

Software Engineering Department

Braude College

**Capstone Project Phase B**

Speech Denoising: a Noise2Noise Approach

23-2-R-10

**Students**

Omri Cohen, [omri.cohen2@e.braude.ac.il](mailto:omri.cohen2@e.braude.ac.il)

Eliav Shabat, Eliav.shabat@e.braude.ac.il

**Supervisors**

Dr. Renata Avros

Prof. Zeev Volkovich

**Rafael Supervisor**

Eli Hundia

Content

[1. General Description 3](#_Toc164170632)

[1.1 Purpose 3](#_Toc164170633)

[1.2 Implementation 3](#_Toc164170634)

[1.3 Users 3](#_Toc164170635)

[2. Solution 3](#_Toc164170636)

[2.1 Structure 3](#_Toc164170637)

[2.2 System 4](#_Toc164170638)

[2.3 Flow 5](#_Toc164170639)

[2.4 Loss function 6](#_Toc164170640)

[2.5 Optimizer 7](#_Toc164170641)

[3. Research process 8](#_Toc164170642)

[3.1 Theoretical research 8](#_Toc164170643)

[3.2 Practical research 9](#_Toc164170644)

[4. Tools and Environments 9](#_Toc164170645)

[5. Challenges 10](#_Toc164170646)

6[. Results and Conclusions 11](#_Toc164170647)

[6.1 Goals and evaluations 11](#_Toc164170648)

[6.2 Train 1 12](#_Toc164170649)

[6.3 Train 2 14](#_Toc164170650)

[6.4 Train 3 15](#_Toc164170651)

[7. General conclusions 18](#_Toc164170652)

[8. Assessment of Project Benchmarks and Outcomes 19](#_Toc164170653)

[9. References 19](#_Toc164170654)

# General Description

## Purpose

The project focuses on creating an AI-based denoising system using Noise2Noise approach to improve the quality of audio data in noisy environments. This system is designed to filter out noise from audio samples, especially those recorded through radio communications, enhancing speaker identification accuracy. The objective is to support Rafael's ongoing efforts to apply machine learning for speaker recognition, ensuring clear audio for effective communication and analysis in environments with high noise levels.

## Implementation

The implementation based on Deep Complex U-Net architecture combined with Noise2Noise approach. This strategy enables effective model training by focusing on enhancing key metrics like Signal-to-Noise Ratio (SNR) and Perceptual Evaluation of Speech Quality (PESQ). The framework of the project covers implementation of the neural network model, the mathematical complexities involved in denoising process, dataset creation and training. By integrating the audio processing techniques with deep learning model, the project aims to meet its objectives efficiently.

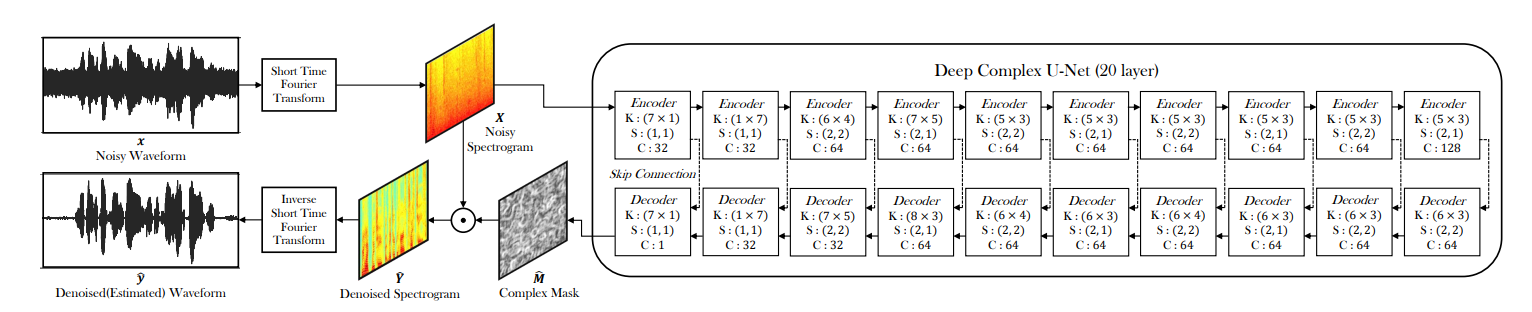
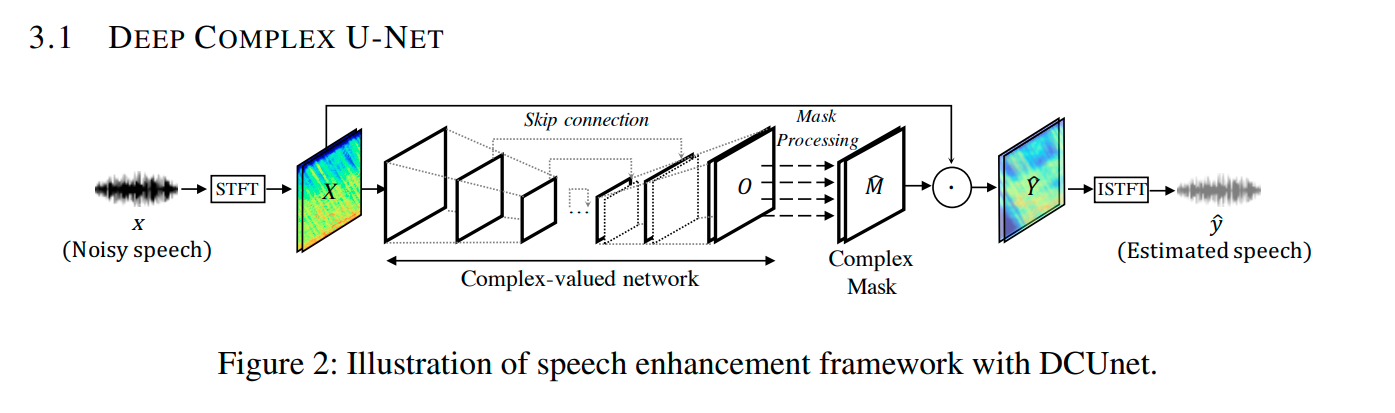
## Users

The primary users of this system are engineers and other professionals in the defense sector, especially those tasked with identifying speakers from audio recordings, such as in military radio communications.

# Solution

## Structure

The core structure of the proposed solution is centered around the Deep Complex U-Net (DCUnet-20) architecture. This advanced model is an extension of the traditional U-Net architecture but is uniquely designed to handle complex-valued data, allowing it to process both magnitude and phase information from audio spectrograms. This complex-valued masking framework is critical for achieving high fidelity in speech enhancement by accurately reconstructing the audio signals from noisy inputs. The DCUnet-20 architecture's design includes 20 layers and utilizes complex convolutional layers, batch normalization, and activation functions to maintain the integrity of the complex data throughout the network.



## System

The system operates on a Noise2Noise (N2N) approach, which is fundamentally different from traditional noise reduction techniques that typically require clean audio samples for reference during the training phase. Instead, the N2N method employs a strategy where the network learns to map noisy inputs directly to denoised outputs without the need for clean target data. This is particularly beneficial in real-world scenarios where acquiring clean audio samples is challenging. The training process leverages noisy data as both input and target, enabling the model to effectively predict and eliminate noise from the audio signals, thus enhancing the clarity and intelligibility of the output.

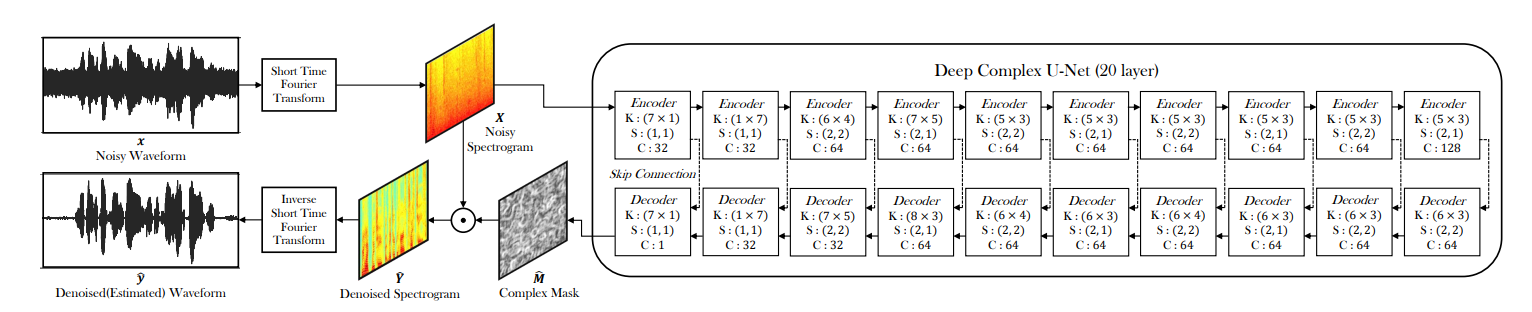
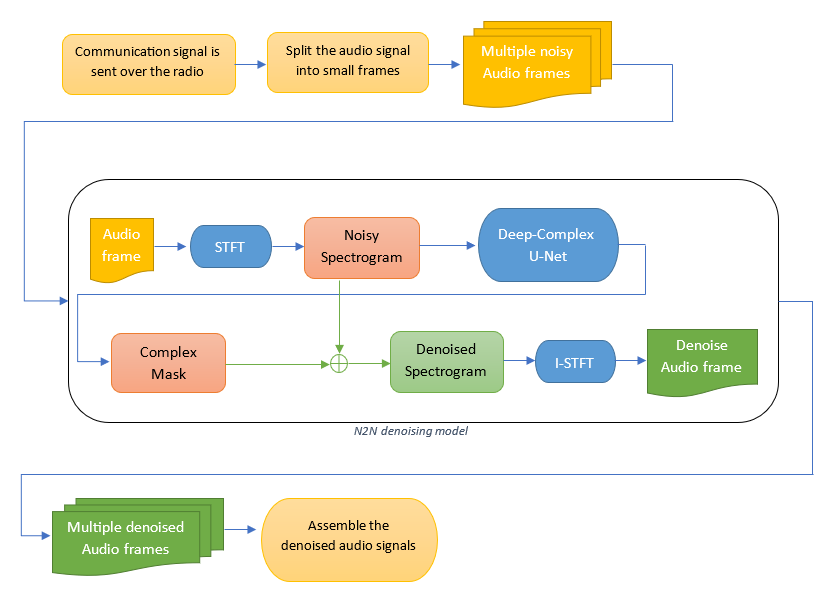


Figure 3 Image represents a use of U-Net as the core of the denoising process

## Flow

To effectively reduce noise in audio signals, we employ the DCUnet-20 architecture, an advanced version of the U-Net model tailored to handle complex data. This process involves several crucial steps designed to enhance the audio quality at every stage. Starting with the initial transformation of audio into a format suitable for analysis, the method progresses through detailed processing. It continuously improves to achieve the best possible clarity. Below is an overview of each phase in this meticulous denoising process:

Figure 4 Solution Flow



* **Conversion to Spectrograms**:
  + The denoising process initiates by converting speech samples into spectrograms, visually representing the audio signal's frequency spectrum.
  + This conversion is essential for the DCUnet-20 architecture to analyze and process the embedded complex information within noisy audio data.
* **Processing Through Convolutional Layers**:
  + Each spectrogram passes through the model’s complex-valued convolutional layers.
  + Multiple kernels within these layers filter and transform the data, isolating and removing noise components.
* **Utilization of Parameters During Training**:
  + During the training phase, the model employs parameters such as Signal-to-Noise Ratio (SNR) and Perceptual Evaluation of Speech Quality (PESQ).
  + These metrics guide the model’s learning process, helping it effectively distinguish between noise and the speech signal.
  + This distinction significantly enhances the clarity and quality of the output audio.
* **Refinement Through Network Layers**:
  + As the audio spectrogram progresses through the network, each layer refines its interpretation of the data, focusing on different aspects of the audio signal.
  + Skip connections within the U-Net architecture transfer information directly across layers, preserving crucial details for accurate noise reduction.
* **Iterative Model for Continuous Evaluation**:
  + The iterative model continuously evaluates the system's performance, adopting a systematic approach to refine the denoising capabilities.
  + Through ongoing cycles of assessment and refinement, the system progressively improves, effectively minimizing background noise while preserving the original audio's quality.
  + This process ensures that the output remains crisp and accurate, suitable for various applications where superior audio quality is crucial.

## Loss function

The loss function using the Weighted Signal-to-Distortion Ratio (wSDR) to evaluate the performance of the model. The implementation includes functions to resample audio signals to different sample rates and to evaluate various audio quality metrics using a neural network model. The key function calculates the wSDR, which is a metric designed to assess the quality of audio signals, particularly useful in scenarios such as audio source separation or enhancement. It is a more comprehensive measure than the traditional Signal-to-Distortion Ratio (SDR) because it accounts for both the fidelity of the target signal and the suppression of the interference (noise).

The loss calculation process includes several steps:

1. Resampling: Adjustment of the sample rate of audio data using linear interpolation. This is crucial when the input and output sample rates differ, ensuring that the audio data fits the expected format of the model or the evaluation metrics.

2. wSDR Computation:

* Convert audio signals that initially in the frequency domain back to the time domain using the Inverse Short-Time Fourier Transform (ISTFT).
* Calculate the actual target signal and the noise
* Computed the energy ratio of the target signal relative to the combined energy of the target and the noise.
* Calculate the wSDR as a weighted sum of the SDRs for the target and the noise, where the weights are based on the energy ratio.

3. Audio Quality Metrics Calculation:

* Processes audio data and computes metrics such as PESQ (Perceptual Evaluation of Speech Quality), SNR (Signal-to-Noise Ratio), SSNR (Segmental SNR), and STOI (Speech Intelligibility Index).
* The metrics are aggregated to provide an overall assessment of the audio quality.

The Weighted Signal-to-Distortion Ratio (wSDR) loss function offers a comprehensive evaluation of audio quality by considering both the accuracy of the predicted target signal and the effectiveness of noise suppression, which is particularly beneficial in complex audio environments. The model's ability to resample audio signals ensures seamless operation across different audio contexts and standards, enhancing its versatility. Additionally, the inclusion of various metrics such as PESQ, SNR, SSNR, and STOI provides a broad perspective on audio quality, enabling detailed analysis and comparisons of audio enhancement techniques. The practical implementation of metric calculation, leveraging a neural network and direct audio manipulation, suits real-world applications where diverse audio samples and conditions are common. Together, this structured and detailed approach not only boosts the performance assessment of denoising tasks but also aids in the development of more robust and effective audio processing technologies.

## Optimizer

The optimizer used in the model is Adam, which stands for Adaptive Moment Estimation. Adam is a popular choice in training deep learning models because it combines the advantages of two other extensions of stochastic gradient descent: Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp). Essentially, it adjusts the learning rate of each parameter based on estimates of first and second moments of the gradients. This adjustment helps in handling sparse gradients on noisy problems, common in many deep learning applications.

The model use Adam optimizer during training, for each epoch, and for each batch within the epoch as follows:

* Zero Gradient: The gradients are zeroed out to prevent accumulation of gradients from previous forward passes, which can adversely affect the learning.
* Forward Pass: The model makes predictions based on the noisy input data.
* Loss Calculation: wSDR function computes the loss between the predicted and true outputs.
* Backward Pass: The loss is propagated back through the network by allowing the calculation of gradients for each parameter.
* Parameter Update: Updates the model parameters based on the current gradients. This step is where Adam's adaptive learning rate mechanism comes into play, adjusting each parameter based on its estimated first and second moments of the gradients.
* Additionally, a learning rate scheduler (`StepLR`) is employed to adjust the learning rate at each epoch by multiplying it with a predefined factor after a specified number of epochs. This helps in fine-tuning the model as training progresses.

Adam optimizer is highly regarded for its efficiency in converging, as it adapts learning rates for different parameters, typically resulting in faster convergence compared to other stochastic optimization methods. It excels across a wide range of deep learning applications, especially those involving large datasets or instances with noisy or sparse gradients. Adam automates learning rate tuning by adjusting rates based on the averages of recent gradient magnitudes, enhancing performance in practice. Despite its sensitivity to the initial learning rate setting, Adam is less affected by other hyperparameters, offering robustness and versatility for a variety of deep learning tasks. These features make Adam an exceptionally reliable and effective choice for training deep neural networks, as demonstrated in complex audio signal processing tasks.

# Research process

## Theoretical research

The research process began with defining the problem alongside the RAFAEL team. The initial step involved exploring various machine learning architectures known for their efficacy with audio files, signals, and frequency, including RNN, CNN, and UNET. Concurrently, challenges such as limited human resources and the difficulty of acquiring clean recordings for training were assessed. Given these constraints, the U-Net architecture, paired with the Noise2Noise approach, was selected for its ability to train models effectively without clean data, achieving results comparable to those trained with clean recordings. Subsequent phases involved a thorough review of literature on audio denoising, understanding the necessary architecture, and assembling the tools required for model implementation, training, and execution. Additionally, further investigations addressed critical aspects relevant to the project:

* Audio structures – nparrays, tensors, waveforms, frequency
* Mainly evaluation parameters – SNR, PESQ
* Suitable development environment – Required libraries, memory needs, GPU etc.
* Datasets – number/length of samples, data augmentation

At the conclusion of the research process, a theoretical document, referred to as Part A of the project book, was composed. The document provided a theoretical basis intended for practical application in Part B. Throughout the implementation phase, the focus shifted to fine-tuning neural network configurations, including adjustments to kernels and modifications to layer structures, and to enhancing model training through careful selection and management of datasets.

## Practical research

This section details a practical study that involved developing various datasets, modifying the model, training it, and analyzing the outcomes. Datasets were generated using diverse software and scripts to evaluate the model's performance across various noise types and levels. Adjustments were made to the kernels and layers tasked with learning frequency, speech, and noise characteristics. The model underwent multiple training iterations, with each session followed by a thorough analysis of the results. These evaluations aimed to refine the model progressively, moving closer to the objective of effectively removing noise from radio communications.

# Tools and Environments

In this project, we employed a comprehensive toolkit, including both hardware and software components, to develop and refine our denoising AI model. Initially, we utilized Visual Studio Code coupled with Git for version control, ensuring our project's progress remained consistent and independent of the workstation used. Our development began on a local computer running WSL (Ubuntu) paired with an NVIDIA GTX 4070ti GPU, leveraging the CUDA toolkit for enhanced processing capabilities. As the project evolved, we transitioned to AWS EC2 instances equipped with Tesla T4 GPU, providing the computational power necessary for more extensive training sessions.

Our methodology included the creation of Python scripts for data pre-processing, enabling us to prepare the audio samples for training and generate new, denoised samples from each audio file. Specifically, we developed:

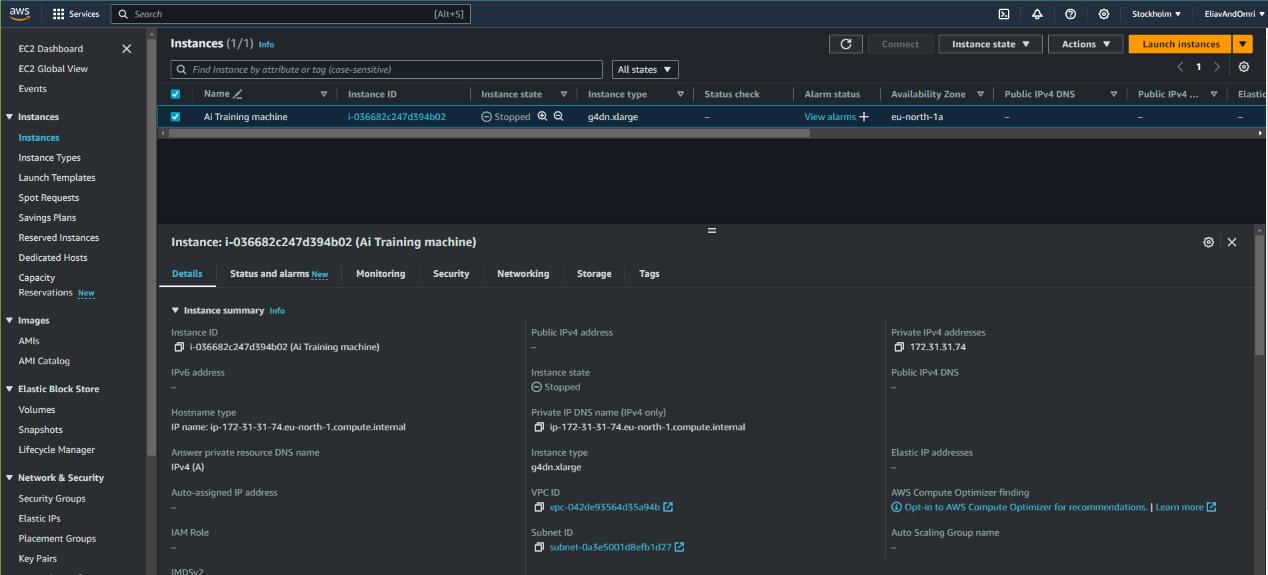


Figure 5 AWS Dashboard

* **audio\_recorder.py**: A script for recording and saving audio files, crucial for accumulating raw data.
* **audio\_recorder\_gui.py**: An interface script providing a graphical user interface for the audio recorder, enhancing user interaction.
* **generate\_file\_to\_merge\_list.py**: This script was designed to prepare lists of audio files for merging, facilitating batch processing.
* **get\_snr\_spectrogram\_analysis.py**: A utility for analyzing signal-to-noise ratio (SNR) and spectrograms, essential for evaluating audio quality.
* **split\_audio\_files.py**: A tool to split audio files into smaller segments, aiding in the creation of a more diverse dataset.

For the backend and data manipulation, we relied on a suite of Python libraries, installed via pip, including **sox** for Linux backend support, **torchaudio**, **torch** for deep learning, **pydub** for audio file manipulation, **numpy** for numerical computations, and **matplotlib**, **scipy** for data visualization and scientific computing respectively. Furthermore, **pesq** and **pystoi** were utilized for audio quality and intelligibility measurement, **numba** for accelerating Python functions, **tqdm** for progress bars, **IPython** for interactive computing, and **pysoundfile** and **soundfile** for reading and writing sound files.

This integrated approach, combining robust computational resources with tailored scripts and a broad range of Python libraries, allowed us to efficiently process and denoise audio data, ultimately leading to the successful development of the AI denoising model. The interface with the client was managed through intuitive scripts like **audio\_recorder\_gui.py**, ensuring ease of use and accessibility for end-users looking to contribute data or utilize the denoising capabilities of our model.

# Challenges

In developing our AI model for denoising audio signals, we faced several challenges that improved our problem-solving and technical abilities:

* **Scarcity of Human Resources**: A major challenge was the limited availability of personnel needed to record and label a large dataset of audio samples. We addressed this by developing automated scripts for recording and labeling, which boosted our efficiency and improved the accuracy of our data.
* **Sourcing Compatible Noises**: Aligning the noises with the model’s requirements and the assumptions of our project's foundational article was critical. We conducted a detailed search and selection process to ensure the noises were compatible and effective for training.
* **Computational Resource Constraints**: Initially, model training was conducted on a private computer with an NVIDIA GPU, leading to training sessions lasting 5-6 hours. To enhance our efficiency, we switched to using Amazon's EC2 equipped with Tesla T4 GPUs. This not only shortened our training times but also allowed us to work remotely.
* **Familiarization with Audio Signal Processing**: Transitioning from image and text-based models to audio processing presented a steep learning curve. We needed to quickly grasp concepts like Signal-to-Noise Ratio (SNR), frequency domain analysis, and spectrograms to evaluate and refine our model effectively.

These challenges were daunting but also invaluable learning opportunities. Through dedicated study and practical application, we developed a robust understanding of audio signal processing. This knowledge, combined with our automated scripts, empowered us to fine-tune our model more precisely and interpret the outcomes effectively, marking substantial progress in our project.

# Results and Conclusions

## Goals and evaluations

The main goal of the project is to remove noise from radio communication recordings for future use in speaker identification. This objective was evaluated through various methods during the model's training process: listening to recordings, analyzing waveform plots for direct comparisons, and utilizing parameters like Signal-to-Noise Ratio (SNR) to assess audio quality and noise removal. While SNR analyses helped gauge the effectiveness of the noise reduction, achieving this goal required iterative refinement and adjustments based on feedback from these evaluations.

A secondary goal of the project was to train a model using only noisy datasets to simplify the dataset creation/search process. Initial training sessions used small datasets sourced from the Internet to familiarize with the model’s training process and necessary adjustments. These early results indicated some success in noise filtering, though the SNR was poor and sometimes negative, suggesting the model was not yet well-adapted to radio communication recordings. This realization led to the decision to train the model on a larger, more specialized dataset.

The project culminated in a model that effectively removes noise and meets the expected standards, serving both Rafael's requirements and our project objectives. The model's training solely on noisy recordings successfully demonstrated this approach's viability. However, it was observed that extremely noisy recordings or those with strongly interfered speech did not achieve optimal SNR. The team believes that with further training, these challenges can also be overcome, thus further fulfilling the project's goals. The following section of the book details the next three primary training sessions we conducted and their outcomes.

## Train 1

The initial training session specifically targeted at white noise, aimed to assessing feasibility, familiarizing with training requirements, and conducting preliminary evaluation of the model's effectiveness. The dataset comprised 20,000 samples, each noised with white noise and ranging in duration from 3 to 7 seconds.

Outcomes of noise reduction from a recording featuring an English speaker with high level white noise in the background:





* Initial SNR between noisy and clean recordings: 6.959dB
* Improved SNR after denoising process between denoised and clean recordings: 18.475dB
* Improvement of 11.516dB

Figure 7 Denoised audio recording spectrogram – The black parts are the denoised areas. Most of the noise removed successfully

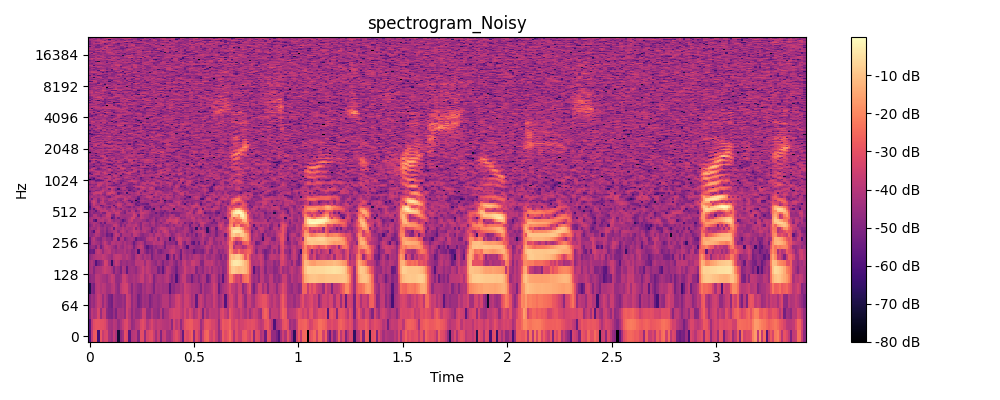
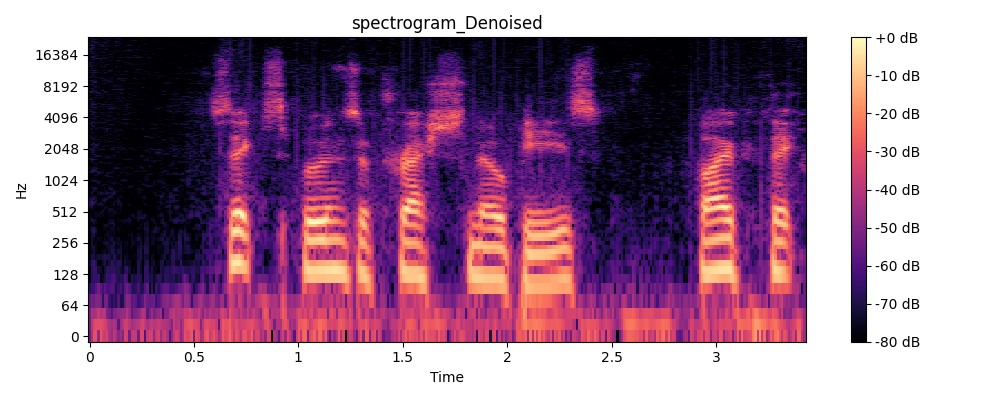


Figure 6 Noisy audio recording spectrogram

Figure 11 Combined waveforms - The denoised waveform (green) closely follows the clean waveform (orange) indicating successful denoising (blue)

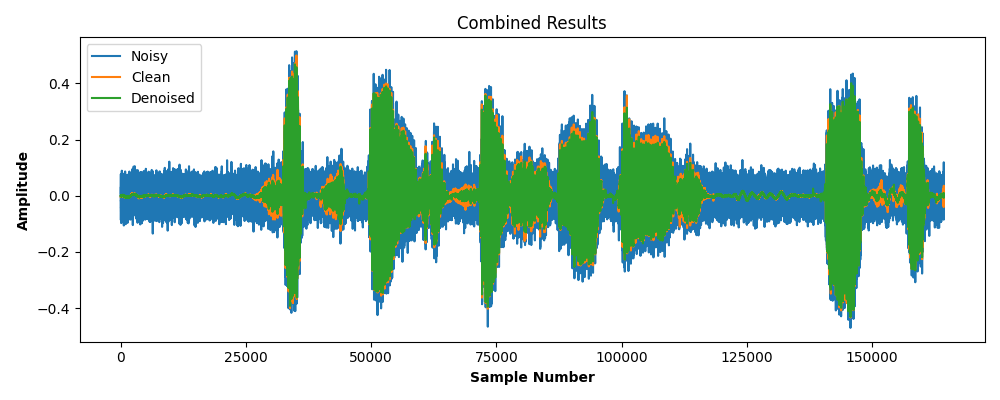


Figure 10 Denoised waveform

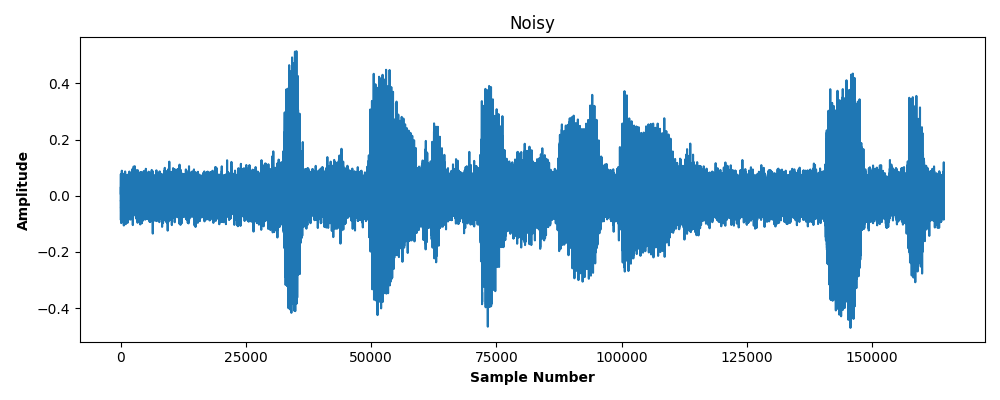
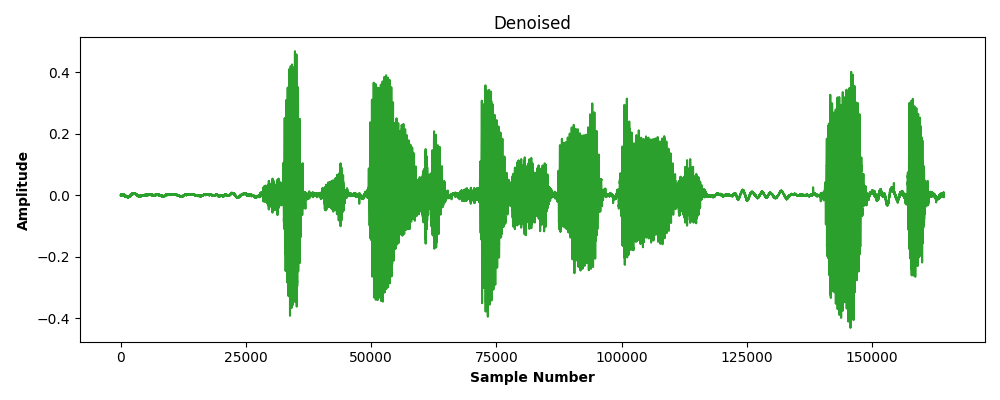


Figure 9 Noisy waveform

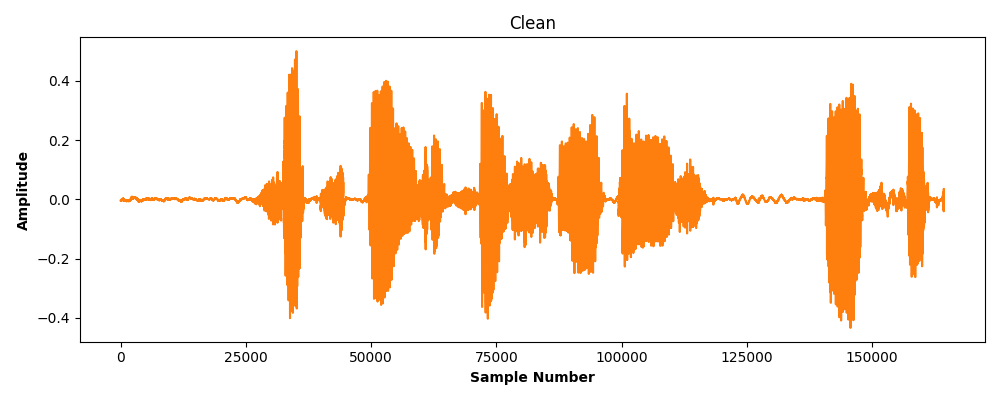


Figure 8 Clean waveform

## Train 2

The next training session targeted at radio communication signals for the purpose of training the model on task-specific noises. The dataset comprised 7,000 samples. Each sample was artificially tuned to the frequency of a radio communication using audacity software

Outcomes of noise reduction from a recording featuring a Russian speaker with gunshot noise in the background:





* Initial SNR between noisy and clean recordings: -4.209dB
* Improved SNR after denoising process between denoised and clean recordings: -3.288dB
* Improvement of 0.921dB

Figure 13 Denoised audio recording spectrogram – The black parts are the denoised areas. Only some of the noise removed.

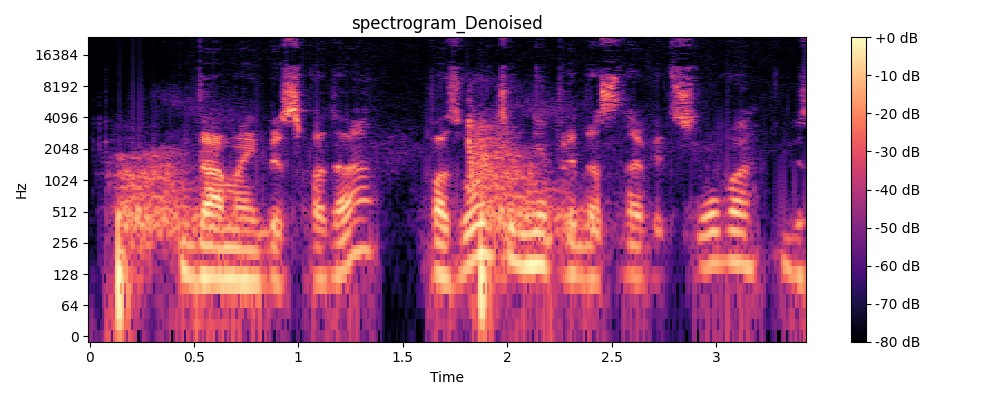


Figure 12 Noisy audio recording spectrogram

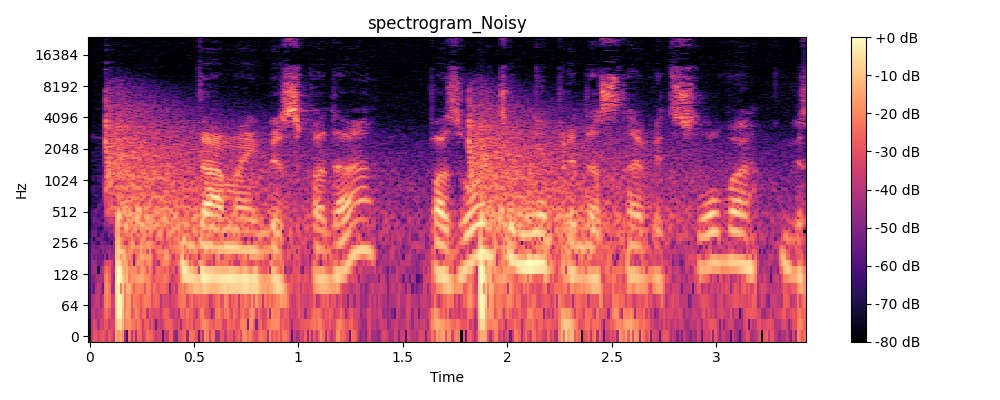


Figure 17 Combined Waveforms - The denoised waveform (green) does not follow the clean waveform (orange) well indicating an unsuccessful denoising.

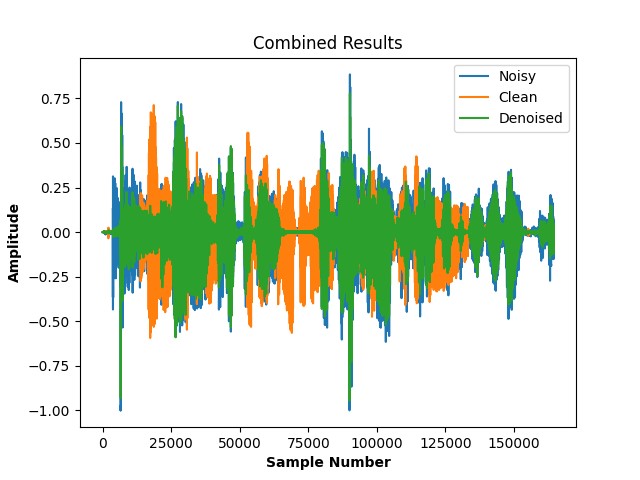


Figure 16 Denoised waveform

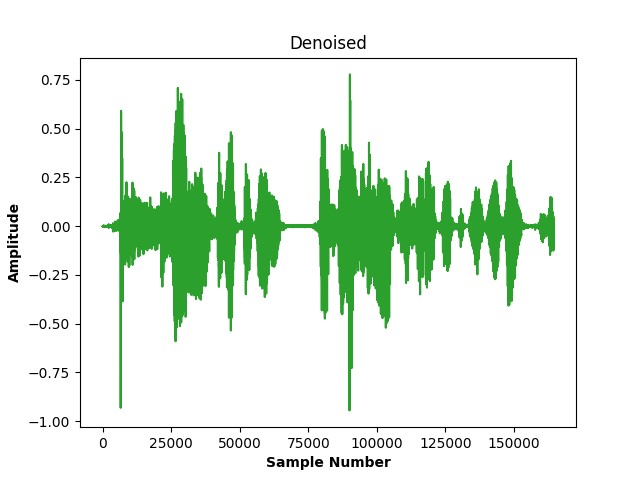


Figure 15 Noisy waveform

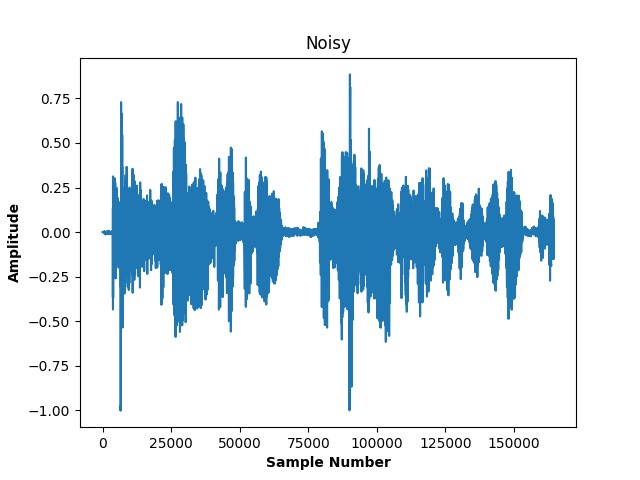
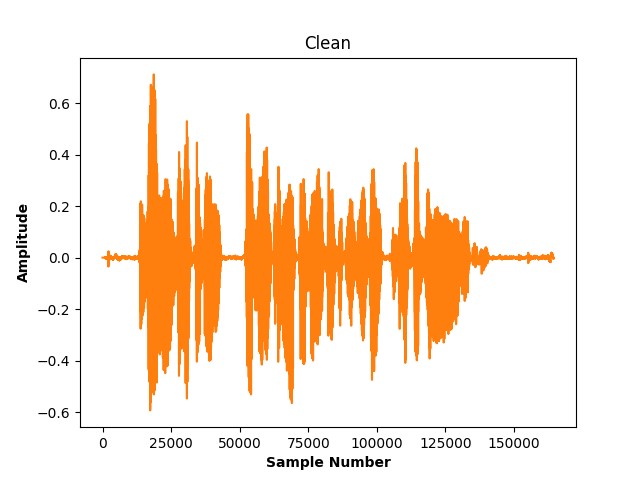


Figure 14 Clean waveform



After reviewing the outcomes of this training session and conducting further research on the dataset used, it was concluded that training with 7,000 samples was insufficient. Additionally, the model requires training on various background noises to enhance its noise removal capabilities.

## Train 3

The last training session targets noises of different types to make the model learn to denoise diverse background noises (radio communication, white noise, gunshots, etc.). The data set consisted of 23,000 samples. Each sample consisted of a speech segment noised with a different noise.

Results of noise reduction from a recording that includes an English speaker with train stops noise in the background:





* Initial SNR between noisy and clean recordings: -1.119dB
* Improved SNR after denoising process between denoised and clean recordings: 10.005dB
* Improvement of 11.124dB

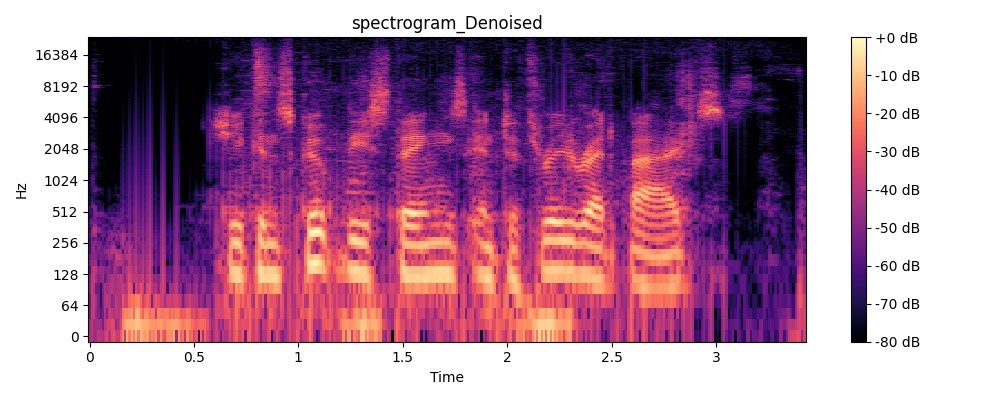


Figure 19 Denoised audio recording spectrogram – The black parts are the denoised areas. Most of the noise removed successfully

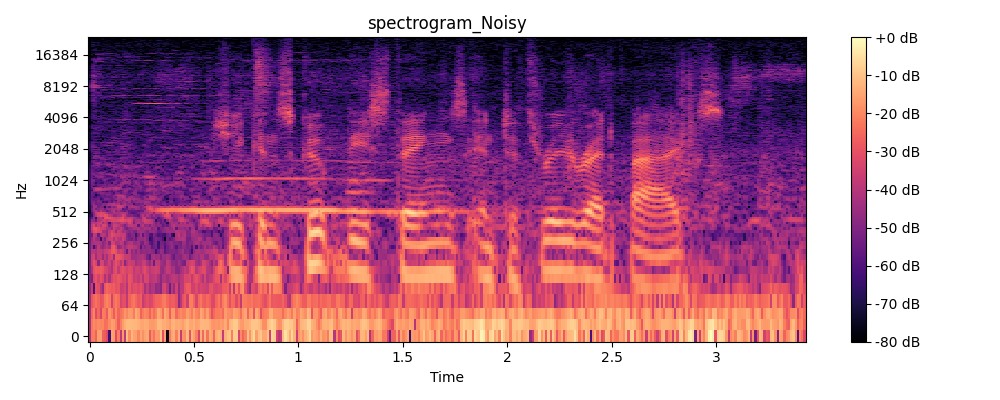


Figure 18 Noisy audio recording spectrogram

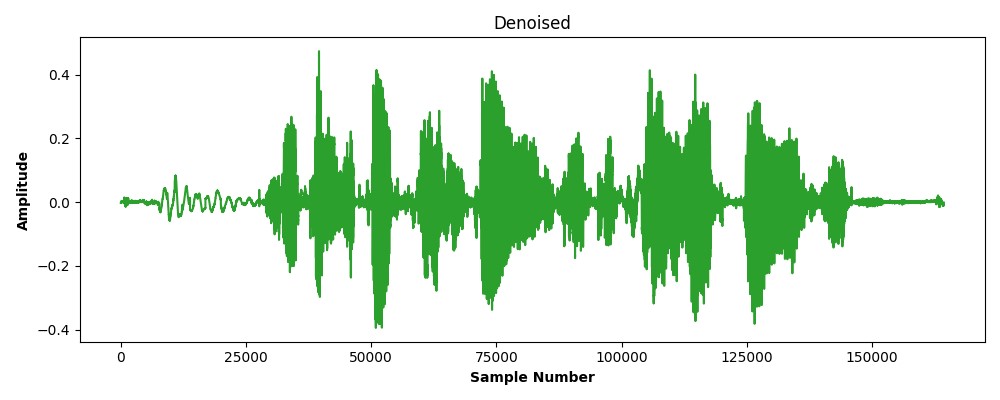


Figure 21 Denoised waveform

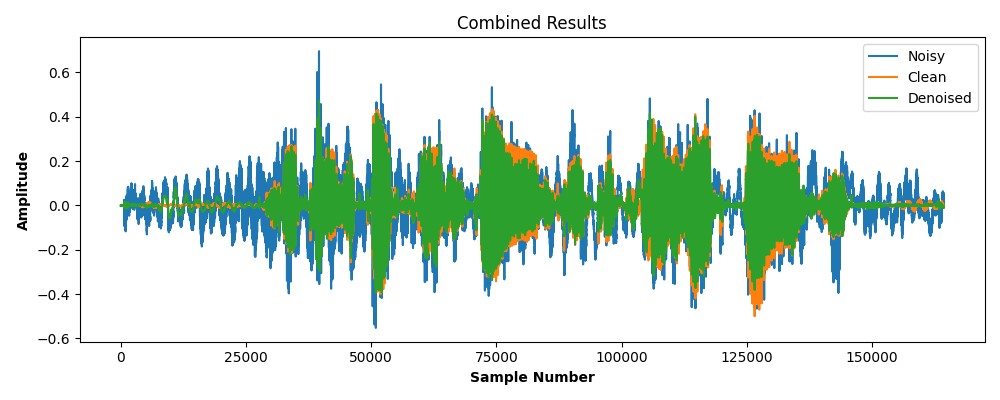


Figure 22 Combined waveforms - The denoised waveform (green) closely follows the clean waveform (orange) indicating successful denoising (blue)

Figure 20 Noisy waveform

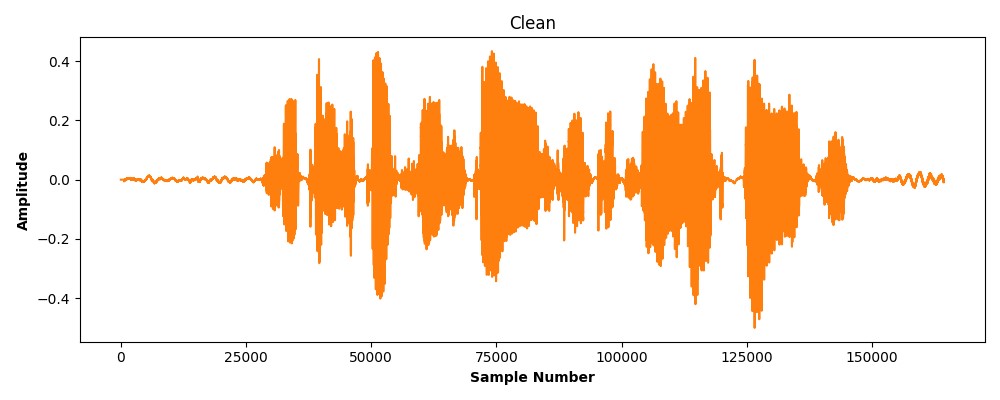
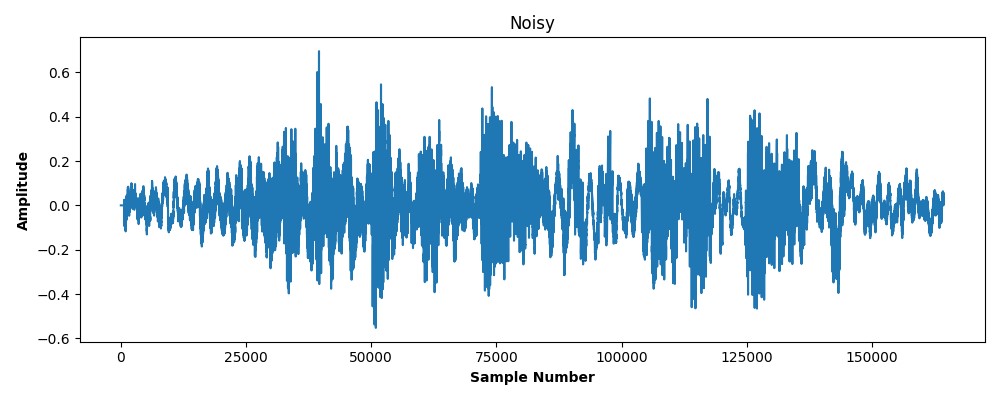


Figure 20 Clean waveform

# General conclusions

The project demonstrated several key lessons learned through its approach to solving the noise denoising problem:

1. Importance of Methodical Planning: The effectiveness of the project process was largely due to thorough initial research and the setup of a necessary working environment before embarking on the practical components. This foundational work highlighted the value of careful planning in setting up a solid basis for project activities, enabling smoother transitions into complex tasks.

2. Version Control Systems: Adopting Git as the version control system was a strategic choice that significantly aided our project management. This tool helped maintain clarity on project progress and track changes effectively.

3. Strategic Task Distribution and Collaboration: Dividing the project's tasks allowed to progress independently on different components, optimizing our efficiency. However, recognizing when collaboration was necessary, regular meetings scheduled to work on tasks that required joint efforts. This balance between independent work and teamwork was crucial in maintaining project momentum and handling intricate tasks efficiently.

4. Choosing the Right Tools and Infrastructure: Initially using a local GPU for model training showed limitations in flexibility and scalability, leading to switch to AWS with a Tesla T4 GPU. Although the transition required setup time, it provided a more scalable and powerful computing environment. This lesson highlighted the importance of selecting appropriate technological solutions to enhance efficiency and scalability.

5. Timing and Preparation for Data Acquisition: Reflecting on the project timeline, starting the generation and acquisition of datasets earlier would have optimized the schedule. This lesson was particularly poignant as external circumstances, such as a team member's emergency military service, impacted our ability to begin this phase as planned. Moving forward, prioritizing preliminary tasks like dataset preparation early in the project timeline is essential, especially to buffer against potential delays from unforeseen events.

These lessons from the project provide valuable insights into effectively managing complex technical projects, particularly those involving significant data and computational resources. Each lesson points to broader principles of project management, such as the importance of planning, the utility of powerful tools, and the need for flexibility in response to changing circumstances.

# Assessment of Project Benchmarks and Outcomes

Throughout the course of the project, key benchmarks were overcome, particularly the primary objective to denoise audio samples from tactical radio communication systems. Significant progress made in noise removal. includes the specific goal of improving the Signal-to-Noise Ratio (SNR) by at least 10dB. Some audio samples saw an increase in SNR exceeding 10dB, and others with initially low quality (SNRs as low as -6), only improved to about +6.

An unforeseen aspect of the project was the quality of speech in the denoised audio samples. Although the overall SNR improved, the clarity and intelligibility of speech did not enhance. This outcome has underscored the importance of including speech quality as a potential metric in future projects. Addressing this in subsequent initiatives could lead to more effective enhancements for tactical communication systems.

# 9. References

1. Madhav Mahesh Kashyap, Anuj Tambwekar, Krishnamoorthy Manohara, S Natarajan, “Speech Denoising without Clean Training Data: a Noise2Noise Approach", Department of Computer Science and Engineering, PES University, India, Sep 2021.

[2] H.-S. Choi, J.-H. Kim, J. Huh, A. Kim, J.-W. Ha, and K. Lee,

“Phase-aware speech enhancement with deep complex u-net,” in

International Conference on Learning Representations, 2018.

[3] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra,

“Perceptual evaluation of speech quality (pesq)-a new method for

speech quality assessment of telephone networks and codecs,” in

2001 IEEE International Conference on Acoustics, Speech, and

Signal Processing. Proceedings (Cat. No.01CH37221), vol. 2,

2001, pp. 749–752 vol.2.

[4] [Noisy speech database for training speech enhancement algorithms and TTS models (ed.ac.uk)](https://datashare.ed.ac.uk/handle/10283/2791)

[5] [UrbanSound8K - Urban Sound Datasets (weebly.com)](https://urbansounddataset.weebly.com/urbansound8k.html)

[6] <https://github.com/pheepa/DCUnet>