BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

# TOWARDS UNSUPERVISED SPEECH ENHANCEMENT USING NEURAL AUDIO CODECS

VYLEPŠENÍ KVALITY ŘEČI BEZ SUPERVIZE ZA POMOCI NEURÁLNÍCH AUDIO KODEKŮ

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR
AUTOR PRÁCE

Bc. DOMINIK KLEMENT

SUPERVISOR
VEDOUCÍ PRÁCE

doc. LUKÁŠ BURGET, Ph.D.

BRNO 2025

# Master's Thesis Assignment

| | | |
|---|---|---|
| Institut: | Department of Computer Graphics and Multimedia (DCGM) | 165614 |
| Student: | **Klement Dominik, Bc.** | |
| Programme: | Information Technology and Artificial Intelligence | |
| Specialization: | Machine Learning | |
| Title: | **Towards Unsupervised Speech Enhancement using Neural Audio Codecs** | |
| Category: | Speech and Natural Language Processing | |
| Academic year: | 2024/25 | |

Assignment:

1. Familiarize yourself with neural audio codecs and architectures for speech quality enhancement.
2. Train and evaluate a neural audio codec that improves speech quality rather than merely reconstructing it.
3. Extend the neural audio codec to enable unsupervised training for speech enhancement.
4. Evaluate the unsupervised training and analyze the trained model and the enhanced speech audio.

Literature:

1. Défossez, A., Copet, J., Synnaeve, G., & Adi, Y. (2022). High Fidelity Neural Audio Compression. arXiv preprint arXiv:2210.13438. https://arxiv.org/pdf/2210.13438
2. Nicholas Babaev, Kirill Tamogashev, Azat Saginbaev, Ivan Shchekotov, Hanbin Bae, Hosang Sung, WonJun Lee, Hoon-Young Cho, & Pavel Andreev. (2024). FINALLY: fast and universal speech enhancement with studio-like quality. https://arxiv.org/pdf/2410.05920
3. Xiaoyu Bie, Simon Leglaive, Xavier Alameda-Pineda, & Laurent Girin (2021). Unsupervised Speech Enhancement using Dynamical Variational Auto-Encoders. CoRR, abs/2106.12271 https://arxiv.org/pdf/2106.12271

Requirements for the semestral defence:
Points 1 and 2

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

| | |
|---|---|
| Supervisor: | **Burget Lukáš, doc. Ing., Ph.D.** |
| Head of Department: | Černocký Jan, prof. Dr. Ing. |
| Beginning of work: | 1.11.2024 |
| Submission deadline: | 21.5.2025 |
| Approval date: | 13.5.2025 |

## Abstract

This thesis investigates approaches to speech enhancement using neural audio codecs (NACs). Traditional supervised methods rely on artificially simulated datasets of paired clean and noisy speech, which might fail to generalize to real-world conditions. To address these limitations, this work introduces a novel dual-branch architecture that enables clean speech/noise decomposition without requiring paired data. The system employs adversarial training with branch-specific discriminators to guide one branch toward generating clean speech and the other toward generating noise. To ensure consistency between the input and the enhanced output, the system also enforces that the sum of the two branches closely resembles the original noisy input. Additionally, vector quantization is used to control latent bandwidth and reduce interference between the branches. The model is evaluated across supervised and unsupervised settings using a wide range of objective, perceptual, and downstream metrics. Extensive experiments validate the effectiveness of the proposed approach, demonstrating its ability to enhance speech quality in both synthetic and real-world noisy environments without explicit supervision. The results show that the model performs comparably to prior work while offering high efficiency, enabling real-time audio enhancement.

## Abstrakt

Táto práca sa zaoberá skúmaním spôsobov na vylepšenie reči pomocou neurálnych audio kodekov. Štandardné supervizované metody sa spoliehajú na umelo vytvorené dáta z párov zašumenej a čistej reči, čo môže sposobiť problémy s generalizáciou na reálne audio. Ako riešenie v tejto práci predstavujeme novú dvoj-vetvovú architektúru, ktorá umožňuje separáciu čistej reči a šumu bez potreby párových dát. Navrhnutý systém využíva adversariálny tréning s vetvovými disrkiminátormi, ktoré zaručujú že jedna vetva bude produkovať čistú reč a druhá šum. Pre zaručenie konzistencie medzi čistou rečou a vstupnou zašumenou nahrávkou, systém je trénovaný tak aby súčet výstupov z daných vetví odpovedal vstupnému zašumenému audiu. Naviac pritom používa vektorovú kvantizáciu na kontrolu priepustnosti latentných reprezentácií pre kontrolu interferencie medzi vetvami. Navrhnutý model je vyhodnotení na supervizovanom aj nesupervizovanom učení za použitie širokej škály objektívnych, percepčných a downstream metrík. Rozsiahle experimenty validujú efektivitu navrhnutého riešenia a demonštrujú jeho schopnosť vylepšiť kvalitu reči syntetických aj skutočných nahrávok bez použitia explicitnej supervízie. Výsledky ukazujú že systém je porovnateľný s predchádzajúcimi riešeniami a umožňuje vylepšenie audia v reálnom čase.

## Keywords

Speech Enhancement, Neural Audio Codecs, Speech Processing, Deep Learning, Unsupervised Learning, Adversarial Training

## Klíčová slova

Vylepšenie Reči, Neurálny Audio Kodek, Spracovanie Reči, Hlboké Učenie, Učenie bez Supervízie, Adversariálne Trénovanie

## Reference

KLEMENT, Dominik. *Towards Unsupervised Speech Enhancement using Neural Audio Codecs.* Brno, 2025. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor doc. Lukáš Burget, Ph.D.

## Rozšířený abstrakt

Táto diplomová práca predstavuje komplexné skúmanie neštruktúrovaného (nesupervizovaného) zlepšovania reči pomocou neurónových audio kodekov (NACs), pričom sa zameriava na zásadné obmedzenia tradičných supervizovaných metód. Supervizované prístupy zvyčajne vyžadujú rozsiahle datasety pozostávajúce zo spárovaných príkladov zašumenej a čistej reči. Tieto datasety sú umelo vytvárané, čo obmedzuje ich schopnosť efektívne generalizovať do reálnych akustických prostredí v dôsledku rozdielov medzi simulovaným a skutočným zašumeným audiom.

Na prekonanie týchto problémov práca navrhuje novú dvojvetvovú neurónovú architektúru, špeciálne navrhnutú na umožnenie efektívnej separácie a rekonštrukcie čistej reči a šumu bez potreby spárovaných dát. Tento dvojvetvový model využíva adversariálne trénovanie, pri ktorom sa dve samostatné vetvy trénujú súčasne—jedna sa výslovne zameriava na produkciu čistej reči, zatiaľ čo druhá sa zameriava na rekonštrukciu šumu. Kľúčovým prvkom tejto metódy je zavedenie rekonštrukčnej konzistencie, ktorá zabezpečuje, že kombinovaný výstup oboch vetiev sa čo najviac približuje pôvodnému zašumenému vstupu. Takéto riešenie umožňuje využitie rozsiahlych dát nezávisle nahratej čistej reči a šumu, čím sa eliminuje potreba explicitného spárovaných príkladov.

Neoddeliteľnou súčasťou úspechu tejto architektúry je implementácia vektorovej kvantizácie (VQ), ktorá slúži na reguláciu šírky pásma latentných reprezentácií a minimalizáciu interferencie medzi oboma vetvami. To zlepšuje schopnosti modelu v oblasti separácie, čím zabezpečuje jasnú a efektívnu dekompozíciu na rečové a šumové zložky. Navrhnutá architektúra modelu je rozsiahlo hodnotená v rôznych scenároch vrátane supervizovaného a plne nesupervizovaného trénovania.

Vyhodnotenie kvality odšumenej reči zahŕňa rôzne signálové metriky, posúdenie vnímanej kvality a downstream úloh, ako napríklad automatické rozpoznávanie reči (ASR), na overenie praktickej využiteľnosti metódy zlepšovania reči. Výsledky z rozsiahlych experimentov ukazujú, že navrhnutá nesupervizovaná metóda dosahuje porovnateľnú kvalitu než predošlé nesupervizované techniky. Zároveň to dosahuje pri podstatne zjednodušených požiadavkách na dáta a výrazne vyššej výpočtovej efektívnosti, čo ju robí mimoriadne vhodnou pre aplikácie v reálnom čase.

Okrem toho boli vykonané rozsiahle ablačné štúdie s cieľom objasniť dopady jednotlivých komponentov modelu a tréningových stratégií. Systematicky sa skúmali faktory ako vplyv pretrénovania, architektúra diskriminátorov, strategické využitie vektorovej kvantizácie a rôzne metódy kombinovania vetiev.

Celkovo táto práca poskytuje robustný, teoreticky podložený a experimentálne overený framework pre nesupervizované odšumenie reči, pričom vykazuje značný potenciál pre efektívne nasadenie v realistických a zložitých šumových prostrediach bez potreby nákladných a ťažko škálovateľných spárovaných dát.

# Towards Unsupervised Speech Enhancement using Neural Audio Codecs

## Declaration

I declare that this thesis is my own work supervised by doc. Lukáš Burget, Ph.D. All sources have been acknowledged by references.

. . . . . . . . . . . . . . . . . . . . . . .
Dominik Klement
September 12, 2025

## Acknowledgements

I would like to express my deepest gratitude to my supervisor, doc. Lukáš Burget, Ph.D., for his invaluable guidance, our discussions during coffee breaks, and his unwavering support throughout the two years I have been involved in speech research as a member of the Speech@FIT group. I am also sincerely grateful to M.Sc. Mireia Diez Sanches, Ph.D., for the countless discussions and her help when I was starting out in research. Furthermore, I would like to extend my thank you to prof. Honza Černocký and the entire Speech@FIT group for fostering a productive and inspiring research environment, and for giving me the opportunity to spend a year on a research stay at Johns Hopkins University.

There, I am especially thankful to Prof. Sanjeev Khudanpur and Matthew Wiesner, Ph.D., for their support and guidance, as well as to all my CLSP friends for making my stay truly unforgettable. A very special thank you goes to Matthew Maciejewski, Ph.D., with whom I had the chance to explore a wide range of research ideas and random topics about practically anything. His feedback and discussions around this project were instrumental—this thesis would have never come to light without him.

Last but certainly not least, I would like to thank my family, girlfriend, and friends for their endless love and support.

### Computation Resources

This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90254).

# Contents

# List of Figures

# Chapter 1

# Introduction

Speech enhancement (SE) is a long-standing problem in signal processing and machine learning, aiming to improve the quality and intelligibility of speech degraded by environmental noise or artifacts caused by transmission or recording. It plays a crucial role in a wide range of applications, including teleconferencing, hearing aids, automatic speech recognition (ASR), and real-time communication systems. While significant progress has been made using supervised deep-learning approaches, these typically require large-scale paired datasets of noisy and clean audio, which are expensive, domain-specific, and challenging to scale in real-world scenarios. Moreover, supervised training usually relies on artificially simulated noisy speech recordings, creating a gap between training and testing conditions, leading to worse generalization to real-world noise conditions.

In recent years, unsupervised and self-supervised learning have emerged as promising alternatives, enabling models to learn from unpaired or weakly labeled data. However, applying these techniques to speech enhancement remains challenging due to the underdetermined nature of the problem: Given a noisy speech signal, infinitely many plausible ways exist to separate the clean speech without explicit supervision. Effective solutions must, therefore, rely on strong inductive biases, signal priors, or structural constraints.

This thesis proposes a novel framework for unsupervised speech enhancement based on neural audio codecs. We introduce a dual-branch architecture that outputs two parallel audio signals that, after summming, reconstruct the input signal. Furthermore, we employ adversarial training to force one branch to output clean speech and the other to output noise by training two generative adversarial network (GAN) discriminators to distinguish between the generated outputs and real clean speech or noise, which allows us to leverage vast amounts of clean speech data and noise corpora without requiring paired examples. We further explore vector quantization (VQ) as a mechanism to limit the bandwidth of the latent representations, which helps to reduce leakage between the two branches and encourages the disentanglement of clean and noisy speech components.

We evaluate our models across various supervised and unsupervised settings using both traditional signal-based metrics and perceptual metrics, as well as downstream tasks such as automatic speech recognition (ASR) performance or speaker similarity. We investigate the impact of pretraining, branch combination methods, discriminator design, and VQ bottlenecks through detailed ablations. We also provide a theoretical justification for our architecture, showing that it can be viewed as an approximate solution to a maximum a posteriori (MAP) inference of clean speech given a noisy input, underpinning the theoretical soundness of our approach.

## Contributions

The main contributions of this thesis are as follows:

- A novel dual-branch neural audio codec architecture for unsupervised speech enhancement, enabling clean/noise decomposition without paired data.
- Adversarial branch-specific discriminators and a reconstruction consistency constraint to guide the learning of clean and noise components.
- A comprehensive experimental evaluation across supervised, semi-supervised, and unsupervised regimes, including ablations and real-world scenarios.

## Thesis Outline

The remainder of this thesis is organized as follows:

- **Chapter 2** introduces the speech enhancement task and prior supervised and unsupervised approaches.
- **Chapter 3** lays down the foundations of neural audio codecs and its components.
- **Chapter 4** presents the architecture, loss functions, and theoretical motivation behind our proposed method.
- **Chapter 5** describes used datasets, implementation and training details.
- **Chapter 6** reports extensive experimental results, including baselines, ablations, and analysis of supervised and unsupervised variants.
- **Chapter 7** concludes with a summary of findings and outlines directions for future work.

# Chapter 2

# Speech Enhancement

This chapter introduces SE background with basic definitions of noises, reverberated signals, and the SE task itself. It also describes prior supervised and unsupervised SE approaches.

## 2.1 Background

Whether it is a noise created by the recording device, environmental background, bandwidth limitation or packet loss, these (and many more) factors corrupt speech audio signal. Noises can be divided into two main categories:

1. Stationary - has constant statistical properties, i.e., $\mathbf{n} = (n_1, n_2, \ldots, n_T), n_t \sim p(\theta)$, usually caused by electronic device imperfections (e.g. Gaussian or Brown noise) or devices like air conditioning,
2. Non-stationary - statistical properties change over time, i.e., $\mathbf{n} = (n_1, n_2, \ldots, n_T)$, $n_t \sim p(\theta(t))$, majority of urban noises (e.g. sirene, wind, music, background speech).

Furthermore, indoor audio usually gets reverberated as a consequence of waveform bouncing of the surrounding walls and other objects, causing the reflected signal to be captured by a recording device with small delay, which can be mathematically defined as:

$$\mathbf{x}'(t) = \mathbf{x}(t) + \sum_{i=1}^{\infty} \mathbf{x}(t - \tau_i)\alpha_i, \tag{2.1}$$

where $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^T$ is the original and reverberated signal respectively, $\tau_i$ is the $i-$th delay and $\alpha_i \in \mathbb{R}$ is the attenuation factor.

As a result of the aforementioned signal corruptions, speech technology performance rapidly decreases, making it barely usable in challenging environments.

## 2.2 Task Definition

Speech enhancement is a speech processing task consisting in estimating clean speech signal from input noisy signal. More formally, let the noisy input signal be defined as:

$$\mathbf{y} = (\mathbf{x} + \mathbf{n}) \star \mathbf{r}, \tag{2.2}$$

where $\mathbf{x}$ is a clean speech signal, $\mathbf{n}$ is arbitrary noise (can be non-stationary), $\mathbf{r}$ is a room impulse response (RIR), $\mathbf{y}$ denotes the noisy signal, all defined as sequences of real numbers, and $\star$ is a convolution operator. The goal of a speech enhancement model is to estimate $\mathbf{x}$ given a noisy recording $\mathbf{y}$ with unknown ground truth noise, nor RIR, which can be mathematically formulated as:

$$f^* = \arg\min_{f \in \mathcal{F}} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}} d(\mathbf{x}, f(\mathbf{y})), \tag{2.3}$$

$$f^* = f_{\theta^*}, \theta^* = \arg\min_{\theta \in \Theta} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}} d(\mathbf{x}, f_\theta(\mathbf{y})), \tag{2.4}$$

$$\hat{\mathbf{x}} = f^*(\mathbf{y}), \tag{2.5}$$

where $d$ is some distance metric (i.e. $L_1$ norm of a difference of the ground truth clean speech signal and the estimated clean speech signal), $f$ is an enhancement model, and $\mathcal{F}$ is a family of functions we consider during optimization, and $\mathcal{D}$ is a dataset of paired examples. Usually, we parametrize the function $f$ by parameters $\theta$ from a set of possible parameters $\Theta$, e.g., a transformer neural network with pre-determined hyperparameters. Also, another plausible definition includes probability densities and maximum a posteriori (MAP) estimation and can be defined as:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}' \in \mathbb{R}^T} p(\mathbf{x}'|\mathbf{y}), \tag{2.6}$$

meaning that we are trying to find the most probable clean speech signal given the noisy input.

## 2.3 Prior Approaches

We provide an overview of prior approaches to speech enhancement, first focusing on traditional methods that predate deep learning. Then, we first describe supervised approaches, which are more common in the literature, and then we focus on unsupervised approaches, which are the main focus of this thesis.

### 2.3.1 Traditional Pre-Deep-Learning Methods

Speech enhancement has been approached through a range of methods, each with distinct theoretical foundations and practical trade-offs. One of the earliest and most widely used techniques is the Wiener filter [36], which estimates the clean speech signal by minimizing the mean-square error between the noisy observation and the desired signal in the frequency domain. While effective at high signal-to-noise ratios (SNRs), the Wiener filter can introduce distortion at low SNRs due to its reliance on accurate a priori SNR estimates. The authors of [7] later introduced spectral subtraction as a computationally efficient alternative that estimates the noise spectrum during nonspeech segments and subtracts it from the noisy signal. Although simple and suitable for real-time applications, this method often introduces various noise artifacts. Building upon these earlier approaches, [15] proposed a statistically optimal method that directly estimates the short-time spectral amplitude (STSA), which is more perceptually relevant than the full complex spectrum estimated by the Wiener filter. By modeling speech and noise as independent Gaussian variables, their MMSE-STSA estimator achieved improved noise suppression and reduced speech distortion, particularly under uncertain speech presence. More recently, [33] combined spectral subtraction and adaptive wavelet thresholding in the wavelet domain by applying spectral subtraction to low-frequency bands and wavelet-based denoising to high-frequency components. Their hybrid method effectively mitigates both noise artifacts and over-smoothing, demonstrating improved performance in non-stationary acoustic environments.

### 2.3.2 Supervised Deep Learning Methods

More recent work has moved beyond traditional signal processing approaches, with a growing focus on data-driven methods based on deep learning. Initial efforts in this direction employed fully connected neural networks trained to predict clean spectral magnitudes from noisy inputs [82]. These models were later extended with recurrent architectures, such as long short-term memory (LSTM) networks, which enabled the modeling of temporal dependencies in speech [76]. The incorporation of phase-sensitive loss functions and time-frequency masking further improved performance, particularly in terms of intelligibility and automatic speech recognition robustness. However, these models typically required more parameters and were computationally intensive, limiting their applicability in low-resource or real-time scenarios.

To address these limitations, convolutional approaches gained popularity. The Redundant Convolutional Encoder-Decoder (R-CED) network introduced a fully convolutional neural network with skip connections and no pooling layers, enabling efficient spectral mapping with a relatively small number of parameters [44]. This made it more suitable for deployment in resource-constrained environments. Similarly, the Temporal Convolutional Neural Network (TCNN) [43] employed causal and dilated convolutions to maintain long receptive fields without the need for recurrence. Its encoder-decoder structure, combined with a temporal convolutional module, supported low-latency inference while maintaining strong enhancement performance.

Furthermore, Transformer-based models have also been explored for speech enhancement. The Two-Stage Transformer Neural Network (TSTNN) proposed a modular architecture that integrates local and global contextual modeling using transformer blocks and gated recurrent units (GRUs) [75]. This design allowed the model to efficiently process long-range dependencies in the time domain, with moderate computational complexity.

Parallel to these developments, generative models began to be applied to speech enhancement. One of the earlier examples, SEGAN [46], used a generative adversarial framework operating directly on raw waveforms to map noisy speech to cleaner outputs without relying on spectral representations. While this end-to-end approach showed the potential of generative modeling in the time domain, it faced challenges with training stability and noise generalization. MetricGAN [18, 19] introduced a different generative formulation, in which the discriminator does not perform binary classification but instead learns to approximate evaluation metrics such as PESQ. This enables the generator to optimize for perceptual quality more directly. Although MetricGAN's training follows an adversarial structure, its objective is closely tied to metric regression, distinguishing it from classical GAN setups focused on data distribution matching.

More recently, diffusion-based models have offered a principled probabilistic approach to generative speech enhancement. The Score-based Generative Model for Speech Enhancement (SGMSE+) [56] models the forward corruption process as a stochastic differential equation (SDE) in the complex STFT domain and learns the reverse process via a noise-conditional score network. The model generates enhanced speech through iterative sampling and has demonstrated effective denoising and dereverberation capabilities across a variety of noise conditions. Its training does not rely on metric-based losses or adversarial objectives, but rather on score matching, aligning it more closely with probabilistic inference frameworks. Evaluation on several benchmarks has shown that diffusion-based models like SGMSE+ provide competitive or superior results compared to both discriminative and earlier generative methods.

Overall, these developments represent a progression from deterministic signal estimators toward probabilistic models that aim to capture the underlying structure of clean speech. This shift has contributed to improvements in enhancement robustness, generalization to unseen conditions, and perceptual quality.

### 2.3.3 Unsupervised Enhancement Methods

While supervised deep learning methods have achieved remarkable performance, they remain constrained by their dependence on paired clean and noisy data. This limitation has prompted increasing interest in unsupervised approaches that eliminate the need for clean references, instead relying on the structure of speech, self-supervision, or generative modeling.

One prominent line of research in unsupervised speech enhancement utilizes deep probabilistic generative models trained only on clean speech. Dynamical Variational Autoencoders (DVAEs) [6] model clean speech as a sequence of latent variables with temporal dependencies, enabling more accurate modeling of speech dynamics than standard Variational Autoencoders (VAEs). The DVAE acts as a prior during inference, where it is combined with a Nonnegative Matrix Factorization (NMF)-based noise model within a variational Expectation-Maximization (EM) framework to perform unsupervised separation of clean speech from noisy mixtures. This structure allows the use of only clean speech data during training, while still enabling enhancement on real-world noisy inputs. To extend this capability, later work by the authors of [34] replaced the handcrafted NMF noise model with a trainable Deep Dynamical Generative Model (DDGM), which more flexibly captures the structure of realistic noise. This shift toward fully learnable components improves generalization and enables both noise-agnostic and noise-specific configurations with faster inference.

Complementing these probabilistic models, adversarial learning techniques have been adapted to the unsupervised setting. MetricGAN-U [20] introduces a novel formulation where the GAN discriminator is trained not to distinguish between real and fake samples, but instead to approximate a perceptual quality metric such as DNSMOS or SRMR. The generator is then optimized to improve the discriminator's predicted score, effectively enhancing the speech signal by maximizing its estimated perceptual quality. Crucially, this setup allows training on noisy speech alone, using the learned metric proxy as supervision, and supports enhancement tasks such as dereverberation in addition to denoising.

Another class of unsupervised methods leverages unpaired datasets through cycle-consistent architectures [80]. CycleGAN-based enhancement models train two generator-discriminator pairs with a cycle-consistency loss to ensure that mappings from noisy to clean speech and back preserve content. However, early versions suffered from unstable clean-to-noisy mapping. This issue was addressed through Noise-Informed Training (NIT) [67], which augments the input with noise-type labels to guide the generation of more realistic noisy signals. This extension improved the consistency and stability of the training cycle, leading to better generalization on unseen noise types and more reliable enhancements.

Other approaches have explored alternative formulations of the training process itself. The Noisy-target Training (NyTT) [21, 22] method trains a model to predict a moderately noisy signal from a more severely degraded version, using only noisy data. A more noisy version is synthesized by adding additional noise to the input, which then serves as the target. This setup exploits the assumption that the clean speech is shared across both versions, enabling the model to learn to suppress the added noise. Despite its simplicity, NyTT has shown strong performance, especially under domain mismatch conditions where traditional supervised models tend to fail.

Further expanding on self-supervised strategies, TF-Cycle-MixIT [73] addresses the challenge of enhancing extremely weak speech signals under severe noise. It integrates Mixture Invariant Training (MixIT) [78], a method for source separation without clean references, with harmonic-aware features and time-frequency fusion. The system operates in a cycle-consistent training loop and employs periodic pseudo-label refinement to stabilize learning. By leveraging harmonic structures common in voiced speech, the model can retain intelligibility even when the signal-to-noise ratio is very low. This makes it particularly effective in harsh conditions such as those found in distributed acoustic sensing or industrial monitoring.

Another recent line of work is QMixCAT [74], which aligns with teacher-student and pseudo-labeling strategies found in other self-supervised learning domains. It simulates a supervised training setup using only noisy data by employing quality-guided pseudo-labeling and a competitive teacher-student framework. Noisy mixtures are scored using a perceptual quality estimator to select high-confidence pseudo-labels, which are used to supervise a student model. A Competitive Alternating Training (CAT) strategy then updates both the student and teacher networks iteratively. This approach complements prior methods such as TF-Cycle-MixIT and MetricGAN-U by focusing on perceptual guidance and dynamic model refinement, demonstrating strong performance without requiring clean targets

Collectively, these methods illustrate a clear shift from reliance on parallel data toward more flexible and data-efficient learning frameworks. Through a variety of architectures and training strategies—probabilistic modeling, adversarial learning, and self-supervised objectives—unsupervised speech enhancement has emerged as a viable and robust alternative to traditional supervised methods, capable of meeting the demands of real-world applications.

# Chapter 3

# Neural Audio Codecs

Learning neural representations of input data remains the main challenge in the field of machine learning. Until recently, most neural representation models operated in continuous latent vector spaces. However, many modalities are inherently discrete, including speech — discrete spoken words with a finite number of speakers that can be characterized by a combination of discrete factors (e.g., gender, emotion, age, $F_0$, to name a few). Hence, producing discrete representations seems better to fit the nature of the true latent speech structure.

Discrete representation models, first successfully introduced as VQ-VAE [42], and later by the authors of VQ-GAN [16], proved that discrete representations are enough to match the performance of continuous counterparts when performing reconstruction and generation, and sparked an interest in learning discrete representations. Furthermore, the authors also proved that learning discrete representation for speech resulted in implicit phoneme-like structures in the representations, proving that the model is indeed learning the important higher-level latent structures. These architectures, among architectures like MelGAN [28] or HiFiGan [27], inspired the birth of end-to-end neural audio codecs (NACs). One of the first architectures of modern NAC is considered to be SoundStream [84], which was later followed by Encodec [12] and many others.

Neural Audio Codec is a neural network-based model that takes audio as input, encodes it into a sequence of discrete representations (tokens), and transforms these tokens back to an output audio that resembles the input as accurately as possible. As depicted in Figure 3.1, the model consists of three main parts:

1. Encoder - Usually a convolutional neural network that encodes input audio into down-sampled high-dimensional continuous representations.
2. Quantization - A clustering module that converts continuous representations into discrete representations with maximum accuracy.
3. Decoder - Usually a neural network consisting of transposed convolutions that decodes discretized latent representations back to a waveform.

## 3.1 Encoder and Decoder

The encoder and decoder are the main parts of the autoencoder network that, along with the quantization module, define the architecture of NAC models. The difference between

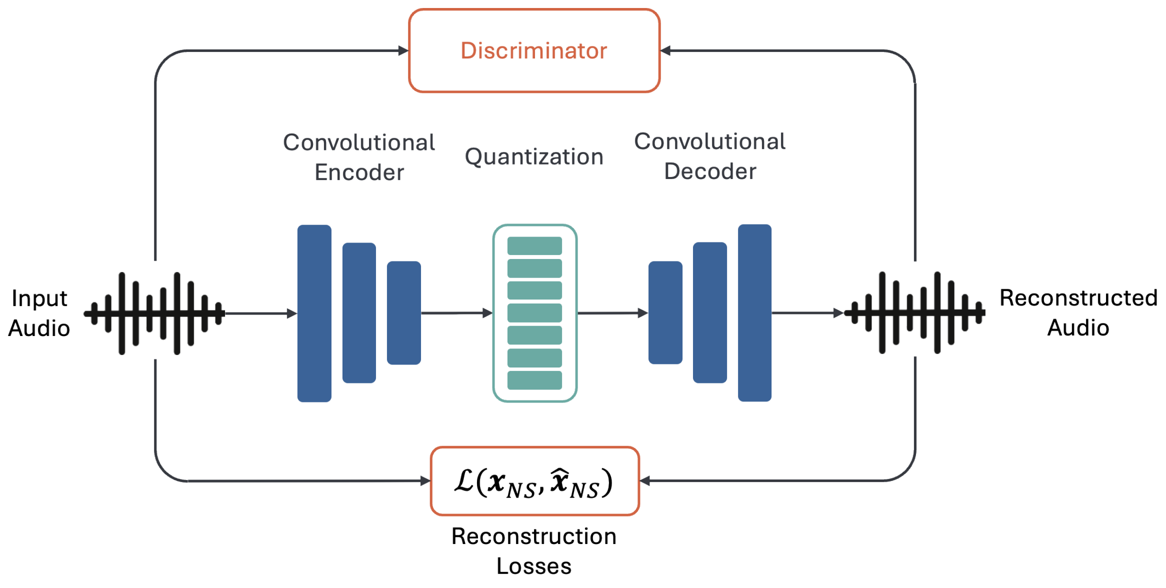
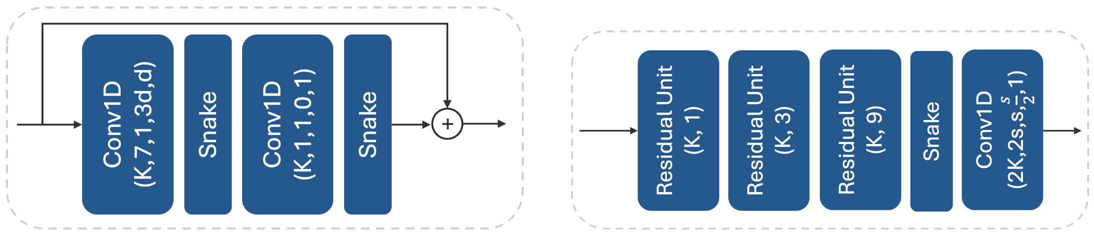


Figure 3.1: Neural Audio Codec schema.

the encoder and the decoder is mainly in the decoder block, which has similar architecture to the encoder block but the last convolutional layer is transposed and placed before the residual units to increase the time resolution and decrease the number of channels (see Figure 3.2b); hence, we describe only the encoder architecture. As we base all our models on Descript Audio Codec (DAC) [29], we tailor the description to the specifics of DAC architecture. However, other convolutional NACs are similar.



(a) Schema of one residual unit.

(b) Schema of one encoder block.

Figure 3.2: Schema of NAC encoder building blocks. Conv1D parameters are: (number of kernel filters, kernel size, stride, padding, dilation), respectively, and the residual unit parameters are: (number of kernels, dilation, and stride).

The encoder consists of a 1D convolution layer with $d$ channels, kernel size of 7, and padding 3 that transforms single-channel waveform to a $d$-channel signal (dictates the latent space dimensionality). Then it is followed by a sequence of encoder blocks (Figure 3.2b), each consisting of 3 residual units (Figure 3.2a) followed by a Snake activation function, which is defined as $f(x) = x + \frac{1}{\alpha}\sin^2(\alpha x)$ and was proposed by [87] to inject a periodic inductive bias into the model. The authors of DAC [29] observed that it results in improved reconstruction audio quality. The last building block of the encoder block is a 1D convolutional layer with stride $s$ reducing the time resolution by a factor of $s$ and increasing the number of channels twice.

### 3.1.1 Normalization

Normalization is a crucial element of most modern neural networks that significantly improves the stability of the training process. The authors of MelGAN [28] observed that the choice of normalization technique was crucial for audio quality. The main problem was rooted in normalization techniques that alter the distribution of the activation functions (i.e., normalize mean), which can lead to removing important pitch information, making the audio sound more metallic. The best-performing technique was weight normalization, which re-parametrizes the weight matrices as:

$$\mathbf{w} = g\frac{\mathbf{v}}{\|\mathbf{v}\|}, \tag{3.1}$$

where $\mathbf{v}$ is the orignal weight matrix and $g \in \mathbb{R}$ defines the new norm of $\mathbf{w}$. Hence, it decouples the norm from the direction of the given parameter. It can be shown that this technique stabilizes the training process without altering internal activations by normalizing the gradients instead. For more information, we refer the reader to the original publication [62].

## 3.2 Transformer

Nowadays, the Transformer is a ubiquitous model used for almost all tasks and was first introduced in 2017 by the authors of [70]. For this work, we only introduce the self-attention, how to encode the positional information, and we will try to give intuition behind the use of Transformer in NAC models. For more details about the Transformer architecture, we refer the reader to the original paper.

Self-attention is a mechanism that increases the model's receptive field to infinity compared to fully convolutional networks by allowing it to combine information from all the input embeddings. For mathematical description, let $\mathbf{E} = (\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N) \in \mathbb{R}^{N\times d}$ be a sequence of $N$ embeddings of dimension $d$. We distinguish between three types of embeddings: query $\mathbf{Q}$, key $\mathbf{K}$, and value $\mathbf{V}$. In self-attention, all three are being computed from the same input sequence $\mathbf{E}$ by applying three different linear transformations as defined below:

$$\mathbf{Q} = \mathbf{E} \cdot W_Q, \quad \mathbf{K} = \mathbf{E} \cdot W_K, \quad \mathbf{V} = \ E \cdot W_V. \tag{3.2}$$

The attention scores are computed as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}. \tag{3.3}$$

It can be seen that the attention mechanism weights all the values by scores computed from the similarity of queries and keys, allowing the model to focus on the most relevant parts of the input but also to use information from the entire input sequence.

It is not hard to observe that self-attention is position invariant, meaning that the order of the input sequence does not matter. We need to inject positional information into the model to overcome this issue. The authors of [70] proposed to add absolute positional encodings to the input embeddings. However, it is not sufficient for audio compression because the position of some sound in the recording should not matter to the task itself. Hence, relative positional information is mostly used, informing the attention mechanism about the distance between the current position in the embedding sequence and the one it is attending to. For this purpose, this and also other works (e.g. [45]) use the rotary positional encoding proposed by [64].

Until recently, the majority of NAC models were fully convolutional. However, researchers started employing transformers in NAC models to increase the receptive field and to increase the efficiency (i.e., decrease the bitrate) [45, 79, 14]. Due to the infinite receptive field, the transformer-based NACs are able to model long-term dependencies in the audio signal and encode it more efficiently, which is beneficial not only for the higher compression rate but also for the subsequent audio language modeling these codecs are usually used for.

## 3.3 Quantization

Let $\mathbf{z} \in \mathbb{R}^{N \times d}$ be a d-dimensional vector sequence of length $N$. Furthermore, let $Q_q : \mathbb{R}^d \rightarrow \mathbb{N}$ be a quantizer, i.e., a function that converts a continuous (possibly high-dimensional) vector into a discrete number (token), and $Q_d : \mathbb{N} \rightarrow \mathbb{R}^d$ be a dequantizer, i.e., a function that converts a token back to a continuous real-valued vector. Then, we can define $Q = Q_d \circ Q_q$ as a quantization module.

The quantization module $Q$ usually stores a set of codes called codebook. It is a look-up table, where the key is an integer token ID and the value is a vector of $d$ dimensional real values (usually called a centroid or a prototype). The quantization process for an arbitrary input vector $\mathbf{x} \in \mathbb{R}^d$ can be defined as:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{c} \in \mathbf{C}} dist(\mathbf{x}, \mathbf{c}), \tag{3.4}$$

where $\mathbf{C} = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K) \in \mathbb{R}^{K \times d}$ represents a codebook and $dist$ is a distance function (usually Euclidean).

There are several approaches to finding a suitable codebook; however, all of them can be described in the following way: Let $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) \in \mathbb{R}^{N \times d}$ be a batch of $d$-dimensional vectors (i.e., a training set). Furthermore, let $Q_{\mathbf{C}}$ denote a quantization function with an internal codebook $\mathbf{C}$. Then, finding an optimal codebook can be defined as a minimization problem:

$$\mathbf{C} = \arg\min_{\mathbf{C}'} \frac{1}{N} \sum_{i=1}^{N} dist(\mathbf{x}_i, Q_{\mathbf{C}'}(\mathbf{x}_i)). \tag{3.5}$$

### 3.3.1 RVQ

Having only one codebook allows us to encode $\log_2(N)$ bits of information. However, if we want to increase the bit rate (especially to increase the quality of the audio produced by NAC), e.g., 80 bits, we would need to store $2^{80}$ unique codes. If we assume 512-dimensional centroids stored in 32-bit floating point numbers, we would need storage of size almost 5000 zeta bytes.

To overcome the foregoing issues, we use residual vector quantization (RVQ), which, instead of a single codebook, contains $L$ codebooks. As the name suggests, the RVQ procedure involves the iterative application of quantization to the preceding quantization errors. We can define the RVQ algorithm as follows:

**Algorithm 1:** Residual Vector Quantization

```
Input: $\mathbf{x}$ - $d$-dimensional vectors, $N_q$ - number of quantizers, $Q_i$ - quantization
       functions
Output: quantized embedding $\hat{\mathbf{x}}$
$\hat{\mathbf{x}} \leftarrow \mathbf{0}$
$\mathbf{e} \leftarrow \mathbf{x}$
for $i = 1$ to $N_q$ do
    $\mathbf{y} \leftarrow Q_i(\mathbf{e})$
    $\hat{\mathbf{x}} \mathrel{+}= \mathbf{y}$
    $\mathbf{e} \mathrel{-}= \mathbf{y}$
end for
return $\hat{\mathbf{x}}$
```

### 3.3.2 Update Methods

Several methods exist for updating the centroids, out of which EMA and gradient-based are most commonly used.

#### EMA

The exponential moving average (EMA) method determines the values of a codebook code as a moving average of data points that were quantized as the particular code during the forward pass (see Equation 3.4). EMA updates resemble the K-means algorithm; however, during neural network training, we use mini-batches instead of the whole training set; hence, we need to update the centroids gradually. More formally, let $\{\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \ldots, \mathbf{x}_{i,n_i}\}$ be a set of embeddings in a mini-batch that were closest to the code $\mathbf{c}_i$ during the forward pass. We define a code value update as follows:

$$N_i^{(t+1)} = N_i^t \cdot \alpha + (1-\alpha) \cdot n_i \tag{3.6}$$

$$m_i^{(t+1)} = m_i^t \cdot \alpha + (1-\alpha) \sum_{j=1}^{n_i} \mathbf{x}_{i,j}, \tag{3.7}$$

$$\mathbf{c}_i^{(t+1)} = \frac{m_i^{(t+1)}}{N_i^{(t+1)}}, \tag{3.8}$$

where $\alpha \in [0,1]$ roughly determines the number of averaged elements. Usually, $\alpha$ is set to 0.99 during EMA updates.

Some code does not get updated because that particular code quantizes no vectors. To increase the codebook utilization, authors of the Encodec model [12] used random restarts—randomly re-initialize the corresponding centroid from a pre-defined distribution (usually Gaussian).

#### Gradient-based

Another popular method for updating codebooks consists of optimizing two losses:

$$\mathcal{L}_q(\mathbf{x}, \hat{\mathbf{x}}) = \underbrace{\|\mathrm{sg}[x] - \hat{x}\|_2^2}_{\mathcal{L}_{cb} \text{ codebook}} + \beta \underbrace{\|x - \mathrm{sg}[\hat{\mathbf{x}}]\|_2^2}_{\mathcal{L}_{cm} \text{ commitment}}, \tag{3.9}$$

where sg means stop-gradient and $\beta$ is a weighting constant that adjusts the importance of the particular loss parts, usually set to 0.25. While codebook loss updates the codebook's centroids to quantize the input as accurately as possible, the commitment loss serves as a regularization to force the encoder to output embeddings close to the codebook centroids (i.e., forces the encoder to stick to the codes the output got quantized to).

### 3.3.3 Curse of Dimensionality

To produce rich discrete input representations, we usually demand high codebook utilization (usually measured as codebook entropy). The codebook entropy is estimated by quantizing sufficiently large data and counting the number of times each code was used in each codebook. The final codebook entropy is calculated as an average of per-codebook entropies. Then, the codebook utilization is usually equal to the normalized entropy—i.e., dividing the entropy by $\log_2(N)$, where $N$ is the number of codes in the codebook.

As the encoder latent space is usually high-dimensional (1024+ dimensional), computing $L_2$ distance becomes less informative about the actual similarity between two embeddings. It has been proven that for some chosen point in a high-dimensional space, the difference between the nearest and the furthest point distances to the selected point converges to 0 in the limit of space dimensionality [1]. Also, the curse of dimensionality says that the volume of multi-dimensional space increases exponentially w.r.t. to the number of dimensions, meaning the latent space becomes sparsely populated. All these facts lead to low codebook entropy, which decreases the bandwidth of discrete representations and the reconstruction quality (we want the NAC to output as accurately reconstructed audio as possible).

To overcome the aforementioned issues, previous works applied linear down projection, performed the quantization in low-dimensional space, and projected up the quantized embeddings back to the original high-dimensional vector space, which can be defined as:

$$\hat{\mathbf{x}} = \mathbf{W}_u \cdot Q(\mathbf{W}_d \cdot \mathbf{x}), \tag{3.10}$$

where $\mathbf{W}_d \in \mathbb{R}^{d \times l}$ is a down-projection matrix, $\mathbf{W}_u \in \mathbb{R}^{l \times d}$ is an up-projection matrix, and $Q$ is a quantization function defined above, and $l$ is the quantization space dimensionality. $l = 8$ has shown to achieve the best compromise between the entropy of the codebook (0.99) and the reconstructed audio quality [29]. Intuitively, we can say that $W_d$ is going to be a PCA-like transformation [49], as we are minimizing $\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2$ and thus need to project the original vector space into a low-rank space where we can measure the embedding similarities more accurately (i.e., it does not make sense to keep dimensions with low variance, as they do not reflect embedding similarities in the original space).

## 3.4 Reconstruction Loss Functions

The reconstruction loss term is comprised of time and frequency domain loss terms. Let $\mathbf{x}, \hat{\mathbf{x}} \in \mathbb{R}^T$ be an input and an output audio signal, respectively. The time-domain loss term is defined as:

$$\mathcal{L}_t(\mathbf{x}, \hat{\mathbf{x}}) = \|x - \hat{x}\|_1 . \tag{3.11}$$

The frequency-domain losses are defined below.

### 3.4.1 MultiScale Spectral Loss

Let $S_{n,h}(\mathbf{x})$ denote a spectrogram computed using short-time Fourier transform (STFT) with window size $n$ and hop length $h$ from some input signal $\mathbf{x}$. The multi-scale spectral loss is defined as:

$$\mathcal{L}_{fs}(\mathbf{x}, \hat{\mathbf{x}}) = \sum_{i=1}^{N_s} \left\| \log_{10} |S_{n_i,h_i}(\mathbf{x})| - \log_{10} |S_{n_i,h_i}(\hat{\mathbf{x}})| \right\|_1 , \tag{3.12}$$

where $N_s$ is the number of different scales/sizes (e.g., $n_i \in \{2048, 1024, 512\}$), and $h_i = \frac{n_i}{4}$. Here, we abuse the notation of $L_1$, as the $S_{n,h}$ returns a matrix and assume the metric is computed over all the elemnents of the matrix by flattening it.

### 3.4.2 MultiScale Mel Loss

Another frequency-domain loss uses a mel-spectrogram to reflect human auditory perception better. Let $M_{n,h,k}(x)$ denote a mel-spectrogram where $n, h, \mathbf{x}$ have the same meaning as previously stated, and $k$ is the number of mel bands. The (multi-scale) Mel-spectrogram loss is defined as:

$$\mathcal{L}_{fm}(\mathbf{x}, \hat{\mathbf{x}}) = \sum_{i=1}^{N_m} \left\| \log_{10}(M_{n_i,h_i,k_i}(\mathbf{x})) - \log_{10}(M_{n_i,h_i,k_i}(\hat{\mathbf{x}})) \right\|_1 . \tag{3.13}$$

## 3.5 GAN

In the generative adversarial network (GAN) paradigm, two networks called generator ($G$) and discriminator ($D$) are trained against each other (adversarially) [24]. The generator $G$ is trained to produce samples indistinguishable from the real samples from $p_{data}$, while the discriminator is trained to determine if a sample is real or generated. The adversarial training process optimizes the generator to minimize the Jensen-Shannon divergence between the distributions of real and generated samples. It has been proven that if this training process converges, and the generator is able to fool the discriminator well, the generator eventually produces samples from the training data distribution $p_{data}$ [24].

However, in the context of NAC models, we do not generate any data, as the codec is a deterministic function of the input. We rather use the adversarial training to increase the perceptual quality of the reconstructed audio—i.e., to guide the model towards focusing on the perceptually most important aspects instead of only satisfying the reconstruction losses [13, 28].

Least Squares generative adversarial network (LS-GAN) [40] is commonly used by the authors of previous work on audio generation instead of the vanilla GAN mainly due to the problem of vanishing gradients in the case of an overpowered generator or discriminator (i.e., generator perfectly fools discriminator or vice-versa). Compared to vanilla GANs, LS-GAN optimizes Pearson divergence instead of Jensen-Shannon divergence, but claim about the convergence described above still holds. In the descriptions below, we use $G$ and $D$ to represent generator and discriminator functions.

### 3.5.1 Discriminator

We first describe the discriminator, as it is later used in the definition of the generator losses. In the case of NACs, the discriminator consists of multiple sub-discriminators operating on various resolutions, periods, or frequency bands (see below). Each sub-discriminator is a convolutional neural network with linear output activation function, which outputs a sequence of scores that are pushed to being close to 0 in the case of generated audio and close to 1 in the case of real audio. The loss function used to train the discriminator is defined as:

$$\mathcal{L}_D = \sum_{k=1}^{K_D} \mathbb{E}_{x\sim p_{data}}[D_k(G(x))^2 + (1 - D_k(x))^2], \tag{3.14}$$

where $p_{data}$ is the training data distribution, $D_k$ is the $k$-th sub-discriminator, and $K_D$ is the number of sub-discriminators.

### Multi Period Discriminator

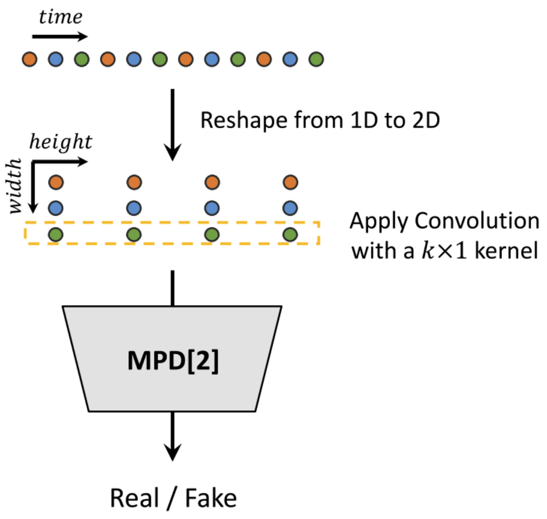


Figure 3.3: Multi Period Discriminator [27].

Multiperiod discriminator (MPD) was proposed by the authors of HiFi Gan [27] and is shown in Figure 3.3. It consists of five sub-discriminators, each focusing on a different portion of periodic input audio signals. It consists of a 6-layer 2D convolutional network with the number of channels $[32, 128, 512, 1024, 1024, 1]$ (the last convolutional layer pools across-channel information to a single channel) where each but the last convolution layer uses the LeakyReLU activation function, kernel size $(5, 1)$, stride $(3, 1)$ and padding $(2, 0)$ — i.e., the length of the input sequence is preserved. The periods for each discriminator are defined as: $p \in [2, 3, 5, 7, 11]$.

### Multi-band Complex Spectrogram Discriminator

To penalize phase shifts, the Multi-band Complex Spectrogram Discriminator (MCSD) utilizes the complex spectrogram that encodes magnitude in the real axis and phase in the imaginary axis. The architecture consists of three 2D convolutional layers, each having 32 kernels, stride 1, and padding 1, followed by the LeakyReLU [81] activation function. The complex spectrogram is treated as a matrix of tuples—real and imaginary axis (i.e., 2 channels). The last, 4$^{\text{th}}$, convolutional layer uses only one kernel and is added to obtain the sub-discriminator scores. The authors of DAC [29] found that splitting the spectrogram into sub-bands improves high-frequency prediction and mitigates aliasing artifacts; hence, each sub-discriminator is replicated (without weight sharing) and applied to the specific frequency band.

### 3.5.2 Generator

The generator in the context of NAC is the encoder-decoder model. The adversarial loss function is defined as:

$$\mathcal{L}_G = \sum_{k=1}^{K_D} \mathbb{E}_{x \sim p_{data}}[(1 - D_k(G(x)))^2], \tag{3.15}$$

where $p_{data}$ is the training data distribution. Furthermore, the generator is trained with an additional feature matching loss. The discriminator learns representations, based on which it is able to distinguish between real and generated audio. Thus, the $L_1$ norm of the difference between the discriminator representations of the NAC input and the generated audio can be viewed as a learned similarity metric we can use to provide the generator with enhanced guidance on how the real audio should sound. Let $F_{i,m}(\mathbf{x})$ represent a feature map produced by the $m$-th convolutional layer in the $i$-th discriminator. Then, the feature matching loss is defined as:

$$\mathcal{L}_{feat}(\mathbf{x}, \hat{\mathbf{x}}) = \sum_{k=1}^{K_D} \sum_{l=1}^{M_k - 1} \left\| F_{k,l}(\mathbf{x}) - F_{k,l}(\hat{\mathbf{x}}) \right\|_1, \tag{3.16}$$

where $M_k$ is the number of conv layers in the given discriminator (we are not using the output of the last convolutional layer as it is treated as the discriminator score and not an internal feature map). Also, when computing the feature matching loss, we do not update the discriminator parameters and only backpropagate through the discriminator to obtain the gradients for the generator parameters update.

## 3.6 Combined Loss

The overall loss function the generator is being optimized with is a weighted combination of the aforementioned losses. However, the authors of DAC [29] did not use the Multi-Scale spectral loss, which was used in prior works (e.g., [12]), but used the Multi-Scale Mel loss instead. The overall loss function is defined as:

$$\mathcal{L} = \lambda_t \mathcal{L}_t + \lambda_{fm} + \lambda_G \mathcal{L}_G + \lambda_{feat} \mathcal{L}_{feat} + \lambda_{cb} \mathcal{L}_{cb} + \lambda_{cm} \mathcal{L}_{cm}. \tag{3.17}$$

Given the number of the loss weights, it is important to tune them carefully or at least to estimate their dynamic range and set the weighs accordingly.

# Chapter 4

# Our Approach

This chapter outlines the details of our proposed methods, including their variations, in-depth description, and intuition behind each design choice. We first present a single-branch baseline model consisting of the convolutional encoder and decoder, transformer, and quantization. Afterward, we present a dual-branch model that extends the single-branch baseline with another parallel branch modeling residual noise, leading to a model that allows us to train SE unsupervised.

## 4.1 Single-branch

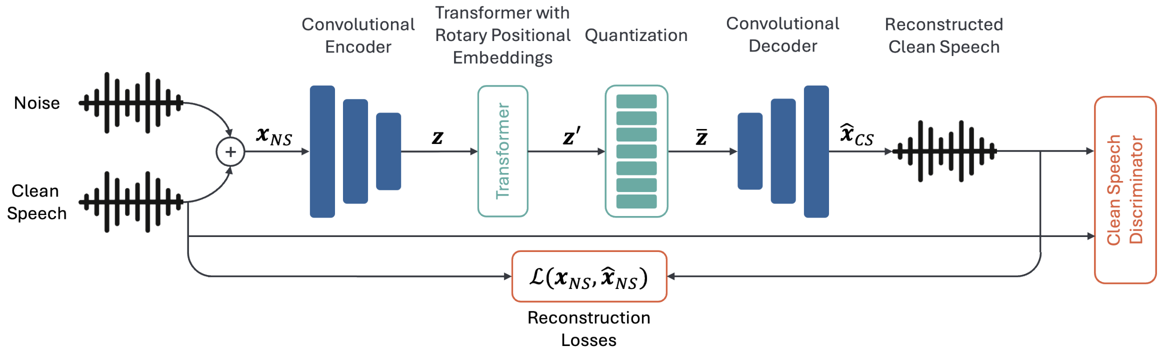


Figure 4.1: Single-branch speech enhancement model scheme.

Prior SE approaches (Section 2.3) have converged mainly to two paradigms: spectrogram-based methods and waveform-based methods. Traditional SE approaches typically estimate a time-frequency mask applied to input noisy spectrogram, while more recent approaches perform enhancement directly on waveforms, which is our case.

The single-branch enhancement model is essentially a NAC model that takes noisy mixture $\mathbf{x}_{NS} \in \mathbb{R}^T$, created as a sum of noise and clean speech as an input, and predicts clean speech (unlike NAC that predicts the input audio).

As shown in Figure 4.1, the noisy audio is passed to the convolutional encoder $\mathbf{z} = E(\mathbf{x}_{NS}) \in \mathbb{R}^{T' \times d}$, extracting local features from the input signal and encoding it into a high-dimensional latent sequence of real-valued vectors. In order to capture global information in the speech signal, like prosody or speaker characteristics, we further employ a transformer [71] model to increase the receptive field of the enhancement model to the entire sequence. We found out during our preliminary experimentation that adding positional information to the input sequence was crucial for model convergence, as a transformer model is invariant to the position in the input latent sequence, which indeed does not correspond to the nature of the audio signal for the enhancement task (i.e., we do want to enhance the input speech while retaining the order of spoken phonemes). Moreover, the absolute position in the sequence does not matter as much as the relative difference between the positions in the sequence (i.e., immediate surroundings of $i$-th element $\mathbf{z}[i]$ explain $\mathbf{z}[i]$ more than a distant element in the sequence $\mathbf{z}$); hence, we utilize relative positional information by using Rotary Positional Embeddings (RoPE) [64]. After the transformer with RoPE (RoFormer) processes the latent sequence $\mathbf{z}_{CS} = \mathcal{R}(\mathbf{z})$, we optionally quantize the embeddings using RVQ, producing quantized embeddings $\bar{\mathbf{z}}_{CS}$. Even though the quantization step is unnecessary, it allows us to control branch-wise bandwidth with dual-branch models described below. The last step of the pipeline is the decoding, which converts the latent sequence back to enhanced waveform $\hat{\mathbf{x}}_{CS}$.

### 4.1.1 Training

We follow the NAC training scheme utilizing both reconstruction and adversarial losses to train the model. However, in addition to the Multi-Scale Mel loss used during NAC training, we also optimize negative SI-SDR [59] (we want to maximize SI-SDR) defined as:

$$\text{SI-SDR}(\mathbf{x}, \hat{\mathbf{x}}) = 10 \log_{10} \left( \frac{\left\| \frac{\hat{\mathbf{x}}^T \mathbf{x}}{\|\mathbf{x}\|_2^2} \mathbf{x} \right\|_2^2}{\left\| \frac{\hat{\mathbf{x}}^T \mathbf{x}}{\|\mathbf{x}\|_2^2} \mathbf{x} - \hat{\mathbf{x}} \right\|_2^2} \right), \tag{4.1}$$

beacuse it resulted in faster model convergence and better speech quality in our preliminary experiments. The overall generator loss function is defined as:

$$\begin{aligned} \mathcal{L}(\mathbf{x}_{CS}, \hat{\mathbf{x}}_{CS}) = & -\lambda_s \text{SI-SDR}(\mathbf{x}_{CS}, \hat{\mathbf{x}}_{CS}) + \lambda_{fm} \mathcal{L}_{fm}(\mathbf{x}_{CS}, \hat{\mathbf{x}}_{CS}) + \\ & + \lambda_{feat} \mathcal{L}_{feat}(\mathbf{x}_{CS}, \hat{\mathbf{x}}_{CS}) + \lambda_G \mathcal{L}_G(\hat{\mathbf{x}}_{CS}) + \\ & + \lambda_{cb} \mathcal{L}_{cb}(\mathbf{z}_{CS}, \bar{\mathbf{z}}_{CS}) + \lambda_{cm} \mathcal{L}_{cm}(\mathbf{z}_{CS}, \bar{\mathbf{z}}_{CS}), \end{aligned} \tag{4.2}$$

where $\lambda_i \in \mathbb{R}_+$ is the particular loss weight, and the other losses are defined in Section 3.4.

Together with the generator, we also train the discriminator $D_{cs}$, which is trained to distinguish between real and generated clean speech audio by optimizing the $\mathcal{L}_D$ loss function defined in Section 3.5 and 3.3.

## 4.2 Dual-branch

The single-branch model alone might be enough for supervised SE but is not for leveraging unpaired data without any supervision available. Even though the architecture is GAN, it lacks the essential part in the case of unpaired data: consistency—i.e., ensuring the output is as close to the input as possible in terms of uttered words or speaker information.

Prior unsupervised SE approaches (Section 2.3) have typically relied on cycle-consistent adversarial training (e.g., CycleGAN-based SE), contrastive learning, or domain adaptation, often requiring complex architectures or additional regularization terms to enforce content preservation.

In contrast to prior work, we propose a more straightforward approach of dividing the latent sequence into two parallel branches, one modeling clean speech and the other noise,

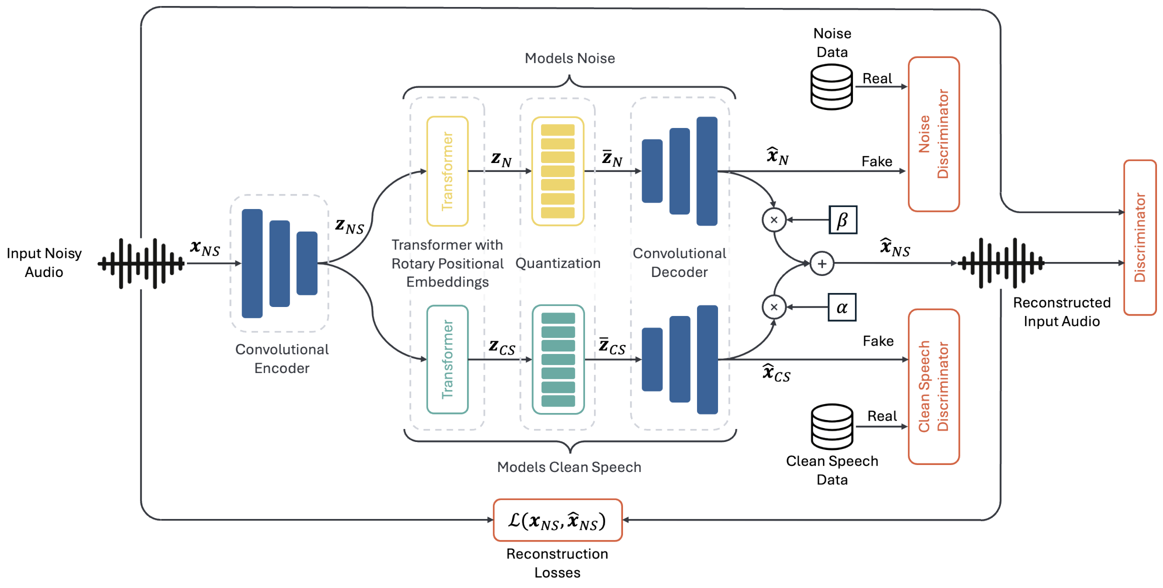


Figure 4.2: Dual-branch speech enhancement model scheme.

and forcing the combined branch output to be as close to the input as possible. The clean speech and noise discriminators ensure that the output of the particular branch follows the distribution of the corresponding data. Furthermore, forcing the combined output to resemble the input closely ensures consistency—i.e., that the produced clean speech is an enhanced version of the input, rather than some audio that sounds like a clean speech but does not correspond to the speech in the noisy input.

As shown in Figure 4.2, the model takes noisy speech $\mathbf{x}_{NS} \in \mathbb{R}^T$ as an input and passes it through the convolutional encoder $E : \mathbb{R}^T \to \mathbb{R}^{T' \times d}$ to obtain a latent sequence $\mathbf{z}_{NS}$. Then, we apply two RoFormers in parallel, producing two latent sequences $\mathbf{z}_{CS} = \mathcal{R}_{CS}(\mathbf{z}_{NS})$ and $\mathbf{z}_N = \mathcal{R}_N(\mathbf{z}_{NS})$, denoting clean speech and noise respectively. Then, we optionally quantize the RoFormer outputs using RVQ, producing quantized embeddings $\bar{\mathbf{z}}_{CS}$ and $\bar{\mathbf{z}}_N$. The last step of the pipeline is the decoding, which converts the latent sequence back to waveform using a shared convolutional decoder $D : \mathbb{R}^{T' \times d} \to \mathbb{R}^T$: $\hat{\mathbf{x}}_{CS} = D(\bar{\mathbf{z}}_{CS})$ and $\hat{\mathbf{x}}_N = D(\bar{\mathbf{z}}_N)$. Finally, we combine the two outputs to obtain the noisy output $\hat{\mathbf{x}}_{NS} = \alpha \cdot \hat{\mathbf{x}}_{CS} + \beta \cdot \hat{\mathbf{x}}_N$, where $\alpha, \beta \in \mathbb{R}$ are scalars defined in Section 4.2.1. If we look at the model as a whole, it can be seen as a NAC model that reconstructs the input noisy audio $\mathbf{x}_{NS}$, while the branch-wise outputs $\hat{\mathbf{x}}_{CS}$ and $\hat{\mathbf{x}}_N$ are used to perform source separation.

### Discriminators

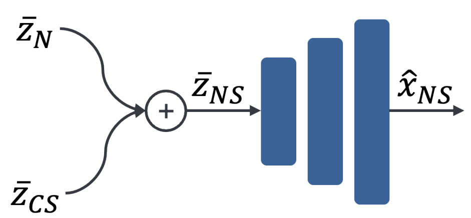
Reconstruction alone is not enough to perform the enhancement, and we need extra supervision to force each branch to perform distinct tasks. We employ two branch-specific discriminators, $D_{CS}$ and $D_N$, trained to distinguish between real and generated clean speech and, and real and generated noise, respectively. As described in Section 3.5.1, our architecture employs least square GAN losses and adversarial training results in minimizing Pearson divergence $\chi^2_{pearson}\left(p_{CS} \parallel \frac{p_{CS}+p_{clean}}{2}\right)$, where $p_{CS}$ is the distribution of the CS branch output, and $p_{clean}$ is the true clean speech distribution (defined by the pool of clean speech data). Anlogous statements are held for the noise branch as well. After the convergence, each branch should produce samples from the corresponding distribution, i.e., $p_{CS} \approx p_{clean}$

and $p_N \approx p_{noise}$, while adhering to the consistency constraint posed by the reconstruction loss.

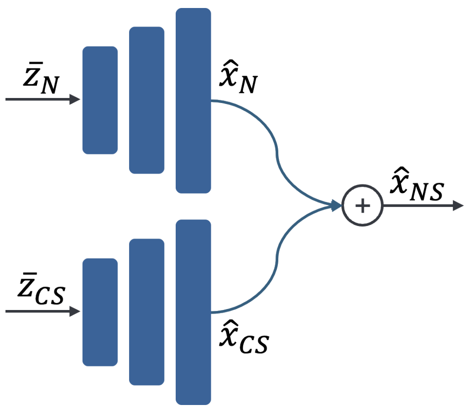
Furthermore, we employ the third discriminator $D_{NS}$ to enhance the quality of the reconstructed speech, consequently enhancing the quality of the audio each branch estimates.

### 4.2.1 Branch Combination Methods

In this work, we explored two main approaches to branch information combination: a combination of branch-wise latent spaces followed by decoding the noisy waveform, and first decoding each branch and then combining the raw waveforms (as already introduced in Section 4.2 and Figure 4.2). We discuss the pros and cons of each of the approaches below.



(a) Latent space combination followed by noisy speech decoding.



(b) Branch-wise latent space decoding followed by waveform combination

Figure 4.3: Branch combination methods: latent space combination, and waveform combination.

#### Latent Space

As Figure 4.3a shows, latent space combination consists in combining noise $\bar{\mathbf{z}}_N$ and clean speech $\bar{\mathbf{z}}_{CS}$ latent sequences into a noisy speech latent sequence $\bar{\mathbf{z}}_{NS}$ and subsequent decoding to obtain noisy waveform $\hat{\mathbf{x}}_{NS}$. In general, we can describe it mathematically as:

$$\hat{\mathbf{x}}_{NS} = D(c(a(\bar{\mathbf{z}}_N), b(\bar{\mathbf{z}}_{CS}))), \tag{4.3}$$

where $a, b : \mathbb{R}^{T' \times d} \to \mathbb{R}^{T' \times d}$ are branch-wise processing functions preparing the latent sequences for combination (i.e., projecting the latent sequences to $K$-dimensional space), $c : \mathbb{R}^{(T' \times d) \times (T' \times d)} \to \mathbb{R}^{T' \times d}$ is the combination function that takes two latent sequences as an input and produces one latent sequence as an output, and $D$ is the decoder that converts latent sequence to a waveform. Notably, simple $a = id, b = id, c(\mathbf{x}, \mathbf{y}) = \frac{1}{2}(\mathbf{x} + \mathbf{y})$, where $id$ is an identity function, works well.

The main advantage of this approach is that we work with latent sequences that represent a subsampled version of the input signal instead of waveforms directly. Also, we leverage the power of convolutional decoder $D$ to process the combined latent sequence before outputting audio (i.e., amplify clean speech or attenuate noise).

On the other hand, combining latent spaces does not directly ensure that the information is complementary between the two branches, nor does it directly push the model towards outputting clean speech consistent with the input noisy speech.

### Waveform

Compared to the latent space combination, as shown in Figure 4.3b, we directly decode both branches into two waveforms: clean speech $\hat{\mathbf{x}}_{CS} = D(\bar{\mathbf{z}}_{CS})$ and residual noise $\hat{\mathbf{x}}_N = D(\bar{\mathbf{z}}_N)$. Then, we combine the two waveforms by summing them. However, pure summation can cause penalizing each branch for incorrectly estimating the amplitude. Therefore, we multiply each branch output by scalars $\alpha, \beta \in \mathbb{R}$, resulting in an estimated noisy input $\hat{\mathbf{x}}_{NS} = \alpha\hat{\mathbf{x}}_{CS} + \beta\hat{\mathbf{x}}_N$. The scalars are estimated by solving the following optimization problem:

$$\alpha, \beta = \arg\min_{\alpha', \beta'} \left\|\mathbf{x}_{NS} - (\alpha'\hat{\mathbf{x}}_{CS} + \beta'\hat{\mathbf{x}}_N)\right\|_2^2, \tag{4.4}$$

and the solution is derived in Appendix A.

The main advantage of waveform combination compared to latent space combination is that it forces each branch to learn complementary information in the time domain. For example, if the noisy branch estimates noise only, the clean speech branch must estimate the corresponding clean speech. Otherwise, the combined noisy speech would not resemble the input noisy speech well during the noisy speech reconstruction. Hence, it allows us to use noisy speech reconstruction to ensure consistency.

However, suppose the noise is not perfectly estimated. In that case, the clean speech is going to be incentivized to estimate some of the noise (i.e., leak noise) to please the input audio reconstruction losses, which puts more weight on the discriminator to punish the enhanced speech for such leakage and forces it not to leak any noise in, potentially requiring advanced discriminator design.

## 4.2.2 Training

We extend the single-branch training scheme by adding the input audio reconstruction losses and adversarial losses from $D_{NS}$ and $D_N$. The overall loss function is defined as:

$$\begin{aligned}\mathcal{L}_{CS}(\mathbf{x}_{CS}, \hat{\mathbf{x}}_{CS}) = -\lambda_{CS,s}\text{SI-SDR}(\mathbf{x}_{CS}, \hat{\mathbf{x}}_{CS}) + \lambda_{CS,fm}\mathcal{L}_{fm}(\mathbf{x}_{CS}, \hat{\mathbf{x}}_{CS}) +\\ + \lambda_{CS,feat}\mathcal{L}_{feat}(\mathbf{x}_{CS}, \hat{\mathbf{x}}_{CS}) + \lambda_{CS,G}\mathcal{L}_G(\hat{\mathbf{x}}_{CS}) +\\ + \lambda_{CS,cb}\mathcal{L}_{cb}(\mathbf{z}_{CS}, \bar{\mathbf{z}}_{CS}) + \lambda_{CS,cm}\mathcal{L}_{cm}(\mathbf{z}_{CS}, \bar{\mathbf{z}}_{CS})\\ + \lambda_{CS,zm}\mathcal{L}_{zm}(\hat{\mathbf{x}}_{CS}) + \lambda_{CS,Emax}\mathcal{L}_{Emax}(\hat{\mathbf{x}}_{CS}),\end{aligned} \tag{4.5}$$

$$\begin{aligned}\mathcal{L}_N(\mathbf{x}_N, \hat{\mathbf{x}}_N) = \lambda_{N,feat}\mathcal{L}_{feat}(\mathbf{x}_N, \hat{\mathbf{x}}_N) + \lambda_{N,G}\mathcal{L}_G(\hat{\mathbf{x}}_N) +\\ + \lambda_{N,cb}\mathcal{L}_{cb}(\mathbf{z}_N, \bar{\mathbf{z}}_N) + \lambda_{N,cm}\mathcal{L}_{cm}(\mathbf{z}_N, \bar{\mathbf{z}}_N),\end{aligned} \tag{4.6}$$

$$\begin{aligned}\mathcal{L}_{NS}(\mathbf{x}_{NS}, \hat{\mathbf{x}}_{NS}) = -\lambda_{NS,s}\text{SI-SDR}(\mathbf{x}_{NS}, \hat{\mathbf{x}}_{NS}) + \lambda_{NS,fm}\mathcal{L}_{fm}(\mathbf{x}_{NS}, \hat{\mathbf{x}}_{NS}) +\\ + \lambda_{NS,feat}\mathcal{L}_{feat}(\mathbf{x}_{NS}, \hat{\mathbf{x}}_{NS}) + \lambda_{NS,G}\mathcal{L}_G(\hat{\mathbf{x}}_{NS}),\end{aligned} \tag{4.7}$$

where $\lambda_{i,j} \in \mathbb{R}_+$ is the particular loss weight, and the other losses are defined in Section 3.4.

Furthermore, during our preliminary experiments, we observed that some models started producing clean speech and noise waveforms with gradually increasing DC offset (i.e., non-zero mean), which resulted in the model collapsing and producing non-zero mean silence as clean speech due to numerical instabilities. This behaviour was observed mainly during the unsupervised training. Therefore, to prevent the collapse, we add a regularization term that penalizes the model for such behavior by employing the following loss:

$$\mathcal{L}_{zm}(\mathbf{x}) = \left| \frac{1}{T} \sum_{i=1}^{T} \mathbf{x}[i] \right| . \tag{4.8}$$

Also, to prevent branches from collapsing and outputting silence, we maximize the energy of the clean speech branch output by minimizing:

$$\mathcal{L}_{Emax}(\mathbf{x}) = -\log \left( \frac{1}{T' \cdot F} \sum_{t=1}^{T'} \sum_{f=1}^{F} |X(t,f)|^2 \right) , \tag{4.9}$$

where $X(t,f) \in \mathbb{C}$ is the STFT (window length 25ms, hop length 10ms) of the signal $\mathbf{x}$.

The final loss is just a summation of all the three loss classes defined by equations 4.5-4.7. However, we do not have paired clean speech, noise, and noisy speech data during unsupervised training. Hence, we cannot use any loss terms, requiring the ground truth clean speech and noise. Therefore, the $\mathcal{L}_{CS}$ and $\mathcal{L}_{N}$ loss functions consist only from $\mathcal{L}_G, \mathcal{L}_{cb}, \mathcal{L}_{cm}$. The input reconstruction loss $\mathcal{L}_{NS}$ stays intact as it does not require any supervision.

Similarly to the single-branch model, we train the discriminators $D_{CS}, D_N, D_{NS}$ using the $\mathcal{L}_D$ loss function defined in Section 3.5 using some clean speech, noise (both unpaired), and input audio as real data samples.

### 4.2.3 Theoretical Justification

We now formally justify that our dual-branch model can be understood as an approximate solution to the maximum a posteriori (MAP) inference of clean speech given the noisy input, proving that the model can perform SE.

Let $\mathbf{x}_{NS} \in \mathbb{R}^T$ be noisy input audio, and let us assume that it is an unknown mixture of clean speech $\mathbf{x}_{CS} \in \mathbb{R}^T$ and noise $\mathbf{x}_N \in \mathbb{R}^T$, i.e., $\mathbf{x}_{NS} = \mathbf{x}_{CS} + \mathbf{x}_N$. Our goal is to find a function $f_\theta$, such that $f_\theta(\mathbf{x}_{NS}) \approx \mathbf{x}_{CS}$, or probabilistically:

$$f_\theta(\mathbf{x}_{NS}) = \underset{\mathbf{x}_{CS}}{\arg\max}\, p(\mathbf{x}_{CS}|\mathbf{x}_{NS}). \tag{4.10}$$

The problem is that we do not have access to the clean speech $\mathbf{x}_{CS}$ during unsupervised training, which makes finding $f_\theta$ by optimizing $\theta$ in a traditional (supervised) way impossible.

Instead of finding $f_\theta$, we transform the problem into finding a function $g_\phi(\mathbf{x}_{NS}) = (\hat{\mathbf{x}}_{CS}, \hat{\mathbf{x}}_N)$, such that $\hat{\mathbf{x}}_{CS} + \hat{\mathbf{x}}_N = \hat{\mathbf{x}}_{NS} \approx \mathbf{x}_{NS}$, where $\hat{\mathbf{x}}_{CS}, \hat{\mathbf{x}}_N \in \mathbb{R}^T$ are two signals (in our architecture produced by the two branches). To obtain back the function $f_\theta$ we initially aimed to find, we need to ensure that $\hat{\mathbf{x}}_{CS} \approx \mathbf{x}_{CS}$ and take the first output of the function $g_\phi$, i.e., $f_\theta(\mathbf{x}_{NS}) = g_\phi(\mathbf{x}_{NS})_1$.

To do so, let us define two probability densities of the particular signals: $\mathbf{x}_{CS} \sim p_{clean}, \mathbf{x}_N \sim p_{noise}$, and let us further assume that $\mathbf{x}_{CS} \perp \mathbf{x}_N$, i.e., the two signals are independent (the same holds for the outputs of $g_\phi$), which is an assumption made by many prior SE or speech separation methods [5, 8]. Also, assume that the outputs of $g_\phi$ satisfy:

$$\begin{gathered}\hat{\mathbf{x}}_{CS}, \hat{\mathbf{x}}_N = \arg\max_{\mathbf{x},\mathbf{x}'} p_{clean}(\mathbf{x}) \cdot p_{noise}(\mathbf{x}') \\ \text{subject to } \hat{\mathbf{x}}_{NS} = \hat{\mathbf{x}}_{CS} + \hat{\mathbf{x}}_N.\end{gathered} \tag{4.11}$$

Authors of [32] have shown that LS-GAN is mode-seeking, meaning that the generator tends to produce samples close to some data distribution mode. Hence, if we assume that $p_{clean}$ and $p_{noise}$ are unimodal, the adversarial training will push the model towards producing samples with maximized clean speech and noise densities.

Following the assumptions above and the consistency property, we can write the following:

$$p(\mathbf{x}_{NS}|\mathbf{x}_{CS}) = p_{noise}(\mathbf{x}_{NS} - \mathbf{x}_{CS}), \tag{4.12}$$

after which we immediately obtain:

$$p(\mathbf{x}_{CS}|\mathbf{x}_{NS}) \propto p(\mathbf{x}_{NS}, \mathbf{x}_{CS}) = p_{noise}(\mathbf{x}_{NS} - \mathbf{x}_{CS}) \cdot p_{clean}(\mathbf{x}_{CS}). \tag{4.13}$$

Hence, for a fixed $\mathbf{x}_{NS}$, if we assume that $\mathbf{x}_{NS} = \hat{\mathbf{x}}_{NS}$ and use our prior assumptions from Equation 4.11, we can write:

$$\begin{aligned}\arg\max_{\mathbf{y}} p(\mathbf{x}_{CS} = \mathbf{y}|\mathbf{x}_{NS}) &= \arg\max_{\mathbf{y}} p(\mathbf{x}_{CS} = \mathbf{y}|\hat{\mathbf{x}}_{NS}) \\ &= \arg\max_{\mathbf{y}} p_{noise}(\underbrace{\hat{\mathbf{x}}_{NS} - \mathbf{y}}_{\hat{\mathbf{x}}_N}) \cdot p_{clean}(\mathbf{y}).\end{aligned} \tag{4.14}$$

As the noisy input $\mathbf{x}_{NS}$ is fixed and we assume we have estimated a noise $\hat{\mathbf{x}}_N$ according to Equation 4.11, we can write:

$$\hat{\mathbf{x}}_{CS} = \arg\max_{\mathbf{y}} p(\mathbf{x}_{CS} = \mathbf{y}|\mathbf{x}_{NS}). \tag{4.15}$$

Therefore, we proved that under the assumptions above, $g_\phi$ is a proxy for finding the function $f_\theta$. This derivation confirms that the proposed dual-branch architecture, appropriate discriminator-based regularization, and waveform-level combination form a consistent and theoretically grounded approach to unsupervised speech enhancement.

# Chapter 5

# Experimental Setup

This chapter thoroughly describes the overall experimental setup, including datasets, preprocessing, model architecture details, hyperparameters, hardware, and implementation details.

## 5.1 Software and Hardware

We based our codebase on descript audio codec (DAC)[1], which is based on pure PyTorch 2.7 [47]. Furthermore, for audio processing, we use Librosa [41], NumPy [25], and Lhotse [88]. We utilize the newly created toolkit VERSA [63] for metrics computation.

We train all our models on Nvidia A40, A100, and H100 GPUs, utilizing distributed data parallel (DDP) when training on multiple GPUs. Lastly, we use automatic mixed precision (AMP) and train our models in bfloat16 to further optimize the training and torch just-in-time (JIT) model compilation[2] to optimize the CUDA computations even further.

## 5.2 Data

We conducted the majority of experiments using noisy data synthetically created on-the-fly using two data pools: clean speech and noise. For both training and validation, we followed the URGENT challenge [86] setup with one change: we did not use all the augmentation/distortion methods as the authors to limit our focus solely on the enhancement rather than audio inpainting or bandwidth extension to name a few. Furthermore, we resampled all audio samples to 16kHz and downmixed them to monochannel. Even though around 5% of the training data is 8kHz, we decided to keep these samples as it does not introduce any significant bias towards not focusing on higher frequencies.

### 5.2.1 Clean Speech

The overall clean speech dataset pool consists of the following datasets[3]:

- **LibriVox data from DNS5 challenge:** Audiobook recordings with sampling rates between 8 kHz and 48 kHz, totaling approximately 350 hours of speech. This dataset,

[1]https://github.com/descriptinc/descript-audio-codec
[2]https://docs.pytorch.org/tutorials/intermediate/torch_compile_tutorial.html
[3]The dataset language is English unless stated otherwise.

used in the 5th Deep Noise Suppression Challenge, comprises clean speech derived from public domain audiobooks, facilitating research in noise suppression and speech enhancement tasks [55].

- **LibriTTS reading speech:** Audiobook speech with sampling rates from 8 kHz to 24 kHz, totaling around 200 hours. LibriTTS is a multi-speaker English corpus designed for text-to-speech research, derived from the original LibriSpeech corpus with improved quality and prosody annotations [85].
- **VCTK reading speech:** Recordings of read speech from newspapers and other texts at 48 kHz sampling rate, approximately 80 hours in duration. The VCTK corpus includes speech data from 110 English speakers with various accents, each reading about 400 sentences, providing a diverse dataset for speech synthesis and recognition research [72].
- **WSJ reading speech:** Read Wall Street Journal news articles recorded at 16 kHz, totaling about 85 hours. The WSJ0 corpus contains recordings from 123 speakers reading excerpts from the Wall Street Journal, commonly used for training and evaluating speech recognition or speech separation systems [9].
- **EARS speech:** Studio-quality speech recordings at 48 kHz, totaling around 100 hours. The EARS dataset comprises expressive anechoic recordings from 107 speakers, covering various speaking styles and emotions, aimed at benchmarking speech enhancement and dereverberation methods [57].
- **Multilingual LibriSpeech (de, en, es, fr):** Audiobook recordings in multiple languages (German, English, Spanish, French), with sampling rates between 8 kHz and 48 kHz. Approximately 450 hours of transcribed speech (total dataset size: 48,600 hours). MLS is a large-scale multilingual corpus derived from LibriVox audiobooks, supporting research in automatic speech recognition across diverse languages [50].
- **CommonVoice 19.0 (de, en, es, fr, zh-CN):** Crowd-sourced voice recordings in several languages (German, English, Spanish, French, Mandarin Chinese) with sampling rates from 8 kHz to 48 kHz. Around 1300 hours of validated speech (total dataset size: 9500 hours). Common Voice is an open-source project by Mozilla, collecting diverse voice data to improve speech recognition technologies [2].

We further processed the speech data pool by removing non-speech and silence-dominated samples and applied DNSMOS [53]-based filtering to remove noisy and low-quality speech samples, the same way as the authors of the challenge.

### 5.2.2 Noise

Similar to clean speech, the overall noise dataset pool consists of multiple datasets described below:

- **Audioset + FreeSound noise in DNS5 challenge:** A diverse collection of noise recordings from YouTube (via AudioSet) and the FreeSound platform, encompassing various real-world acoustic environments. This dataset, totaling approximately 180 hours, was utilized in the 5th Deep Noise Suppression Challenge to train and evaluate noise suppression models [55].

- **WHAM! noise:** Ambient noise recordings captured in urban environments such as coffee shops, restaurants, and bars in the San Francisco Bay Area. These recordings, sampled at 48 kHz and totaling around 70 hours, were used to augment the WSJ0-2mix dataset, creating the WHAM! Corpus for evaluating speech separation in noisy conditions [77].
- **FSD50K (human voice filtered):** An open dataset comprising over 51,000 audio clips manually labeled across 200 classes derived from the AudioSet ontology. For noise-related tasks, clips containing human voice have been filtered out, resulting in approximately 100 hours of diverse sound events suitable for environmental sound classification and related research [17].
- **Free Music Archive (medium):** A curated collection of music tracks from the Free Music Archive, encompassing a wide range of genres and styles. The medium subset includes approximately 200 hours of audio, providing a rich resource for music information retrieval tasks and audio analysis [11].
- **Wind noise simulated by participants:** Synthetic wind noise samples generated using various simulation techniques. These samples, with variable sampling rates and durations, are employed to augment datasets for training and evaluating noise suppression algorithms in wind interference scenarios.

### 5.2.3 Room Impulse Responses

During training, we use around 60k samples of simulated RIRs from DNS5 challenge [55] to simulate reverberation.

## 5.3 Training and Validation Set

The overall training set consists of 1.3 million samples of clean speech of variable length, usually a few seconds, and the noise corpora consist of 110 thousand variable-length noise samples, usually a few seconds long.

For validation, we reserved a held-out set for each clean speech and noise dataset and created randomly simulated data with signal-to-noise ratio (SNR) values ranging from -7 dB to 20 dB. The validation set contains 2368 recordings of variable lengths from 2 to 15 seconds. The distribution of SNR values is shown in Table 5.1.

Table 5.1: Distribution of SNR values in the validation set.

| SNR Range | Number of Samples |
|---|---|
| $< -5$ dB | 23 (0.97%) |
| $[-5, 15)$ dB | 1889 (79.77%) |
| $\geq 15$ dB | 456 (19.26%) |

## 5.4 Test Set

For test and comparison purposes, we did not stick to the URGENT challenge set up, as the setup is recent and there is not enough prior work we would be able to compare our models to. Instead, we use a well-established VCTK-Demand test set from Valentini denoising benchmark [69], widely used for SE models comparison, consisting of 824 simulated noisy utterances from 2 speakers with 4 SNRs—2.5dB, 7.5dB, 12.5dB, 17.5dB.

## 5.5 Metrics

To assess the quality of enhanced speech and noisy input reconstruction in the case of dual-branch models, we use 3 types of metrics: intrusive or reference-based, comparing the enhanced signal to the ground-truth clean speech, non-intrusive or reference-free metrics predicting the mean opinion score (MOS), and downstream metrics assessing the quality of downstream tasks using the enhanced speech as an input (automatic speech recognition or speaker verification). Each of the used metrics is categorized and described below.

### 5.5.1 Intrusive Metrics

- **Scale-Invariant Signal-to-Distortion Ratio (SI-SDR):** Measures the ratio of the target signal's power to the distortion error and is invariant to the scale of the target. It is widely used for evaluating speech enhancement and source separation tasks [31]. SI-SDR is defined in Equation 4.1.
- **Log-Mel Distance:** Computes the $L_1$ distance between log-mel spectrograms of the reference and processed signals, which is a common metric used for assessing spectral fidelity in speech reconstruction tasks [30]. We use 25ms window length, 10ms hop length, and 80 mel bands.
- **Perceptual Evaluation of Speech Quality (PESQ):** PESQ estimates perceived speech quality by simulating how the human ear would compare a clean and a distorted signal, analyzing differences in loudness and timing across frequency bands. It produces a predicted MOS score, making it useful for evaluating codecs and networks in conditions like noise, delay, or packet loss [58].
- **Short-Time Objective Intelligibility (STOI):** STOI predicts speech intelligibility by measuring the correlation between short-time temporal envelopes of clean and degraded speech across frequency bands. It focuses on 384 ms segments to capture important speech modulations relevant to human understanding [66].

### 5.5.2 Non-Intrusive Metrics

- **Deep Noise Suppression Mean Opinion Score (DNSMOS):** DNSMOS uses a deep neural network trained on human-rated speech samples to estimate perceived quality directly from noisy speech, without needing a clean reference. Its self-teaching training process allows it to generalize well across diverse noise types and speech conditions, making it highly reliable for evaluating noise suppression systems [54].
- **UTokyo-SaruLab MOS (UTMOS):** UTMOS predicts speech quality by combining neural and traditional models trained on SSL features and listener-aware representations. Through ensemble learning and contrastive loss, it accurately ranks synthetic speech samples by perceived quality—even in low-data or mismatched evaluation scenarios [60].

### 5.5.3 Downstream Metrics

- **SpeechBERT Score:** Evaluates the semantic similarity between generated and reference speech by computing BERTScore on self-supervised learning (SSL) features from Wav2Vec2 [4] extracted from both signals [61].
- **Speaker Similarity (Spk SIM):** Measures the cosine similarity between the ground truth clean speech and the enhanced speech using embeddings extracted from a pre-trained speaker embedding extractor.
- **Word Error Rate (WER):** Calculates the rate of errors in automatic speech recognition by comparing the transcribed text to the reference using Whisper Large V3 [52].

### 5.5.4 Training Details

The encoder consists of 4 blocks (Figure 3.2) with strides 2, 4, 5, and 8. Input audio is processed by a convolutional layer containing 64 filters, and the output of each block doubles the number of features (i.e., each block uses an output convolution layer that contains twice as many kernels as the previous output convolutional block), resulting in a 1024-dimensional latent space. Before passing the latent sequence to the decoder, we use 1d convolution with 1536 convolutional filters, resulting in higher-dimensional space, giving the decoder more freedom to perform additional latent sequence processing. The convolutional decoder reflects the encoder, i.e., we use transposed convolutions instead of regular convolutions. Its last convolutional layer uses only one kernel, and the output represents an audio sampled at 16kHz.

Furthermore, we employ vanilla RoFormer models as presented in [64] with 8 layers, each operating in 1536-dimensional hidden space. We also explored larger RoFormers, mainly higher hidden spaces or more transformer layers, but it usually resulted in unstable training (exploding gradients and problems with convergence). In contrast to convolutional encoder and decoder, we use GeLU activation function [26] (a default option for RoFormer). Overall, the single-branch model has around 133M parameters, with 21.5M in the encoder, 52.3M in the decoder, and the rest in the RoFormer. The dual-branch model has 191.8M parameters due to the second branch (RoFormer and VQ). However, during inference, we only use the clean speech branch, meaning that there is no speed difference between the two models during inference.

For each of the three discriminators (clean speech, noise, and noisy speech), we use an ensemble of eight discriminators following the DAC [29] architecture, consisting of five multi-period discriminators with periods $[2, 3, 5, 7, 11]$ and three multi-band-multi-scale complex spectrogram discriminators with window lengths $[512, 1024, 2048]$, FFT hop lengths being set to $\frac{1}{4}$ of the window length, and bands set to $[[0, 0.1], [0.1, 0.25], [0.25, 0.5], [0.5, 0.75], [0.75, 1]]$. The bands correspond to the portion of the frequency resolution in the spectrogram (i.e., the first band covers the lowest 10% of the frequency range). During each training step, we first update all the discriminators and then update the generator (enhancement model).

To compute the Multi-scale log-Mel distance loss, we use 7 different STFT window lengths (same as DAC): $[32, 64, 128, 256, 512, 1024]$, and the hop length is set to $\frac{1}{4}$ of the window length. The number of mel filters is set to $[5, 10, 20, 40, 80, 160, 320]$ corresponding to each window length (i.e., window length 32 samples, 8 samples hop length, and 5 mel filters).

To create each training example, we randomly select a 3s long segment from a clean speech utterance[4]. Then, we randomly mix the clean speech with Gaussian noise with randomly sampled SNR from 0 to 25dB with a probability of 0.05 and with noise corpora described above with a probability of 0.95. When mixing the clean speech with noise corpora, we randomly sample one of the three SNR ranges: $[-10, -5), [-5, 20), [20, 30]$, with probabilities: 0.1, 0.8, 0.1, respectively. We then sample an SNR from a uniform distribution over the selected range. This way, we have more control over the SNR distribution of the training data compared to other works, which sample SNR uniformly from one given range. The last step is to convolve the noisy speech with a randomly selected RIR with a probability of 0.5.

We use AdamW [38] with a linear-warmup for 5k steps, max learning rate (lr) equal to $10^{-4}$ with cosine learning schedule [37], and weight decay equal to $10^{-2}$. We trained all our models for 100k iterations except for the ones we compare with prior work, which were trained for 200k steps[5]. We do not use any early stopping nor validation-based best model selection as we observed that usually, the last iteration resulted in the best-performing model, or the variance in the performance was negligible.

[4]We also experimented with longer segments (up to 6s), which usually resulted in better enhancement performance as the roformer can fully utilize the self-attention and information from a broader context. However, we stuck with the shorter segments to reduce the training time.

[5]Even though we could train for more steps, due to computational and time constraints, we kept a lower number of training iterations, than is usually used for NAC training

# Chapter 6

# Experiments

In this chapter, we thoroughly examine the trained models, both single-branch and dual-branch, starting from laying out the single-branch baselines and analyzing if pre-training for audio reconstruction helps. Furthermore, we examine which part of the single-branch model is responsible for the enhancement, and how the enhancement training affects the quantized representations. We then move to dual-branch models, where we first show the results for the supervised training and then for unsupervised training we first show baseline results and continue with ablations. As ablations, we explore the effect of model initialization and branch combination methods and show how removing discriminators affects the model performance. We also theoretically and empirically present the leakage problem and show how VQ can help. We then present a comparison to prior supervised and unsupervised speech enhancement models. Finally, we close the chapter with WER results on the VCTK-Demand computed from Whisper transcriptions, enhancement results on a real-world noisy dataset ATCO2 [23], and results on the inference speed on both CPU and GPU.

## 6.1 Single Branch

We train all single-branch models for 100k training steps using the same training data described in Section 5.2. If not stated otherwise, we initialize the encoder and decoder parameters from a pre-trained DAC [29]. Roformer and VQ parameters are initialized randomly.

We set the loss weights as: $\lambda_s = 1, \lambda_{fm} = 1, \lambda_{feat} = 1, \lambda_G = 1, \lambda_{cb} = 1, \lambda_{cm} = 0.25$, and whenever we do not use VQ, we do not optimize the VQ-related losses.

### 6.1.1 Baselines

Table 6.1 shows per-SNR group results of the single-branch baselines. Overall, the results, as expected, show that performance gets worse as SNR goes down (i.e., recordings are more noisy).

The first two rows of each group show that adding roformer improves enhancement performance over a simple encoder-decoder model. After listening to the simple encoder-decoder model outputs, if the SNR is low, some noise can leak into the clean speech, which does not happen as often in the model with RoFormer. The improvement can be attributed to the extra parameters that further transform the latent sequence and increase the receptive field (to infinity) due to the self-attention mechanism employed in the roformer. As can be seen, the improvement is consistent across various SNR levels.

Table 6.1: Comparison of single-branch baseline models: encoder-decoder only, encoder-decoder with roformer, and encoder-decoder with roformer and VQ. The results are based on the validation set.

| SNR | model | DNSMOS↑ | UTMOS↑ | PESQ↑ | STOI↑ | SpkSim↑ | SBERT↑ | SI-SDR↑ | MEL↓ |
|---|---|---|---|---|---|---|---|---|---|
| $< -5$ | enc-dec | 2.90 | 2.16 | 1.62 | 0.78 | 0.51 | 0.78 | 8.43 | 0.83 |
| | + roformer | 2.94 | 2.26 | 1.75 | 0.79 | 0.49 | 0.79 | 9.49 | 0.82 |
| | + VQ | 2.96 | 2.30 | 1.65 | 0.77 | 0.41 | 0.77 | 7.77 | 0.84 |
| $[-5, 15)$ | enc-dec | 3.03 | 2.49 | 2.07 | 0.85 | 0.61 | 0.85 | 12.26 | 0.76 |
| | + roformer | 3.02 | 2.54 | 2.21 | 0.87 | 0.60 | 0.86 | 12.62 | 0.73 |
| | + VQ | 3.05 | 2.55 | 1.98 | 0.83 | 0.48 | 0.83 | 9.64 | 0.74 |
| $\geq 15$ | enc-dec | 3.04 | 2.55 | 2.29 | 0.89 | 0.66 | 0.88 | 13.83 | 0.64 |
| | + roformer | 3.01 | 2.56 | 2.45 | 0.90 | 0.64 | 0.89 | 14.11 | 0.62 |
| | + VQ | 3.03 | 2.55 | 2.16 | 0.86 | 0.51 | 0.85 | 10.29 | 0.65 |
| all | enc-dec | 2.98 | 2.36 | 1.92 | 0.83 | 0.58 | 0.83 | 10.95 | 0.74 |
| | + roformer | 2.98 | 2.43 | 2.07 | 0.84 | 0.56 | 0.84 | 11.59 | 0.71 |
| | + VQ | 3.01 | 2.45 | 1.88 | 0.81 | 0.46 | 0.81 | 8.98 | 0.72 |

Furthermore, the last row shows that adding VQ results in worse performance, as we use less amount of bits to encode the audio. However, even though SI-SDR shows decreased performance by almost 3dB, Mel-distance, MOS scores, PESQ, and STOI are comparable with the no-VQ model, proving that the audio is comparably intelligible despite the lower quality.

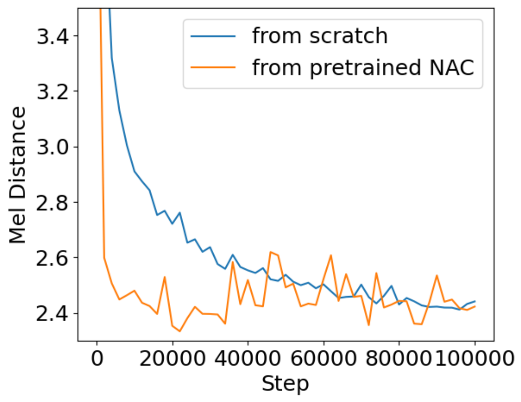
### 6.1.2 Does Codec Pre-Training Help?

To assess the impact of encoder and decoder audio reconstruction pre-training on the performance and convergence of the single-branch model, we trained one model with randomly initialized weights and the other with encoder and decoder weights initialized from a pre-trained DAC model.

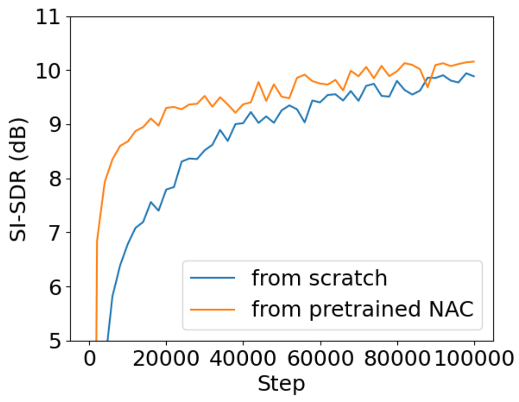
Table 6.2: Comparison of single-branch models with and without pre-training. The results are based on the validation set.

| Pre-trained | DNSMOS↑ | UTMOS↑ | PESQ↑ | STOI↑ | SpkSim↑ | SBERT↑ | SI-SDR↑ | MEL↓ |
|---|---|---|---|---|---|---|---|---|
| No | 2.91 | 2.20 | 1.81 | 0.81 | 0.52 | 0.82 | 10.01 | 0.75 |
| Yes | 2.98 | 2.43 | 2.07 | 0.84 | 0.56 | 0.84 | 11.59 | 0.71 |

Table 6.2 compares the performance of single-branch models with and without pre-training. It can be observed that the pre-training encoder and decoder consistently boost the final performance on all the metrics. Furthermore, Figure 6.1 shows both models' convergence curves of mel distance and SI-SDR. The pre-trained model converges faster than the model trained from scratch. However, Figure 6.1a also shows that the pre-trained model quickly adapts to the speech enhancement task and stays at some local optima, causing the validation loss to oscillate, which can be mitigated by, e.g., learning rate tuning or heavier regularization. Nonetheless, the pre-trained model achieves better performance



(a) Mel distance



(b) SI-SDR

Figure 6.1: Comparison of mel distance and SI-SDR convergence between the different initialization methods of a single branch model without RoFormer.

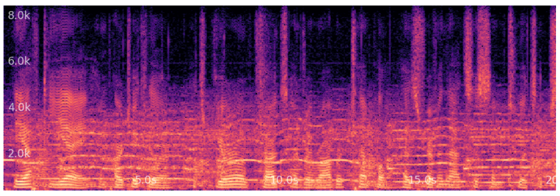
way sooner than the one initialized randomly. It proves that using pre-trained components for building an SE model is beneficial, as the model can leverage the knowledge learned during pre-training for input audio reconstruction, as it does not learn audio representations from scratch.

### 6.1.3 Which part of the model is responsible for enhancement?

When training SE systems, a question arises: Do internal representations contain only enhanced speech-related information, and can they serve as robust features for downstream tasks? To answer this question, we trained two single-branch models without roformer and VQ, both initialized from a pre-trained NAC: one with the frozen encoder and the other with the frozen decoder.

Table 6.3: Comparison of single-branch models initialized from NAC where only the encoder is trained (i.e., the decoder is frozen) and vice versa on the validation set.

| Trained | DNSMOS↑ | UTMOS↑ | PESQ↑ | STOI↑ | SpkSim↑ | SBERT↑ | SI-SDR↑ | MEL↓ |
|---|---|---|---|---|---|---|---|---|
| enc | 2.90 | 2.27 | 1.80 | 0.81 | 0.57 | 0.81 | 9.04 | 0.79 |
| dec | 2.92 | 2.29 | 1.89 | 0.82 | 0.57 | 0.82 | 9.66 | 0.74 |


(a) Noisy recording.

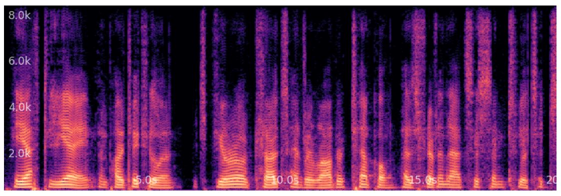

(b) Enhanced speech.

Figure 6.2: Spectrograms of noisy speech (left) and enhanced speech by the single-branch model with frozen encoder (right).

Figure 6.2 shows the spectrograms of the noisy speech and enhanced speech produced by the single-branch model with the frozen encoder. As can be seen, the model can remove most of the noise while preserving clean speech, which suggests that the majority of the enhancement happens in the decoder, as we were able to successfully train the decoder to perform enhancement while the encoder was fixed—i.e., the decoder can enhance a latent sequence containing both clean speech and noise information, as the model was initialized from a pre-trained NAC performing audio reconstruction. Also, our findings show that the latent space produced by neural audio codecs is separable and can be used for speech enhancement or source separation.

Furthermore, Table 6.3 shows similar performances when either part of NAC is trained while the other is frozen. However, when only the encoder is trained, it does not necessarily mean that the latent sequence does not contain any noise-related information, as the encoder might exploit the decoder to perform SE, e.g., by projecting noise to a space orthogonal to what the decoder was trained to reconstruct.

Hence, the decoder is likely the main part of the model responsible for speech enhancement, as it can also enhance the latent sequence containing noise[1]. Still, we do not rule out the hypothesis that the encoder might be trying to get rid of as much noise information as possible despite being unlikely.

### 6.1.4 VQ Stability

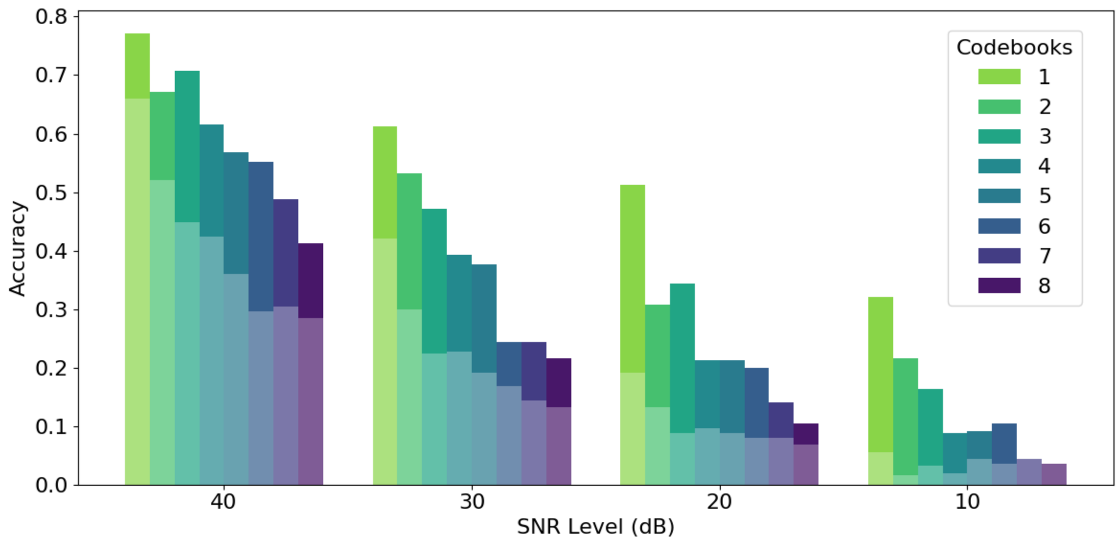


Figure 6.3: VQ stability: per-codebook accuracy of SE (darker color) and NAC models (lighter color). The accuracy is computed as a ratio of correctly classified codes of encoded noisy input audio (i.e., matching noisy and clean speech codes) to the total number of codes.

Despite our previous experiments showing that we do not need to train both encoder and decoder to train an SE system as decoder-only training is sufficient, we further explored

[1]We also successfully trained RoFormer and decoder to reconstruct the input noise on top of the frozen encoder representations pre-trained for SE.

how enhancement task affects the produced codes when VQ is employed, and the entire model is trained.

We trained two single-branch models with VQ using the same dataset for 100K training steps: one for SE and the other for input reconstruction (i.e., standard NAC training). We then artificially created a noisy recording by adding Gaussian noise with specified SNR db values (10, 20, 30, 40) and performed forward passes with and without the noise. Then, we computed per-codebook accuracy, where the clean speech codes serve as a reference and the noisy speech codes as a hypothesis. Figure 6.3 shows that SE codes are more robust—i.e., accuracy is higher; hence, they are less affected by the added noise. It can also be seen that the lower the SNR (i.e., the more noise is added), the more significant the gap between SE and NAC code accuracy is, mainly for the earlier codebooks. The lower difference between later codebooks can be explained by the fact that these codebooks encode some residual information that is not affected by the added noise that much.

Therefore, once the codes are used as features for downstream tasks (e.g., speaker verification or diarization), using the ones produced by the SE model is more beneficial, as they bring extra robustness for the downstream model without training on noisy task-specific data.

## 6.2 Dual Branch - Supervised

In this section, we show the experiments with dual-branch models trained in a supervised manner. Similarly to the single-branch models, we train all dual-branch models for 100k training steps using the same training data described in Section 5.2. If not stated otherwise, we initialize the encoder and decoder parameters from a pre-trained single-branch enhancement model (i.e., encoder, decoder, clean speech RoForme $\mathcal{R}_{CS}$ and clean speech discriminator are pre-trained). All other parameters are initialized randomly. We also use the same training hyperparameters as in the single-branch models.

During supervised dual-branch model training, we set the loss weights as: $\lambda_{CS,s} = 1$, $\lambda_{CS,fm} = 1$, $\lambda_{CS,feat} = 2$, $\lambda_{CS,G} = 4$, $\lambda_{CS,cb} = 1$, $\lambda_{CS,cm} = 0.25$, $\lambda_{CS,zm} = 10$, $\lambda_{CS,Emax} = 1$, $\lambda_{N,feat} = 2$, $\lambda_{N,G} = 1$, $\lambda_{N,cb} = 1$, $\lambda_{N,cm} = 0.25$, $\lambda_{NS,s} = 1$, $\lambda_{NS,fm} = 1$, $\lambda_{NS,feat} = 2$, $\lambda_{NS,G} = 1$. The loss terms are defined in equations 4.5, 4.6, and 4.7.

### 6.2.1 Supervised Baselines

We trained two dual-branch baseline models we use later on for initialization: one with and the other without VQ.

Table 6.4 shows the per-SNR group results of the baselines. It can be seen that both models perform comparably on the MOS-based metrics, proving that VQ does not affect the subjective quality of the enhanced speech too much. However, VQ can affect speaker similarity, as the clustering can slightly alter the speaker's voice and produce unwanted artifacts. The most significant difference can be seen in the SI-SDR and MEL distance, which assess similarity on the waveform level. The difference is mainly caused by distortions created by limiting the bandwidth and lossly compressing the latent sequence. Also, it can be seen that the dual-branch model performs slightly worse than the single-branch RoFormer models shown in Table 6.1. It can be attributed to the fact that both the encoder and decoder must be able to reconstruct clean speech and a large spectrum of noise, which is a more difficult task.

Table 6.4: Comparison of dual-branch model with and without VQ. The results are based on the validation set.

| SNR | model | DNSMOS↑ | UTMOS↑ | PESQ↑ | STOI↑ | SpkSim↑ | SBERT↑ | SI-SDR↑ | MEL↓ |
|---|---|---|---|---|---|---|---|---|---|
| $< -5$ | No-VQ | 2.93 | 2.14 | 1.66 | 0.78 | 0.45 | 0.78 | 9.20 | 0.86 |
| | VQ | 2.89 | 2.17 | 1.52 | 0.74 | 0.37 | 0.75 | 6.66 | 0.91 |
| $[-5, 15)$ | No-VQ | 3.01 | 2.47 | 2.15 | 0.86 | 0.57 | 0.86 | 12.65 | 0.76 |
| | VQ | 3.01 | 2.48 | 1.89 | 0.83 | 0.47 | 0.82 | 9.42 | 0.80 |
| $\geq 15$ | No-VQ | 3.01 | 2.51 | 2.40 | 0.90 | 0.63 | 0.89 | 14.21 | 0.63 |
| | VQ | 3.00 | 2.49 | 2.09 | 0.86 | 0.50 | 0.85 | 10.30 | 0.68 |
| all | No-VQ | 2.98 | 2.34 | 1.99 | 0.84 | 0.53 | 0.83 | 11.50 | 0.74 |
| | VQ | 2.96 | 2.35 | 1.77 | 0.80 | 0.43 | 0.80 | 8.43 | 0.79 |

Furthermore, Figure 6.4 shows the spectrograms of separated clean speech and noise by both of the presented models, proving that both models can separate the two sources. It is also worth noting that we do not use any reconstruction losses between the estimated noise and the ground-truth noise. However, the feature loss $\mathcal{L}_{N,feat}$ operates on top of the complext spectrogram and ensures that the reconstructed noise is as close to the ground-truth noise as possible.

### 6.2.2 Branch combination comparison

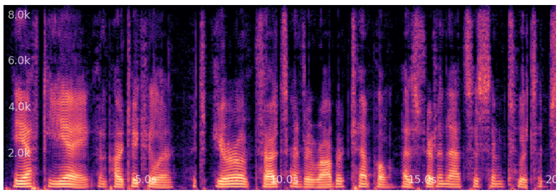
Table 6.5: Comparison of branch combination methods of dual-branch models trained in a supervised manner.

| Method | DNSMOS↑ | UTMOS↑ | PESQ↑ | STOI↑ | SpkSim↑ | SBERT↑ | SI-SDR↑ | MEL↓ |
|---|---|---|---|---|---|---|---|---|
| latent | 2.97 | 2.34 | 1.98 | 0.83 | 0.53 | 0.83 | 11.34 | 0.75 |
| waveform | 2.98 | 2.34 | 1.99 | 0.84 | 0.53 | 0.83 | 11.50 | 0.74 |

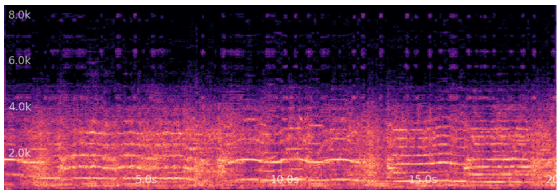
Table 6.5 compares latent and waveform combination methods described in Section 4.2.1. As can be seen, both methods perform comparably, with a slight advantage of the waveform combination method. However, the difference is negligible, proving that both methods result in similar performance when training with supervision.

## 6.3 Dual Branch - Unsupervised

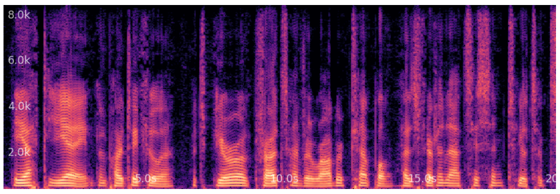
We continue the dual-branch model experiments with unsupervised training. The only difference between supervised and unsupervised training is that we do not use paired clean speech supervision. Hence, we cannot use the discriminator feature losses (Equation 3.16) or reconstruction losses (Section 3.4) between the ground-truth clean speech and the enhanced speech. Therefore, as stated in Section 4.2.2, we drop all the losses requiring reference clean speech and noise. We keep the loss weights the same as during the supervised training. The particular loss terms are defined in Section 3.4.



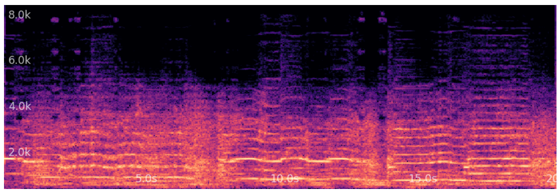
(a) Enhanced Speech



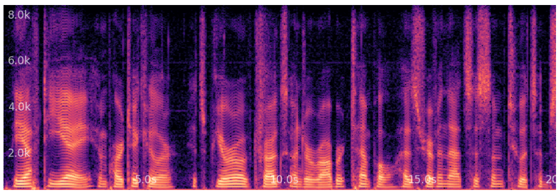
(b) Estimated Noise



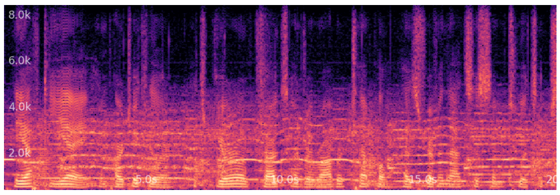
(c) Enhanced Speech (VQ)



(d) Estimated Noise (VQ)



(e) Ground truth clean speech.



(f) Input noisy speech

Figure 6.4: Spectrograms of input noisy speech, enhanced speech, and estimated noise by the dual-branch model with (first row) and without (second row) VQ. The last row shows the input noisy speech.

### 6.3.1 Baselines

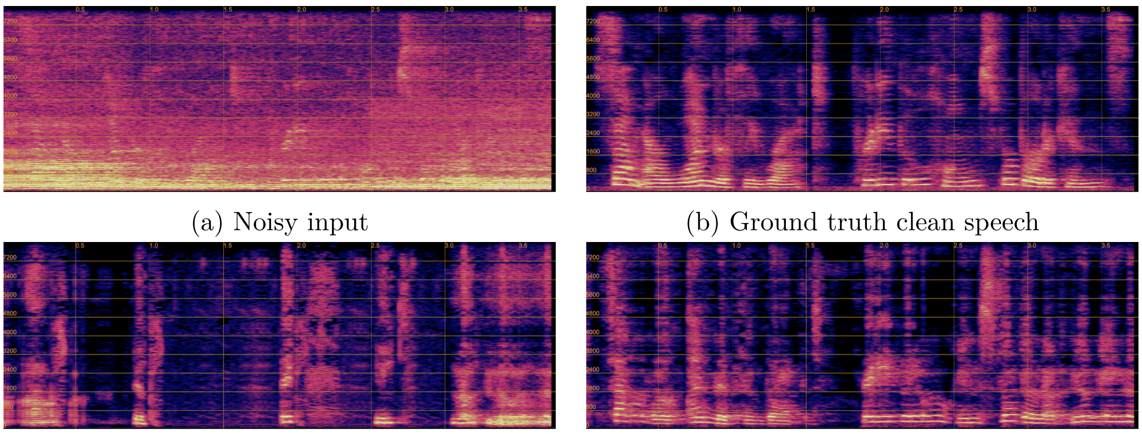
We trained the same model configurations as in Section 6.2.1 without using paired data. We initialized all the parameters from NAC pre-trained for audio reconstruction and trained for another 100k iterations.

We can see from Table 6.6 that, as with the supervised models, no-VQ setup beats the VQ setup in terms of signal-level metrics and also speaker similarity. However, other metrics do not differ much, proving that VQ does not significantly affect subjective intelligibility. Furthermore, we can see that the performance drops significantly for low-SNR recordings, achieving SI-SDR 3.34dB compared to 9.2dB for supervised models (Table 6.4). Similar performance drops can be observed on other metrics as well. After listening to low-SNR recordings, we observed that the model cannot correctly identify speech regions and outputs only a tiny part of the clean speech mixed with input noise. However, it suppresses other background noise rather than leaking it to the clean speech branch. An example with SNR -6.75dB is shown in Figure 6.5, where we can see that the supervised model (Figure 6.5d) can at least reconstruct lower frequency harmonics, despite not reconstructing the higher frequency harmonics, compared to the unsupervised model (Figure 6.5c) that is unable to reconstruct most of the clean speech.

On the other hand, Figure 6.6 shows the spectrograms of enhanced speech and estimated noise from an input noisy recording with 1.92dB SNR by VQ and No-VQ models. It can be seen that both models are capable of separating the sources from the noisy recording. However, compared to the supervised counterparts depicted in Figure 6.4, neither noise nor clean speech is as well separated as in the supervised case. This is mainly caused by the fact

Table 6.6: Comparison of dual-branch model with and without VQ. The results are based on the validation set.

| SNR | model | DNSMOS↑ | UTMOS↑ | PESQ↑ | STOI↑ | SpkSim↑ | SBERT↑ | SI-SDR↑ | MEL↓ |
|---|---|---|---|---|---|---|---|---|---|
| $< -5$ | No-VQ | 2.59 | 1.63 | 1.24 | 0.65 | 0.36 | 0.72 | 3.34 | 1.44 |
|  | VQ | 2.58 | 1.55 | 1.16 | 0.62 | 0.27 | 0.65 | 1.44 | 1.28 |
| $[-5, 15)$ | No-VQ | 2.93 | 2.08 | 1.60 | 0.81 | 0.50 | 0.82 | 10.28 | 1.09 |
|  | VQ | 2.93 | 2.03 | 1.44 | 0.77 | 0.39 | 0.76 | 7.69 | 1.13 |
| $\geq 15$ | No-VQ | 3.02 | 2.30 | 1.92 | 0.87 | 0.57 | 0.87 | 13.04 | 0.78 |
|  | VQ | 3.01 | 2.22 | 1.65 | 0.81 | 0.43 | 0.80 | 9.27 | 0.81 |
| all | No-VQ | 2.80 | 1.94 | 1.51 | 0.76 | 0.45 | 0.79 | 7.90 | 1.03 |
|  | VQ | 2.80 | 1.87 | 1.36 | 0.72 | 0.35 | 0.72 | 5.37 | 1.07 |


(a) Noisy input

(b) Ground truth clean speech

(c) Enhanced speech (unsupervised)

(d) Enhanced speech (supervised)

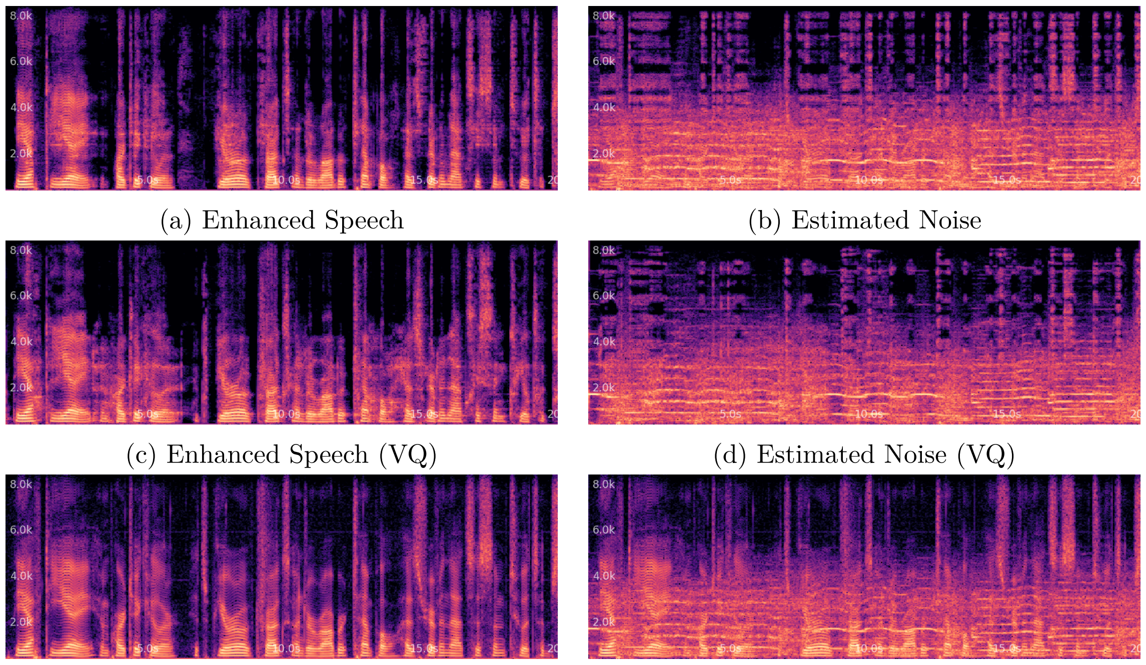
Figure 6.5: Spectrograms of input noisy speech, ground truth clean speech, and enhanced speech by the dual-branch model trained in unsupervised and supervised manner.

that unsupervised models are receiving a lot less information about how the output should look. Also, it is caused by noise leakage, which is thoroughly explained in Section 6.3.5.

### 6.3.2 Initialization

To explore how sensitive is the unsupervised training to initialization or pre-training, we trained three dual-branch models for 100k iterations: randomly initialized (i.e., from scratch), encoder and decoder initialized from a pre-trained NAC, and all the parameters initialized from a dual-branch model pre-trained in a supervised manner.

Figure 6.7 shows the convergence curves of the mel distance and SI-SDR metrics. Similarly to single-branch models, the model trained entirely from scratch converges slower than the one initialized from NAC. Furthermore, we can see that MEL distance and SISDR are increasing over time, converging to similar values as without dual-branch pre-training, which is caused by the progressive noise leakage (explained thoroughly in Section 6.3.5). In addition, Table 6.7 shows the performance after 100k training steps on validation data. As can be seen, we could train the dual-branch model without any paired data from scratch



(a) Enhanced Speech (b) Estimated Noise

(c) Enhanced Speech (VQ) (d) Estimated Noise (VQ)

(e) Ground truth clean speech. (f) Input noisy speech

Figure 6.6: Spectrograms of input noisy speech, enhanced speech, and estimated noise by the dual-branch model trained in an unsupervised manner with (first row) and without (second row) VQ. The last row shows the input noisy speech.

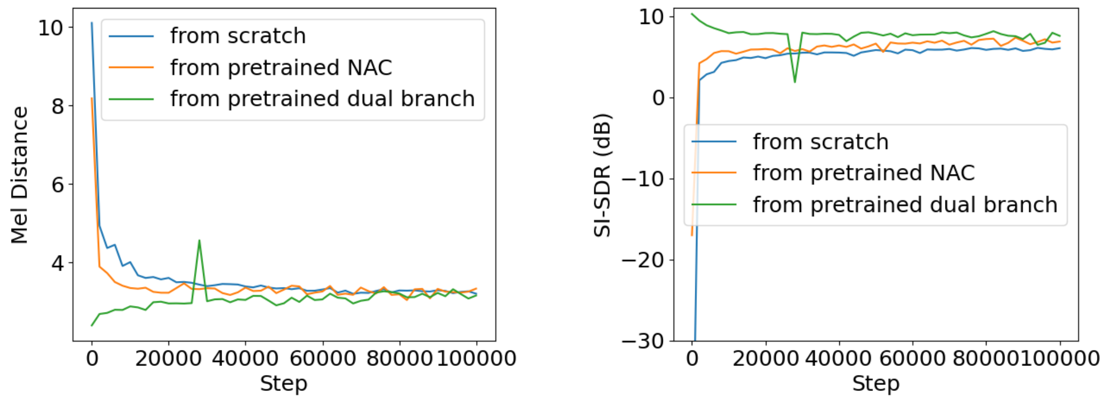
Table 6.7: Comparison of initialization methods of dual-branch models trained in an unsupervised manner. 2B stands for dual-branch.

| Init Method | DNSMOS↑ | UTMOS↑ | PESQ↑ | STOI↑ | SpkSim↑ | SBERT↑ | SI-SDR↑ | MEL↓ |
|---|---|---|---|---|---|---|---|---|
| From Scratch | 2.73 | 1.83 | 1.42 | 0.74 | 0.40 | 0.75 | 6.95 | 1.06 |
| Pre-trained NAC | 2.80 | 1.94 | 1.51 | 0.76 | 0.45 | 0.79 | 7.90 | 1.03 |
| Pre-trained 2B | 2.86 | 2.04 | 1.55 | 0.76 | 0.45 | 0.78 | 8.49 | 0.99 |

and without any model collapse or other instabilities. However, the figure also reveals that more pre-training results in better convergence and ebetter final model performance. Also, the experiment suggests that if we do not have any paired clean and noisy speech data, we can at least pre-train the encoder and decoder to perform audio reconstruction to improve the SE performance. However, if we do have at least some paired data (or can simulate some), it is beneficial to pretrain all the parameters.

Furthermore, Figure 6.8 shows the spectrograms of the enhanced speech example produced by the three models after being trained for 100k iterations. As can be seen, pre-training improves the enhanced speech quality, mainly in the high-frequency areas where the harmonics are better reconstructed. Hence, we can see that the unsupervised training does not result in complete forgetting of the abilities gained during pre-training.

To conclude this section, pre-training is important for better enhancement performance when training without supervision. However, it is not necessary as the model can be trained from scratch without pre-training.



(a) Mel distance (b) SI-SDR

Figure 6.7: Comparison of mel distance and SI-SDR convergence between the different initialization methods of dual-branch models trained in an unsupervised manner on the validation set.

### 6.3.3 Branch Combination Comparison

Compared to supervised training where waveform and latent space combination methods performed comparably, in the case of unsupervised training, the latent combination method resulted in a collapsed model due to not being able to properly ensure consistency compared to the waveform combination method (also described as a disadvantage in Section 4.2.1). The Mel distance progress throughout the training can be seen in Figure 6.9, where the model with latent combination suddenly diverged and started producing silence instead of enhanced speech.

The main reason behind the inability of the latent combination method to ensure the consistency is that we do not enforce the complementarity of the encoded information in neither time domain nor latent space. This can also be seen in Figure 6.10 that compares latent latent vs waveform combination method outputs. Figure 6.10a shows the enhanced speech of one example to be silence, which is likely one of the modes of the clean speech discriminator that the clean speech branch is sampling—a schoolbook example of mode collapse. In contrast, Figure 6.10b shows an enhanced speech spectrogram corresponding to the correctly separated clean speech.

Similar findings apply to the separated noise, as shown in Figure 6.10c, where the model produces a deformed output not corresponding to the noise in the recording when the latent combination is used, whereas the waveform combination method produces an audio containing the correct input noise (Figure 6.10d), although with some artifacts caused by the discriminator architecture.

Lastly, Figures 6.10e and 6.10f show the reconstructed input audio spectrograms. As can be seen, both methods output audio close to the original noisy input, which shows that even though each of the branches alone reconstructs nonsense, the combined latent space indeed results in a proper reconstruction of the input audio when the latent combination is used. This also proves that when latent space combination is employed, the overall input reconstruction does not ensure consistency between the estimated sources and the input in the exact way that waveform combination does. Hence, we can conclude that the waveform combination is the right choice for unsupervised training, as it forces both branches to reconstruct audio consistent with the input and that the discriminators can force source separation simultaneously.

### 6.3.4 Discriminator Ablations

To compare what discriminators are essential for the unsupervised training and if they can enforce the separation of the clean speech and noise, we trained three dual-branch models for 100k iterations, all initialized from a pre-trained dual-branch model: one with all discriminators, one with only clean speech and noisy speech discriminators (i.e., without noise discriminator), and one with only noisy speech discriminator.

Table 6.8: Comparison of employed discriminators during dual-branch models training in an unsupervised manner. The first row shows the model with all discriminators, while the last row show the model without noise and clean speech discriminators.

| Discriminators | DNSMOS↑ | UTMOS↑ | PESQ↑ | STOI↑ | SpkSim↑ | SBERT↑ | SI-SDR↑ | MEL↓ |
|---|---|---|---|---|---|---|---|---|
| All | 2.86 | 2.04 | 1.55 | 0.76 | 0.45 | 0.78 | 8.49 | 0.99 |
| - Noise | 2.88 | 2.06 | 1.55 | 0.78 | 0.45 | 0.77 | 8.68 | 1.00 |
| - Clean Speech | 1.75 | 1.57 | 1.36 | 0.78 | 0.63 | 0.78 | 4.50 | 1.67 |

First, we compare the models based on the validation metrics shown in Table 6.8. It can be seen that using all discriminators results in almost identical performance to the one without noise discriminator in all but the signal metrics (SI-SDR, Mel distance). Regarding the signal metrics, we can see that the model trained with all discriminators achieves worse SI-SDR but better MEL distance than the one without noise discriminator. This suggests that the former model perceptually reconstructs the audio equally well but might introduce some waveform distortions, causing SI-SDR and Mel distance to be slightly worse (although not significantly). Furthermore, comparing Figure 6.11a and Figure 6.11c shows that training with all discriminators yields more accurate clean speech reconstruction (comparing against ground truth depicted in Figure 6.8d) including breathing sounds present in the ground truth clean speech, which are not present in the speech enhanced by the model trained without noise discriminator. It results from forcing the noise branch to focus on noise modeling as much as possible, which causes the overall reconstruction to push the clean speech branch towards outputting more accurate clean speech.

In addition, Figure 6.11b and 6.11f show how the noise discriminator affects the output of the noise branch. When we do use the noise discriminator, we can see that the noise branch mainly focuses on noise modeling, albeit with some artifacts (e.g., spectral stripes) that are caused, as we previously stated, by suboptimal discriminator design [3]. On the contrary, if we do not use the noise discriminator, clean speech leaks in, causing the branch to reconstruct mainly noise and some residual clean speech that is not well reconstructed by the CS branch. However, even though the noise branch containing some clean speech seems not to affect the clean speech branch too much, it is beneficial to force the noise branch to model the contained noises as well as possible (i.e., to use the noise discriminator), as the clean speech branch is the forced to reconstruct the clean speech more accurately.

Lastly, reconstructed audio by the model trained without branch-specific discriminators is shown in Figure 6.11c and 6.11d. It can be observed that the noise and clean speech are not separated at all. Each branch just reconstructs some part of the input noisy signal (i.e., both reconstruct the noisy signals with different artifacts), which is also reflected in the metrics, where the model without $D_{CS}, D_N$ performs the worst.

To sum up, the clean speech discriminator is essential for unsupervised training, as it forces the model to reconstruct the clean speech well, proving our intuition behind the design right. The noise discriminator is unnecessary, but it helps improve the overall performance and reduce the leakage of clean speech in the noise branch. However, it is not essential for unsupervised training, as we can train a model without it, achieving comparable performance. This also shows that having a clean speech data pool with unpaired noisy speech examples is sufficient for training the dual-branch model in an unsupervised manner, as the branch can learn to separate the clean speech and the residual signal without additional supervision.

### 6.3.5 Leakage

We observed that when training without any paired noisy and clean speech data, the model cannot perfectly enhance the clean speech, and some noise can leak in, usually depending on the amount of present noise in the input recording. In this section, we also analyze the leakage from a theoretical point of view and show two examples of leakage spectrograms.

For the theoretical description, let us reuse prior notation and denote $\mathbf{x}_{NS} \in \mathbb{R}^T$ as noisy input signal, $\hat{\mathbf{x}}_{CS} = f_\theta(\mathbf{x}_{NS}) \in \mathbb{R}^T$ as enhanced speech, and $\hat{\mathbf{x}}_N = g_\phi(\mathbf{x}_{NS}) \in \mathbb{R}^T$ as estimated noise signal, where $f_\theta, g_\phi$ are parametrized functions (i.e., neural networks), representing the two branches in the dual-branch model. Furthermore, as a reminder, $D_{CS}$ and $D_{NS}$ are the clean speech and the noise discriminators, respectively. As defined before, the reconstructed input audio is:

$$\hat{\mathbf{x}}_{NS} = \alpha \hat{\mathbf{x}}_{CS} + \beta \hat{\mathbf{x}}_N. \tag{6.1}$$

Let us further define two gradient vectors coming from:

- the clean speech discriminator $D_{CS}$ as $\nabla_{\hat{\mathbf{x}}_{CS}} D_{CS}(\hat{\mathbf{x}}_{CS})$,
- the overall reconstruction loss as $\nabla_{\hat{\mathbf{x}}_{NS}} \mathcal{L}(\hat{\mathbf{x}}_{NS})$ including the adversarial losses coming from $\nabla_{\hat{\mathbf{x}}_N} D_{NS}(\hat{\mathbf{x}}_N)$.

If we look at the gradient that is being used during updating $\theta$, we can see that:

$$\nabla_\theta f(\mathbf{x}_{NS}) = (\alpha \nabla_{\hat{\mathbf{x}}_{CS}} \mathcal{L}(\hat{\mathbf{x}}_{NS}) + \nabla_{\hat{\mathbf{x}}_{CS}} D_{CS}(\hat{\mathbf{x}}_{CS}))^\top \mathbf{J}_{f_\theta}, \tag{6.2}$$

where $\mathbf{J}_{f_\theta}$ is the Jacobian of the output of $f_\theta$ w.r.t. to the parameters $\theta$. Hence, the clean speech branch gets affected by both, the gradient coming from the overall reconstruction losses ensuring that $\hat{\mathbf{x}}_{NS}$ is reconstructed well and the gradient coming from the clean speech discriminator that forces $f$ to produce signals that have similar statistical properties as clean speech.

However, this introduces a few issues. Firstly, the magnitude of each gradient path can differ quite a lot (if $\|\nabla_{\hat{\mathbf{x}}_{CS}} \mathcal{L}(\hat{\mathbf{x}}_{NS})\|_2^2 \gg \|\nabla_{\hat{\mathbf{x}}_{CS}} D_{CS}(\hat{\mathbf{x}}_{CS})\|_2^2$, then CS branch gradients can be biased towards the overall reconstruction rather than outputting clean-speech-like signal). Also, the gradients are most likely not aligned and sometimes might point in the opposite direction, creating so-called gradient conflict during multitask optimization [35]. A possible solution might be using PCGrad [83] or gradient normalization similar to the authors of Encodec [12], who proposed to normalize the gradients coming from different losses and then multiply them by the corresponding loss weights. We even re-implemented this balancing; however, further weight loss tuning would be needed to achieve the same performance as without it. Also, branch-wise gradient normalization differs from overall gradient normalization and is left as a future work.

Secondly, the discriminator gradients might not be as informative about nuances in the enhanced speech signal, and some noise leaks might not be penalized enough, resulting in noise leakage. It can be solved by increasing the discriminator capacity similarly to [45] and introducing self-supervised features and various auxiliary tasks during training. We also leave the solution as a future work as it is a non-trivial problem to solve and requires a lot of further research and experimentation.

Moreover, Figure 6.12 shows two examples of leakage produced by the model trained with all discriminators. The first row depicts the $\hat{\mathbf{x}}_{CS}$ for two different recordings, while the latter two represent the ground-truth clean speech and the input noisy audio. Example 1 (Figure 6.12a) clearly shows noise leakage in the higher frequencies. Example 2 (Figure 6.12b) demonstrates a similar pattern, albeit with less pronounced leakage mainly present in the lower frequencies due to the noise being guitar music mixed with higher SNR (i.e., quieter noise). Nevertheless, it can also be seen that the leakage happens only when the speech is present, showing that the model correctly identified non-speech parts of the input audio and suppressed the underlying noise almost entirely.

### VQ vs No-VQ

Training the dual-branch model with continuous latent space allows the model to learn arbitrary continuous representations of the input audio. Even though we showed that the model can learn to separate the clean speech from the residual noise, no explicit mechanism prevents the noise leakage. As we discussed before, the discriminator might not be optimal and may overlook some nuances in the enhanced speech, allowing the reconstruction losses to force the clean speech branch to contain some noise to achieve as good input reconstruction as possible; hence, introducing redundancies in the representations. In this case, a natural solution is to limit the model's bandwidth by quantizing the latent space using RVQ and allowing the model to use a certain number of bits to represent the input audio. This way, to achieve good input reconstruction and satisfy the adversarial losses, the model must not model redundant information (e.g., noise present in the clean speech), as it would decrease the reconstructed input quality.

To explain why VQ helps mitigate leakage, we analyze our dual-branch model using the Information Bottleneck (IB) framework [68]. IB describes representation learning as a trade-off between compressing the input and preserving task-relevant information. The objective is given by:

$$\mathcal{L}_{IB} = I(X; Z) - \beta I(Z; Y), \tag{6.3}$$

where $X$ is the input, $Y$ is the target, $Z$ is the representation, and $I$ denotes the mutual information between two random variables [10]. We can state that our dual-branch model is indirectly trying to optimize the following two IB objectives simultaneously[2], one for the clean speech branch and one for the noise branch:

$$\mathcal{L}_{IB}^{CS} = I(X_{NS}; Z_{CS}) - \beta_{CS} I(Z_{CS}; X_{CS}), \tag{6.4}$$

$$\mathcal{L}_{IB}^{N} = I(X_{NS}; Z_N) - \beta_N I(Z_N; X_N). \tag{6.5}$$

[2]We do not explicitly optimize the information bottleneck objectives.

This means that the clean speech branch is trying to preserve as little information about the input as possible while retaining only the information about the clean speech, while the noise branch is trying to preserve as little information about the input as possible while retaining only the information about the noise.

Focusing on the clean speech branch, we can rewrite the first term in Equation 6.4 by applying the mutual information chain rule as:

$$I(Z_{CS}; X_{NS}) = I(Z_{CS}; X_{CS}) + I(Z_{CS}; X_N | X_{CS}). \tag{6.6}$$

Employing VQ limits the bandwidth of the latent space and obtain the following upper bound:

$$I(Z_{CS}; X_{NS}) \leq L \log_2(|C|), \tag{6.7}$$

where $\log_2(|C|)$ is the number of bits used to represent the latent space embedding and $L$ is the length of $Z_{CS}$. By combining the two equations, we get:

$$I(Z_{CS}; X_{CS}) \leq L \log_2(|C|) - I(Z_{CS}; X_N | X_{CS}). \tag{6.8}$$

This reveals a fundamental trade-off: with a fixed latent bandwidth, any noise-related information retained in the clean speech representation reduces the capacity available for encoding actual speech features. Vector quantization enforces this bandwidth constraint, forcing the model to prioritize what information to preserve. The adversarial loss from the clean speech discriminator encourages the model to allocate its limited capacity toward speech-like structure and forces outputs to resemble real clean speech, making it increasingly costly for the model to let noise leak into the clean speech branch.

To verify this claim, we trained five dual-branch models, all initialized from a pre-trained dual-branch model: four with 1, 2, 4, 8 RVQ codebooks, and one without VQ. However, relying on metrics as before does not work well when assessing if VQ helps with leakage. The main reason is that using VQ decreases the audio quality due to limiting the bandwidth. Thus, the metrics are worse when VQ is employed, even though the leakage is reduced. Therefore, we rely purely on visual inspection of spectrograms (or listening) and leave the rigorous analysis and development of a proper validation approach to future work.

Figure 6.13 shows two noise leakage examples for the abovementioned models. The first example (with higher SNR) shows slight low-frequency noise leaks (stripes reflecting the traces of guitar noise), on which we can see that lower bandwidth results in less leakage, proving our intuition about the model being forced not to model the redundant information. The second example (with lower SNR) also shows that some higher frequency noise leaks to the first few seconds of the audio when no VQ is employed and does not leak as much when VQ is employed. However, the leakage is still present but suppressed.

To conclude, VQ can help with leakage, but at the expense of worse reconstruction quality (proved before by the metrics), which is mainly caused by the limited amount of bits the model can use to represent the audio. However, there has been research into low-bitrate neural audio codecs [45], showing it is possible to decrease the number of codes and sampling rate, while not sacrificing the audio quality as much, and we are planning to explore such architectures as future work.

## 6.4 Comparison with Prior Work

As mentioned before, we utilize the VCTK-Demand test set for metrics computation to compare to previous works. Due to the unavailability of all metrics' prior work results, we only compare the models based on DNSMOS, UTMOS, PESQ, STOI, and SI-SDR.

For supervised speech enhancement, we compare our models with the following prior works: MetricGAN+ [19], HiFi-GAN2 [65], and FINALLY [3]. As shown in Table 6.9, all our models improve upon the noisy input speech in all metrics. Moreover, we can see that all our models outperform the MetricGAN+ and HiFi-GAN2 models in both MOS metrics while being 0.06 points behind the well-performing FINALLY model. Also, our non-VQ models achieve the best SI-SDR scores and do not lag behind HiFi-GAN2 in STOI, achieving second-to-best scores. However, we can see that our models are behind the prior work in PESQ, likely caused by a slight loss of fine-grained details (it can be seen that VQ models have significantly worse PESQ scores) or artifacts that are barely noticeable by human ears. Nonetheless, various studies have reported that reference-based metrics do not correlate well with human perception [39, 48]. Hence, judging by the MOS scores, our models are comparable to the prior work and achieve second to best performance. Also, models employing VQ, while worse in reference-based metrics, do not lose much perceptual quality according to both MOS metrics. Lastly, we compare the WER of our models. To be comparable to the prior work numbers, we utilized a wav2vec2-based xlsr model chosen by the authors of FINALLY [3]. It can be seen that the best-performing models match the input noisy speech performance, while our models and MetricGan+ slightly lag behind. It can be seen that VQ models perform worse, likely due to the introduction of artifacts that confuse the ASR model.

Table 6.9: Comparison of our supervised models with prior work. Our models are shown in the last four rows.

| Model | DNSMOS↑ | UTMOS↑ | PESQ↑ | STOI↑ | SI-SDR↑ | WER↓ |
|---|---|---|---|---|---|---|
| Input Speech | 2.54 | 3.10 | 1.97 | 0.92 | 8.54 | 0.07 |
| Ground Truth | 3.15 | 4.09 | 4.64 | 1.00 | - | 0.05 |
| MetricGAN+ | 2.95 | 3.62 | 3.14 | 0.93 | 8.6 | 0.10 |
| HiFi-GAN2 | 3.12 | 3.99 | 3.14 | 0.95 | 17.9 | 0.07 |
| FINALLY (16kHz) | 3.22 | 4.32 | 2.94 | 0.92 | 4.6 | 0.07 |
| SingleBranch (Ours) | 3.16 | 4.00 | 2.86 | 0.94 | 19.26 | 0.09 |
| SingleBranch VQ (Ours) | 3.16 | 3.98 | 2.59 | 0.92 | 14.86 | 0.13 |
| DualBranch (Ours) | 3.16 | 3.93 | 2.83 | 0.94 | 19.60 | 0.10 |
| DualBranch VQ (Ours) | 3.16 | 3.94 | 2.52 | 0.92 | 14.67 | 0.14 |

Furthermore, we focus on unsupervised model comparison and compare our models with the following prior works: Wiener [36], NyTT [22], MetricGAN-U (half), and MetricGAN-U (full) [20]. The difference between half and full MetricGAN-U is the number of training epochs, as the half model was trained using early stopping [51] based on the average PESQ score, while the full model was trained for 600 epochs on VCTK-Demand [69] train set.

Table 6.10 shows that both our models improve on the noisy input speech in all metrics, except for STOI, where the model with VQ performs slightly worse. However, as we mentioned before, MOS metrics correlate with human perception better, and we significantly beat the noisy speech in both. Furthermore, let us compare our models to the Wiener filter, a signal processing baseline without any training data. Our models outperform it by a large margin (all metrics except for STOI for the VQ model). Also, our no-VQ model matches the performance of NyTT in PESQ, but lacks behind in terms of SI-SDR.

Table 6.10: Comparison of our unsupervised models with prior work. Our models are shown in the last two rows.

| Model | DNSMOS↑ | UTMOS↑ | PESQ↑ | STOI↑ | SI-SDR↑ | WER↓ |
|---|---|---|---|---|---|---|
| Input Speech | 2.54 | 3.10 | 1.97 | 0.92 | 8.54 | 0.07 |
| Ground Truth | 3.15 | 4.09 | 4.64 | 1.00 | - | 0.05 |
| Wiener | 2.54 | 3.05 | 1.93 | 0.92 | 8.42 | - |
| NyTT | - | - | 2.30 | - | 17.66 | |
| MetricGAN-U (half) | 2.89 | - | 2.45 | - | - | - |
| MetricGAN-U (full) | 3.15 | - | 2.13 | - | - | - |
| Unsup Dual Branch (Ours) | 3.04 | 3.61 | 2.29 | 0.93 | 14.51 | 0.10 |
| Unsup Dual Branch VQ (Ours) | 3.03 | 3.60 | 2.12 | 0.89 | 12.22 | 0.17 |

Furthermore, both our models beat the MetricGAN-U (half) model in DNSMOS and the full model in PESQ (half model was selected to achieve the highest PESQ). Although the MetricGAN-U model is trained to indirectly optimize DNSMOS, our approach relies solely on data-driven learning without the incorporation of any such non-intrusive metrics during training. As a result, our model is not constrained by the limitations of optimizing toward a potentially sub-optimal proxy objective. Lastly, we present WER results using the same setup as in Table 6.9. It can be seen that the no-VQ unsupervised model performs comparably to the supervised models. On the other hand, the VQ model does not perform as well as both supervised and unsupervised models. We observed that the pronunciation could be slightly altered, causing the ASR model to output words like assengial or fipe instead of essential and five, respectively. Hence, we conclude that our unsupervised models, while not beating all the prior work, are comparable in the SE metrics and to the supervised models in WER computed from the wav2vec2-based ASR transcripts.

### 6.4.1 Whisper Large-V3 WER

Even though the authors of FINALLY [3] computed all the WER scores using the wav2vec2-based ASR model, we also decided to use our enhancement system in conjunction with Whisper Large-V3 [52] as it is a default choice for ASR not only as a baseline model but also as a production-ready system.

Table 6.11 shows that all but the unsupervised dual-branch VQ model improves upon the noisy input, proving that enhanced audio can assist an already robust ASR model and improve the WER.

## 6.5 Real World Noisy Data

Lastly, to prove that our method can handle real-world noisy data, we trained a dual-branch model on ATCO2 [23]—an air traffic control dataset containing noisy communications between pilots and air traffic controllers. We used 4000h for training and the official test set for evaluation. The model was initialized from a pre-trained NAC and trained for 100k iterations. We used the same clean speech and noise data pools as in the previous experiments.

Table 6.11: Comparison of WER inferred by Whisper Large-V3 ASR model using the enhanced VCTK-Demand dataset.

| Model | WER↓ |
|---|---|
| Input Speech | 0.07 |
| Ground Truth | 0.02 |
| Single Branch | 0.04 |
| Single Branch VQ | 0.07 |
| Dual Branch | 0.04 |
| Dual Branch VQ | 0.06 |
| Unsupervised Dual Branch | 0.06 |
| Unsupervised Dual Branch VQ | 0.08 |

Table 6.12: MOS results on ATCO2 dataset comparing noisy input and enhanced speech by dual-branch model.

| | DNSMOS↑ | UTMOS↑ |
|---|---|---|
| Noisy Input | 1.99 | 1.30 |
| Enhanced | 3.07 | 1.89 |

Table 6.12 demonstrates that our model is capable of learning SE from real-world noisy data, which is reflected primarily in the DNSMOS improvement from 1.99 to 3.07. In contrast, improvement in UTMOS is not as significant as DNSMOS, as it focuses more on intelligibility than the quality of denoising. We observed that on one hand, the enhanced audio is a lot cleaner than the input, but, on the other hand, it sometimes does not contain all the phonemes reconstructed well enough, making it harder to understand the uttered words. An example of the noisy input and enhanced audio is shown in Figure 6.14.

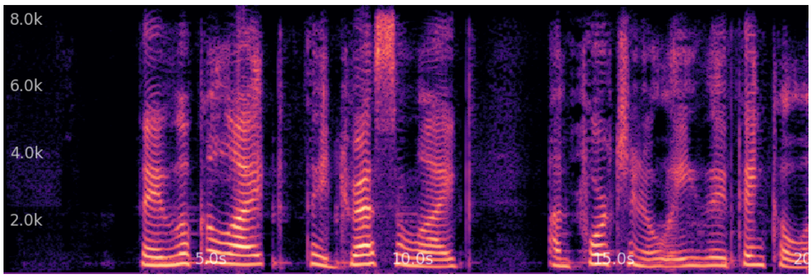
## 6.6 Inference Speed and Model Complexity

To evaluate the practicality of the proposed models, we measure inference time on AMD EPYC 9454 48-Core CPU and NVIDIA H100 NVL GPU. We pass 10s of audio through the model 10x and take an average of the elapsed time. The real-time factor (RTF) is then calculated by dividing the average elapsed time by 10 (audio length in seconds). To compare with the prior work, we assume that Nvidia V100 is 10x slower than H100 NVL and report the re-calibrated RTF. Also, for CPU RTF, we limit the number of cores to 1 to be comparable with other works and also report RTF using eight cores to report RTF close to what one can get when using a standard desktop.

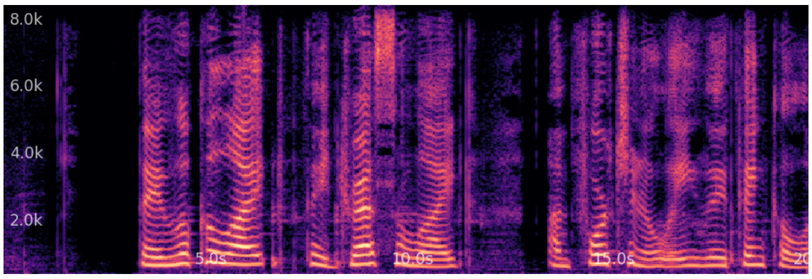
Table 6.13 shows the performance comparison between our models and the prior work. Our models achieve the best RTF under the assumption of the relative speed between the GPUs. However, we can conclude that our models are significantly faster than HiFi-GAN2 and comparable to FINALLY. Also, we can see that when using 1 CPU core, it takes around 1.4s to process 1s of audio and 0.37s when using eight cores. This shows that our model can be used for real-time enhancement even without GPU when multiple cores are used.

Table 6.13: Real time factor comparison. Prior work RTF numbers were taken from [3].

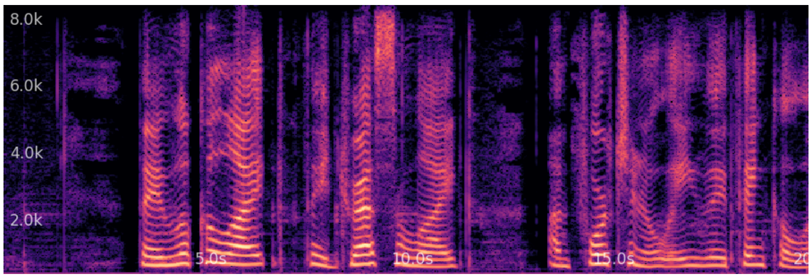
| Model | RTF (CPU)↓ | RTF (GPU)↓ |
|---|---|---|
| HiFi-GAN2 | - | 0.500 |
| FINALLY | - | 0.030 |
| Ours (1 CPU) | 1.340 | **0.023** |
| Ours (8 CPU) | 0.378 | **0.023** |



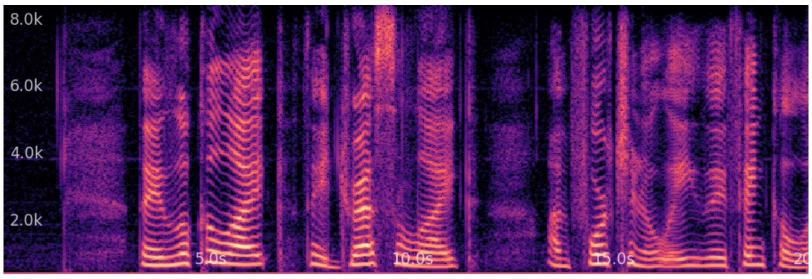
(a) Enhanced speech, trained from scratch.



(b) Enhanced speech, initialized with a pre-trained NAC.



(c) Enhanced speech, initialized with a pre-trained dual-branch model.



(d) Ground truth clean speech.

Figure 6.8: Comparison of initialization methods of dual-branch models trained in an unsupervised manner on the validation set.

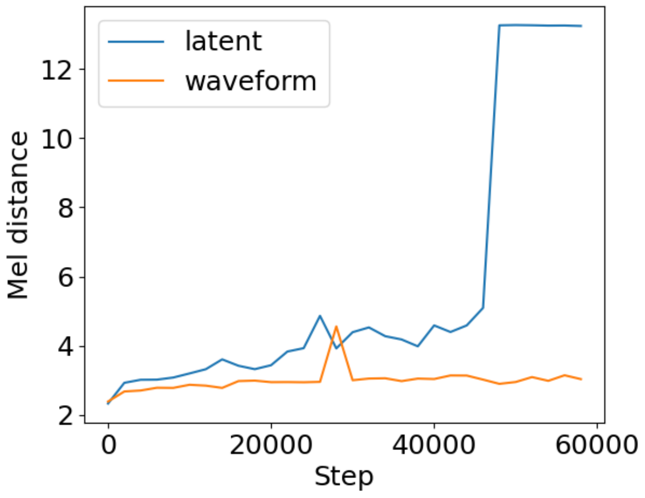


Figure 6.9: Comparison of latent and waveform combination methods of dual-branch models trained in an unsupervised manner.

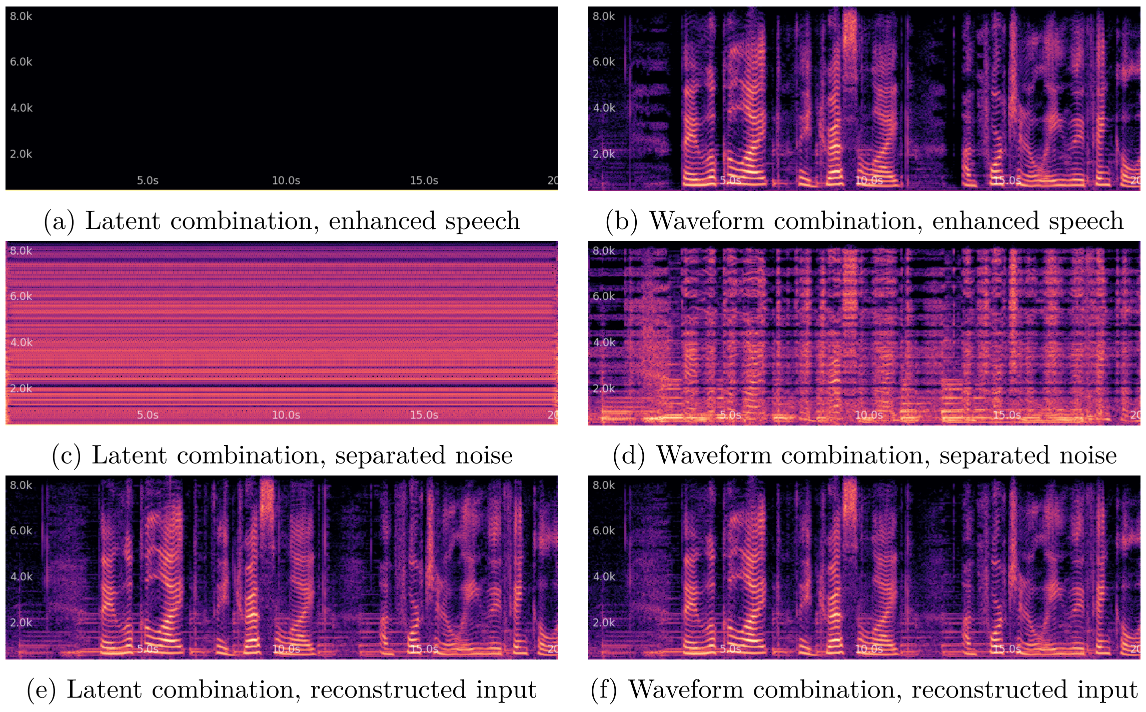
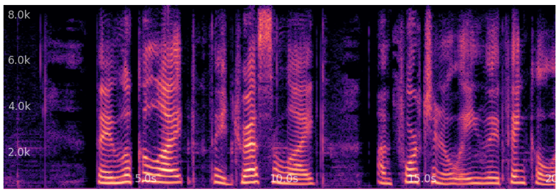
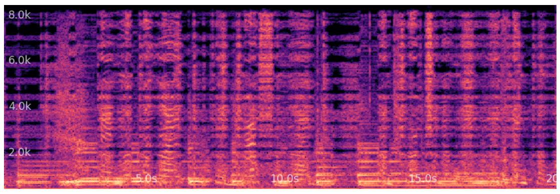


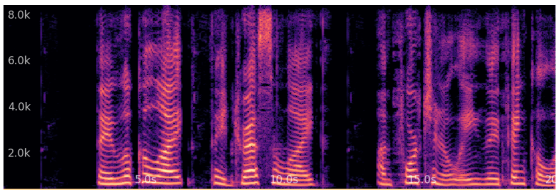
Figure 6.10: Comparison of latent and waveform combination methods of dual-branch models trained in an unsupervised manner.



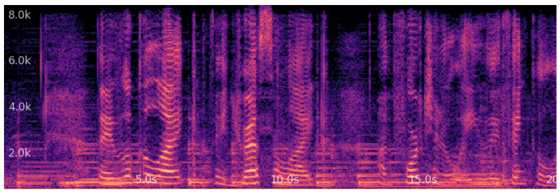
(a) All discriminators, enhanced speech



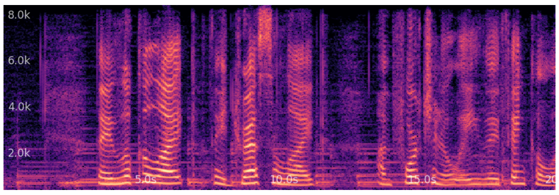
(b) All discriminators, estimated noise



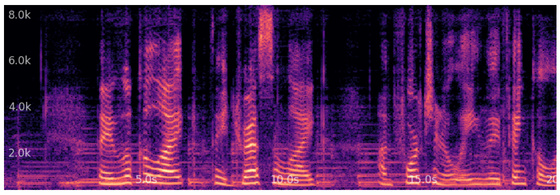
(c) Without noise discriminator, enhanced speech



(d) Without noise discriminator, estimated noise

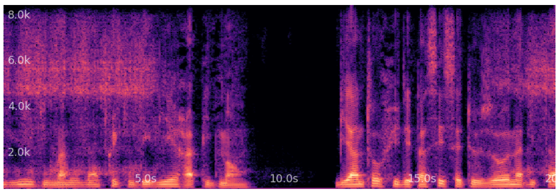


(e) Noisy speech discriminator only, enhanced speech

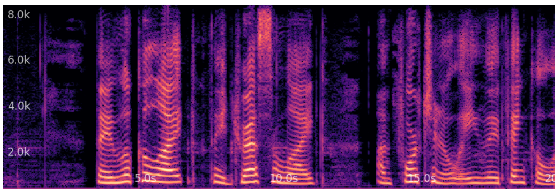


(f) Noisy speech discriminator only, estimated noise

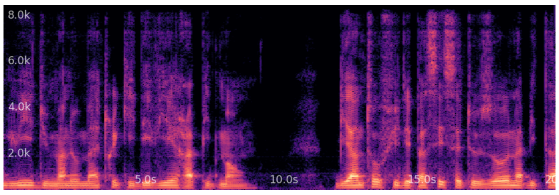
Figure 6.11: Comparison of dual-branch models trained in an unsupervised manner with different discriminators. The first row shows the model with all discriminators, the second row shows the model with clean and noisy speech discriminators, and the third row shows the model with only noisy speech discriminator.



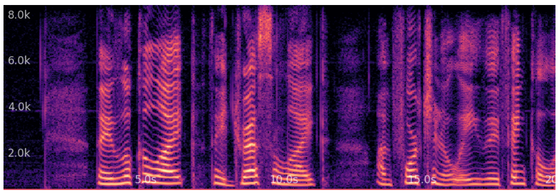
(a) Example 1, Enhanced speech.



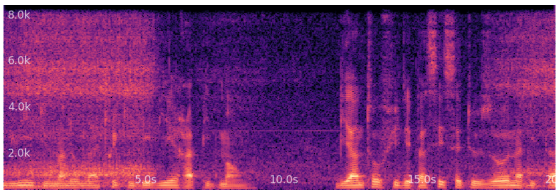
(b) Example 2, Enhanced speech.



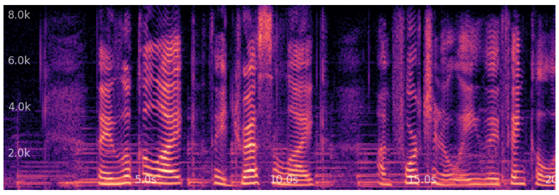
(c) Example 1, Ground truth clean speech.



(d) Example 2, Ground truth clean speech.

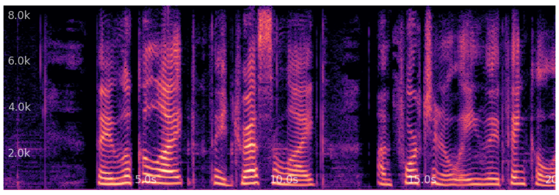


(e) Example 1, Input audio.

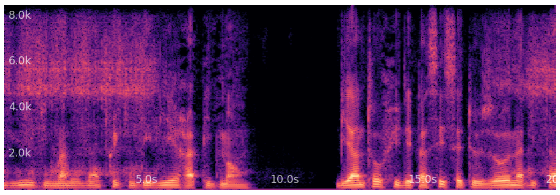


(f) Example 2, Input audio.

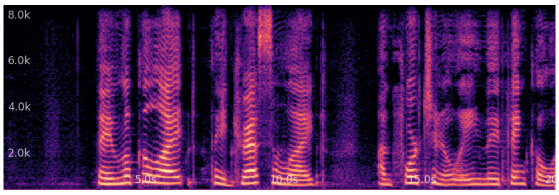
Figure 6.12: Spectrograms of enhanced, ground truth clean speech and input audio, showing leakage.



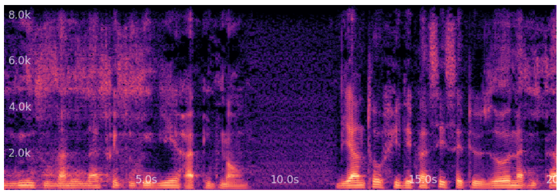
(a) Example 1, no VQ



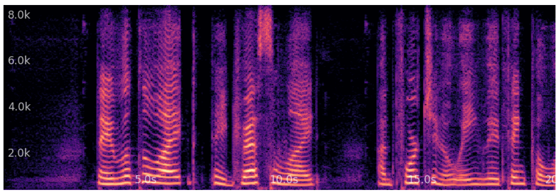
(b) Example 2, no VQ



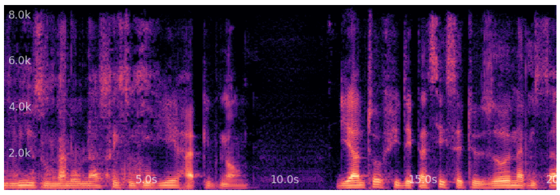
(c) Example 1, 8 Codebooks



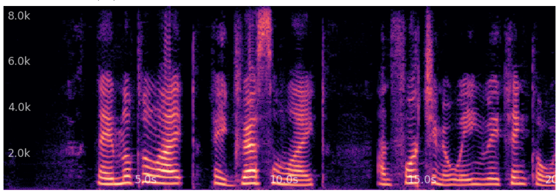
(d) Example 2, 8 VQ Codebooks



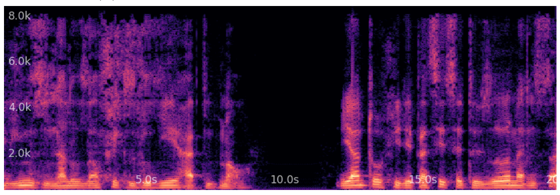
(e) Example 1, 4 VQ Codebooks



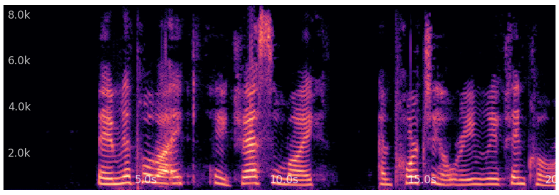
(f) Example 2, 4 VQ Codebooks



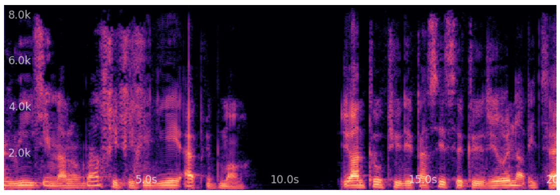
(g) Example 1, 2 VQ Codebooks



(h) Example 2, 2 VQ Codebooks

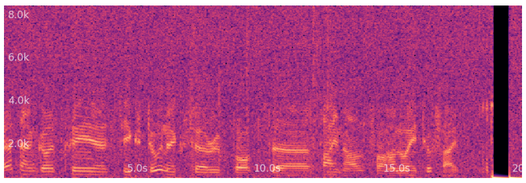


(i) Example 1, 1 VQ Codebook

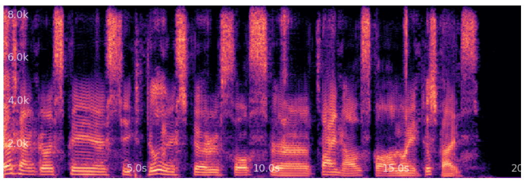


(j) Example 2, 1 VQ Codebook

Figure 6.13: Noise leakage comparison of dual-branch models trained in an unsupervised manner without VQ, with 8, 4, 2, and 1 VQ codebook.



(a) Noisy example.



(b) Enhanced example.

Figure 6.14: Example of noisy input audio and its enhanced version.

# Chapter 7

# Conclusion

In this thesis, we first explored a single-branch SE baseline—a NAC model trained for SE instead of audio reconstruction. Our baseline results proved the ability of the original NAC model to perform SE. Furthermore, we showed that adding a RoFormer model between the encoder and decoder improves the performance on all the metrics consistently and observed that it usually leaks less noise in low SNR scenarios compared to the original NAC model, mainly due to the employed self-attention module, allowing the model to utilize the entire context of the input audio. Also, adding VQ to the model did not degrade the perceptual performance. Furthermore, we explored the effect of initializing the encoder and decoder with parameters from a pre-trained NAC model trained for audio reconstruction. The results showed that initializing the model with a pre-trained NAC resulted in better convergence and final performance according to all the metrics, as the model does not need to learn the speech representations from scratch.

Furthermore, we showed that training the decoder using a fixed encoder from a pre-trained NAC model is sufficient to perform SE. This means that SE models do not necessarily produce latent features that are already noise-free, as the majority of the enhancement can happen in the decoder. Furthermore, we explored code stability when training with VQ and showed that training for SE instead of audio reconstruction results in more stable codes. This suggests that codes from an SE model might be better suited for speech downstream tasks and even provide some degree of robustness for free.

We then extended the single-branch model to a dual-branch model that separates the latent space into two parallel branches, applies transformations and quantization to each of them, and reconstructs two audio streams. The two streams are subsequently combined, and the model is trained to reconstruct the input audio. Furthermore, we introduced two branch-specific discriminators that utilize unpaired real clean speech and noise data pools to force information disentanglement. This way, we force one branch to produce noise-like and the other to produce clean speech-like signals. Combining discriminators with the overall input audio reconstruction ensures that the produced clean speech is the enhanced version of the input signal. This way, we could train the SE model without paired noisy-clean data. We also laid down a theoretical explanation to back our dual-branch model design and showed that the model indeed learns to enhance the noisy speech input.

We first experimented with supervised training of the dual-branch models. The results showed that the model could learn to separate clean speech and noise well. Also, we explored branch combination methods and concluded that waveform-level and latent-space combination methods performed similarly. We continued by exploring the models trained without supervision (i.e., unpaired noisy and clean speech data) and showed that the models enhanced the speech. However, the quality was worse compared to the supervised training as the model received less supervision about how the enhanced version of the input noisy speech should look. We then analyzed the effect of pre-training and, similarly to single-branch models, concluded that the more pre-training is done, the better the final performance. Subsequently, we explored the branch combination methods.

In contrast to the supervised training, we found that the latent-space combination method caused the model to diverge, as it did not force consistency between the enhanced speech and the input enough, proving that waveform-level combination is the key to success. Next, we performed ablations on the number of branch-wise discriminators used and observed that noise discriminator might not be necessary, as both with and without achieved similar results. However, training without noise discriminator leaked more clean speech to the noise branch. Also, we showed that the discriminators force the outputs to follow the corresponding real data distribution, as the model did not perform any information separation without the branch-wise discriminators, dividing the noise and clean speech information almost equally between the branches. Finally, we analyzed the leakage phenomenon (i.e., slight noise leaking into the clean speech) that occurs during unsupervised training and concluded that it might be happening due to imbalanced reconstruction and adversarial gradients or due to the discriminator not being able to capture all the nuances of clean speech.

Furthermore, we showed that our models are comparable to the prior work in supervised and unsupervised settings and can improve an already robust ASR system Whisper Large-V3, even with unsupervised training without VQ. We also showed that we could train a dual-branch model on real-world noisy data without supervision, which is not usually done by prior unsupervised SE works, proving that our method can learn from real-world noisy recordings. Lastly, we showed that our models can perform SE in real-time on CPU and GPU, making them suitable for real-world applications.

## Future work

We have proven that our dual-branch model can learn to enhance clean speech without paired noisy and clean speech data. However, the enhanced speech quality is still not state-of-the-art, mainly in the unsupervised setting. Also, we showed that the model can leak some noise into the clean speech, degrading the quality substantially.

As part of future work, we intend to investigate alternative encoder-decoder architectures based entirely on transformer neural networks, which have been shown to scale effectively and to compress audio into low bit-rate representations with minimal quality degradation. Additionally, we plan to transition from a purely time-domain approach to a time-frequency domain representation. This shift is expected to reduce computational demands and enable training on longer utterances.

We also aim to conduct a more in-depth exploration of the discriminator design. Specifically, we will seek architectural improvements that minimize noise leakage and enhance the fidelity of the reconstructed speech. Furthermore, we plan to investigate low-resource scenarios in which only limited paired noisy-clean data are available (i.e., data captured using close-talk microphones). In this context, we will examine the potential benefits of unsupervised pre-training and supervised fine-tuning. Conversely, we also intend to explore the utility of simulated data in improving model performance during unsupervised training.

Finally, the dual-branch framework proposed in this work can be extended to a multi-branch architecture. In such a configuration, each branch can be tasked with enhancing a different type of source (such as speech, musical instruments, or environmental noise) provided that these sources are distinguishable by distinct prior distributions. This approach is particularly relevant for music source separation, where individual data pools for each instrument can be provided, allowing the model to learn to isolate specific instruments. Beyond audio, we also envision extending this methodology to other modalities, including unsupervised image or video enhancement. This would position our framework as a general-purpose solution for unsupervised source separation across various domains.

# Bibliography

[1] Aggarwal, C. C.; Hinneburg, A. and Keim, D. A. On the Surprising Behavior of Distance Metrics in High Dimensional Spaces. In: *Proceedings of the 8th International Conference on Database Theory*. Berlin, Heidelberg: Springer-Verlag, 2001, p. 420–434. ICDT '01. ISBN 3540414568.

[2] Ardila, R.; Branson, M.; Davis, K.; Kohler, M.; Meyer, J. et al. Common Voice: A Massively-Multilingual Speech Corpus. In: Calzolari, N.; Béchet, F.; Blache, P.; Choukri, K.; Cieri, C. et al., ed. *Proceedings of the Twelfth Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, May 2020, p. 4218–4222. ISBN 979-10-95546-34-4. Available at: `https://aclanthology.org/2020.lrec-1.520/`.

[3] Babaev, N.; Tamogashev, K.; Saginbaev, A.; Shchekotov, I.; Bae, H. et al. *FINALLY: fast and universal speech enhancement with studio-like quality*. 2024. Available at: `https://arxiv.org/abs/2410.05920`.

[4] Baevski, A.; Zhou, Y.; Mohamed, A. and Auli, M. Wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. In: Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. and Lin, H., ed. *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2020, vol. 33, p. 12449–12460. Available at: `https://proceedings.neurips.cc/paper_files/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf`.

[5] Bando, Y.; Mimura, M.; Itoyama, K.; Yoshii, K. and Kawahara, T. Statistical Speech Enhancement Based on Probabilistic Integration of Variational Autoencoder and Non-Negative Matrix Factorization. In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, April 2018, p. 716–720. Available at: `http://dx.doi.org/10.1109/ICASSP.2018.8461530`.

[6] Bie, X.; Leglaive, S.; Alameda Pineda, X. and Girin, L. Unsupervised Speech Enhancement Using Dynamical Variational Autoencoders. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2022, vol. 30, p. 2993–3007.

[7] Boll, S. F. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on acoustics, speech, and signal processing*. IEEE, 1979, vol. 27, no. 2, p. 113–120.

[8] Cohen, I. and Berdugo, B. Speech enhancement for non-stationary noise environments. *Signal Processing*. Elsevier BV, november 2001, vol. 81, no. 11, p. 2403–2418.

[9] Consortium, L. D. *CSR-I (WSJ0) Complete* https://catalog.ldc.upenn.edu/LDC93S6A. 1993. Accessed: 2025-05-04.

[10] Cover, T. M. and Thomas, J. A. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. USA: Wiley-Interscience, 2006. ISBN 0471241954.

[11] Defferrard, M.; Benzi, K.; Vandergheynst, P. and Bresson, X. FMA: A Dataset for Music Analysis. In: *Proc. ISMIR*. 2017, p. 316–323. Available at: https://arxiv.org/abs/1612.01840.

[12] Défossez, A.; Copet, J.; Synnaeve, G. and Adi, Y. High Fidelity Neural Audio Compression. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. Available at: https://openreview.net/forum?id=ivCd8z8zR2. Featured Certification, Reproducibility Certification.

[13] Donahue, C.; McAuley, J. and Puckette, M. Adversarial Audio Synthesis. In: *International Conference on Learning Representations*. 2019. Available at: https://openreview.net/forum?id=ByMVTsR5KQ.

[14] Défossez, A.; Mazaré, L.; Orsini, M.; Royer, A.; Pérez, P. et al. *Moshi: a speech-text foundation model for real-time dialogue*. 2024. Available at: https://arxiv.org/abs/2410.00037.

[15] Ephraim, Y. and Malah, D. Speech enhancement using a minimum mean-square error short-time spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech, and Signal Processing*. IEEE, 1984, vol. 32, no. 6, p. 1109–1121.

[16] Esser, P.; Rombach, R. and Ommer, B. Taming transformers for high-resolution image synthesis. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, p. 12873–12883.

[17] Fonseca, E.; Favory, X.; Pons, J.; Font, F. and Serra, X. FSD50K: An Open Dataset of Human-Labeled Sound Events. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2022, vol. 30, p. 829–852. Available at: https://arxiv.org/abs/2010.00475.

[18] Fu, S.-W.; Tsao, Y.; Lu, X. and Kawai, H. MetricGAN: Generative adversarial networks based black-box metric scores optimization for speech enhancement. In: PMLR. *International Conference on Machine Learning*. 2019, p. 2031–2041.

[19] Fu, S.-W.; Yu, C.; Hsieh, T.-A.; Plantinga, P.; Ravanelli, M. et al. *MetricGAN+: An Improved Version of MetricGAN for Speech Enhancement*. 2021. Available at: https://arxiv.org/abs/2104.03538.

[20] Fu, S.-W.; Yu, C.; Hung, K.-H.; Ravanelli, M. and Tsao, Y. MetricGAN-U: Unsupervised Speech Enhancement/ Dereverberation Based Only on Noisy/ Reverberated Speech. In: *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2022, p. 7412–7416.

[21] Fujimura, T.; Koizumi, Y.; Yatabe, K. and Miyazaki, R. Noisy-target Training: A Training Strategy for DNN-based Speech Enhancement without Clean Speech. In: *2021 29th European Signal Processing Conference (EUSIPCO)*. 2021, p. 436–440.

[22] Fujimura, T. and Toda, T. Analysis Of Noisy-Target Training For Dnn-Based Speech Enhancement. In: *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2023, p. 1–5.

[23] Gomez, J. P. Z.; Veselý, K.; Szöke, I.; Blatt, A.; Motlicek, P. et al. ATCO2 corpus: A Large-Scale Dataset for Research on Automatic Speech Recognition and Natural Language Understanding of Air Traffic Control Communications. *Journal of Data-centric Machine Learning Research*, 2024. ISSN XXXX-XXXX. Available at: `https://openreview.net/forum?id=3CiWvVQVfw`.

[24] Goodfellow, I.; Pouget Abadie, J.; Mirza, M.; Xu, B.; Warde Farley, D. et al. Generative adversarial networks. *Communications of the ACM*. ACM New York, NY, USA, 2020, vol. 63, no. 11, p. 139–144.

[25] Harris, C. R.; Millman, K. J.; Walt, S. J. van der; Gommers, R.; Virtanen, P. et al. Array programming with NumPy. *Nature*. Springer Science and Business Media LLC, september 2020, vol. 585, no. 7825, p. 357–362. Available at: `https://doi.org/10.1038/s41586-020-2649-2`.

[26] Hendrycks, D. and Gimpel, K. *Gaussian Error Linear Units (GELUs)*. 2023. Available at: `https://arxiv.org/abs/1606.08415`.

[27] Kong, J.; Kim, J. and Bae, J. HiFi-GAN: generative adversarial networks for efficient and high fidelity speech synthesis. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2020. NIPS '20. ISBN 9781713829546.

[28] Kumar, K.; Kumar, R.; Boissiere, T. de; Gestin, L.; Teoh, W. Z. et al. MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis. In: Wallach, H.; Larochelle, H.; Beygelzimer, A.; Alché Buc, F. d'; Fox, E. et al., ed. *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019, vol. 32. Available at: `https://proceedings.neurips.cc/paper_files/paper/2019/file/6804c9bca0a615bdb9374d00a9fcba59-Paper.pdf`.

[29] Kumar, R.; Seetharaman, P.; Luebs, A.; Kumar, I. and Kumar, K. *High-Fidelity Audio Compression with Improved RVQGAN*. 2023. Available at: `https://arxiv.org/abs/2306.06546`.

[30] Langman, R.; Jukić, A.; Dhawan, K.; Koluguri, N. R. and Ginsburg, B. *Spectral Codecs: Spectrogram-Based Audio Codecs for High Quality Speech Synthesis*. 2024. Available at: `https://arxiv.org/abs/2406.05298`.

[31] Le Roux, J.; Wisdom, S.; Erdogan, H. and Hershey, J. R. SDR–half-baked or well done? *ArXiv preprint arXiv:1811.02508*, 2019.

[32] Li, C. T. and Farnia, F. Mode-Seeking Divergences: Theory and Applications to GANs. In: Ruiz, F.; Dy, J. and Meent, J.-W. van de, ed. *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*. PMLR, 25–27 Apr 2023, vol. 206, p. 8321–8350. Proceedings of Machine Learning Research. Available at: `https://proceedings.mlr.press/v206/ting-li23a.html`.

[33] Li, R.; Bao, C.; Xia, B. and Jia, M. Speech enhancement using the combination of adaptive wavelet threshold and spectral subtraction based on wavelet packet decomposition. In: *2012 IEEE 11th International Conference on Signal Processing*. 2012, vol. 1, p. 481–484.

[34] Lin, X.; Leglaive, S.; Girin, L. and Alameda Pineda, X. Unsupervised speech enhancement with deep dynamical generative speech and noise models. In: *Interspeech 2023*. 2023, p. 5102–5106. ISSN 2958-1796.

[35] Liu, B.; Liu, X.; Jin, X.; Stone, P. and Liu, Q. Conflict-Averse Gradient Descent for Multi-task learning. In: Ranzato, M.; Beygelzimer, A.; Dauphin, Y.; Liang, P. and Vaughan, J. W., ed. *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2021, vol. 34, p. 18878–18890. Available at: `https://proceedings.neurips.cc/paper_files/paper/2021/file/9d27fdf2477ffbff837d73ef7ae23db9-Paper.pdf`.

[36] Loizou, P. C. *Speech Enhancement: Theory and Practice*. 2ndth ed. USA: CRC Press, Inc., 2013. ISBN 1466504218.

[37] Loshchilov, I. and Hutter, F. SGDR: Stochastic Gradient Descent with Warm Restarts. In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. Available at: `https://openreview.net/forum?id=Skq89Scxx`.

[38] Loshchilov, I. and Hutter, F. *Decoupled Weight Decay Regularization*. 2019. Available at: `https://arxiv.org/abs/1711.05101`.

[39] Manocha, P.; Jin, Z. and Finkelstein, A. Audio similarity is unreliable as a proxy for audio quality. In: *Interspeech 2022*. ISCA: ISCA, September 2022, p. 3553–3557.

[40] Mao, X.; Li, Q.; Xie, H.; Lau, R. Y. K.; Wang, Z. et al. Least Squares Generative Adversarial Networks. In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, p. 2813–2821. Available at: `https://doi.org/10.1109/ICCV.2017.304`.

[41] McFee, B.; McVicar, M.; Faronbi, D.; Roman, I.; Gover, M. et al. *Librosa/librosa: 0.10.1*. Zenodo, 2023. Available at: `https://zenodo.org/record/8252662`.

[42] Oord, A. van den; Vinyals, O. and Kavukcuoglu, K. Neural discrete representation learning. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6309–6318. NIPS'17. ISBN 9781510860964.

[43] Pandey, A. and Wang, D. TCNN: Temporal convolutional neural network for real-time speech enhancement in the time domain. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, p. 6875–6879.

[44] Park, S. R. and Lee, J. M. A fully convolutional neural network for speech enhancement. In: *Interspeech*. 2017, p. 1993–1997.

[45] Parker, J. D.; Smirnov, A.; Pons, J.; Carr, C.; Zukowski, Z. et al. Scaling Transformers for Low-Bitrate High-Quality Speech Coding. In: *The Thirteenth International Conference on Learning Representations*. 2025. Available at: `https://openreview.net/forum?id=4YpMrGfldX`.

[46] Pascual, S.; Bonafonte, A. and Serrà, J. *SEGAN: Speech Enhancement Generative Adversarial Network*. 2017. Available at: `https://arxiv.org/abs/1703.09452`.

[47] Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E. et al. Automatic differentiation in PyTorch. In: *NIPS-W*. 2017.

[48] Pirklbauer, J.; Sach, M.; Fluyt, K.; Tirry, W.; Wardah, W. et al. Evaluation Metrics for Generative Speech Enhancement Methods: Issues and Perspectives. In: *Speech Communication; 15th ITG Conference*. 2023, p. 265–269.

[49] Plaut, E. *From Principal Subspaces to Principal Components with Linear Autoencoders*. 2018. Available at: `https://arxiv.org/abs/1804.10253`.

[50] Pratap, V.; Xu, Q.; Sriram, A.; Liptchinsky, V.; Synnaeve, G. et al. MLS: A Large-Scale Multilingual Dataset for Speech Research. In: *Proc. Interspeech*. 2020, p. 2757–2761. Available at: `https://www.isca-archive.org/interspeech_2020/pratap20_interspeech.html`.

[51] Prechelt, L. Early stopping - but when? In: *Lecture Notes in Computer Science*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, p. 55–69. Lecture notes in computer science.

[52] Radford, A.; Kim, J. W.; Xu, T.; Brockman, G.; McLeavey, C. et al. *Robust Speech Recognition via Large-Scale Weak Supervision*. 2022. Available at: `https://arxiv.org/abs/2212.04356`.

[53] Reddy, C. K. A.; Gopal, V. and Cutler, R. *DNSMOS: A Non-Intrusive Perceptual Objective Speech Quality metric to evaluate Noise Suppressors*. 2021. Available at: `https://arxiv.org/abs/2010.15258`.

[54] Reddy, C. K.; Gopal, V. and Cutler, R. DNSMOS: A Non-Intrusive Perceptual Objective Speech Quality Metric to Evaluate Noise Suppressors. *ArXiv preprint arXiv:2010.15258*, 2021.

[55] Research, M. *Deep Noise Suppression Challenge 5 (DNS5)* `https://github.com/microsoft/DNS-Challenge`. 2023. Accessed: 2025-05-04.

[56] Richter, J.; Welker, S.; Lemercier, J.-M.; Lay, B. and Gerkmann, T. Speech Enhancement and Dereverberation With Diffusion-Based Generative Models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023, vol. 31, p. 2351–2364.

[57] Richter, J.; Wu, Y.-C.; Krenn, S.; Welker, S.; Lay, B. et al. EARS: An Anechoic Fullband Speech Dataset Benchmarked for Speech Enhancement and Dereverberation. In: *Proc. Interspeech*. 2024. Available at: `https://www.isca-archive.org/interspeech_2024/richter24_interspeech.pdf`.

[58] Rix, A. W.; Beerends, J. G.; Hollier, M. P. and Hekstra, A. P. Perceptual evaluation of speech quality (PESQ)–a new method for speech quality assessment of telephone networks and codecs. *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2001, p. 749–752.

[59] Roux, J. L.; Wisdom, S.; Erdogan, H. and Hershey, J. R. SDR – Half-baked or Well Done? In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019, p. 626–630.

[60] Saeki, T. et al. UTMOS: UTokyo-SaruLab System for VoiceMOS Challenge 2022. In: *Proceedings of Interspeech*. 2022, p. 2022–2026.

[61] Saeki, T. et al. SpeechBERTScore: Reference-Aware Automatic Evaluation of Speech Generation Leveraging NLP Evaluation Metrics. *ArXiv preprint arXiv:2401.16812*, 2024.

[62] Salimans, T. and Kingma, D. P. Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks. In: Lee, D.; Sugiyama, M.; Luxburg, U.; Guyon, I. and Garnett, R., ed. *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2016, vol. 29. Available at: `https://proceedings.neurips.cc/paper_files/paper/2016/file/ed265bc903a5a097f61d3ec064d96d2e-Paper.pdf`.

[63] Shi, J.; Shim, H. jin; Tian, J.; Arora, S.; Wu, H. et al. VERSA: A Versatile Evaluation Toolkit for Speech, Audio, and Music. In: *2025 Annual Conference of the North American Chapter of the Association for Computational Linguistics – System Demonstration Track*. 2025. Available at: `https://openreview.net/forum?id=zU0hmbnyQm`.

[64] Su, J.; Ahmed, M.; Lu, Y.; Pan, S.; Bo, W. et al. RoFormer: Enhanced transformer with Rotary Position Embedding. *Neurocomputing*, 2024, vol. 568, p. 127063. ISSN 0925-2312. Available at: `https://www.sciencedirect.com/science/article/pii/S0925231223011864`.

[65] Su, J.; Jin, Z. and Finkelstein, A. HiFi-GAN-2: Studio-quality Speech Enhancement via Generative Adversarial Networks Conditioned on Acoustic Features. In: *WASPAA 2021*. October 2021.

[66] Taal, C. H.; Hendriks, R. C.; Heusdens, R. and Jensen, J. An algorithm for intelligibility prediction of time–frequency weighted noisy speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 2011, vol. 19, no. 7, p. 2125–2136.

[67] Ting, W.-Y.; Wang, S.-S.; Chang, H.-L.; Su, B. and Tsao, Y. Speech Enhancement Based on CycleGAN with Noise-informed Training. In: *2022 13th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. 2022, p. 155–159.

[68] Tishby, N.; Pereira, F. C. and Bialek, W. *The information bottleneck method*. 2000. Available at: `https://arxiv.org/abs/physics/0004057`.

[69] Valentini Botinhao, C. *Noisy speech database for training speech enhancement algorithms and TTS models*. University of Edinburgh. School of Informatics. Centre for Speech Technology Research (CSTR), 2017.

[70] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L. et al. Attention is All you Need. In: Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R. et al., ed. *Advances in Neural Information Processing Systems.* Curran Associates, Inc., 2017, vol. 30. Available at: `https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

[71] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L. et al. Attention is All you Need. In: Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R. et al., ed. *Advances in Neural Information Processing Systems.* Curran Associates, Inc., 2017, vol. 30. Available at: `https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

[72] Veaux, C.; Yamagishi, J. and MacDonald, K. *CSTR VCTK Corpus: English Multi-speaker Corpus for CSTR Voice Cloning Toolkit* `https://datashare.ed.ac.uk/handle/10283/2950`. 2017. Accessed: 2025-05-04.

[73] Wang, M.; Bai, X.; Zhang, C. and Zhong, Y. Unsupervised weak speech enhancement using periodic mixing invariant training. *Circuits Systems Signal Process.* Springer Science and Business Media LLC, may 2025.

[74] Wang, S.; Guan, H. and Long, Y. QMixCAT: Unsupervised Speech Enhancement Using Quality-guided Signal Mixing and Competitive Alternating Model Training. In: *Interspeech 2024.* 2024, p. 642–646. ISSN 2958-1796.

[75] Wang, Z.; Tan, Z.-H. and Yu, H. TSTNN: Two-stage transformer neural network for speech enhancement in the time domain. *IEEE Signal Processing Letters.* IEEE, 2021, vol. 28, p. 1080–1084.

[76] Weninger, F.; Erdogan, H.; Watanabe, S.; Vincent, E.; Le Roux, J. et al. Speech enhancement with LSTM recurrent neural networks and its application to noise-robust ASR. In: Springer. *International Conference on Latent Variable Analysis and Signal Separation.* 2015, p. 91–99.

[77] Wichern, G.; Le Roux, J.; Wisdom, S.; Erdogan, H. and Hershey, J. R. WHAM!: Extending Speech Separation to Noisy Environments. In: *Proc. Interspeech.* 2019, p. 1368–1372. Available at: `https://arxiv.org/abs/1907.01160`.

[78] Wisdom, S.; Tzinis, E.; Erdogan, H.; Weiss, R.; Wilson, K. et al. Unsupervised Sound Separation Using Mixture Invariant Training. In: Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. and Lin, H., ed. *Advances in Neural Information Processing Systems.* Curran Associates, Inc., 2020, vol. 33, p. 3846–3857. Available at: `https://proceedings.neurips.cc/paper_files/paper/2020/file/28538c394c36e4d5ea8ff5ad60562a93-Paper.pdf`.

[79] Wu, H.; Kanda, N.; Eskimez, S. E. and Li, J. *TS3-Codec: Transformer-Based Simple Streaming Single Codec.* 2025. Available at: `https://arxiv.org/abs/2411.18803`.

[80] Xiang, Y. and Bao, C. A Parallel-Data-Free Speech Enhancement Method Using Multi-Objective Learning Cycle-Consistent Generative Adversarial Network. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2020, vol. 28, p. 1826–1838.

[81] Xu, B.; Wang, N.; Chen, T. and Li, M. *Empirical Evaluation of Rectified Activations in Convolutional Network.* 2015. Available at: https://arxiv.org/abs/1505.00853.

[82] Xu, Y.; Du, J.; Dai, L. and Lee, C.-H. A regression approach to speech enhancement based on deep neural networks. In: IEEE. *IEEE/ACM Transactions on Audio, Speech, and Language Processing.* 2014, vol. 23, no. 1, p. 7–19.

[83] Yu, T.; Kumar, S.; Gupta, A.; Levine, S.; Hausman, K. et al. Gradient surgery for multi-task learning. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems.* Red Hook, NY, USA: Curran Associates Inc., 2020. NIPS '20. ISBN 9781713829546.

[84] Zeghidour, N.; Luebs, A.; Omran, A.; Skoglund, J. and Tagliasacchi, M. SoundStream: An End-to-End Neural Audio Codec. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.* IEEE Press, november 2021, vol. 30, p. 495–507. ISSN 2329-9290. Available at: https://doi.org/10.1109/TASLP.2021.3129994.

[85] Zen, H.; Dang, V. R.; Clark, R.; Zhang, Y.; Weiss, R. J. et al. LibriTTS: A Corpus Derived from LibriSpeech for Text-to-Speech. In: *Proc. Interspeech.* 2019, p. 1526–1530. Available at: https://www.isca-archive.org/interspeech_2019/zen19_interspeech.pdf.

[86] Zhang, W.; Scheibler, R.; Saijo, K.; Cornell, S.; Li, C. et al. URGENT Challenge: Universality, Robustness, and Generalizability For Speech Enhancement. In: *Interspeech 2024.* ISCA, September 2024, p. 4868–4872. Interspeech_2024. Available at: http://dx.doi.org/10.21437/Interspeech.2024-1239.

[87] Ziyin, L.; Hartwig, T. and Ueda, M. Neural Networks Fail to Learn Periodic Functions and How to Fix It. In: Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. and Lin, H., ed. *Advances in Neural Information Processing Systems.* Curran Associates, Inc., 2020, vol. 33, p. 1583–1594. Available at: https://proceedings.neurips.cc/paper_files/paper/2020/file/1160453108d3e537255e9f7b931f4e90-Paper.pdf.

[88] Żelasko, P.; Povey, D.; Trmal, J. Y. and Khudanpur, S. *Lhotse: a speech data representation library for the modern deep learning ecosystem.* 2021. Available at: https://arxiv.org/abs/2110.12561.

# Appendix A

# Derivation of Branch-wise Scaling

Let $\mathbf{x}_{NS} \in \mathbb{R}^T$ be a noisy speech, $\hat{\mathbf{x}}_{CS} \in \mathbb{R}^T$ an output of clean speech branch, and $\hat{\mathbf{x}}_N$ an output of noise branch.

We want to find $\alpha, \beta \in \mathbb{R}$, such that:

$$\alpha^*, \beta^* = \underset{\alpha,\beta}{\arg\min} \underbrace{\|\mathbf{x}_{NS} - (\alpha\hat{\mathbf{x}}_{CS} + \beta\hat{\mathbf{x}}_N)\|_2^2}_{L(\alpha,\beta)}. \tag{A.1}$$

To find $\alpha^*, \beta^*$, as the function $L(\alpha, \beta)$ is convex w. r. t. both parameters, we take the derivative w. r. t. input, set it to 0 and solve the system of equations:

$$\frac{\partial L}{\partial \alpha} = 2(\mathbf{x}_{NS} - \alpha\hat{\mathbf{x}}_{CS} - \beta\hat{\mathbf{x}}_N)^\top(-\hat{\mathbf{x}}_{CS}) = 0 \Leftrightarrow \alpha\hat{\mathbf{x}}_{CS}^\top\hat{\mathbf{x}}_{CS} + \beta\hat{\mathbf{x}}_N^\top\hat{\mathbf{x}}_{CS} = \mathbf{x}_{NS}^\top\hat{\mathbf{x}}_{CS}, \tag{A.2}$$

$$\frac{\partial L}{\partial \beta} = 2(\mathbf{x}_{NS} - \alpha\hat{\mathbf{x}}_{CS} - \beta\hat{\mathbf{x}}_N)^\top(-\hat{\mathbf{x}}_N) = 0 \Leftrightarrow \alpha\hat{\mathbf{x}}_{CS}^\top\hat{\mathbf{x}}_N + \beta\hat{\mathbf{x}}_N^\top\hat{\mathbf{x}}_N = \mathbf{x}_{NS}^\top\hat{\mathbf{x}}_N. \tag{A.3}$$

Hence, we obtain a system of linear equations:

$$\underbrace{\begin{bmatrix} \hat{\mathbf{x}}_{CS}^\top\hat{\mathbf{x}}_{CS} & \hat{\mathbf{x}}_N^\top\hat{\mathbf{x}}_{CS} \\ \hat{\mathbf{x}}_{CS}^\top\hat{\mathbf{x}}_N & \hat{\mathbf{x}}_N^\top\hat{\mathbf{x}}_N \end{bmatrix}}_{\mathbf{A}} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{NS}^\top\hat{\mathbf{x}}_{CS} \\ \mathbf{x}_{NS}^\top\hat{\mathbf{x}}_N \end{bmatrix}. \tag{A.4}$$

Finally, after multiplying both sides by $\mathbf{A}^{-1}$, we get:

$$\begin{bmatrix} \alpha^* \\ \beta^* \end{bmatrix} = \frac{1}{(\hat{\mathbf{x}}_{CS}^\top\hat{\mathbf{x}}_{CS})(\hat{\mathbf{x}}_N^\top\hat{\mathbf{x}}_N) - (\hat{\mathbf{x}}_N^\top\hat{\mathbf{x}}_{CS})^2} \begin{bmatrix} \hat{\mathbf{x}}_N^\top\hat{\mathbf{x}}_N & -\hat{\mathbf{x}}_{CS}^\top\hat{\mathbf{x}}_N \\ -\hat{\mathbf{x}}_{CS}^\top\hat{\mathbf{x}}_N & \hat{\mathbf{x}}_{CS}^\top\hat{\mathbf{x}}_{CS} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{NS}^\top\hat{\mathbf{x}}_{CS} \\ \mathbf{x}_{NS}^\top\hat{\mathbf{x}}_N \end{bmatrix}. \tag{A.5}$$