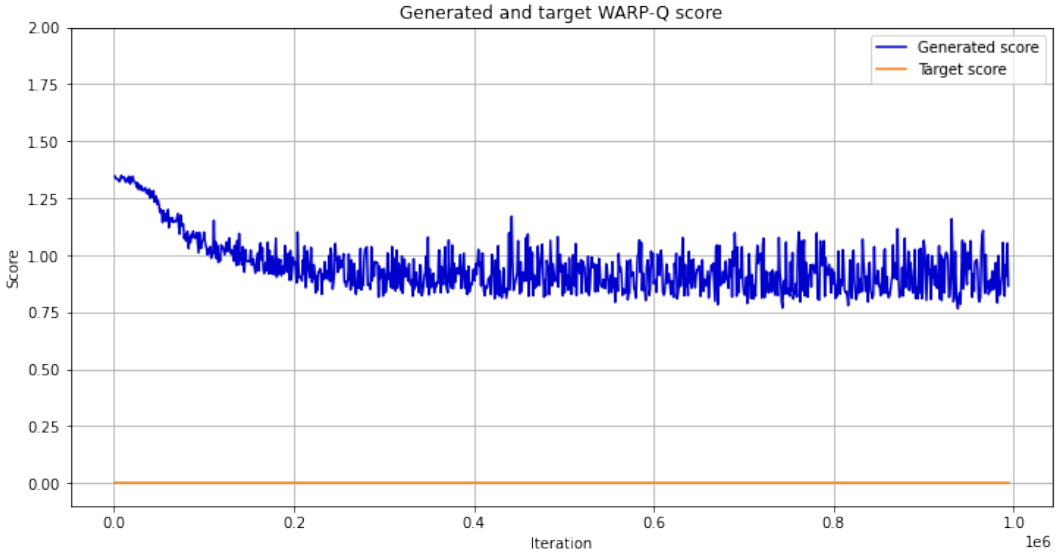


Figure B.7: WARPQ score for Demucs with time-embedding as in figure B.4

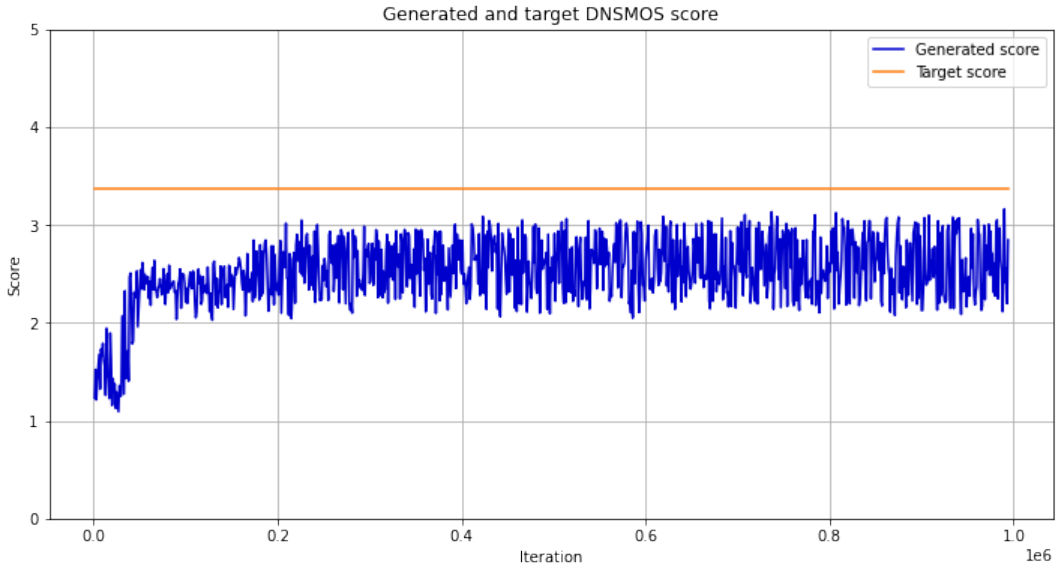


Figure B.8: DNSMOS score for Demucs with time-embedding as in figure B.4

B.3 Time-embedding with positional encoder inputted to each encoder and decoder

This section shows the results from increasing the complexity of the time-embedding, by taking the time-embedding from the last experiment but inputting the output into each encoder and decoder. The idea is show in figure B.9. The results from these embeddings are shown in figure B.10 - B.14. Please note, that the model has the posibilty to improve further, as it is stopped while the loss still improved.

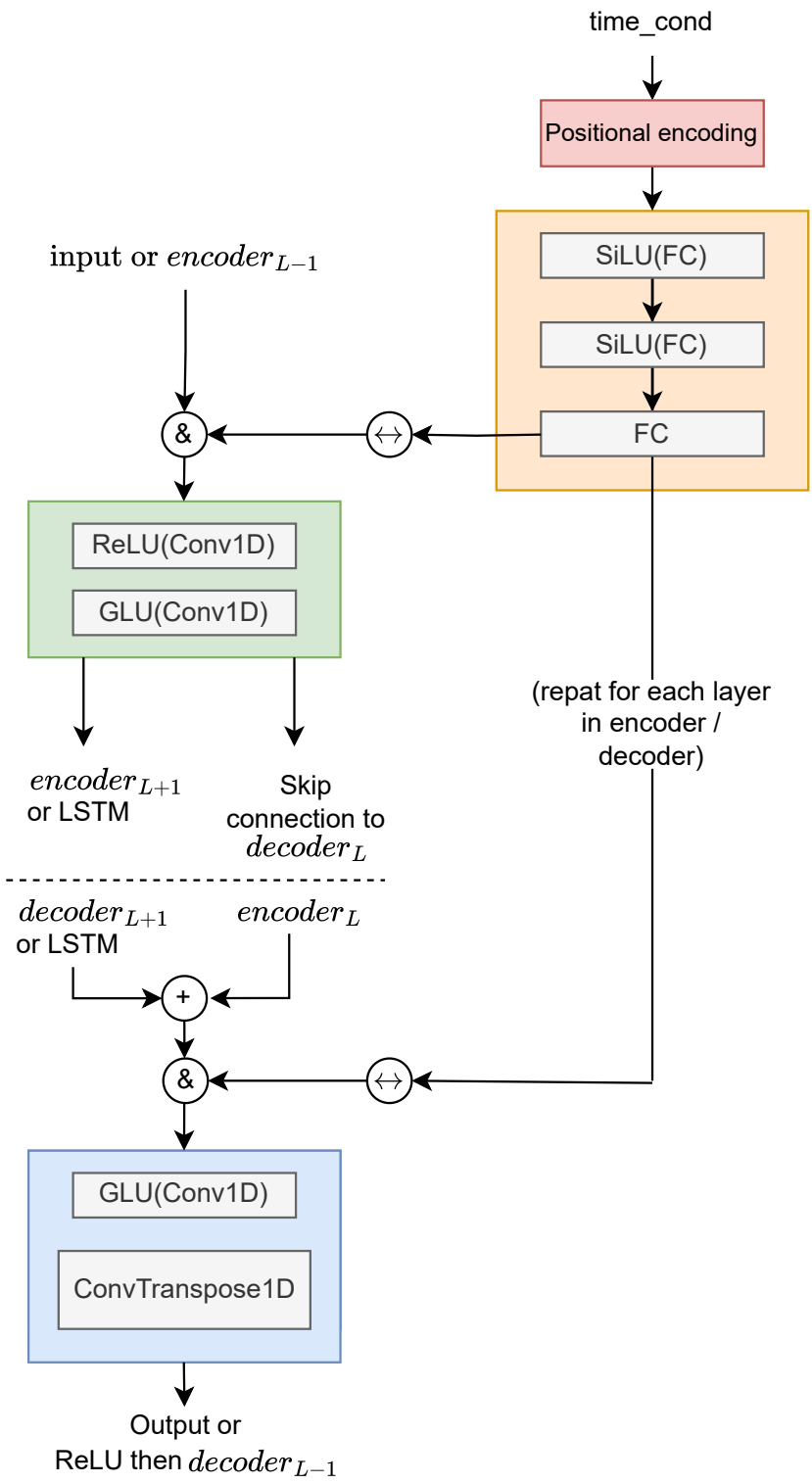


Figure B.9: An embedding of timestep "t" that uses positional encoding, three fully connected layers, and concatenation into each encoder and decoder.

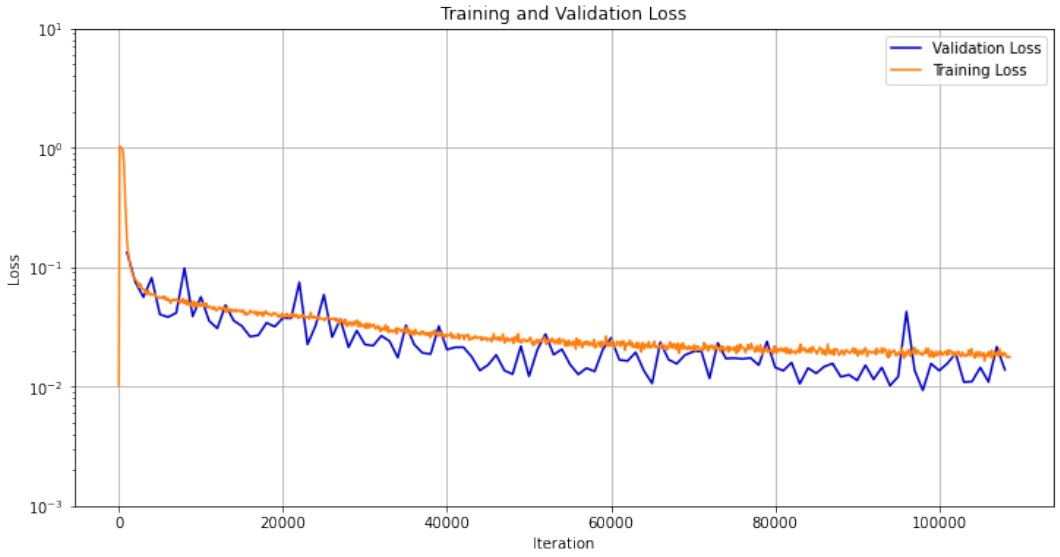


Figure B.10: training and validation loss for Demucs with time-embedding as in figure B.9

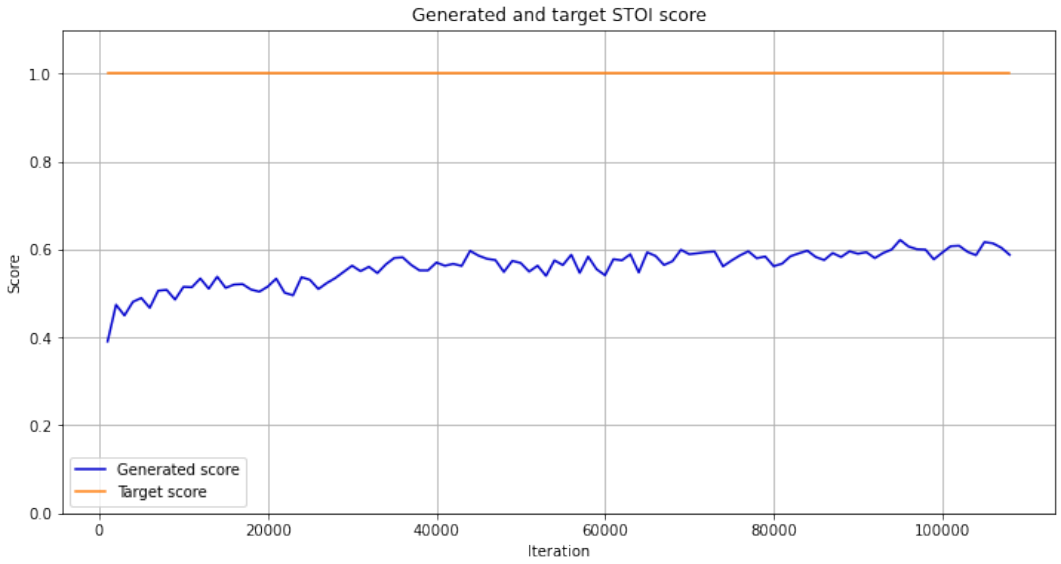


Figure B.11: STOI score for Demucs with time-embedding as in figure B.9

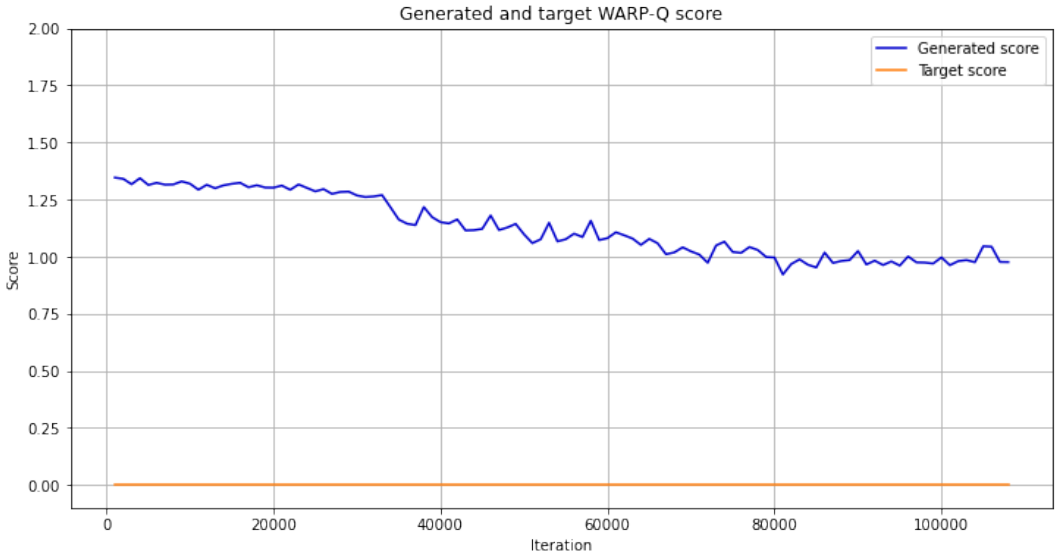


Figure B.12: WARPQ score for Demucs with time-embedding as in figure B.9

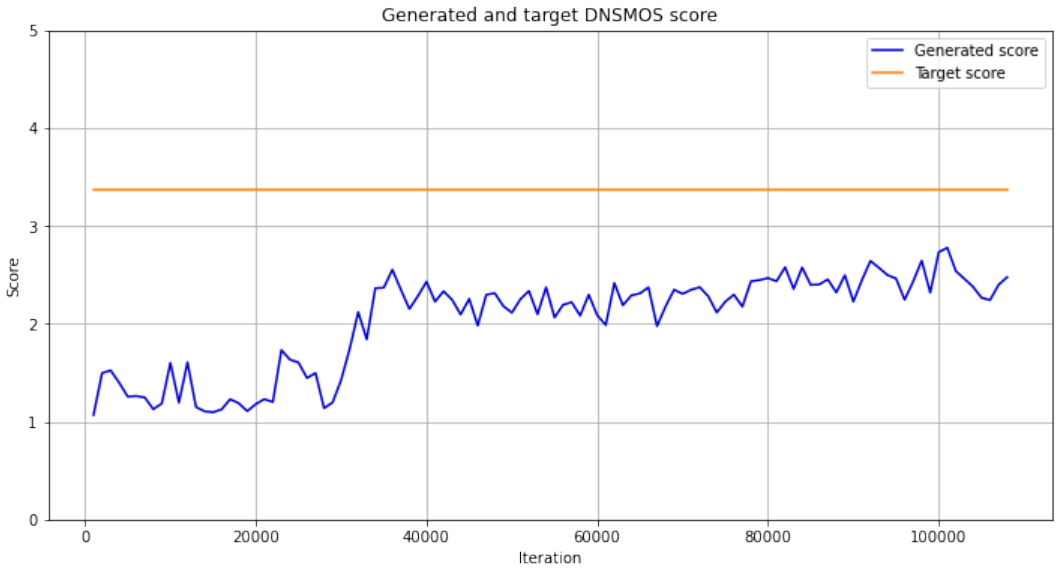


Figure B.13: DNSMOS score for Demucs with time-embedding as in figure B.9

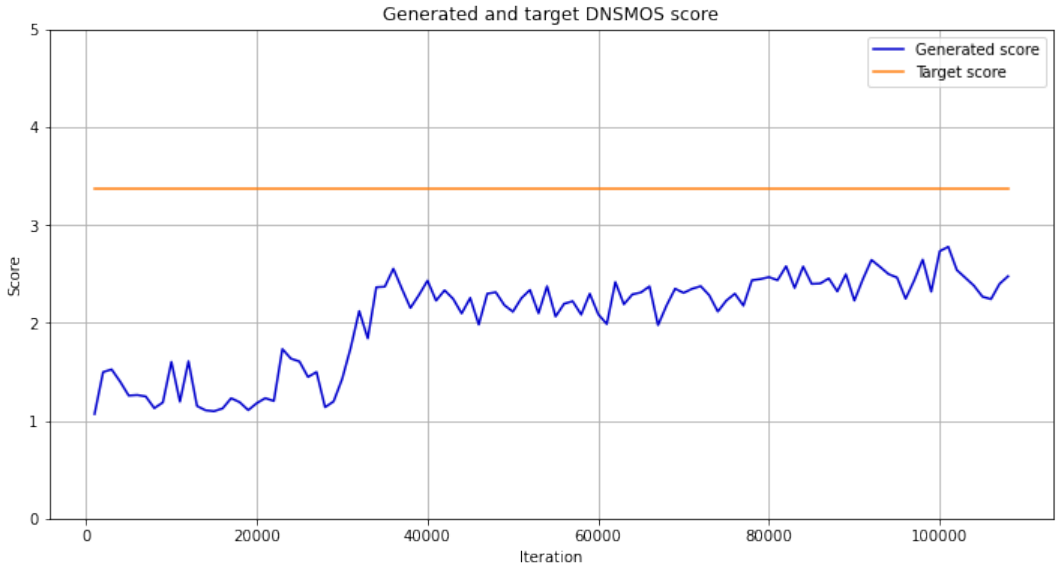


Figure B.14: DNSMOS score for Demucs with time-embedding as in figure B.9

B.4 Comparison between simple time embedding, and the complex time embedding

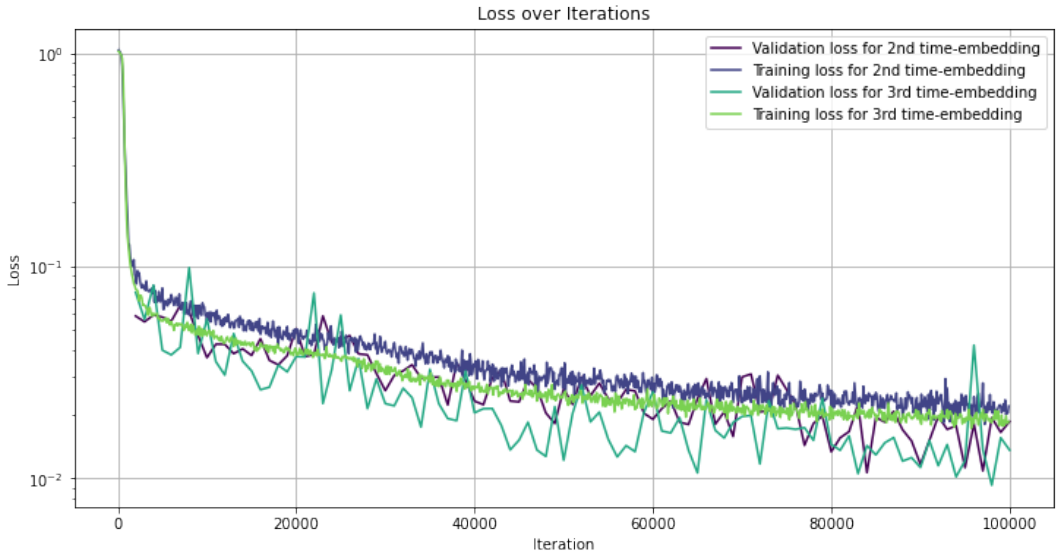
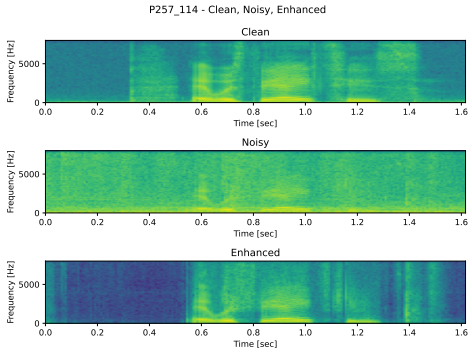


Figure B.15: Loss curves for simple time-embedding, and the complex time-embedding.

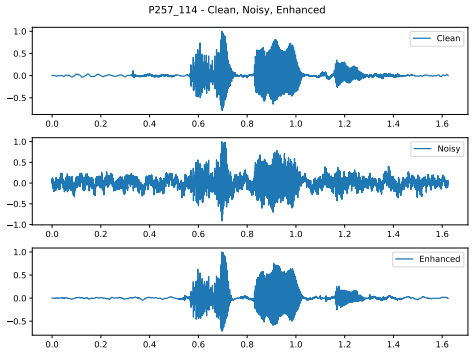
C | Visual impression of enhanced data

In this section, a visual impression of the results is shown. For each model, the same samples, chosen randomly, with 2.5 dB SnR and 17.5 dB SnR is shown.

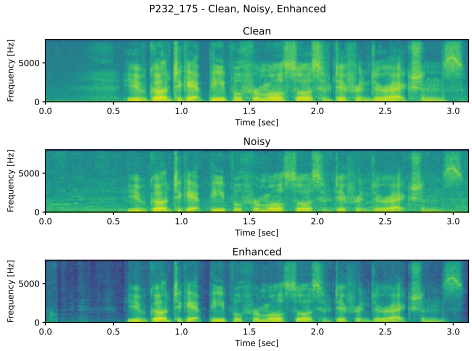
C.0.1 Facebook Demucs



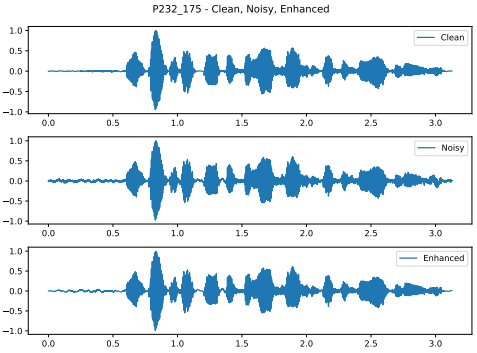
(a) Figure showing the signal in the frequency domain, where the input signal has a SnR of 2.5 dB.



(b) Figure showing the signal in the time domain, where the input signal has a SnR of 2.5 dB.

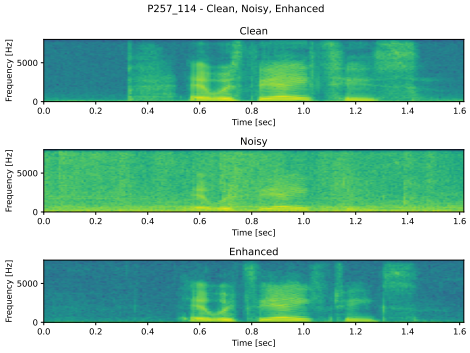


(c) Figure showing the signal in the frequency domain, where the input signal has a SnR of 17.5 dB.

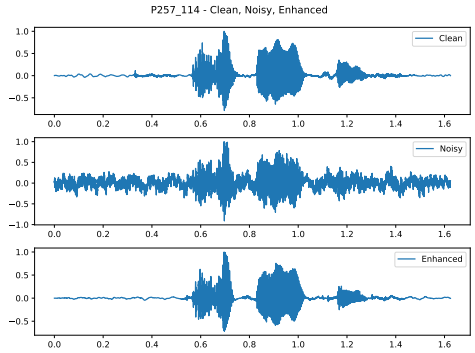


(d) Figure showing the signal in the time domain, where the input signal has a SnR of 17.5 dB.

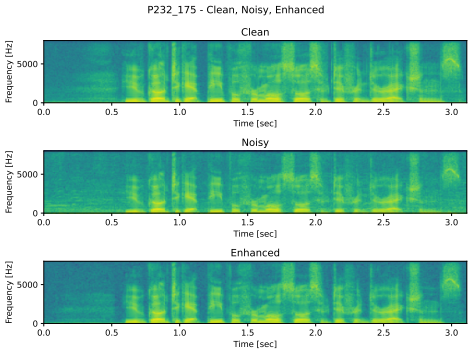
C.0.2 SGMSE



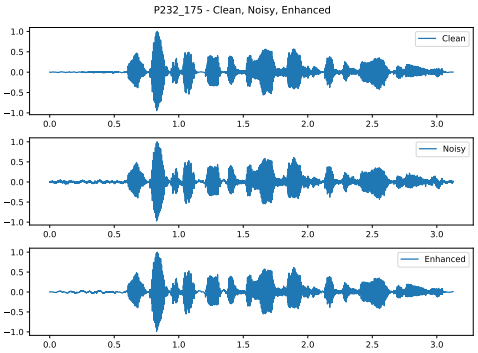
(a) Figure showing the signal in the frequency domain, where the input signal has a SnR of 2.5 dB.



(b) Figure showing the signal in the time domain, where the input signal has a SnR of 2.5 dB.

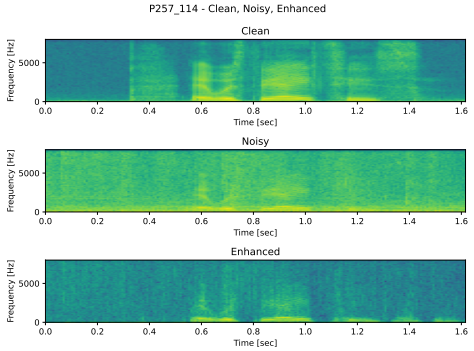


(c) Figure showing the signal in the frequency domain, where the input signal has a SnR of 17.5 dB.

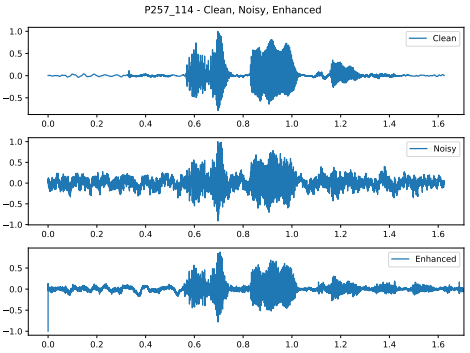


(d) Figure showing the signal in the time domain, where the input signal has a SnR of 17.5 dB.

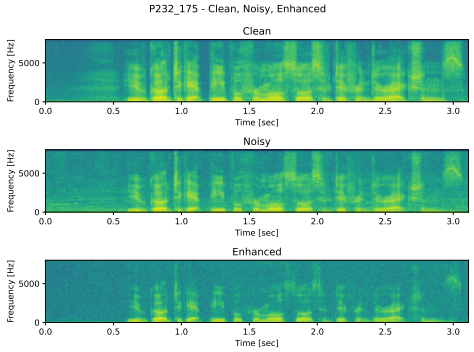
C.0.3 Non diffusion aware E2E model



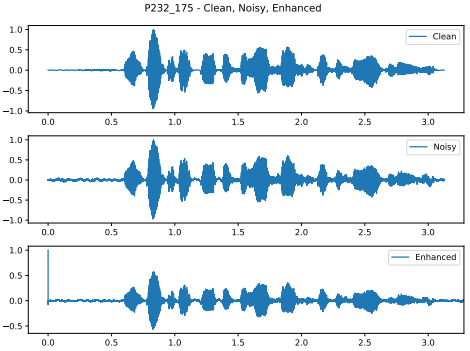
(a) Figure showing the signal in the frequency domain, where the input signal has a SnR of 2.5 dB.



(b) Figure showing the signal in the time domain, where the input signal has a SnR of 2.5 dB.

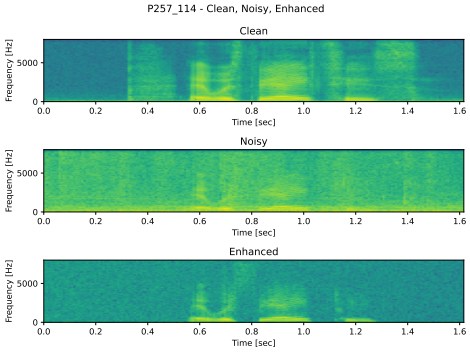


(c) Figure showing the signal in the frequency domain, where the input signal has a SnR of 17.5 dB.

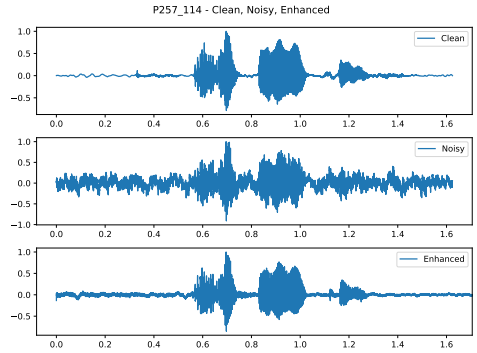


(d) Figure showing the signal in the time domain, where the input signal has a SnR of 17.5 dB.

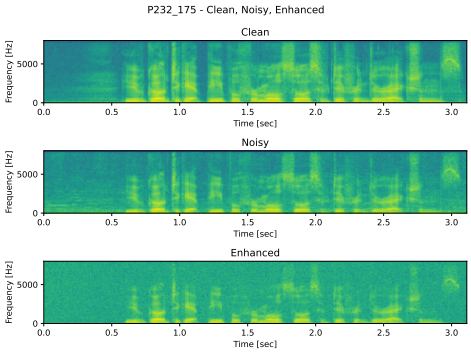
C.0.4 E2E model



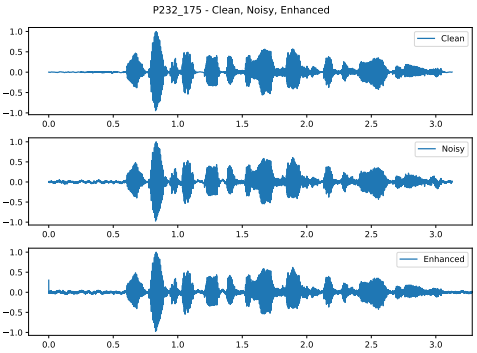
(a) Figure showing the signal in the frequency domain, where the input signal has a SnR of 2.5 dB.



(b) Figure showing the signal in the time domain, where the input signal has a SnR of 2.5 dB.



(c) Figure showing the signal in the frequency domain, where the input signal has a SnR of 17.5 dB.



(d) Figure showing the signal in the time domain, where the input signal has a SnR of 17.5 dB.

D | Postprocessing filter

To remove any residual white noise, a post filter is tested. The filter of choice is a spectral subtraction filter due to simple implementation, and improved performance on this exact data compared to a wiener filter implemented with Scipy [Virtanen et al., 2020].
The implementation of the filter, assumes that there is no speech in the first part of the signal. This is tested randomly in the dataset, but not verified on all data. The filter is implemented as shown in code D.1, where the noisy data is transformed into the STFT domain. The part of the data that is assumed to be speech less is then used to calculate the power spectrum. The power spectrum is then subtracted from the entire noisy signal's power spectrum. The phase is then reatined in the last step before doing the inverse STFT turning it back into the time domain.

Pseudocode D.1: The code used to remove the last white noise

```
def spectral_subtraction(noisy_signal, noise_estimation):
    # Compute the STFT of the noisy signal
    f, t, Zxx = signal.stft(noisy_signal, fs=16000, nperseg
       =256)

    # Estimate the noise power spectrum
    noise_spectrum = np.mean(np.abs(Zxx[:, :noise_estimation
       ])**2, axis=1, keepdims=True)

    # Subtract the noise power spectrum from the noisy signal
       's power spectrum
    signal_power = np.abs(Zxx)**2
    signal_power_subtracted = np.maximum(signal_power -
       noise_spectrum, 0)

    # Construct the denoised STFT
    Zxx_denoised = np.sqrt(signal_power_subtracted) * np.exp
       (1j * np.angle(Zxx))

    # Compute the inverse STFT to obtain the denoised signal
    _, denoised_signal = signal.istft(Zxx_denoised, fs=16000)

    return denoised_signal
```

E | Listening experiment

E.1 Experimental listening test

The measurement is performed by Magnus Munk Jensen as part of hes Master thesis at AAU. The measurements take place in May of 2024 and demonstrate the quality of the proposed end-to-end diffusion model compared to the state-of-the-art SGMSE model.
The experimental setup undergoes testing in a pilot experiment, where subjects can indicate any discomforts or areas they find confusing.
The experiment is designed as a variation of the Mean Opinion Score (MOS) test, as described by the International Telecommunication Union, Telecommunication standardization sector [International Telecommunication Union (ITU), 1996, 2003].
It differs from the standard, as the Absolute Category Scaling (ACR) is replaced with a open-ended Visual Analog Scale (VAS). This is to avoid problems, described in [Le Maguer et al., 2024], where the ACR is not accurate enough for modern neural networks. The scale is open-ended to avoid problems with lumping.
The data used for the listening test stem from a dataset published by the University of Edinburgh [Valentini Botinhao et al., 2016]. The data is recordings from 1 women, and 1 man, both from England, and with their native language being English. The noise in the dataset is background noise from a living room, an office space, a bus, and 2 different streets. The Signal to Noise Ratio (SnR) of the test data is between 2.5 dB to 17.5 dB. The original sampling frequency is 46 kHz. For the dataset to work with the model, this has been lowered to 16 kHz with Librosa [McFee et al., 2015]. In the dataset there is 824 speech recordins, where the speech that the subjects are listening to has been chosen randomly, ensuring all SnR's is represented.

E.1.1 Procedure

The measurements are made in a sound treated room, at Aalborg University, specificly 'Cabin B' in the 'B5' section at Frb. Vej 7. Before the actual listening test, the user get to familiarize with the GUI, and the speech.

E.1.2 Setup

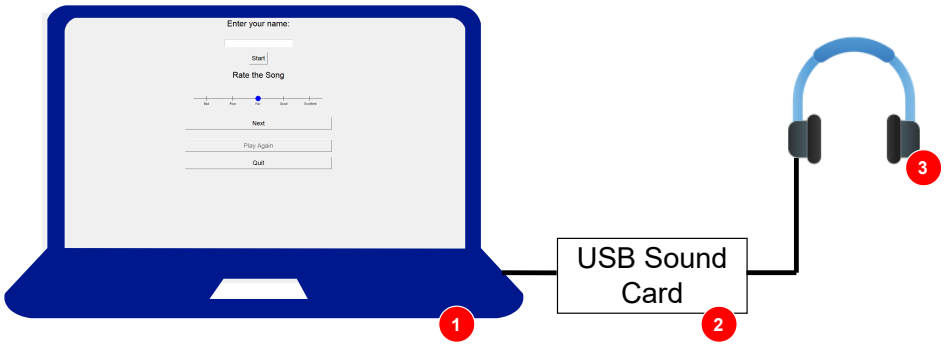


Figure E.1: A sketch showing an OAE probe, a coupler, and a tube with known length.

E.1.3 Equipment

	Equipment	**AAU number**
1	Lenovo T14S	
2	RME-AUDIO Fireface UFX II	108228
3	Beyer Dynamic DT 990 Pro	

Table E.1: Equipment used for listening test

Further, a graphical user interface (GUI) is made with Python. The GUI is shown in figure E.2. The user must input their name, but this is then randomly changed, such that the actual name is not stored. This is to ensure, different naming in the saved result, while still complying with GPDR.

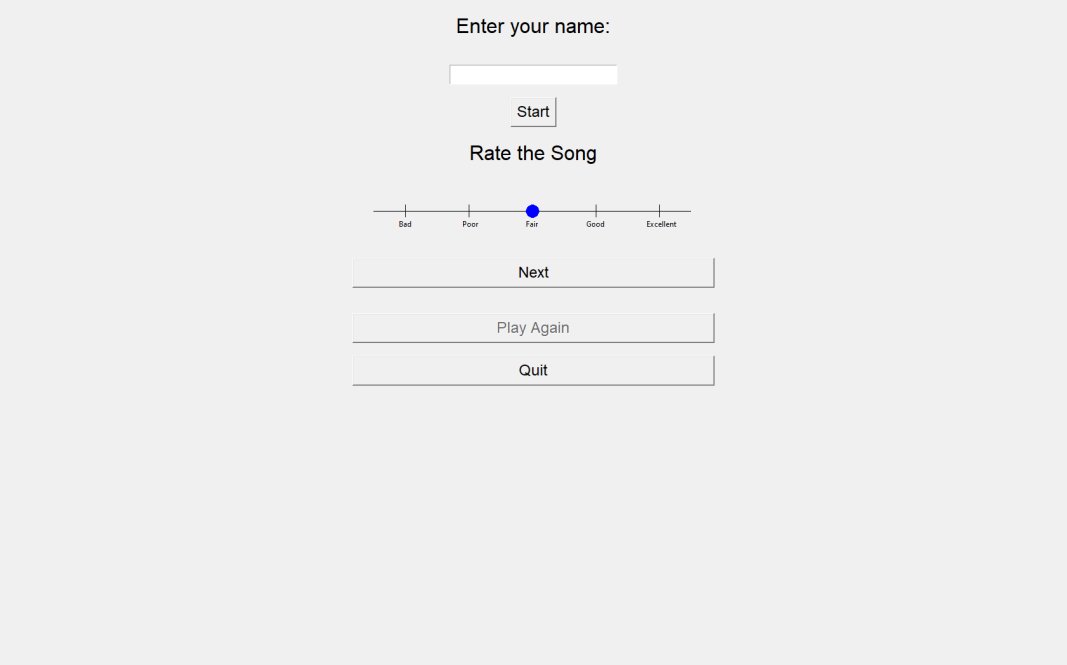


Figure E.2: The graphical user interface for the listening test

E.1.4 Calibration of equipment

In order to increase reproduceability, the played speech level is calibrated. As human speech is in the area of 60 dB A-Weighted [for Disease Control og , CDC], the calibration is made to ensure the speech is between 60 dB, and 66 dB. The speech used for the calibration, is 6 randomly chosen signals from the clean dataset [Valentini Botinhao et al., 2016]. 3 of the speech signals is by the male speaker, and 3 is from the female speaker. Each speech signal is played on repeat for 30 seconds, while the average is measured in periods of 1 second.
After the speech is calibrated, a 440 Hz test tone is also measured, in order to more easily calibrate a similar setup again. The test tone is made with Numpy, in Python. Also this is measured as an average of 1 second measurements.
The calibration is performed on a head and torso simulator, equipped with GRAS 40 AD microphones. The measurement are made in MatLab on the RME-AUDIO USB interface.

E.1.5 Results

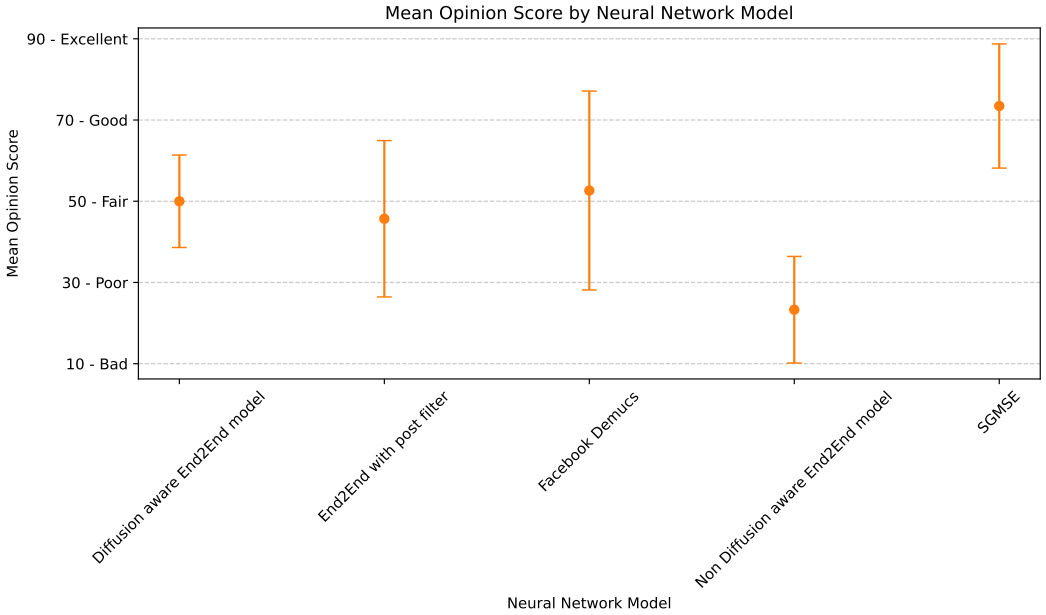


Figure E.3: The mean results in the listening experiment. The dots are the respective mean results, and the errorbars shows the standard deviation.

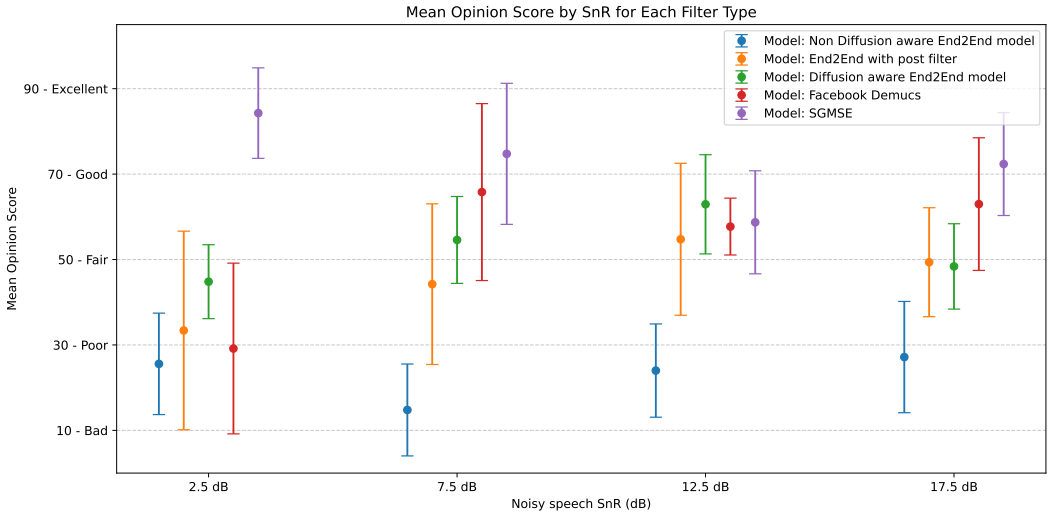


Figure E.4: A detailed version of the results from the listening experiment, showing the result split into the SnR of the noisy signal, that has been enhanced. The dots are the respective mean results, and the errorbars shows the standard deviation.