Sandro Eitzinger, BSc

Machine Learning Based Speech Separation

master's thesis

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Information and Computer Engineering

submitted to

Graz University of Technology

Supervisors

Univ.-Prof. Dipl.-Ing. Dr.mont. Franz Pernkopf

Signal Processing and Speech Communication Laboratory

Graz, January, 2024

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

date	(signature)

Acknowledgements

First and foremost, I would like to express my deepest appreciation to my supervisor, Franz Pernkopf, who endured countless meetings, encouraged me at times of dread and pushed me through the ups and downs of writing and finishing a thesis. This endeavour would not have been possible without him.

Next, I would like to thank Shahar, a researcher at the school of Computer Science, Tel Aviv University, for answering my questions with regard to the intricacies of his publications.

Furthermore, I want to thank my dad, Christian Eitzinger, for supporting me in everything I do over the last almost 3 decades.

I am also grateful for my friends Simon and Alex for being excellent teammates over the years of study.

Finally, and most importantly I want to thank my partner Sarah Granig for her never ending patience when I worked late at night, her stern encouragement when I was too demoralized to write and for everything she does for me and has done over last 15 years. I would certainly not be the person I am today and in this position without her support.

Abstract (English)

The human brain's ability to discern a single speaker within a mixture of speakers and noise is unprecedented. Teaching a computer to do the same is a challenging task where research is often inspired by human perception of auditory stimuli. Various applications rely on such speech separation models, ranging from hearing aids to voice assistants. In this work, the practical aspects of speech separation using state-of-the-art machine learning models are discussed, focusing on the Separate and Diffuse model. The required submodels, i.e., a deterministic separator model and a diffusion probabilistic based vocoder model, are introduced. The diffusion model is trained on mel-spectrogram representations of single-speaker audio utterances. Then, a convolutional neural network that aligns and merges the separator's estimation and the noisy estimation of the diffusion model is trained. For this purpose, several standard source separation datasets, including WSJ0-2/3mix, libri2/3mix, WHAM! and WHAMR!, were generated. A two stage training approach was necessary: First, the diffusion and separator models were trained on the raw datasets, i.e., WSJ0 and LibriSpeech. Then, the alignment network was trained with the diffusion model and separator as pretrained models in the machine learning pipeline. This approach resulted in SI-SDR values of 22.9dB and 21.0dB for WSJ0-2 and WSJ0-3mix benchmarks, respectively exceeding benchmarks of the baseline models. Next, an ensemble approach, that is, replacing the diffusion model with another deterministic separator and merging the two estimations was implemented and evaluated. The diffusion based model produced satisfactory results but deviated from the benchmarks described in the original paper. The ensemble based approach lead to underperforming results of 19.7dB on WSJ0-2mix and 13.7dB on WHAMR!, respectively. The ensemble model showed potential but was not able to produce satisfactory results. Overall, the improvement of the Separate and Diffuse model does not justify the additional complexity and computational overhead. Therefore, it is recommended to employ the baseline models (i.e. SepFormer) in real world applications.

Abstract (German)

Die Fähigkeit des menschlichen Gehirns, einen einzelnen Sprecher in einer Mischung aus Sprechern und Geräuschen zu erkennen, ist beispiellos. Einem Computer dasselbe beizubringen, ist eine schwierige Aufgabe, bei der sich die Forschung oft an der menschlichen Wahrnehmung von auditorischen Reizen inspiriert. Verschiedene Anwendungen, von Hörgeräten bis hin zu Sprachassistenten, stützen sich auf solche Modelle zur Sprachseparation. In dieser Arbeit werden die praktischen Aspekte der Sprachtrennung mit modernsten maschinellen Lernmodellen diskutiert. Der Schwerpunkt liegt dabei auf dem Separate and Diffuse Modell. Die erforderlichen Teilmodelle, d.h. ein deterministisches Separatormodell und ein auf Diffusion basierendes Vocodermodell, werden vorgestellt. Das Diffusionsmodell wird auf Mel-Spektrogramm-Darstellungen von Ein-Sprecher-Audio-Äußerungen trainiert. Anschließend wird ein neuronales convolutional Netzwerk trainiert, das die Schätzung des Separators und die verrauschte Schätzung des Diffusionsmodells abgleicht und zusammenführt. Zu diesem Zweck wurden mehrere Standarddatensätze für die Quellenseparation, darunter WSJ0-2/3mix, libri2/3mix, WHAM! und WHAMR! generiert. Ein zweistufiger Trainingsansatz war erforderlich: Zunächst wurden die Diffusions- und Separatormodelle auf den Rohdatensätzen, d.h. WSJ0 und LibriSpeech, trainiert. Dann wurde das Alignment-Netzwerk mit dem Diffusionsmodell und dem Separator als vortrainierte Modelle in der maschinellen Lernpipeline trainiert. Dieser Ansatz führte zu SI-SDR-Werten von 22,9 dB und 21,0 dB für die WSJ0-2mix bzw. WSJ0-3mix-Benchmarks, die die Benchmarks der Basismodelle. Als Nächstes wurde ein Ensemble-Ansatz, d.h. die Ersetzung des Diffusionsmodells durch ein anderes deterministisches Separatormodell und die Zusammenführung der beiden Schätzungen implementiert und bewertet. Das diffusionsbasierte Modell lieferte zufriedenstellende Ergebnisse, wich jedoch von den in der ursprünglichen Arbeit beschriebenen Benchmarks ab. Der auf dem Ensemble basierende Ansatz führte zu schlechteren Ergebnissen. Das Ensemble-Modell zeigte zwar Potenzial, konnte aber keine zufriedenstellenden Ergebnisse liefern. Auch wenn die Ergebnisse auf dem Papier zufriedenstellend sind, wird empfohlen, einfach die die Basismodelle zu verwenden, um den Rechenaufwand und die Komplexität zu vermeiden, die durch die in dieser Arbeit diskutierte Architektur entstehen. Der Ensemble-basierte Ansatz führte zu unterdurchschnittlichen Ergebnissen von 19,7 dB bei WSJ0-2mix bzw. 13,7 dB bei WHAMR!. Das Modell zeigte zwar Potenzial, war aber nicht in der Lage, zufriedenstellende Ergebnisse zu erzielen. Insgesamt rechtfertigt die Verbesserung des Separate- und Diffuse-Modells nicht die zusätzliche Komplexität und den zusätzlichen Rechenaufwand. Daher wird empfohlen, in realen Anwendungen die Basismodelle (d. h. SepFormer) zu verwenden.[1]

[1] Abstract translated with DeepL [1] and corrected manually.

Contents

1 Introduction

Disentangling speech signals with machine learning based approaches can be divided into two paradigms: Single-channel source separation handles the extraction of speaker utterances from monophonic recordings, i.e., recordings captured with a single microphone. Multichannel source separation extends this challenge to settings with multiple microphones. This thesis focuses primarily on singe channel source separation (SCSS).

1.1 Scope of the thesis

Speech separation, commonly referred to as the cocktail party problem [2], is a source separation problem in audio processing where only the overlapping speech signal sources are considered [3]. Applications of speech separation are wide-ranging from voice assistants to hearing aids.
To achieve more effective and efficient speech separation methods, various machine learning techniques have been explored in the past [4]–[7] each one striving to push the envelope of what is possible in terms of quality and performance.

In this work, the practical aspects of speech separation using state-of-the-art Machine Learning (ML) models are explored. Utilizing deep learning models, we aim to untangle the complex auditory scenes of multi speaker mixtures to provide cleaner and more intelligible speech.
The core of this journey is the recently suggested "Separate and Diffuse" model introduced in [8]. This model combines the capabilities of deterministic separation models such as Separation Transformer (SepFormer) [4] with a generative diffusion model.

The Separate and Diffuse model is the culmination of cutting edge research in the field. Deterministic models such as SepFormer excel at capturing intricate patterns in audio mixtures to create clean seperated audio sources. However, deterministic separation models such as SepFormer, MossFormer [5] or Gated-Long Short Term Memory (LSTM) [7] are restricted by the single channel speech separation bound introduced in [6].
DiffWave, a diffusion based generative vocoder model [9], is introduced to create realistic audio samples. The amalgamation of deterministic source separation and generative sample creation leads to state-of-the-art performance, exceeding the previously mentioned boundaries for speech separation.

In the subsequent chapters, the technical intricacies of the implementation will be discussed. The architecture of all models used in this work is elucidated. Further, a crucial aspect lies in the comparisons of the Separate and Diffuse model with the respective baseline models.
Several evaluation metrics are employed to analyze to capabilities and the improvement of the Separate and Diffuse model. Additionally, the model is tested in various standardized benchmark scenarios. This will provide a better understanding and be the basis for further models which introduce diffusion based models into their architecture.

Additionally, an alternative approach, based on multiple deterministic models is implemented. Replacing the diffusion model with another separator model, in this case SUccessive DOwnsampling and Resampling of Multi-Resolution Features (SuDoRM-RF) [10], [11] allows the reduction

of computational overhead introduced by the diffusion process. The aim is to build an ensemble of separation models outperforming a single model.

1.2 Structure and Organization

Chapter 2 presents the necessary fundamentals needed to comprehend the subsequent chapters. The machine learning techniques used in speech separation are discussed. Further, the separator models used in this work (SepFormer & SuDoRM-RF) are elaborated. Afterward, diffusion and the DiffWave model are introduced. Finally, the Separate and Diffuse model architecture and its inner workings and the ensemble model are presented.

Chapter 3 introduces the ML framework, the datasets and the evaluation metrics used, and discusses the training setup. First, we present the datasets that were utilized and the preprocessing techniques applied to them. Subsequently, the Asteroid framework and setup to train the models is introduced. Finally, the metrics that were used to measure the separation capabilities of the models are discussed.

The experimental results of the implementation are presented within Chapter 4. First, the baseline models are evaluated to give a common ground for comparison. Next, a suite of pertinent, de facto standard evaluation metrics and methodologies are employed to analyze to capabilities and the improvement of the Separate and Diffuse model. To do so, we initially determine the evaluation metrics at the input of the model i.e., the mixture against the clean signal and the estimated separation against the clean signal. Afterward, we juxtapose the results against state-of-the-art techniques, i.e., the utilized baseline models, to gauge the improvement of the speech separation method by including diffusion into the pipeline. Then, the ensemble model is evaluated and analyzed in similar fashion. Finally, the utilized models are compared in terms of model size and inference time, and a closing conclusion is drawn.

All results where obtained utilizing evaluation datasets containing 3000 data samples created directly from the datasets described in Section 3.1.2 and Section 3.1.6.

In the final chapter, Chapter 5, the methodology and main findings of this work are summarized. A final verdict on the feasibility of real world applications of the models presented in this thesis is given. Lastly, we provide an outlook for possible future research topics and expansions of this work.

2 Related Work and Fundamentals

The initial chapter of this work presents the necessary fundamentals needed to comprehend the subsequent chapters. The machine learning techniques used in speech separation are discussed. Further, the separator models used in this work (SepFormer & SuDoRM-RF) are elaborated. Afterward, diffusion and the DiffWave model is introduced. Finally, the Separate and Diffuse model architecture, and its inner workings, and the ensemble model are presented.

2.1 Machine Learning in Speech Separation

The emergence of machine learning introduced several deep learning models for single channel source separation, which lead to significant improvements regarding speed and performance in comparison to traditional signal processing methods. Several models will be covered in this chapter, elucidating the progression of machine learning based speech separation models up to the current state-of-the-art.

2.1.1 Clustering and PIT

The first successful deep learning based model, deep clustering, utilizes deep learning architecture to discriminately learn embeddings used as the basis to achieve separation [12]. A Recurrent Neural Network (RNN) [13] models the embeddings for each time-frequency bin. Applying a basic clustering algorithm, e.g., k-means [14] clusters the embeddings into S speaker clusters. Following the success of deep clustering, several novel ideas and methods were introduced [15]–[21]. All of them further improving separation, addressing limitations of previous methods and posing new challenges in speech separation, one paper at a time. Yu et al. [22] introduced Permutation Invariant Training (PIT), a novel deep learning training criterion. PIT attempts to minimize the separation error directly as opposed to deep clustering. This solves two problems posed by earlier methods:
First, estimation of a mask is problematic in segments containing silence. Here the mask is not well-defined. Secondly, a smaller error on masks does not necessarily imply a smaller error between the estimated magnitude of the separation and the ground truth magnitude of each respective source. Methods mentioned earlier interpret speech separation as a multi-class regression problem, where a set number of frames N of the mixture is used, to feed a deep learning model to generate a frame of masks for each speaker present. Using the masks, a single frame of speech utterance is generated for every source. To train these models, the corresponding target magnitude has to be provided to each output layer to calculate a loss. Due to the fact that multiple output layers are necessary in the model (a separate one for each speaker source), the assignment of correct reference magnitudes is challenging. This is commonly referred to as the label ambiguity or permutation problem [12], [23]. Figure 2.1 shows a schematic overview of PIT in the context of two-speaker separation. Since PIT operates on frame-level, the problem of speaker tracking arises when attempting to concatenate consecutive frames together. A solution to this was Utterance-level Permutation Invariant Training (uPIT) [19]. uPIT extends PIT with

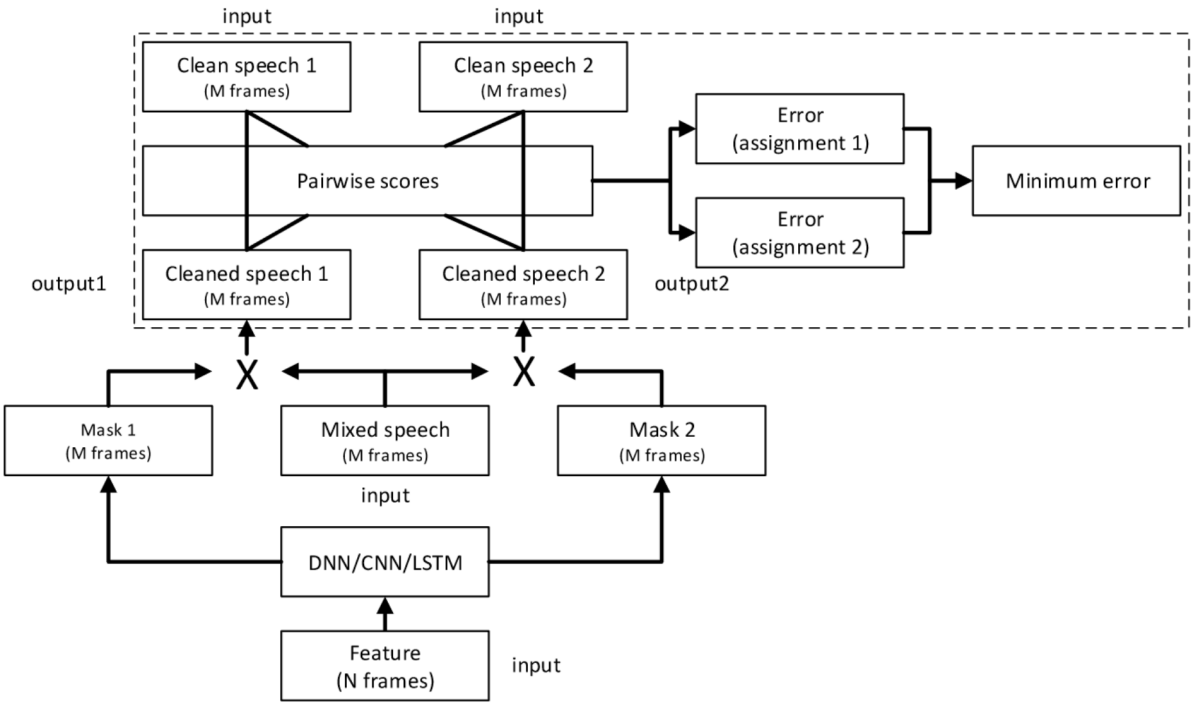


Figure 2.1: PIT in a two-speaker speech separation setting. Image courtesy of [22].

an utterance-level cost function. This eliminates the need of solving an additional permutation problem during inference, which is necessary due to the frame-level nature of PIT.

Parallel to these developments, Wang et al. [16] partially addressed the issue of reconstruction errors introduced by phase inconsistencies. This approach does not rely on surrogate loss functions on the ground truth short-time Fourier Transform (STFT) magnitudes, but instead defines the loss function directly on the reconstructed speech waveform.
A commonality of these techniques lies therein, that the methods are applied to the signals in the time-frequency domain. To do so, the STFT of the magnitude and phase components of the input mixture is calculated. The separator models are then trained on the magnitude component alone. During reconstruction of the signal, the improved phase component (or the phase of the mixture signal) is used in addition to the obtained magnitude component to reassemble the output waveform for each speaker using inverse short-time Fourier Transform (iSTFT).

2.1.2 ConvTasNet

Time-domain Audio Separation Network (TasNet) employed an encoder-decoder framework to model the signal in the time-domain directly. This removes the need to perform frequency decomposition and reduces the separation task to an estimation of source masks on encoder outputs alone, which are then synthesized by the decoder [24]. This approach greatly improved the minimum latency of the model and reduced the computational cost. Figure 2.2 shows the encoder-decoder architecture introduced by TasNet. The encoder models the input signal in time-domain to a high-dimensional representation. Separation is performed by a (in this case) deep LSTM model on the non-negative outputs of the encoder to calculate source masks. A source mask is a multiplicative function for each target source. To reconstruct the source, the source masks are applied to the mixture weights, which then get reconstructed by the decoder module. Convolutional Time-Domain Audio Separation Network (Conv-TasNet) [25] is an improvement to TasNet. Similarly to TasNet, Conv-TasNet operates in the time-domain but

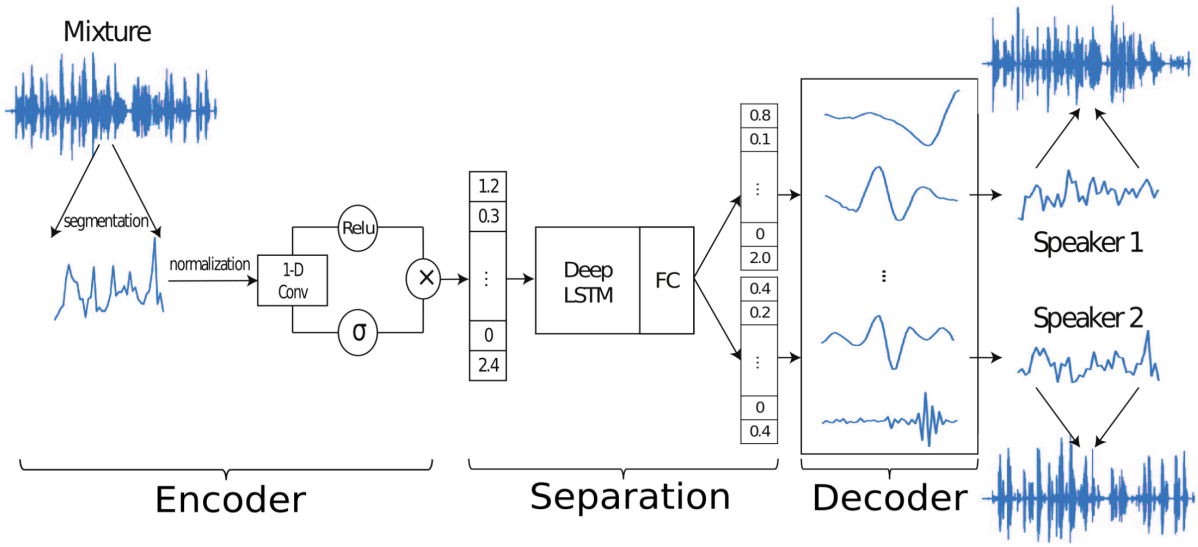


Figure 2.2: The encoder-decoder framework employed by TasNet. Image courtesy of [24].

predominantly employs Convolutional Neural Networks (CNNs) to process the audio waveforms. Further, it employs a more sophisticated architecture comprising, several convolution layers as shown in Figure 2.3. The input waveform is modelled by a 1-D convolutional autoencoder [26] and fed into a temporal convolutional network. The temporal convolutional network then estimates masks in the form of an over-complete set of filters for analysis and synthesis dependent on input received from the encoder. A detailed view of the 1-D convolutional block is shown

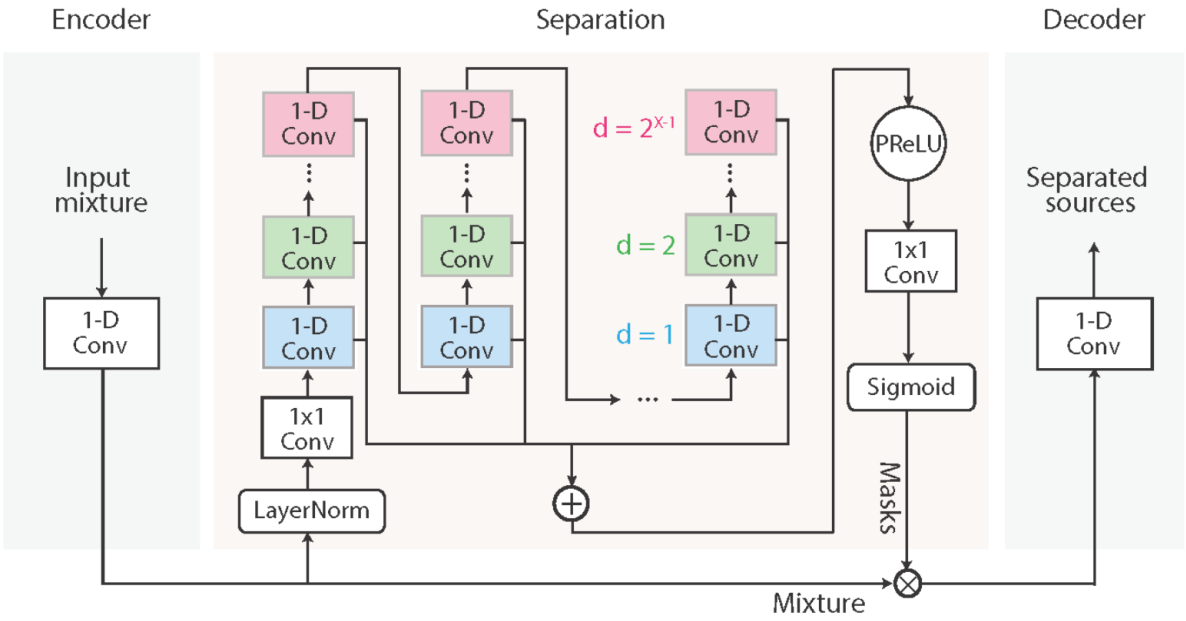


Figure 2.3: The system flowchart of Conv-TasNet. Image courtesy of [25].

in Figure 2.4. This method improves modeling of long term-dependencies in speech utterances whilst maintaining a small amount of model parameters. While these methods focused on the time-domain signal processing Liu et al. [27] introduced a deep computational auditory scene analysis (CASA) approach that optimizes frame level speaker tracking and separation.

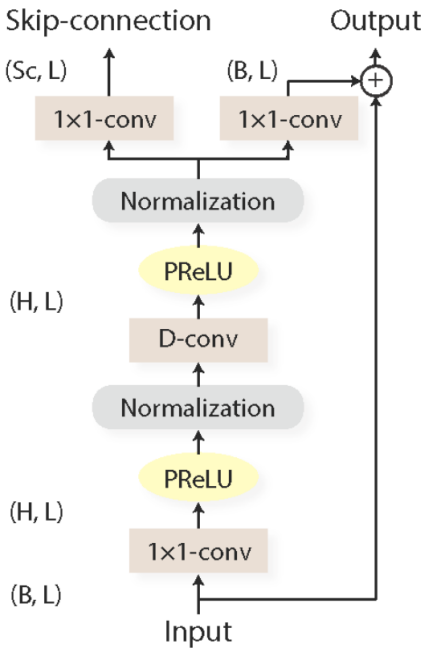


Figure 2.4: The internal layout of a 1-D convolutional block. Image courtesy of [25].

2.1.3 Dual-path RNNs

Inspired by the time-domain approaches such as TasNet, Luo et al. [28] proposed dual-path RNNs. Conventional RNNs struggle with optimization issues due to long time step sequences, while one dimensional convolutional approaches such as Conv-TasNet, fail to model utterance level sequences due to the restricted receptive field. Dual-path RNNs split long sequential input into small chunks to then iteratively apply inter- and intra-chunk operations. Figure 2.5 shows the system flowchart of a dual-path RNN:

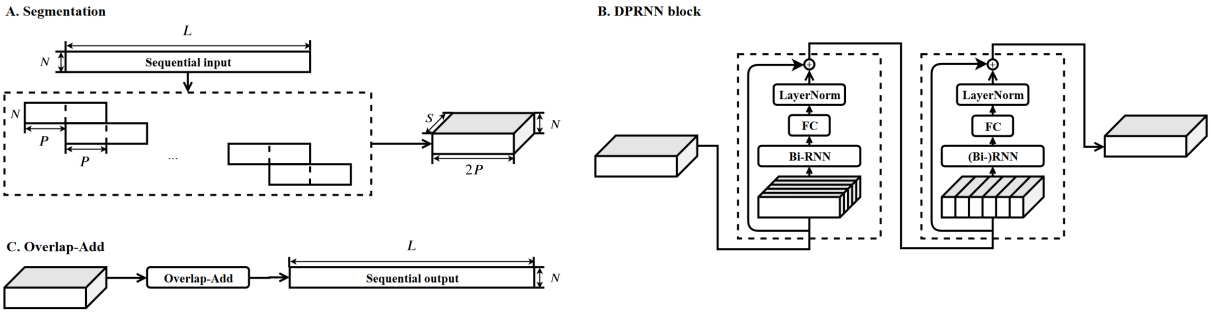


Figure 2.5: Architecture of dual-path RNNs. Image courtesy of [28].

Figure 2.5 A shows the segmentation stage of the dual-path RNN, where the sequential input gets split into overlapping chunks and concatenated into a 3-dimensional tensor.[2]
In Figure 2.5 B, two RNNs with recurrent connections in different dimensions are used inside the dual-path RNN block. To process information on the local level, a bidirectional RNN is applied on intra-chunk level, i.e., in parallel to individual chunks. Afterward, the inter-chunk RNN is used to acquire dependencies of a global nature. To increase the complexity of the architecture, several dual-path RNN blocks can be chained together. By applying an overlap-add operation to the 3-dimensional output of the dual-path RNN block (see Figure 2.5 C), the final sequential

[2] Overlapping of chunks is optional and the overlap ratio considered a hyperparameter.

output is obtained.

2.1.4 Transformers

Transformer architectures, as introduced by Vaswani et al. [29], avoid recurrent connections and focus entirely on so-called attention mechanisms instead. This allows Transformer based networks to achieve greater parallelization during training, leading to improved results whilst reducing computational time. Transformers were initially utilized in machine translation, language modeling and english constituency parsing tasks. However, due to the sequence-to-sequence nature of these tasks, Transformers proved vital in source separation tasks, as discussed in Section 2.1.5, where we focus on the SepFormer. Figure 2.6 and Figure 2.7 show the overall architecture, i.e., the encoder and decoder, employed by the Transformer.

The Encoder: The Encoder comprises six identical layers of multi-head attention with normalization (usually layer normalization) [30] and a fully connected feed forward (FFW) layer, followed by a normalization layer. Both sub-layers employ a residual connection [31].

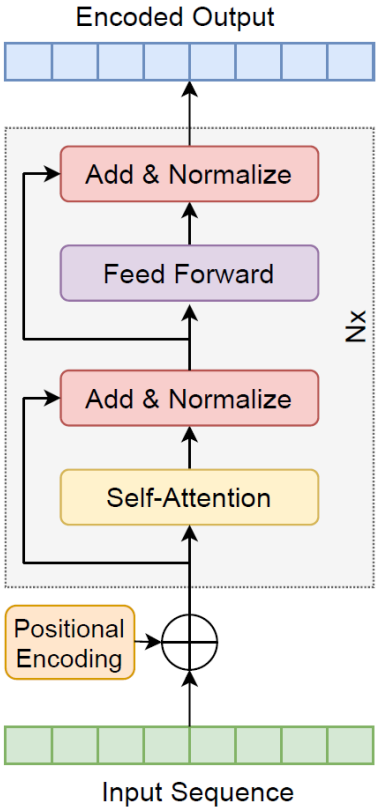


Figure 2.6: The Encoder Architecture. Image courtesy of [32].

The Decoder: The Decoder mirrors the Encoder with the addition of another sublayer consisting of another attention layer which performs attention across the Encoder's outputs as shown in Figure 2.7. Moreover, the initial self-attention mechanism is adjusted with masking to guarantee that predictions for a given position n rely solely on known inputs at positions preceding n [29].

Multi-Head Attention: Attention functions are mapping function, which map sets of key-value pairs and a query to an output [29]. The output is obtained by the computation of a

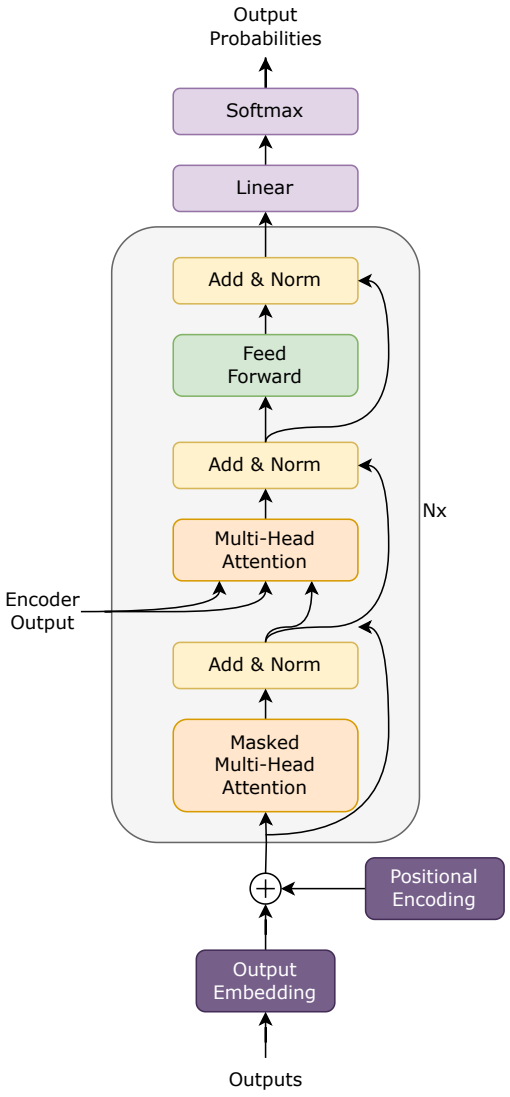


Figure 2.7: The Decoder Architecture. Created with [33].

weighted sum of the values. Hereby, the weights of each value is determined by a so called compatability function. Figure 2.8 shows the multi-head attention layout consisting of layers running in parallel, hence multi-head. The multi-head attention layer can be described as:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O, \tag{2.1}$$

where

$$\text{head}_i = \text{SDPA}(QW_i^Q, KW_i^K, VW_i^V). \tag{2.2}$$

Q, K and V are the queries, keys and values packed into matrices and W_i^Q, W_i^K, W_i^V and W^O represent the parameter matrices.

The attention function used per head, Scaled Dot-Product Attention (SDPA), is shown in Figure 2.9 and is defined as:

$$\text{SDPA}(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V, \tag{2.3}$$

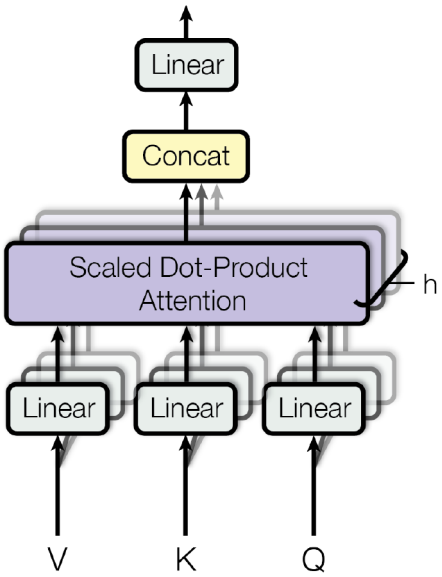


Figure 2.8: Multi-Head Attention. Image courtesy of [29].

where d_k describes the dimensionality of queries and keys.

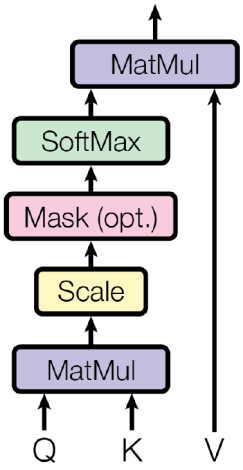


Figure 2.9: Scaled Dot-Product Attention. Image courtesy of [29].

2.1.5 SepFormer

Due to the sequential nature of source separation models that rely on RNNs, parallelization of computations in an effective manner is not possible. As mentioned in Section 2.1.4 transformer-based architectures avoid this issue by employing attention-based mechanisms instead of recurrent connections [29]. SepFormer [4] utilizes a Transformer based neural network instead of RNNs. This allows for processing of entire sequences of data which is in turn significantly less memory consuming and considerably faster than previous state-of-the-art models. Further, long-term dependencies, as are common in speech utterances, can be more easily learned by the model leading to better separation performance. SepFormer is based on the masking based source separation pipeline [24], [25] and constitutes a similar encoder-decoder based architecture

as in Section 2.1.4, but with a transformer based approach to estimate filter masks.

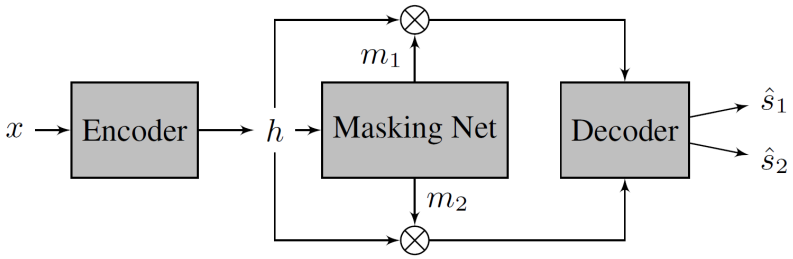


Figure 2.10: High-level overview of the model architecture. Image courtesy of [4].

Figure 2.10 shows the overall architecture from a high level. The model consists of three major components:

The Encoder Block: The encoder block utilizes a single convolutional layer to learn an STFT like representation $h \in \mathbb{R}^{F \times T'}$ of the input mixture [32], i.e.

$$h = \text{ReLU}(\text{conv1D}(x)), \tag{2.4}$$

where the input mixture is given as time-domain signal $x \in \mathbb{R}^T$, F and T' are the number of frequency and time bins, respectively.
It is worth mentioning that the stride factor of the convolutional operation has a significant impact on the memory requirements, speed, and performance of the model.

The Decoder Block: The decoder block utilizes a transposed convolutional layer with the same parameters as the encoder's convolution, to obtain the separated sources in time domain [4]. To do so, the decoder computes the element-wise product of the encoded representation of the input h and the corresponding mask m_k for source k, i.e.

$$\hat{s}_k = \text{Conv1DTranspose}(m_k \odot h)), \tag{2.5}$$

where $\odot$ denotes the element-wise product, m_k the mask obtained by the masking network and $\hat{s}_k \in \mathbb{R}^T$ is the k^{th} separated speech utterance.

The Masking Network: The masking network consists of normalization and linear layers, chunking, two Transformers inside a dual-path processing block (the SepFormer block), Parametric Rectified Linear Unit (PReLU) [34] and linear layers, an overlap-add scheme [28], two FFW layers and a final Rectified Linear Unit (ReLU) layer [35]. Figure 2.11 shows the overall architecture of the masking net. The masking net receives the encoded representations of the

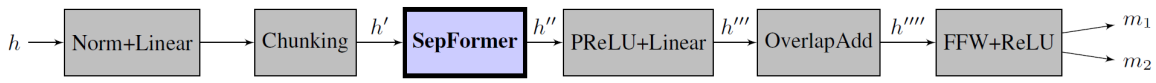


Figure 2.11: Architecture of the masking net. Image courtesy of [4].

input $h \in \mathbb{R}^{F \times T'}$ to predict a mask m_k for each of the N_s speakers present in the given mixture. To do so the input h is normalized via layer normalization as introduced by Ba et al. [30] and fed into a linear layer with a dimensionality matching the number of filters F defined in the encoder section of the model. Secondly, chunks of size C are created by segmenting h along the time axis, with an overlap factor of 50%, resulting in the output denoted as $h' \in \mathbb{R}^{F \times T' \times C \times N_C}$, where

N_C is the amount of chunks and C is the length of an individual chunk. Next, h' is fed into the dual-path processing block shown in Figure 2.12. This block learns short and long-term dependencies in the data. The output of the dual-path processing block is denoted as $h'' \in \mathbb{R}^{F \times C \times N_C}$. h'' then undergoes processing through PReLU activations which are subsequently followed by linear layer to produce output $h''' \in \mathbb{R}^{(F \times N_s) \times N_C \times C}$, where N_s represents the number of speakers present in the input mixture. After completing this step, the same overlap-add scheme as used in dual-path RNNs [28] is applied to obtain $h'''' \in \mathbb{R}^{F \times N_s \times T'}$. In the final stage, h'''' is processed by two FFW layers and a ReLU function to ultimately acquire the masks m_k.

The SepFormer Block: Figure 2.12 gives an overview of the architecture of the SepFormer block. The IntraTransformer and InterTransformer blocks are full self-attention Transformer encoders as introduced by Vaswani et al. [29]. The structure of both Transformer Encoder

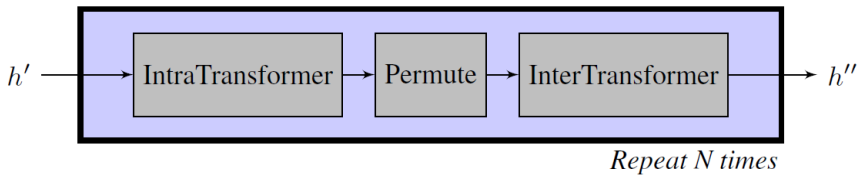


Figure 2.12: Architecture of the SepFormer block. Image courtesy of [4].

blocks is shown in Figure 2.13 and operates as follows. Initially, sinusoidal positional encoding denoted e is added to the input z, where,

$$z' = z + e. \tag{2.6}$$

The sinusoidal positional encoding [29] is defined as

$$PE_{pos,2i} = sin(pos/10000^{2i/d_{model}}), \tag{2.7}$$

where i is the dimension, pos is the position and d_{model} is the output dimension. The addition of positional encoding introduces information about the order of elements in a sequence, contributing to an improved ability to discern and separate individual elements, ultimately enhancing the overall performance. Next, K Transformer layers are applied. To do so, z' is fed into layer

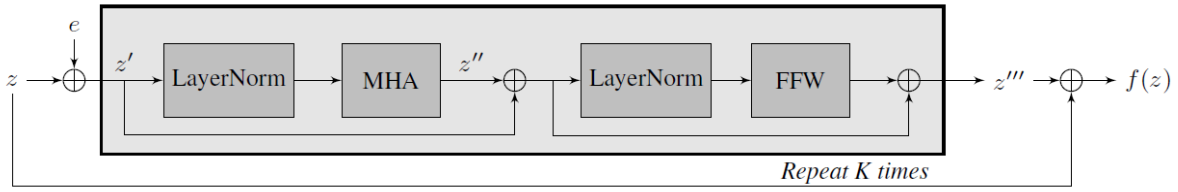


Figure 2.13: IntraTransformer and InterTransformer Architecture. Image courtesy of [4].

normalization and multi-head attention as described in Section 2.1.4:

$$z'' = \text{MultiHeadAttention}(\text{LayerNorm}(z')). \tag{2.8}$$

Afterward, another layer of normalization is utilized and the output ultimately fed into a FFW network that is applied independently to each position present:

$$z''' = \text{FeedForward}(\text{LayerNorm}(z'' + z')) + z'' + z'. \tag{2.9}$$

This results in an overall function of the Transformer block that can be defined as:

$$f(z) = g^K(z + e) + z, \tag{2.10}$$

where $g^K(\cdot)$ denotes the K transformer layers. The SepFormer used in this work employs 8 transformer layers in both the Intra- and InterTransformers, chunks of size 250 with 50% overlap. Eight attention heads and a positional FFW network with a dimension of 1024 are present in each Transformer layer. The entire dual-path pipeline is duplicated twice. It is worth mentioning that residual connections between Transformer layers are added across the entire architecture to improve backpropagation of gradients and avoid vanishing gradients [36].

Figure 2.14 shows the internal processing pipeline of the dual path SepFormer block. The dual path nature of this block, inspired by dual path RNNs [28], allows the SepFormer the modelling of long and short term dependencies. The latent representation of the input denoted as $\overline{h} \in \mathbb{R}^{F \times T'}$ (equivalent to h' of the previous section), is subdivided into overlapping chunks along the time axis. The chunks are subsequently concatenated and fed into the IntraTransformer encoder block which attempts to model the interactions in between all elements present in a chunk separately, effectively bounding the attention memory. In the next stage, the InterTransformer encoder is applied, modelling the interactions between elements which occupy the same position per chunk. The IntraTransformer can be interpreted as a block diagonal attention structure, whereas the InterTransformer acts as a stride attention structure [32], i.e. the output h_i

$$h_i = f_{inter}(f_{intra}(\overline{h}_i)). \tag{2.11}$$

SepFormer is a core component of the model explored in this work.

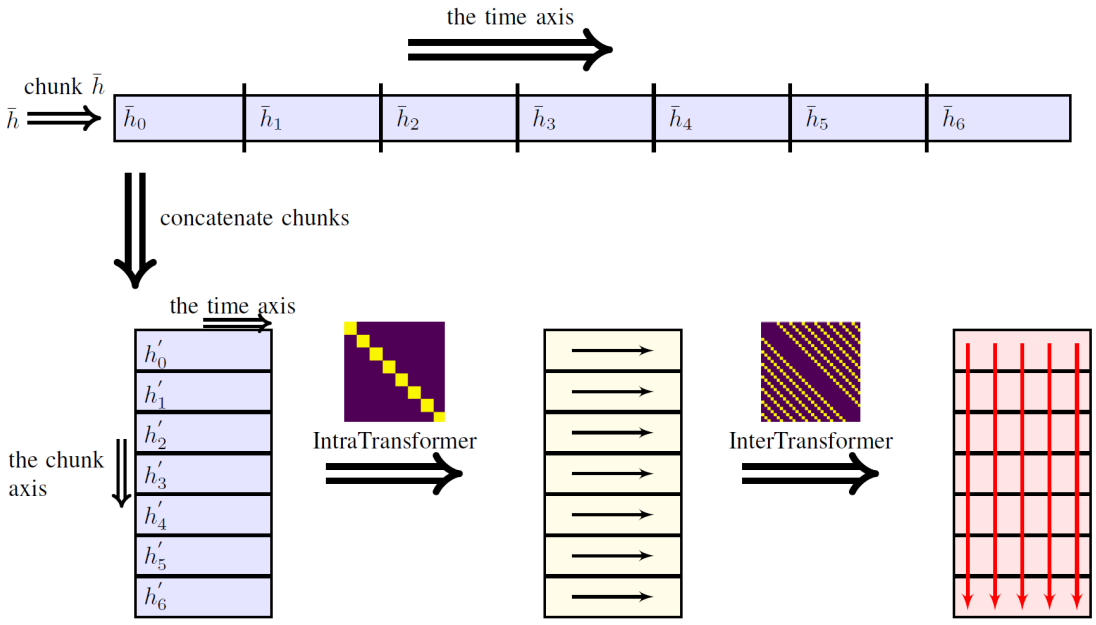


Figure 2.14: Architecture of the dual-path processing pipeline. Image courtesy of [37].

2.1.6 SuDoRm-Rf

Another model used as part of this work is SuDoRM-RF introduced by Tzinis et al. [10]. SuDoRM-RF limits memory usage, number of parameters, latency and number of floating point operations by employing a convolutional structure based on successive downsampling and re-

sampling of multi-resolution features performed by simple one dimensional convolutions. Figure 2.15 shows the architecture employed by SuDoRM-RF. Similar to Conv-TasNet and dual path RNNs [25], [28], SuDoRM-RF relies on a masking based approach using adaptive encoders and decoders:

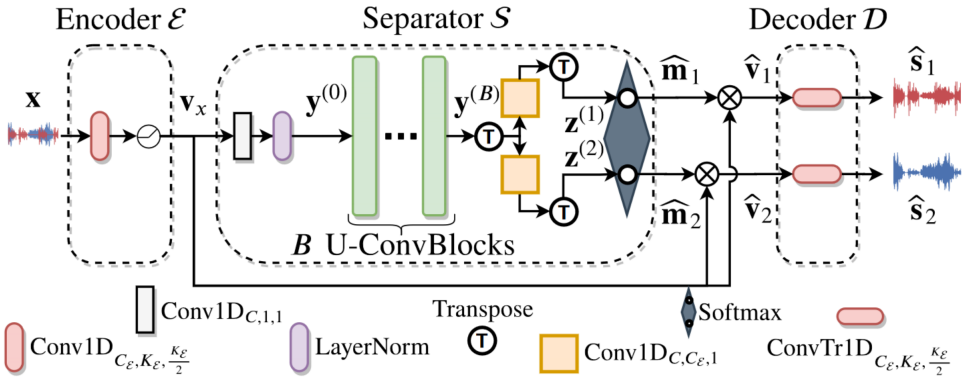


Figure 2.15: The overall architecture of SuDoRM-RF. Image courtesy of [10].

The Encoder Block: The Encoder block consists of a one dimensional convolution similar to Conv-TasNet, i.e.

$$v_x = \text{ReLU}\left(\text{Conv1D}_{C_\varepsilon,K_\varepsilon,\frac{K_\varepsilon}{2}}(x)\right) \in \mathbb{R}^{C_\varepsilon \times L}, \tag{2.12}$$

where C_ε represents the number of output channels in the convolution, K_ε the amount of samples present in the input mixture and $\frac{K_\varepsilon}{2}$ the amount of stride.

The Decoder Block: The Decoder block transforms the masked representation of separate audio utterances by applying the transposed convolution utilized in the Encoder stage:

$$\hat{s}_i = \text{Conv1DTranspose}_{C_\varepsilon,K_\varepsilon,\frac{K_\varepsilon}{2}}(\hat{v}_i). \tag{2.13}$$

The masking network: The masking network S is the main differentiator of the SuDoRM-RF model and operates as follows: Initially, the encoder output is projected to a new channel space via layer normalization and point-wise convolution Conv1D$_{C,1,1}$, i.e.:

$$y_0 = \text{Conv1D}_{C,1,1}\left(\text{LayerNorm}(v_x)\right). \tag{2.14}$$

Next, B layers of U-convolutional blocks are applied. The structure of the U-convolutional block is shown in Figure 2.16. The U-ConvBlock features temporal downsampling to extract information at multiple resolutions, The receptive field of the network is increased in each sampling operation. This block utilizes Q successive temporal down and upsampling operations resembling a U-Net [38] to extract information from the source at multiple resolutions, leaving the temporal resolution intact. [11].

amsmath mathtools

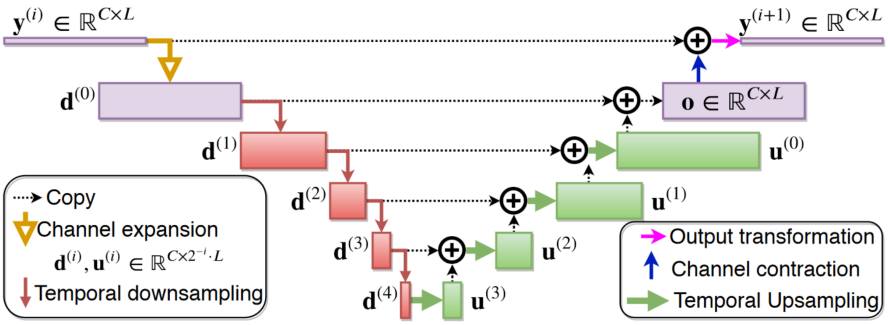


Figure 2.16: Architecture of a U-ConvBlock. Image courtesy of [10].

2.2 Diffusion in Speech Separation

2.2.1 Diffusion

Diffusion probabilistic models are generative machine learning models introduced by Sohl-Dickstein et al. [39] based on non equilibrium statistical physics. These diffusion models can beat state-of-the-art Generative Adverserial Networks (GANs) on Image Synthesis tasks [40] and have gained massive popularity in recent years following the release of OpenAI's DALL-E 2 and 3 models [41], [42].

In diffusion models, the structure of a data distribution is perturbed step by step by adding Gaussian noise via an iterative process. Afterward, the model learns to revert this process to recover the data from the noise. The trained model can then be utilized to generate data from passing noise samples through the reverse diffusion process. The conversion from noise to data is achieved through a Markov chain that adds noise gradually, whilst transitions from state to state are learned to reverse the diffusion [9], [43]. This process is depicted in Figure 2.17. The

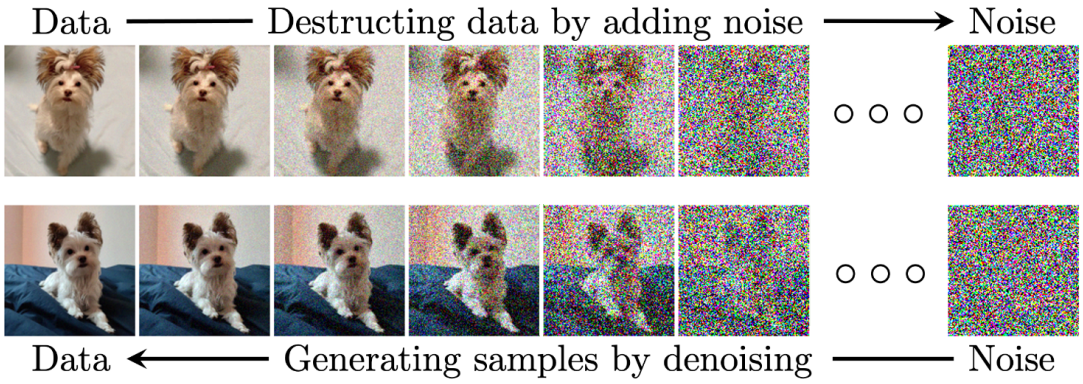


Figure 2.17: Diffusion and reverse diffusion process. Image modified from [44].

diffusion process [43] can be written as:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) \coloneqq \mathcal{N}(\mathbf{x}_t;\ \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}), \tag{2.15}$$

where $\beta_1, \ldots, \beta_T$ describes a variance schedule controlling the Gaussian noise added by the process. The reverse diffusion process is defined as:

$$p_\theta(\mathbf{x}_{0:T}) \coloneqq p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \qquad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \coloneqq \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \mathbf{\Sigma}_\theta(\mathbf{x}_t, t)), \tag{2.16}$$

where the joint distribution $p_\theta(\mathbf{x}_{0:T})$ represents the reverse process with learned Gaussian transitions beginning at $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$ [43], [44].

Diffusion models are trained by obtaining the Markov transitions which maximize the likelihood of the training data, i.e., minimizing the variational upper bound [45] on the negative log likelihood:

$$\mathbb{E}\left[-\log(p)_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_q\left[-\log\left(\frac{p_\theta(\mathbf{x}_0, T)}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right)\right] = \mathbb{E}_q\left[-\log(p)(\mathbf{x}_T) - \sum_{t\geq 1}\log\left(\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_t - 1)}\right)\right] =: L. \tag{2.17}$$

2.2.2 DiffWave

DiffWave [9] is a diffusion probabilistic model to generate waveforms and is a major component of the model explored in this work. Figure 2.19 depicts the diffusion and reverse diffusion process utilizing a Markov chain. The model supports conditional and unconditional generation of waveforms, whereas only the first is used as part of this work. Acting as a neural vocoder, DiffWave synthesizes waveforms conditioned on mel spectrograms [46]. DiffWave generates audio samples from Gaussian noise via iterative refinement. The speech generation can be controlled by providing mel spectrograms which are incorporated into each residual block of the model architecture. To ensure the compatibility at every layer, the mel spectrogram has to undergo upsampling via 2D convolutions to match the dimensionality of the respective residual layer. The sampling and training algorithms can be summarized as:

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$\nabla_\theta \left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Figure 2.18: Diffusion algorithms. Image courtesy of [43].

The training algorithm repeatedly samples a datapoint from a distribution, a time step uniformly from a set $1, \dots, T$ and a noise vector ϵ from a normal distribution. Then a gradient descent step that minimizes the expected value is performed. The process is repeated until convergence. The sampling algorithm is a generative process that starts by sampling a point from a normal distribution and iteratively applies a transformation using another noise vector, where $\alpha_t := 1 - \beta_t$, $\bar{\alpha}_t := \prod_{s=1}^{t}(1 - \beta_s)$, σ_t is a positive scaling factor that controls the amount of noise added to the system at time t and ϵ_θ is a WaveNet [47] model that predicts the input noise ϵ from x_t. This iterative process continues until x_0 is obtained, which is the output of the algorithm.

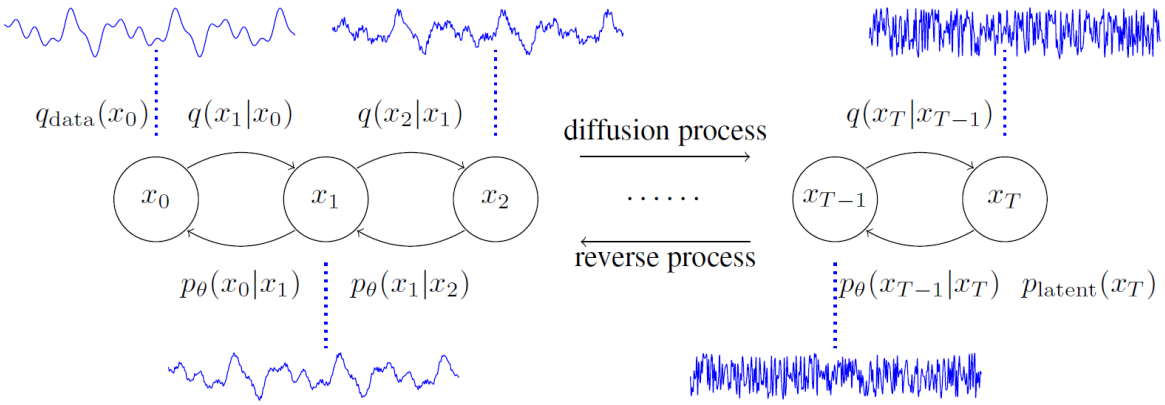


Figure 2.19: The processes employed by DiffWave. Image courtesy of [9].

2.3 Separate and Diffuse

The Separate and Diffuse model proposed by Lutati et al. [8] serves as the foundation for this study, enabling the exploration of advanced techniques for speech separation and surpassing the previously established performance bounds by utilizing generative diffusion models as part of the ML pipeline.

2.3.1 Architecture

The model comprises three key components as show in Figure 2.22:

i A deterministic source separator network (B)

ii A generative diffusion model (GM)

iii An alignment network (F)

The separator network B, serving as a pretrained model, can either be a transformer-based model such as a SepFormer [4], MossFormer [5] or a gated LSTM model [7].
The deterministic separator network B is given an input mixture m to generate a deterministic estimation $\overline{v}_d$ of the individual speech utterances. Afterward, the mel-spectrogram is computed from all estimations $\overline{v}_d$. A mel-spectrogram is a spectrogram using the mel scale transformation. The mel scale aims to model human frequency perception to facilitate consistency with the human ear's response to different ranges of frequencies [48].

The mel scale can be computed as described by O'Shaughnessy [49]:

$$m = 2595 \log\left(1 + \frac{f}{700}\right). \tag{2.18}$$

The parameters used to compute the spectrogram are explored in more detail in Section 3.2.

The computed mel-spectrograms are used to obtain a noisy estimation $\overline{v}_g$ via the generative vocoder network. Since the diffusion process does not guarantee the same number of samples in the output as was given as input, padding is very often necessary to ensure compatible dimensions. Any neural vocoder that achieves an acceptable error range is suitable for the diffusion model. In this particular case DiffWave, introduced by Kong et al. [9] was utilized. To generate speech signals, DiffWave was trained on the corresponding datasets.

Both estimations $\overline{v}_d$ and $\overline{v}_g$ are transformed into the frequency domain $\overline{V}_d$ and $\overline{V}_g$ respectively. The resulting spectrograms are concatenated and fed into the alignment network F. The

alignment network F is implemented using two 6-layer CNNs with residual connections as shown in Figure 2.21 [31].

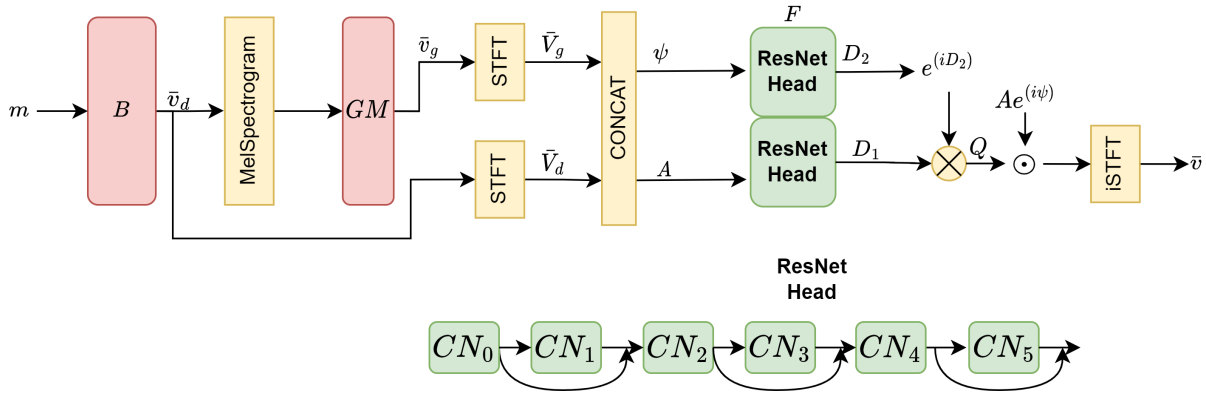


Figure 2.20: The model architecture. Image courtesy of [8].

The magnitude of the concatenation of $\overline{V}_d$ and $\overline{V}_g$, A (a two channel tensor), is used as the input to the first ResNet head leading to a complex factor D_1. Obtaining the second factor D_2 is slightly more complex. The phase of $\overline{V}_d$ is concatenated with the phase of the element-wise product of $\overline{V}_g$ and the complex conjugate of $\overline{V}_d$ resulting in a 2-channel tensor ψ. The 2-channels of the tensors represent the deterministic and generative estimations, respectively. For both components it is paramount to ensure the phase lies in the interval $[0, \pi]$. ψ is fed into the second ResNet resulting in the complex factor D_2. Figure 2.21 displays the structure of the ResNet heads in more detail. The left hand side shows the overall architecture whereas the right side gives a more detailed view of the individual residual network blocks. It needs to be guaranteed that D_2 lies within the interval $[-\pi, \pi]$. Both factors D_1 and D_2 have a dimensionality of $2 \times S$ where the two channels are corresponding to the deterministic estimation v_d and the diffused estimation v_g. S denotes the dimensions of the spectrograms $\overline{V}_d$ and $\overline{V}_g$. To calculate the weight factor α, i.e., the weight for deterministic estimation, the respective channels of D_1 and D_2 are combined into a complex number:

$$\alpha = D_{1,0} \cdot (\cos(D_{2,0}) + i \sin(D_{2,0}))\,, \tag{2.19}$$

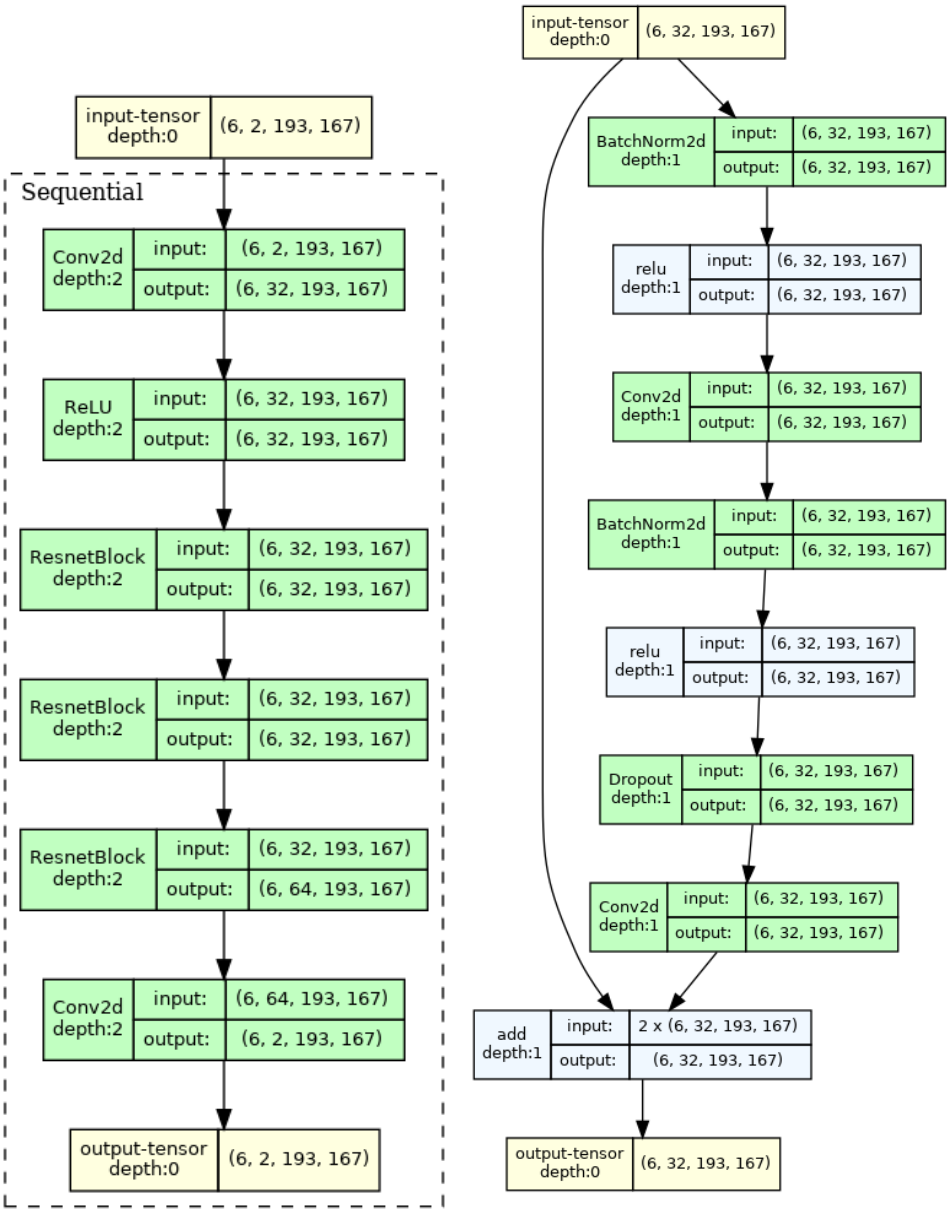
whereas β, the weight for the generative estimation, is defined as:

$$\alpha = D_{1,1} \cdot (\cos(D_{2,1}) + i \sin(D_{2,1}))\,. \tag{2.20}$$

Finally, the separated speech signal $\overline{V}$ is computed as weighted sum between the spectral representation of the separator network B and the diffusion model GM, i.e.:

$$\overline{V} = \alpha \odot \overline{V}_d + \beta \odot \overline{V}_g. \tag{2.21}$$

where $\odot$ denotes the element-wise product. Ultimately, the time-domain signal representing the separated speakers is derived through the application of the inverse short-time Fourier transform.



(a) ResNet head

(b) The residual block used in the ResNet head

Figure 2.21: The architecture of the ResNet head. Visualized using torchview [50].

2.4 Ensemble Model

Due to the complexity introduced by the diffusion model in Section 2.3, an approach is presented which avoids the need for diffusion probabilistic models in the ML pipeline. To do so, the model is set up as a stacked ensemble model [51] by replacing the diffusion model with a deterministic separator model. The architecture of this ensemble model is shown in Figure 2.22. Hereby, the input mixture m is fed into two separator models, $B1$ and $B2$, in parallel. $B1$ is the SepFormer model, whereas $B2$ is implemented as a SuDoRM-RF model, as described in Section 2.1.6. The specific implementation was adapted from [52]. The estimations of both models are transformed into the frequency domain and fed into the same alignment network as depicted in Figure 2.21.

Any separator model with decent errors on the desired task or dataset can be used in place of the SepFormer and SuDoRM-RF, however both are chosen due to their performance and availability of pretrained weights.

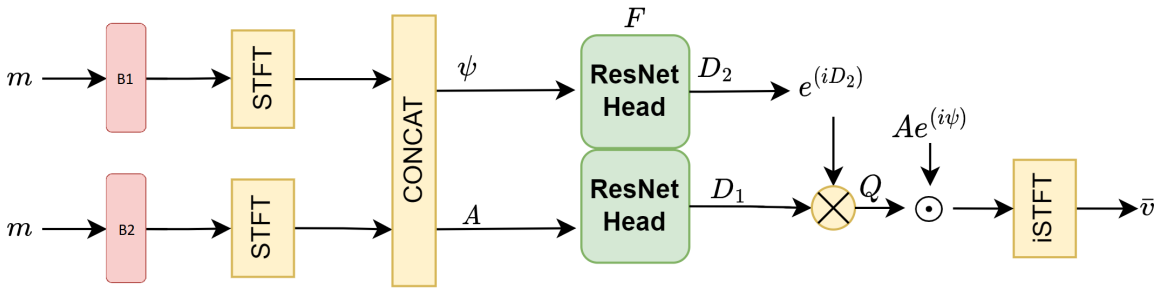


Figure 2.22: The ensemble model architecture. Image modified from [8].

Several different alignment architectures were implemented including complex convolutional neural networks [53], direct weight estimation and time domain based models. Finally, the model displayed in Figure 2.21 showed the best performance to complexity tradeoff and was evaluated in the upcoming chapters.

3 Experimental Setup

This chapter introduces the ML framework, the datasets and the evaluation metrics used and discusses the training setup. First, we present the datasets that were utilized and the preprocessing techniques applied to them. Subsequently, the asteroid framework and setup to train the models is introduced. Finally, the evaluation metrics that were used to assess the separation capabilities of the models are discussed.

3.1 Datasets

Following the exploration of the model's architecture, we shift our focus to the datasets used in this work. This subchapter introduces the sources of the data and provides a comprehensive overview of the preprocessing techniques applied. We discuss how the raw data was transformed, cleaned, and prepared to ensure its compatibility with the model.

3.1.1 WSJ0

The Wall Street Journal (WSJ)0 dataset [54] consists of three main subsets:
SI-64 comprises read speech utterances from the WSJ including 84 speakers. This is considered the training set. It contains approximately 20 hours of speech.
Further two additional datasets (Dev-93 and Eval-92) are used for evaluation and validation. Dev-93 contains around 5 hours of speech, whereas Eval-92 contains around 20 hours.

The dataset serves as the training data for the diffusion models, i.e. DiffWave in Section 2.2.2 used in this work.
To prepare the data, the original raw wavpack1 (wv1) files undergo a conversion process, transforming them into WAV files. Subsequently, mel spectrograms are computed from the converted WAV files. These computed mel spectrograms are then paired with their corresponding WAV files, creating WAV/spectrogram pairs. These pairs form the training set for the diffusion model. By utilizing this data preparation pipeline, the diffusion models can learn from the WAV/spectrogram pairs to effectively model the underlying probability distribution of the audio data. This enables the DiffWave model to generate realistic audio samples used in the Separate and Diffuse model.

3.1.2 WSJ0-mix

The WSJ-Mix dataset is an expansion of the original WSJ0 dataset, introduced by Hershey et al. [12]. This enhanced dataset was created by combining utterances from two or three randomly chosen speakers, resulting in WSJ0-2Mix and WSJ0-3Mix datasets, respectively. The mixing process was performed at various Signal-to-Noise ratios (SNRs) ranging from 0 dB to 5 dB. In total, four different configurations of the dataset are possible:

1. Min mode: The mixture gets trimmed to the length of the shortest utterance present during generation of the mixture.
2. Max mode: The mixture matches the length of the longest utterance present.
3. 8k Sampling Rate
4. 16k Sampling Rate

In this work, the dataset was generated using min mode and 8000 Hertz sampling rate, resulting in a total of 20000, 5000 and 3000 mixtures for training, validation and evaluation datasets respectively. The addition of WSJ0-2Mix and WSJ0-3Mix datasets provided a larger and more diverse set of audio mixtures, making it viable for the model in question.
Mixtures shorter than 32000 samples are dropped from the training, validation and evaluation dataset.

3.1.3 WHAM!

The WSJ0 Hipster Ambient Mixtures (WHAM!) dataset, introduced by Wichern et al. [55], extends the WSJ0-2Mix dataset to provide for more realistic and challenging separation scenarios. The dataset consists of two speaker mixtures taken from the WSJ0-2Mix, overlapped with real world noise samples. These samples include recordings of ambience of restaurants, bars and coffee shops in the San Francisco Bay Area. High SNR speech which would be intelligible was removed from the noise samples to not interfere with the separation process. This set of noisy mixtures can be used to test the robustness of separation models to noise. [55] To create the dataset, every mixture in the WSJ0-2Mix dataset is assigned a randomly sampled noise excerpt. Then, a random SNR value between -6 and +3 decibels (dB) is sampled from a uniform distribution. Finally, gain is applied to the first speakers such that the SNR between the first speaker and the noise sample matches the randomly sampled SNR value.

3.1.4 WHAMR!

WSJ0 Hipster Ambient Mixtures with Reverb (WHAMR!) is a dataset that extends WHAM! by introducing synthetic reverberations to the speaker mixtures in addition to the ambience added by WHAM! [56]. The reverberations are generated and convolved by using pyroomacoustics [57] room impulse response generator. Parameters used to generate the room impulse responses are shown in Figure 3.1. T_{60} denotes the reverberation time in seconds. All other parameters are given in meters except angles denoted by Θ are in radians. For each utterance in the training

Room	L	$\mathcal{U}(5, 10)$
	W	$\mathcal{U}(5, 10)$
	H	$\mathcal{U}(3, 4)$
T60	high	$\mathcal{U}(0.4, 1.0)$
	med.	$\mathcal{U}(0.2, 0.6)$
	low	$\mathcal{U}(0.1, 0.3)$

Mic. Center	L	$\frac{L_{\text{Room}}}{2} + \mathcal{U}(-0.2, 0.2)$
	W	$\frac{W_{\text{Room}}}{2} + \mathcal{U}(-0.2, 0.2)$
	H	$\mathcal{U}(0.9, 1.8)$
Mic. Array	sep.	noise mic. separation
	θ	$\mathcal{U}(0, 2\pi)$
Sources	H	$\mathcal{U}(0.9, 1.8)$
	dist.	$\mathcal{U}(0.66, 2)$
	θ	$\mathcal{U}(0, 2\pi)$

Figure 3.1: Parameters of the sampling distributions for room impulse response generation. Table courtesy of [56].

(tr), validation (cv), and testing (tt) set folders, the following wav files are obtained [58]:

1. `noise`: contains the isolated background noise from WHAM!.
2. `s1_anechoic`: isolated data from speaker 1 without reverb, but with appropriate delays to align with `s1_reverb`.
3. `s2_anechoic`: isolated data from speaker 2 without reverb, but with appropriate delays to align with `s2_reverb`.
4. `s1_reverb`: isolated data from speaker 1 with reverberation.
5. `s2_reverb`: isolated data from speaker 2 with reverberation.
6. `mix_single_anechoic`: for speech enhancement, contains mixture of `s1_anechoic` and `noise`.
7. `mix_clean_anechoic`: clean speech separation for two speakers, contains mixture of `s1_anechoic` and `s2_anechoic`. The relative levels between speakers should match the original WSJ0-2Mix dataset, but the overall level of the mix will be different.
8. `mix_both_anechoic`: contains mixtures of `s1_anechoic`, `s2_anechoic`, and `noise`.
9. `mix_single_reverb`: for speech enhancement, contains mixture of `s1_reverb` and `noise`.
10. `mix_clean_reverb`: clean speech separation for two reverberant speakers, contains a mixture of `s1_reverb` and `s2_reverb`. The relative levels between speakers should match the original WSJ0-2Mix dataset, but the overall level of the mix will be different.
11. `mix_both_reverb`: contains mixtures of `s1_reverb`, `s2_reverb`, and `noise`.

In our case only the noisy and reverberant task is of use, i.e., during training, validation and testing only data from `mix_both_reverb` is considered.

3.1.5 LibriSpeech

LibriSpeech is a de facto rival corpus of read English speech to the WSJ0 dataset, derived from public domain audiobooks that are part of the LibriVox project [59]. It contains 1000 hours of speech utterances sampled at 16000 Hertz split into several subsets shown in Table 3.1

Subset	**Hours**	**Min/speaker**	**Female Speakers**	**Male Speakers**	**Total Speakers**
dev-clean	5.4	8	20	20	40
test-clean	5.4	8	20	20	40
dev-other	5.3	10	16	17	33
test-other	5.1	10	17	16	33
train-clean-100	100.6	25	125	126	251
train-clean-360	363.6	25	439	482	921
train-other-500	496.7	30	564	602	1166

Table 3.1: Subset hours per speaker in LibriSpeech dataset.

To train the models in this work, the train-360 subset containing roughly 360 hours of speech is used. `dev-clean` provides 5.4 hours of speech for validation purposes and `test-clean` serves as the basis for the evaluation dataset.

3.1.6 LibriMix

The LibriMix dataset offers an open source alternative to the WSJ0-2/3Mix and its noisy derivative, WHAM! [60]. As depicted in Table 3.2, LibriMix provides vastly more utterances and a greater variety of speakers in the individual sub datasets, addressing the generalization issues present when training models on WSJ0-Mix datasets and reducing the generalization error overall [60]. LibriMix offers a diverse training set with approximately 1,000 distinct speakers, a significant leap compared to the 101 speakers in wsj0. Furthermore, LibriMix outshines WSJ0 not only in speaker diversity but also in lexical richness. With a staggering 600,000 unique words, LibriMix surpasses WSJ0-Mix, which comprises only 5,000 words.

Dataset	Split	# Utterances	Hours
WSJ0-2/3mix	train	20,000	30
	dev	5,000	8
	test	3,000	5
Libri2Mix	train-360	50,800	212
	train-100	13,900	58
	dev	3,000	11
	test	3,000	11
Libri3Mix	train-360	33,900	146
	train-100	9,300	40
	dev	3,000	11
	test	3,000	11

Table 3.2: Statistics of derived speech separation datasets.

To generate the utterance mixtures, LibriSpeech samples are augmented using a speed-perturbation technique introduced by Ko et. al [61]. In the frequency domain, this warping corresponds approximately to a shift of the spectrum to a mel spectrogram [62]. In our scenario, a perturbation factor α of 0.8 and 1.2 was applied. As opposed to WSJ0-Mix, scaling individual samples does not utilize signal power but instead relies on Loudness Units relative to Full Scale (LUFS) [63]. LUFS measures the perceived loudness of an audio sample and correlates more closely with human perception than ordinary SNRs. The mixtures are then generated from randomly selected, speed augmented raw samples which loudness of each utterance sampled uniformly between -25 and -30 LUFS. Further, random noise samples with loudness between -25 and -33 LUFS are added for the WHAM! equivalent of the dataset. For the train dataset each utterance is used only once, whereas for evaluation and test the procedure is repeated a couple of times to generate 3000 data samples.

3.2 Training Setup

The effectiveness and performance of the model heavily depends on the training setup employed. In this subchapter, we detail the specific configuration used to train the model. This includes the choice of optimization algorithms, hyperparameter tuning, training data partitioning, and any additional training considerations. By elucidating the training setup, we provide a clear understanding of how the model was trained to achieve optimal performance.

3.2.1 The Asteroid Framework

The Separate and Diffuse model was implemented within the asteroid framework, a PyTorch-based audio source separation toolkit for researchers [64]. Using Asteroid we can implement the entire ML pipeline from data preparation to evaluation in a single so-called recipe as shown in Figure 3.2 which is easily configurable via the command line. From a single entry point, the `run.sh` file, the following stages can be executed:

- **Stage 1**: Generate mixtures with the official scripts, optionally perform data augmentation.
- **Stage 2**: Gather data information into json files expected by the corresponding `DataLoader`.
- **Stage 3**: Train the model.
- **Stage 4**: Separate test mixtures and evaluate the model.

Figure 3.2: The stages of the recipe. Image courtesy of [64].

3.2.2 DiffWave

The diffusion model GM in the Separate & Diffuse model in Section 2.3 and architecture was taken from [9] and implemented using [65]. To better fit the needs of our model, some modifications to the original architecture were performed. First and foremost, the number of frequency bins used in the creation of the mel spectrogram was changed to 64. Further, the number of samples in each STFT window was adapted to 384. Finally, the hop size, i.e., the amount of samples between the start of consecutive STFT windows, was set to 192. Reducing this value from the original 256 leads to a higher temporal resolution at the potential cost of larger feature dimensionality.

These changes directly influence the model topology. To meet the requirements of upsampling data sample spectrums during the model's forward pass according to the specified parameters, the `SpectrogramUpsampler` PyTorch module underwent adaptations (see Listing 3.1), wherein the convolutional transpose layers' were fine-tuned, including adjustments to kernel size, stride, and padding, ensuring the output spectrums align with the desired parameters.

```
class SpectrogramUpsampler(nn.Module):
    """
    Upsample single-channel spectrograms using transposed convolutional layers.
    Returns:
        torch.Tensor: Upsampled spectrogram tensor.
    Architecture:
        - Two transposed convolutional layers:
            - First layer: kernel size [3, 32], stride [1, 16], padding [1, 8]
            - Second layer: kernel size [3, 32], stride [1, 12], padding [1, 10]
        - Leaky ReLU activation (slope = 0.4) applied after each convolutional layer.
    Input:
        torch.Tensor: Input spectrogram tensor with shape [batch_size, time_frames, frequency_bins].
    Output:
        torch.Tensor: Upsampled spectrogram tensor with the same shape as the input.
    Note:
        Assumes single-channel (1) input and output spectrograms.
    """
```

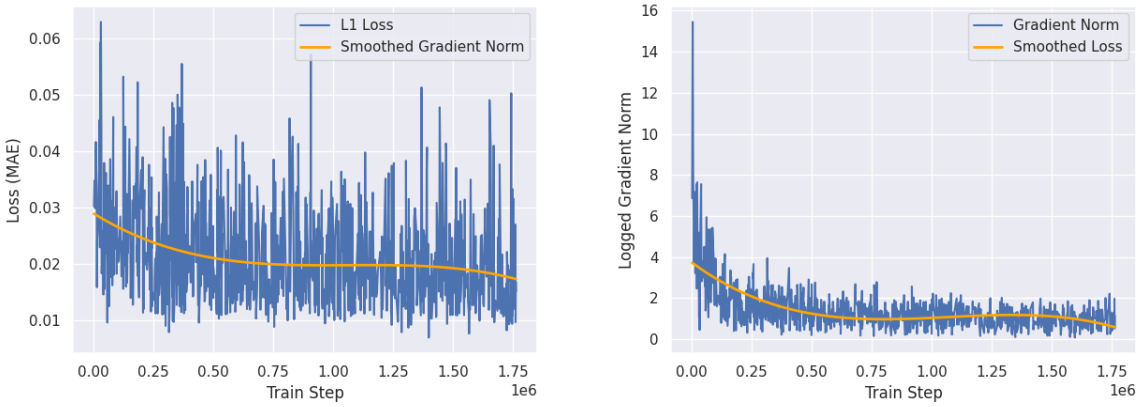
```
18
19      def __init__(self):
20          super().__init__()
21          self.conv1 = ConvTranspose2d(1, 1, [3, 32], stride=[1, 16], padding=[1, 8])
22          self.conv2 = ConvTranspose2d(1, 1, [3, 32], stride=[1, 12], padding=[1, 10])
23
24      def forward(self, x):
25          x = torch.unsqueeze(x, 1)
26          x = self.conv1(x)
27          x = F.leaky_relu(x, 0.4)
28          x = self.conv2(x)
29          x = F.leaky_relu(x, 0.4)
30          x = torch.squeeze(x, 1)
31          return x
```

Listing 3.1: The adapted upsampler code.

The model was trained on the raw WSJ0 and LibriSpeech datasets described in Section 3.1.1 and Section 3.1.5 using Adam [66] with a batch size of 16 and learning rate of $2e-4$. Throughout the training process, the model utilized the L1 loss function, that is Mean Absolute Error (MAE) to calculate the discrepancy between predicted and target values [67]. MAE is defined as:

$$\text{MAE} = \frac{1}{n_{samples}} \sum_{i=1}^{n_{samples}} |y_i - \hat{y}_i| \tag{3.1}$$

where y_i and $\hat{y}_i$ are the targets and predicted targets, respectively. We use a total of 1750000 and 1500000 steps respectively to optimize the model's performance on the given dataset.

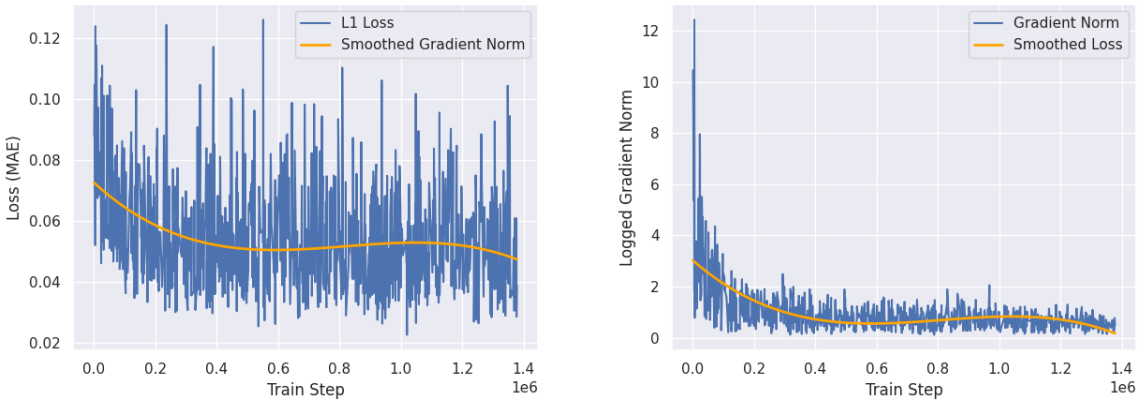


(a) L1 loss per Step

(b) The gradient norm per Step

Figure 3.3: Loss and gradient norm per step. Subfigures showing the loss (left) and gradient norm (right) values per step during training on WSJ0.

Figure 3.3 and Figure 3.4 show the loss and the gradient norm over the duration of the training. The gradient norm refers to a scalar that represents the magnitude of the gradients with respect to the model's parameters during training. Due to the higher variety of speakers and overall increased complexity present in the LibriSpeech corpus, the loss achieved by the diffwave model is considerably worse at around 0.05 as opposed to 0.02 when training on WSJ0.



(a) L1 loss per Step

(b) The gradient norm per Step

Figure 3.4: Loss and gradient norm per step. Subfigures showing the loss (left) and gradient norm (right) values per step during training on LibriSpeech.

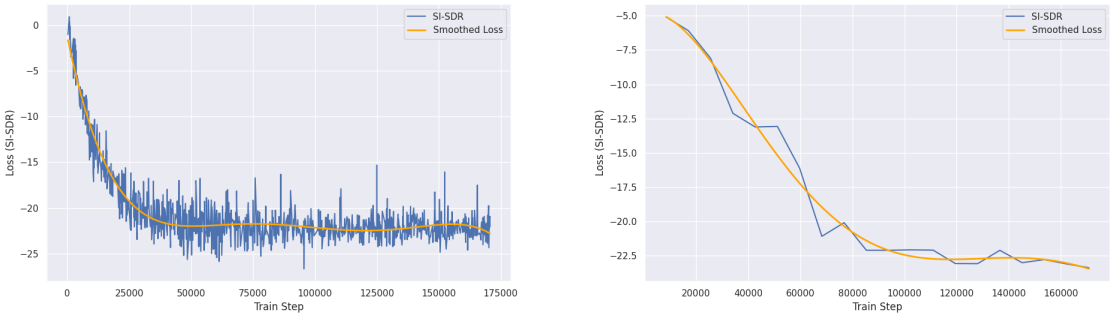
3.2.3 Alignment Network

For the deterministic separator network B, as described in Section 2.3.1, pretrained models were utilized from HuggingFace (implemented in speechbrain [68]). The pretrained separator B and the diffusion model GM are integrated as part of the Separate and Diffuse model in Asteroid. The actual training at this point only changes parameters of the alignment network F. For this purpose, several techniques were employed to improve the optimization of the network's weights.

To control the complexity of the model and avoid overfitting of the alignment model, regularization is necessary. We use early stopping, that is stopping the training process at the smallest error with respect to the validation data set, even if the train error continues to decrease [69]. In our case the patience of the early stopping callback is set to 15 epochs, i.e., if the validation loss does not improve over 15 epochs, training is halted. This leads to more generalized networks.

Further, a learning rate scheduler is implemented, which reduces the learning rate by a factor of 0.5 if no improvement is detected over a period of 5 epochs. For optimization, we employed the Adam algorithm with a learning rate of 0.001. Scale-Invariant Signal-to-Distortion ratio (SI-SDR) as described in Section 3.3.4 was utilized as the target loss function in a permutation invariant manner [22].

Figure 3.5 shows the train and validation loss values over the train steps and provides a smoothed approximation of both.



(a) Negative SI-SNR loss per step

(b) Validation loss per step

Figure 3.5: Train and validation loss per step. Subfigures showing the train loss (left) and validation loss (right) values per step during training.

3.3 Evaluation Metrics

To assess the performance of the model, it is essential to employ appropriate evaluation metrics. Here, we discuss the evaluation metrics utilized to measure the model's effectiveness in speech separation tasks. We explain the rationale behind the chosen metrics and highlight their relevance and significance in evaluating the model's performance.

3.3.1 SDR

Signal-to-Distortion ratio (SDR) is an objective evaluation metric commonly used in source separation tasks to measure the quality of separated audio signals. It assesses the separation performance by quantifying the ratio of the target signal power to the power of the distortion, which includes interference and artifacts present in the separated signal [70].

Given the estimated speech signal s_{target} and the error terms for interference, noise and artifacts e_{interf}, e_{noise} and e_{artif} respectively, the SDR is calculated as follows:

$$\text{SDR} = 10 \log \left(\frac{\|s_{\text{target}}\|^2}{\|e_{\text{interf}} + e_{\text{noise}} + e_{\text{artif}}\|^2} \right). \tag{3.2}$$

where $\|s_{\text{target}}\|^2$ is the power of the target speech signal in time domain, and $\|e_{\text{interf}} + e_{\text{noise}} + e_{\text{artif}}\|^2$ is the power of the combined distortion components (interference, noise and artifacts). To understand the process of obtaining these terms in detail, the reader is encouraged to refer to the original paper [71] as the computation involved is rather complex and goes beyond the scope of this work.

SDR is expressed in dB, and a higher SDR value indicates better separation performance, meaning that the target signal is more prominent compared to the distortion.

3.3.2 SIR

Similarly to Section 3.3.1, the Source-to-Interferences ratio (SIR) is a metric that quantifies the quality of the separation process by measuring the ratio of the power of the target source s_{target} to the power of the interference e_{interf} present in the separated signal [71].

The SIR is defined in a similar fashion to Equation (3.2):

$$\mathrm{SIR} = 10 \log \left(\frac{\|s_{\mathrm{target}}\|^2}{\|e_{\mathrm{interf}}\|^2} \right). \tag{3.3}$$

Higher SIR values generally indicate better separation and less interference. It can be interpreted as the amount of other sources which can be heard in the source estimation. This is also known as bleed or leakage in traditional sound recording situations [72], [73].

3.3.3 SAR

To complete the trifecta of evaluation metrics introduced in [71], the Source-to-Artifacts ratio (SAR) emerges as a crucial measure for assessing the efficacy of source separation models.
It is obtained as follows:

$$\mathrm{SAR} = 10 \log \left(\frac{\|s_{\mathrm{target}} + e_{\mathrm{interf}} + e_{\mathrm{noise}}\|^2}{\|e_{\mathrm{artif}}\|^2} \right). \tag{3.4}$$

Higher SAR values indicate better separation. The common interpretation of this metric entails measuring the extent of undesired artifacts present in a source estimate concerning the true source [74].

3.3.4 SI-SDR

SI-SDR is a modification of SDR resulting in a more robust measure [75]. Some arguments have been put forth suggesting that the methodology employed by the SDR in computing the error terms can potentially result in artificially inflated outcomes.
Figure 3.6 illustrates the disparity between SDR and SI-SDR. The ground truth is illustrated

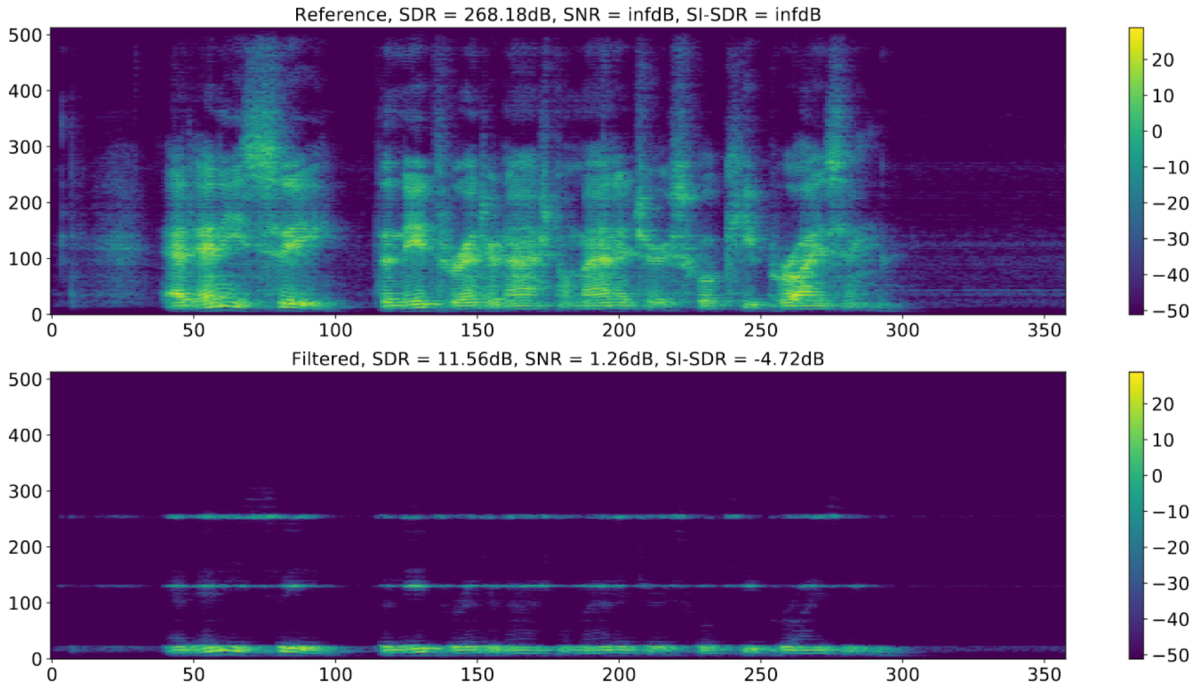


Figure 3.6: The top shows the spectrogram of the original signal. Even though the bottom spectrogram is degraded massively the SDR metric is very high. Image courtesy of [75].

by the top spectrogram. The metrics SDR and SI-SDR in the figure show, as is expected for the original ground truth signal, infinite values. Even though the bottom spectrogram is massively degraded, compared to the ground truth signal, (SNR and SI-SDR are fairly low) the SDR metric is quite high at 11.56 dB. To rectify this, SI-SDR attempts to remove the dependency of SDR on the amplitude scaling of the signal.

Given the estimated signal $\hat{s}$ and the reference clean signal s, the SI-SDR is calculated as follows:

$$\text{SI-SDR} = 10 \log\left(\frac{\|\alpha \cdot s\|^2}{\|\alpha \cdot s - \hat{s}\|^2}\right), \tag{3.5}$$

where α is the scaling factor that minimizes the mean squared error between the estimated signal and the reference signal:

$$\alpha = \underset{\alpha}{\text{argmin}} |\alpha s - \hat{s}|^2. \tag{3.6}$$

For a detailed explanation of the equation, the reader is referred to [75]. SI-SDR is expressed in dB, and a higher SI-SDR value again indicates better separation performance.

3.3.5 STOI

As opposed to metrics mentioned above, Short-time Objective Intelligibility measure (STOI) evaluates speech intelligibility at a short-time scale by analyzing local segments of speech. This makes it suitable for assessing the dynamic changes in intelligibility over time [76].

STOI operates by comparing the short-time magnitude spectra of the clean reference speech signal and the degraded speech signal [76]. It analyzes the similarity between these spectra to determine the level of speech intelligibility. The metric is designed to mimic human perception, making it valuable for evaluating the impact of various signal processing techniques on speech quality and intelligibility. By assessing speech intelligibility at a local level, STOI provides insights into the fluctuating quality of speech throughout an audio signal [77].

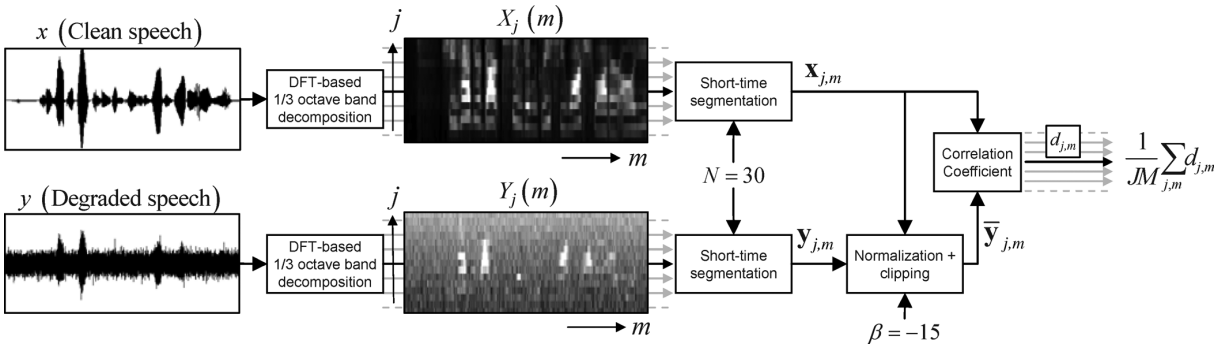


Figure 3.7: Illustration of the STOI calculation. Image courtesy of [76].

Figure 3.7 shows the process of obtaining the metric. Both the clean signal and the degraded signal, i.e., the separated estimation are initially decomposed into DFT [78] based on one-third octave bands. Afterward, the speech temporal envelopes of the degraded short-time segmented signal are normalized and clipped. Then, temporal segments of length 384ms of both signals are compared via the use of a correlation coefficient. Finally, the short-time intermediate intelligibility measures $d_{j,m}$ are averaged into a single scalar value [77].

STOI produces scores within the range of 0 to 1, where 1 represents perfect intelligibility and 0 represents complete loss of intelligibility.

3.3.6 PESQ

Perceptual Evaluation of Speech Quality (PESQ) [79] is a widely used objective metric in the field of speech and audio processing, designed to assess the perceived quality of speech signals by mimicking human auditory perception.

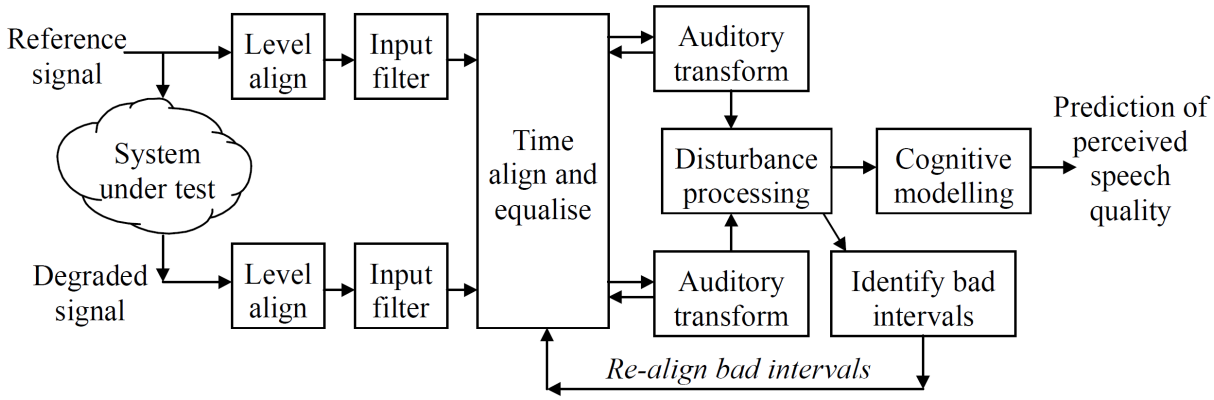


Figure 3.8: Illustration of the PESQ calculation. Image courtesy of [79].

Figure 3.8 illustrates the architecture of PESQ computation. The model initiates by aligning both signals to a standard listening level and then uses Fast Fourier Transform (FFT) with an input filter simulating a standard telephone handset. The signals are temporally aligned and processed through an auditory transform akin to Perceptual Speech Quality Measurement (PSQM)[80]. This transformation also includes equalization for linear filtering in the system and gain variation. From the disturbance (i.e., the difference between the transforms of the signals), two distortion parameters are derived, which are then combined in frequency and time and mapped to predict the subjective Mean Opinion Score (MOS) [79].

The typical range for PESQ is between -0.5 and 4.5 whereas a higher PESQ score indicates better perceived speech quality. A score close to 4.5 suggests little to no degradation of the original signal, while values close to 0 indicate that the perceived quality is similar to a severely distorted or noisy speech signal.

3.3.7 Variants

For ease of use and to facilitate straightforward comparisons, we do employ a set of improvement metrics denoted by the **_imp** postfix in Asteroid (ofter only a lower case 'i' is used to indicate the improvement instead). These metrics compare the quality of the separated sources with the original input mixture to quantify how much the algorithm has improved. An example for SDR is given as:

$$SDRi = SDR_{separated} - SDR_{input}. \tag{3.7}$$

By calculating the improvements in metrics we gain valuable insights into the performance enhancements of our speech separation system. These improvement scores directly highlight the effectiveness of the separation process, allowing us to gauge the system's ability to better discern individual models. This allows easy comparison between baseline models and tweaked models.

4 Results and Discussion

The experimental results of the research are presented and discussed. First, the baseline models are evaluated to give a common ground for comparison. Next, a suite of pertinent, de facto standard evaluation metrics and methodologies are employed to analyze to capabilities and the improvement of the Separate and Diffuse model. Afterward, we juxtapose the results against state-of-the-art baseline models, to compare the efficacy of the speech separation method. Then, the ensemble model is evaluated and analyzed. Finally, all utilized models are compared in terms of model size and inference time and a closing conclusion is drawn.
All results where obtained utilizing evaluation datasets containing 3000 data samples created directly from the datasets described in Section 3.1.2 and Section 3.1.6.

4.1 Results of the Baseline Models

In this subchapter, the utilized baseline models, i.e. SepFormer and SuDoRM-RF, are evaluated on several available benchmarks.

Table 4.1 shows the SDRi and SI-SDRi separation performance of SepFormer on several WSJ0 and LibriSpeech based separation tasks:

Benchmark	SI-SDRi	SDRi
WSJ0-2mix	22.3	22.5
WSJ0-3mix	19.6	19.9
libri2mix	20.2	20.5
libri3mix	18.2	18.6
WHAM!	16.4	16.7
WHAMR!	14.0	13.0

Table 4.1: The SepFormer performance obtained on several benchmarks.

All results except the WSJ0-3mix are congruent with the results presented in [4], [32], [37]. The WSJ0-3mix benchmark performed slightly better with respect to both metrics, however the improvement is almost negligible in magnitude.

As for SuDoRM-RF, the results for the available pretrained models are given in Table 4.2: The WSJ0-2mix benchmark exceeds the results presented in [10], [11] by about 0.6 dB whereas

Benchmark	SI-SDRi	SDRi
WSJ0-2mix	19.5	19.7
WHAMR!	13.5	13.6

Table 4.2: The SuDoRM-RF performance obtained on WSJ0-2Mix and WHAMR!.

WHAMR! results are not available in the respective publication. This improvement might

be caused by the quality of data/preprocessing used or the utilization of the entire dataset. Interestingly, the results reported on GitHub [52] align with the obtained results.

4.2 Separate and Diffuse

We analyze the evaluation metrics at the input of the model, i.e., the mixture against the clean signal and the estimated separation against the clean signal for the WSJ0-2Mix benchmark of 3000 data samples.

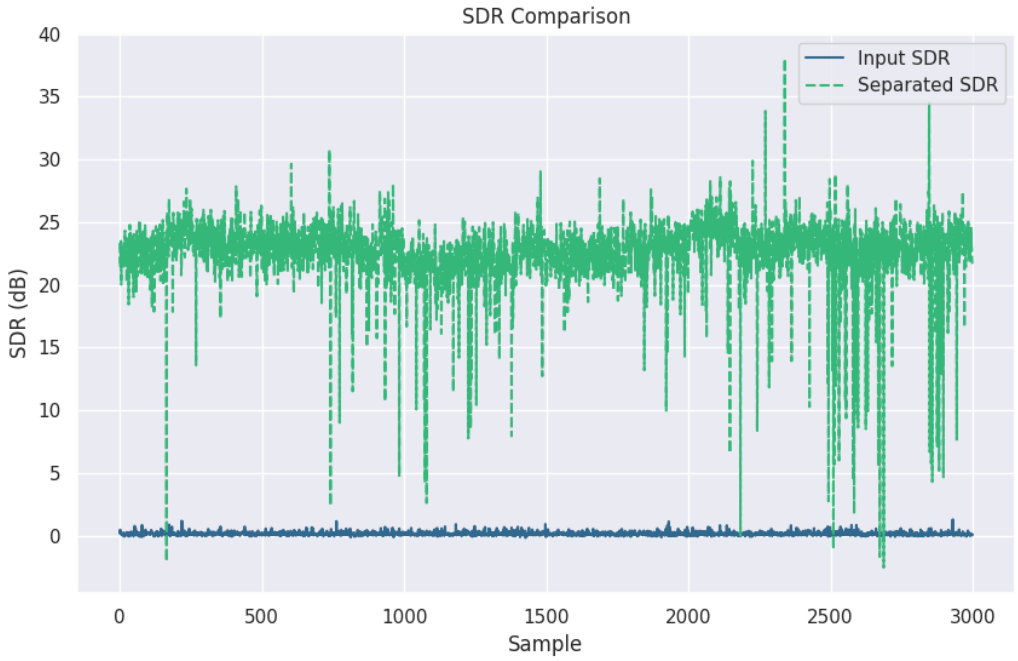


Figure 4.1: Comparison of SDR.

As is to be expected, Figure 4.1 and Figure 4.2 show good separation of the mixture as opposed to the mixture present at the input whereas for some samples the score is very low as shown by the spikes.

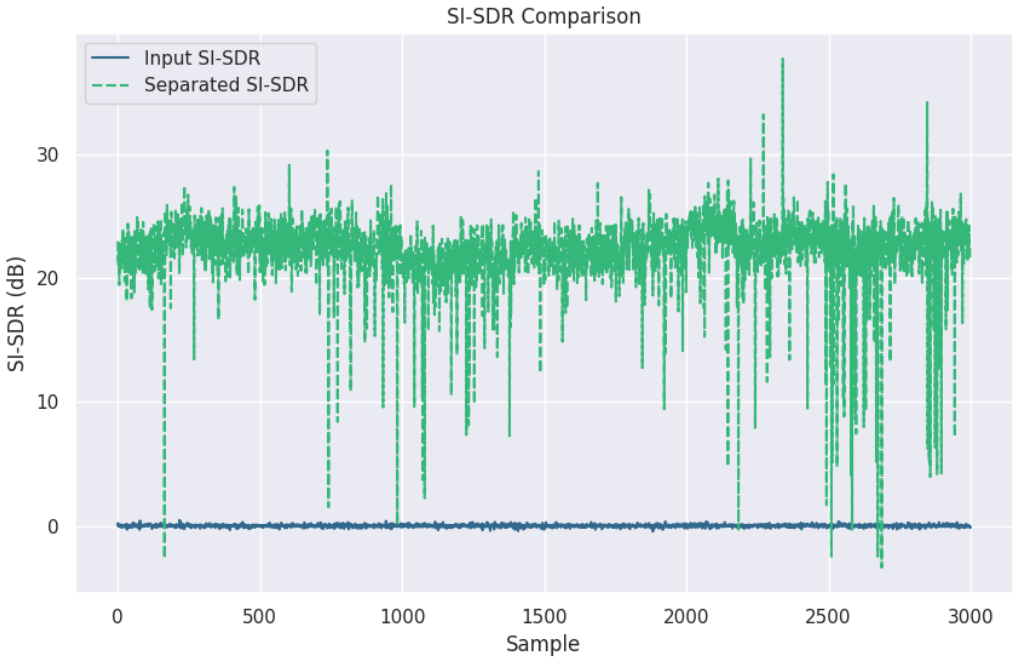


Figure 4.2: Comparison of SI-SDR.

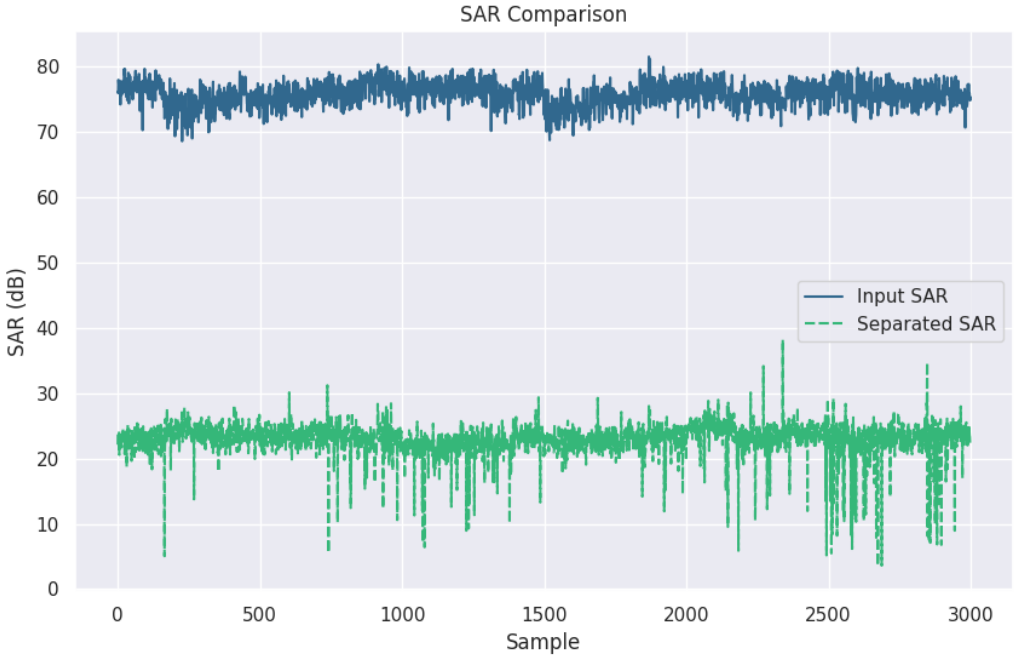


Figure 4.3: Comparison of SAR.

Interestingly, the SAR values as shown in Figure 4.3 decrease from around 75 dB at the input to just above 20 dB.
As described in Section 3.3.3, SAR describes the amount of artifacts introduced by the separation process. This discrepancy could suggest that the model even though effective in reducing interference in the estimation, might not preserve the quality and energy of the target source.

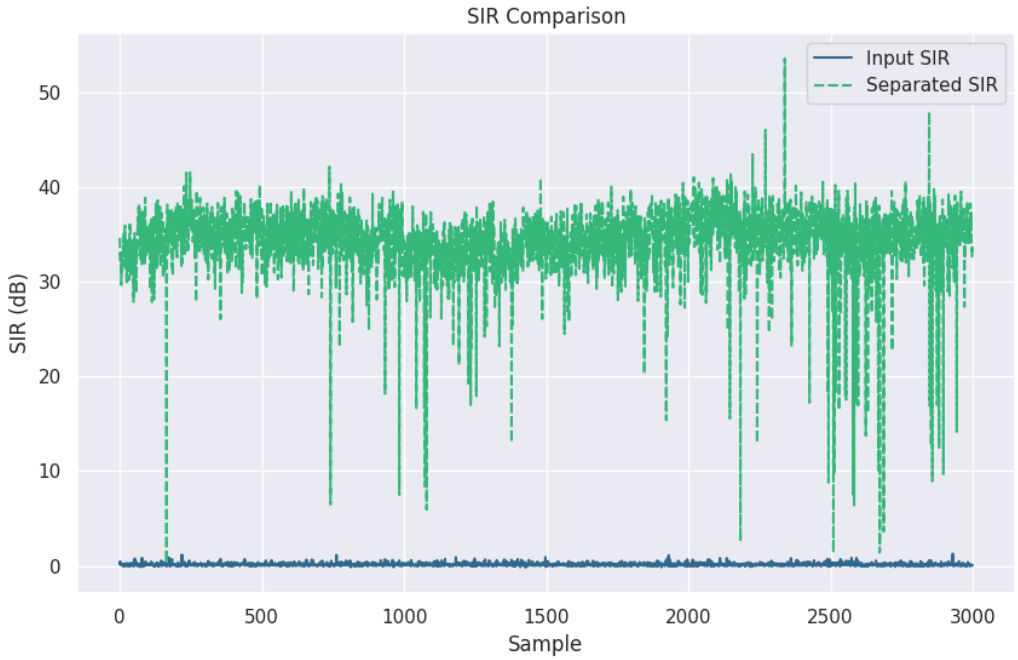


Figure 4.4: Comparison of SIR.

The SIR values depicted in Figure 4.4 collectively suggest that the model is fairly effective at reducing interference and capturing the desired separated utterances, therefore closely resembling the clean signal. However, even though high SIR values indicate successful separation in terms of interference, perceptual issues like distortion can still remain.

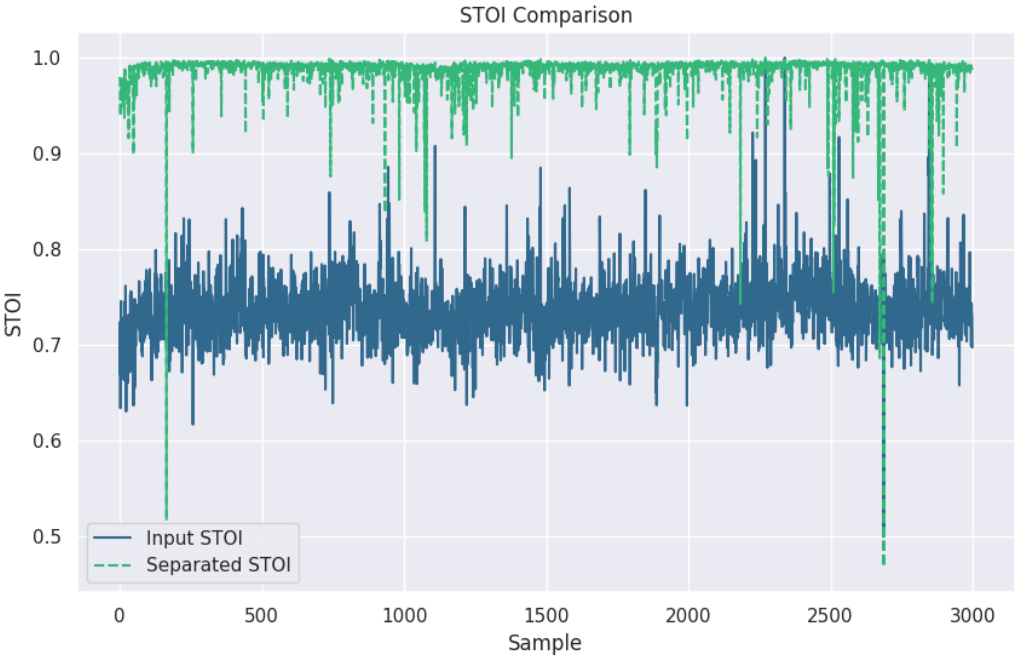


Figure 4.5: Comparison of STOI.

Lower STOI values at the input as portrayed in Figure 4.5 are to be expected since the intelligibility of the target speaker in question is reduced due to the multiple speakers present in the input mixture. Values close to one suggest a very high intelligibility however it is noteworthy that for some samples STOI drops to $0.9-0.8$ and in two occasions even as low as 0.46 suggesting greatly degraded speech intelligibility.

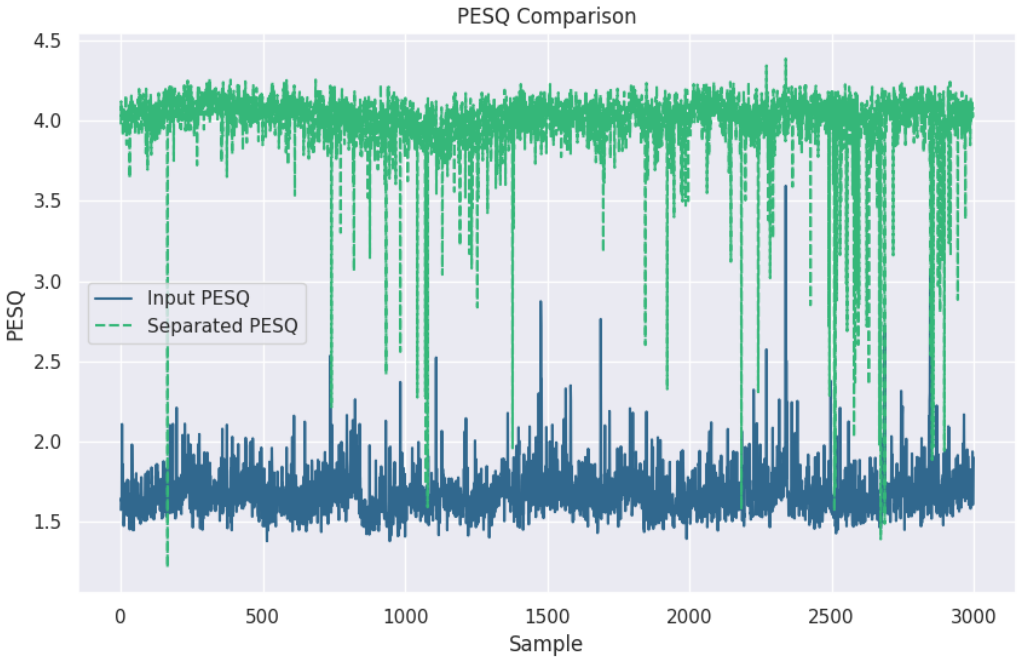


Figure 4.6: Comparison of PESQ.

Figure 4.6 shows the PESQ values across the evaluation samples and paints a similar picture. Throughout, the performance is good however more outliers with higher magnitudes are present in the results.

To conclude this section, we present an overview of the averaged evaluation metrics over the entire evaluation set in Table 4.3.

	SI-SDR	**SDR**	**SIR**	**SAR**	**STOI**	**PESQ**
Mixture	0.00±0.09	0.15±0.14	0.15±0.14	75.75±1.80	0.74±0.03	1.68±0.15
Separated Seq.	22.91±0.85	23.26±0.80	34.29±3.78	22.99±2.56	0.99±0.02	4.00±0.25

Table 4.3: Overview of Evaluation Metrics.

4.3 Separate and Diffuse vs the Baseline Model

In this chapter we compare several evaluation metrics of the separate and diffuse model against the baseline SepFormer [4] separator model in the context of WJS0-2mix. Additionally, we provide an overview of SDRi and SI-SDRi metrics over various clean and noisy WSJ0 and LibriSpeech based separation benchmarks.

Figure 4.7 shows the improvements achieved by the separate and diffuse model as opposed to just using the baseline SepFormer model. For most metrics a noticeable positive change

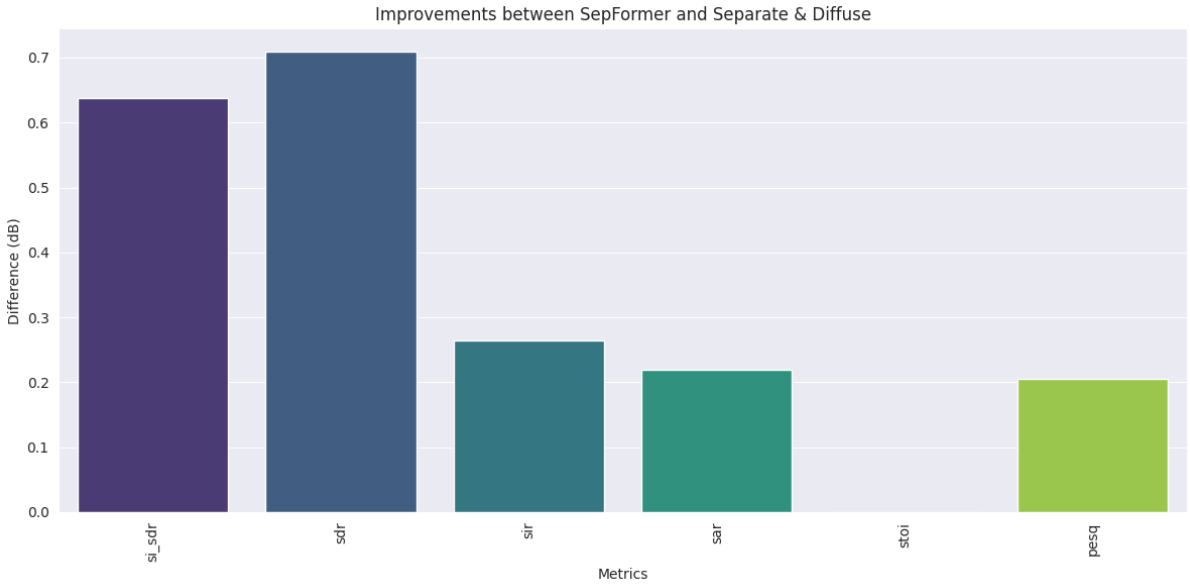


Figure 4.7: The improvements per evaluation metric in comparison to the plain SepFormer.

can be observed. However, it is noteworthy that the magnitude of change does not align with the extraordinary results presented in [8]. Further, metrics concerning the intelligibility and perceived quality, specifically STOI, have remained more or less unchanged.

Table 4.4 shows the separation performance of the model in various settings. Overall, the

Benchmark	SI-SDRi	SDRi
WSJ0-2mix	22.9	23.2
WSJ0-3mix	20.5	20.6
libri2mix	21.0	21.0
libri3mix	20.1	19.9
WHAM!	16.5	16.8
WHAMR!	4.5	3.9

Table 4.4: The separation performance of Separate and Diffuse obtained on several benchmarks.

model outperforms SepFormer with respect to WSJ0-2/3mix by up to 0.9dB. The librimix benchmark involving 3 speakers showed the most improvement, with gains of around 1.9dB and 1.3dB for SI-SDRi and SDRi metrics, respectively. 2 speaker separation based on LibriSpeech outperforms SepFormer in a similar magnitude as WSJ0-2/3mix benchmarks. The WHAM! benchmark, that is, WSJ0-2mix with ambient noise, showed minimal, almost negligible improvements. Adding the challenge of reverberated mixtures completely derails the model, leading to performances way below the baseline benchmarks. Hereby, the diffusion model is not able to cope with reverberated separations, leading to deteriorated utterances. The metrics perform 6dB and 8.3dB worse than the respective SepFormer model alone rendering Separate and Diffuse useless for applications with reverberant mixture separation. One approach to rectify and possibly improve the results on this benchmark could be to train a neural vocoder such as diffwave on raw reverberated single speech utterances to decrease the possible error introduced by the diffusion model. However, no guarantee is given that diffusion-based neural vocoders are capable of generating reverberant audio samples with small enough errors. This exploration exceeds the scope of this thesis and is left for future work.

4.4 Ensemble Model

In this subchapter, the ensemble model in Section 2.4 is evaluated on the WJS0-2mix and WHAMR! benchmarks.

Table 4.5 shows a clear lack of improvement achieved by the ensemble model.

Benchmark	SI-SDRi	SDRi
WSJ0-2mix	19.7	19.9
WHAMR!	13.7	13.9

Table 4.5: The performance of the ensemble model obtained on several benchmarks.

For both separation tasks, the model performs worse than a stand alone SepFormer but better than a SuDoRM-RF model. This suggests that a second deterministic separation model does not sufficiently cover the information missed by SepFormer that a diffusion model seems to provide.

Due to SepFormer already pushing the boundaries of possible separation performance achievable by deterministic models(see [6]), the integration of another, weaker deterministic model does not net the desired results. The slim discrepancy between SepFormer and the achievable upper bound suggests that the potential for improvement via this approach rather minimal. The inclusion of another deterministic model therefore does not enhance the separation performance likely due to an overlap in information gain by both models. It is left to discover if the introduction of another separator with separation results closer to SepFormer e.g. MossFormer [5], would provide information missed by SepFormer. To train and integrate a MossFormer model into the ensemble approach is left for future work. Further, expanding the architecture to more than two models could also net improvements.

4.5 Model Comparison

To conclude this chapter, the various models' parameters are shown. Further, the inference speed of the models is compared.

Table 4.6 shows the model sizes. The deterministic Separator models show very similar amounts of trainable parameters, whereas DiffWave is much smaller. The alignment network with only 200,000 parameters is deemed negligible in comparison.

Model	**Parameters**
SepFormer	26,000,000
DiffWave	6,910,000
Alignment Network	200,000
SuDoRM-RF WSJ0-2mix	25,240,000
SuDoRM-RF WHAMR!	26,610,000

Table 4.6: Number of parameters of utilized models.

The time a model takes to run inference, i.e., the process of making predictions based on unseen data, is vastly different throughout the models. Table 4.7 clearly shows SepFormer as the fastest model with around 22 milliseconds per inference. The diffusion model, despite being over four times smaller takes almost 25 times longer than SepFormer. This leads to a fairly sluggish inference time for the Separate and Diffuse model of over 620 milliseconds. SuDoRM-RF also falls of in comparison to SepFormer with inference times nearly eight times slower. The ensemble model's performance suffers from slower inference of SuDoRM-RF with timings

Model	Inference Time (ms)
SepFormer	22.4
SuDoRM-RF	178.4
DiffWave	546.9
Separate & Diffuse	620.4
Ensemble Model	244.9

Table 4.7: Averaged inference time of utilized models

of around 245 milliseconds. The inference times reported were generated over 1500 iterations on an NVIDIA GeForce RTX 2060 SUPER.

5 Conclusion and Future Work

In this final chapter, the methodology and main findings of this work are summarized. A final verdict of the feasibility of real world applications of the models presented in this thesis is given. Lastly, we provide an outlook for possible future research topics and expansions of this work.

5.1 Conclusion

This work investigated a sequence to sequence machine learning model based on Separate and Diffuse to separate audio mixtures into the individual utterances. The Separate and Diffuse model architecture improved upon existing state-of-the-art models by introducing diffusion into the ML pipeline. The system is composed of a deterministic separator model, a diffusion model and a residual CNN for alignment. The separator was taken as is from the publication and produces separated audio samples from input mixtures. The diffusion model generates noisy audio samples from given mel-spectrograms of the separator output. The alignment network merges the separator's estimation and the diffused, noisy estimation in the frequency domain to obtain a spectrum for each individual speaker in the input mixture.
In addition, another approach, an ensemble model, was introduced which utilized another deterministic separator model (SuDoRM-RF) instead of the diffusion model to reduce inference time and model complexity. Several architectures of alignment networks were tested with the ensemble model, whereas the best performing architecture was evaluated on the WHAMR! and WSJ0-2Mix benchmarks.

To train the models presented in this work, the separator and diffusion models had to be trained before the ensemble and Separate and Diffuse models. The separators were taken as is from the official publications where available. DiffWave, the diffusion model used, was trained on WAV/mel-spectrogram pairs of the WSJ0 and LibriSpeech datasets, depending on the separation task. To do so, the WSJ0 dataset was converted into WAV files and resampled to a sampling rate of 8kHz, resulting in 20 hours of training data for the diffusion model. LibriSpeech, the dataset derived from the LibriVox project, was converted from Free Lossless Audio Codec (FLAC) to WAV and down sampled from 16 to 8kHz. To train DiffWave on LibriSpeech, only the train-360 subset of data was utilized, netting approximately 360 hours of data. Afterward, the alignment networks for both Separate and Diffuse and the ensemble model were trained directly on the mixture datasets with the pretrained separator and diffusion models in the ML pipeline. Noisy and reverberant mixture datasets were introduced in form of WHAM! and WHAMR! to evaluate more realistic separation scenarios.

Several evaluation metrics were introduced and employed to provide results. The Separate and Diffuse model showed promising results on clean separation benchmarks, i.e., WSJ0-2/3mix and librimix. However, in noisy settings the model achieved no significant improvement over the SepFormer whereas in noisy and reverberant settings, the model performed poorly. As for the ensemble model, no improvement was achieved over the baseline SepFormer model due to the overlap in extracted information in both separator models, regardless of the type of alignment procedure used.

Considering the slow inference time and complexity of the Separate and Diffuse approach, which leads to perceptually deceptive results, i.e., the evaluation metrics indicate a clear improvement, whereas a listener cannot distinguish this improvement. Even though the results are satisfactory on paper, in real world applications it is recommended to simply employ SepFormer to avoid the computational overhead and complexity introduced by Separate and Diffuse. The ensemble model showed potential but was not able to produce satisfactory results.

5.2 Future Work

During this work, we encountered several opportunities for improvement and future research.

Firstly, different diffusion based vocoder models such as SpecGrad [81] and WaveGrad [82] or the diffusion model of the recent DiffSound [83] could be utilized. Further, employing a diffusion model specifically designed and trained on noisy and reverberant data would likely improve the performance in WHAM! and WHAMR! benchmarks.

Secondly, replacing SuDoRM-RF with a more powerful deterministic separator, such as MossFormer [5] or the recent state-of-the-art MossFormer2 [84] could improve overall performance of the ensemble approach. Due to the time constraint and computational demand of MossFormer, training a MossFormer model was not feasible.

Another unexplored approach is to insert a model to denoise and dereverberate the WHAM! and WHAMR! estimations before feeding them into the diffusion model.

Bibliography

[1] D. Technologies. "Deepl translator." (Year of access, 2024), [Online]. Available: https://www.deepl.com/translator.

[2] J. H. McDermott, "The cocktail party problem," *Curr Biol*, vol. 19, no. 22, R1024–7, 2009, ISSN: 1879-0445 (Electronic), 0960-9822 (Linking). DOI: 10.1016/j.cub.2009.09.005. [Online]. Available: https://doi.org/10.1016/j.cub.2009.09.005.

[3] F. Bahmaninezhad, S.-X. Zhang, Y. Xu, M. Yu, J. H. L. Hansen, and D. Yu, *A unified framework for speech separation*, 2019. arXiv: 1912.07814 [cs.LG].

[4] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong, *Attention is all you need in speech separation*, 2021. arXiv: 2010.13154 [eess.AS].

[5] S. Zhao and B. Ma, *Mossformer: Pushing the performance limit of monaural speech separation using gated single-head transformer with convolution-augmented joint self-attentions*, 2023. arXiv: 2302.11824 [cs.SD].

[6] S. Lutati, E. Nachmani, and L. Wolf, *Sepit: Approaching a single channel speech separation bound*, 2023. arXiv: 2205.11801 [eess.AS].

[7] E. Nachmani, Y. Adi, and L. Wolf, *Voice separation with an unknown number of multiple speakers*, 2020. arXiv: 2003.01531 [eess.AS].

[8] S. Lutati, E. Nachmani, and L. Wolf, *Separate and diffuse: Using a pretrained diffusion model for improving source separation*, 2023. arXiv: 2301.10752 [eess.AS].

[9] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, *Diffwave: A versatile diffusion model for audio synthesis*, 2021. arXiv: 2009.09761 [eess.AS].

[10] E. Tzinis, Z. Wang, and P. Smaragdis, "Sudo rm -rf: Efficient networks for universal audio source separation," in *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, IEEE, Sep. 2020. DOI: 10.1109/mlsp49062.2020.9231900. [Online]. Available: http://dx.doi.org/10.1109/MLSP49062.2020.9231900.

[11] E. Tzinis, Z. Wang, X. Jiang, and P. Smaragdis, "Compute and memory efficient universal sound source separation," *Journal of Signal Processing Systems*, vol. 94, no. 2, pp. 245–259, Jul. 2021, ISSN: 1939-8115. DOI: 10.1007/s11265-021-01683-x. [Online]. Available: http://dx.doi.org/10.1007/s11265-021-01683-x.

[12] J. R. Hershey, Z. Chen, J. L. Roux, and S. Watanabe, *Deep clustering: Discriminative embeddings for segmentation and separation*, 2015. arXiv: 1508.04306 [cs.NE].

[13] T. e. a. Mikolov, "Recurrent neural network based language model," in *Interspeech*, 2010. [Online]. Available: https://api.semanticscholar.org/CorpusID:17048224.

[14] X. Jin and J. Han, "K-means clustering," in *Encyclopedia of Machine Learning*, C. Sammut and G. I. Webb, Eds. Boston, MA: Springer US, 2010, pp. 563–564, ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_425. [Online]. Available: https://doi.org/10.1007/978-0-387-30164-8_425.

[15] Z. Chen, Y. Luo, and N. Mesgarani, "Deep attractor network for single-microphone speaker separation," *CoRR*, vol. abs/1611.08930, 2016. arXiv: 1611.08930. [Online]. Available: http://arxiv.org/abs/1611.08930.

[16] Z.-Q. Wang, J. L. Roux, D. Wang, and J. R. Hershey, *End-to-end speech separation with unfolded iterative phase reconstruction*, 2018. arXiv: 1804.10204 [cs.SD].

[17] D. S. Williamson, Y. Wang, and D. Wang, "Complex ratio masking for monaural speech separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 3, pp. 483–492, 2016. DOI: 10.1109/TASLP.2015.2512042.

[18] H. Erdogan, J. R. Hershey, S. Watanabe, and J. Le Roux, "Phase-sensitive and recognition-boosted speech separation using deep recurrent neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 708–712. DOI: 10.1109/ICASSP.2015.7178061.

[19] C. Xu, W. Rao, X. Xiao, E. S. Chng, and H. Li, "Single channel speech separation with constrained utterance level permutation invariant training using grid lstm," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 6–10. DOI: 10.1109/ICASSP.2018.8462471.

[20] Y. Z. Isik, J. L. Roux, Z. Chen, S. Watanabe, and J. R. Hershey, "Single-channel multi-speaker separation using deep clustering," *CoRR*, vol. abs/1607.02173, 2016. arXiv: 1607.02173. [Online]. Available: http://arxiv.org/abs/1607.02173.

[21] Z.-Q. Wang, J. L. Roux, and J. R. Hershey, "Alternative objective functions for deep clustering," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 686–690. DOI: 10.1109/ICASSP.2018.8462507.

[22] D. Yu, M. Kolbæk, Z.-H. Tan, and J. Jensen, *Permutation invariant training of deep models for speaker-independent multi-talker speech separation*, 2017. arXiv: 1607.00325 [cs.CL].

[23] C. Weng, D. Yu, M. L. Seltzer, and J. Droppo, "Deep neural networks for single-channel multi-talker speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 10, pp. 1670–1679, 2015. DOI: 10.1109/TASLP.2015.2444659.

[24] Y. Luo and N. Mesgarani, "Tasnet: Time-domain audio separation network for real-time, single-channel speech separation," *CoRR*, vol. abs/1711.00541, 2017. arXiv: 1711.00541. [Online]. Available: http://arxiv.org/abs/1711.00541.

[25] Y. Luo and N. Mesgarani, "Tasnet: Surpassing ideal time-frequency masking for speech separation," *CoRR*, vol. abs/1809.07454, 2018. arXiv: 1809.07454. [Online]. Available: http://arxiv.org/abs/1809.07454.

[26] D. Paper, "Convolutional and variational autoencoders," in *State-of-the-Art Deep Learning Models in TensorFlow: Modern Machine Learning in the Google Colab Ecosystem*. Berkeley, CA: Apress, 2021, pp. 219–241, ISBN: 978-1-4842-7341-8. DOI: 10.1007/978-1-4842-7341-8_9. [Online]. Available: https://doi.org/10.1007/978-1-4842-7341-8_9.

[27] Y. Liu and D. Wang, "Divide and conquer: A deep CASA approach to talker-independent monaural speaker separation," *CoRR*, vol. abs/1904.11148, 2019. arXiv: 1904.11148. [Online]. Available: http://arxiv.org/abs/1904.11148.

[28] Y. Luo, Z. Chen, and T. Yoshioka, *Dual-path rnn: Efficient long sequence modeling for time-domain single-channel speech separation*, 2020. arXiv: 1910.06379 [eess.AS].

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. arXiv: 1706.03762. [Online]. Available: http://arxiv.org/abs/1706.03762.

[30] J. L. Ba, J. R. Kiros, and G. E. Hinton, *Layer normalization*, 2016. arXiv: 1607.06450 [stat.ML].

[31] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV].

[32] C. Subakan, M. Ravanelli, S. Cornell, F. Grondin, and M. Bronzi, *On using transformers for speech-separation*, Feb. 2022.

[33] *Diagrams.net (formerly draw.io)*, https://www.diagrams.net/.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *CoRR*, vol. abs/1502.01852, 2015. arXiv: 1502.01852. [Online]. Available: http://arxiv.org/abs/1502.01852.

[35] K. Fukushima, "Cognitron: A self-organizing multilayered neural network," *Biological Cybernetics*, vol. 20, pp. 121–136, 1975. [Online]. Available: https://api.semanticscholar.org/CorpusID:28586460.

[36] S. Hochreiter, "Untersuchungen zu dynamischen neuronalen netzen," M.S. thesis, TU Munich, 1991.

[37] C. Subakan, M. Ravanelli, S. Cornell, F. Grondin, and M. Bronzi, *Exploring self-attention mechanisms for speech separation*, 2023. arXiv: 2202.02884 [eess.AS].

[38] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015. arXiv: 1505.04597. [Online]. Available: http://arxiv.org/abs/1505.04597.

[39] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," *CoRR*, vol. abs/1503.03585, 2015. arXiv: 1503.03585. [Online]. Available: http://arxiv.org/abs/1503.03585.

[40] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *CoRR*, vol. abs/2105.05233, 2021. arXiv: 2105.05233. [Online]. Available: https://arxiv.org/abs/2105.05233.

[41] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, *Hierarchical text-conditional image generation with clip latents*, 2022. arXiv: 2204.06125 [cs.CV].

[42] Z. Shi, X. Zhou, X. Qiu, and X. Zhu, "Improving image captioning with better use of captions," *CoRR*, vol. abs/2006.11807, 2020. arXiv: 2006.11807. [Online]. Available: https://arxiv.org/abs/2006.11807.

[43] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *CoRR*, vol. abs/2006.11239, 2020. arXiv: 2006.11239. [Online]. Available: https://arxiv.org/abs/2006.11239.

[44] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, W. Zhang, B. Cui, and M.-H. Yang, *Diffusion models: A comprehensive survey of methods and applications*, 2023. arXiv: 2209.00796 [cs.LG].

[45] B. Poole, S. Ozair, A. van den Oord, A. A. Alemi, and G. Tucker, "On variational bounds of mutual information," *CoRR*, vol. abs/1905.06922, 2019. arXiv: 1905.06922. [Online]. Available: http://arxiv.org/abs/1905.06922.

[46] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. J. Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu, "Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions," *CoRR*, vol. abs/1712.05884, 2017. arXiv: 1712.05884. [Online]. Available: http://arxiv.org/abs/1712.05884.

[47] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *CoRR*, vol. abs/1609.03499, 2016. arXiv: 1609.03499. [Online]. Available: http://arxiv.org/abs/1609.03499.

[48] S. S. Stevens, J. Volkmann, and E. B. Newman, "A Scale for the Measurement of the Psychological Magnitude Pitch," *The Journal of the Acoustical Society of America*, vol. 8, no. 3, pp. 185–190, Jun. 2005, ISSN: 0001-4966. DOI: 10.1121/1.1915893. eprint: https://pubs.aip.org/asa/jasa/article-pdf/8/3/185/11828615/185_1_online.pdf. [Online]. Available: https://doi.org/10.1121/1.1915893.

[49] D. O'Shaughnessy, *Speech Communications: Human And Machine (ieee)*. Universities Press (India) Pvt. Limited, ISBN: 9788173713743. [Online]. Available: https://books.google.at/books?id=UWmD8aY_448C.

[50] M. Kurttutan, *Torchview*, https://github.com/mert-kurttutan/torchview, 2023.

[51] D. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, pp. 241–259, Dec. 1992. DOI: 10.1016/S0893-6080(05)80023-1.

[52] E. Tzinis, *Sudo$_r m_r f$*, https://github.com/etzinis/sudo_rm_rf, 2020.

[53] C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal, "Deep complex networks," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=H1T2hmZAb.

[54] J. S. Garofolo and et al., *CSR-I (WSJ0) Complete LDC93S6A*, Web Download, Philadelphia: Linguistic Data Consortium, 1993. [Online]. Available: https://catalog.ldc.upenn.edu/LDC93S6A.

[55] G. Wichern, J. Antognini, M. Flynn, L. R. Zhu, E. McQuinn, D. Crow, E. Manilow, and J. Le Roux, "Wham!: Extending speech separation to noisy environments," in *Proc. Interspeech*, Sep. 2019.

[56] M. Maciejewski, G. Wichern, and J. Le Roux, "Whamr!: Noisy and reverberant single-channel speech separation," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020.

[57] A. C. Laboratory, *Pyroomacoustics*, https://github.com/LCAV/pyroomacoustics, 2022.

[58] M. Maciejewski, G. Wichern, and J. Le Roux. "WHAM! readme." (2020), [Online]. Available: http://wham.whisper.ai/WHAMR_README.html.

[59] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210. DOI: 10.1109/ICASSP.2015.7178964. [Online]. Available: https://www.openslr.org/12/.

[60] J. Cosentino, M. Pariente, S. Cornell, A. Deleforge, and E. Vincent, *Librimix: An open-source dataset for generalizable speech separation*, 2020. arXiv: 2005.11262 [eess.AS].

[61] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Interspeech*, 2015, pp. 3586–3589.

[62] J. Andén and S. Mallat, "Deep scattering spectrum," *CoRR*, vol. abs/1304.6763, 2013. arXiv: 1304.6763. [Online]. Available: http://arxiv.org/abs/1304.6763.

[63] ITU-R, "Recommendation itu-r bs.1770-4: Algorithms to measure audio programme loudness and true-peak audio level," ITU-R Recommendation BS.1770-4, 2015.

[64] M. Pariente, S. Cornell, J. Cosentino, S. Sivasankaran, E. Tzinis, J. Heitkaemper, M. Olvera, F.-R. Stöter, M. Hu, J. M. Martín-Doñas, D. Ditter, A. Frank, A. Deleforge, and E. Vincent, "Asteroid: The PyTorch-based audio source separation toolkit for researchers," in *Proc. Interspeech*, 2020.

[65] S. Nanavati, *Diffwave*, https://github.com/lmnt-com/diffwave, 2023.

[66] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017. arXiv: 1412.6980 [cs.LG].

[67] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance," *Climate Research*, vol. 30, pp. 79–82, 2005. [Online]. Available: https://api.semanticscholar.org/CorpusID:120556606.

[68] M. Ravanelli, T. Parcollet, P. Plantinga, A. Rouhe, S. Cornell, L. Lugosch, C. Subakan, N. Dawalatabad, A. Heba, J. Zhong, J.-C. Chou, S.-L. Yeh, S.-W. Fu, C.-F. Liao, E. Rastorgueva, F. Grondin, W. Aris, H. Na, Y. Gao, R. D. Mori, and Y. Bengio, *Speechbrain: A general-purpose speech toolkit*, 2021. arXiv: 2106.04624 [eess.AS].

[69] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006, ISBN: 0387310738.

[70] R. Scheibler, *Sdr – medium rare with fast computations*, 2021. arXiv: 2110.06440 [eess.AS].

[71] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, pp. 1462–1469, 2006.

[72] "Front matter," in *Music Engineering*, Elsevier, 2001, p. iii.

[73] T. Crich, *Recording tips for engineers: for cleaner, brighter tracks*, en, 2nd ed. Oxford, England: Focal Press, Apr. 2005.

[74] *Open Source Tools & Data for Music Source Separation*. https://source-separation.github.io/tutorial, 2020. [Online]. Available: https://source-separation.github.io/tutorial.

[75] J. L. Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, "SDR - half-baked or well done?" *CoRR*, vol. abs/1811.02508, 2018. arXiv: 1811.02508. [Online]. Available: http://arxiv.org/abs/1811.02508.

[76] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. R. Jensen, "A short-time objective intelligibility measure for time-frequency weighted noisy speech," *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4214–4217, 2010.

[77] C. Taal, R. Hendriks, R. Heusdens, and J. Jensen, "An algorithm for intelligibility prediction of time–frequency weighted noisy speech," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, pp. 2125–2136, Oct. 2011. DOI: 10.1109/TASL.2011.2114881.

[78] A. V. Oppenheim, R. W. Schafer, M. A. Yoder, and W. T. Padgett, *Discrete-time signal processing*, en, 3rd ed. Upper Saddle River, NJ: Pearson, Aug. 2009.

[79] A. W. Rix, J. G. Beerends, M. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs," *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, vol. 2, 749–752 vol.2, 2001.

[80] J. G. Beerends and J. A. Stemerdink, "A perceptual speech-quality measure based on a psychoacoustic sound representation," *Journal of the Audio Engineering Society*, vol. 42, no. 3, pp. 115–123, 1994.

[81] Y. Koizumi, H. Zen, K. Yatabe, N. Chen, and M. Bacchiani, *Specgrad: Diffusion probabilistic model based neural vocoder with adaptive noise spectral shaping*, 2022. arXiv: 2203.16749 [eess.AS].

[82] N. Chen, Y. Zhang, H. Zen, R. J. Weiss, M. Norouzi, and W. Chan, *Wavegrad: Estimating gradients for waveform generation*, 2020. arXiv: 2009.00713 [eess.AS].

[83] D. Yang, J. Yu, H. Wang, W. Wang, C. Weng, Y. Zou, and D. Yu, *Diffsound: Discrete diffusion model for text-to-sound generation*, 2023. arXiv: 2207.09983 [cs.SD].

[84] S. Zhao, Y. Ma, C. Ni, C. Zhang, H. Wang, T. H. Nguyen, K. Zhou, J. Yip, D. Ng, and B. Ma, *Mossformer2: Combining transformer and rnn-free recurrent network for enhanced time-domain monaural speech separation*, 2023. arXiv: 2312.11825 [cs.SD].

List of Figures

List of Tables

Acronyms

CASA computational auditory scene analysis. 17

CNN Convolutional Neural Network. 16, 32, 55

Conv-TasNet Convolutional Time-Domain Audio Separation Network. 16, 17, 26, 27, LXIII

dB decibels. 36, 42, 43, 47, 49, 52

FFT Fast Fourier Transform. 44

FLAC Free Lossless Audio Codec. 55

GAN Generative Adverserial Networks. 28

iSTFT inverse short-time Fourier Transform. 16

LSTM Long Short Term Memory. 13, 16, 31

LUFS Loudness Units relative to Full Scale. 38

MAE Mean Absolute Error. 40

ML Machine Learning. 13, 14, 31, 34, 35, 38, 55

MOS Mean Opinion Score. 45

PESQ Perceptual Evaluation of Speech Quality. 44, 45, 51, LXIII

PIT Permutation Invariant Training. 15, 16, LXIII

PReLU Parametric Rectified Linear Unit. 24

PSQM Perceptual Speech Quality Measurement. 44

ReLU Rectified Linear Unit. 24

RNN Recurrent Neural Network. 15, 17–19, 23–26, LXIII

SAR Source-to-Artifacts ratio. 42, 49, LXIII

SDPA Scaled Dot-Product Attention. 19

SDR Signal-to-Distortion ratio. 41–43, 47, 48, 51, 52, LXIII

SepFormer Separation Transformer. 13, 15, 23–25, 31, 34, 45, 47, 51–56, LXIII, LXV

SI-SDR Scale-Invariant Signal-to-Distortion ratio. 41–43, 47, 49, 51, 52, LXIII

SIR Source-to-Interferences ratio. 42, 50, LXIII

SNR Signal-to-Noise ratio. 35, 36, 38, 43

STFT short-time Fourier Transform. 16, 23, 39

STOI Short-time Objective Intelligibility measure. 43, 44, 50–52, LXIII

SuDoRM-RF SUccessive DOwnsampling and Resampling of Multi-Resolution Features. 13, 15, 26, 27, 34, 47, 53–56, LXIII, LXV

TasNet Time-domain Audio Separation Network. 16, 17, LXIII

uPIT Utterance-level Permutation Invariant Training. 15

WHAM! WSJ0 Hipster Ambient Mixtures. 36–38, 47, 52, 55, 56

WHAMR! WSJ0 Hipster Ambient Mixtures with Reverb. 36, 47, 52, 53, 55, 56, LXV

WSJ Wall Street Journal. 35, 37, 38, 40, 41, 47, 51, 55