**Maintenance and Implementation of Future Changes in Noise2Noise Audio Denoising System**

# System Overview

The system consists of Python files that include the architecture of UNET, various Python libraries for audio processing, mathematical functions, convolutional network, and plotting. For example, torch, torchaudio, numpy, matplotlib. The model supports WAV audio files for training and inference.

# Requirements

For future training, the model requires a CPU with minimum of 16GB RAM, NVIDIA GTX 4070ti GPU/Tesla T4 GPU with minimum 8GB RAM memory and minimum of 150GB SSD memory. For future inference (only denoising), the model requires a standard work environment and a CPU to run the software.

Python is required, with the installation of the requirements.txt file from git.

# Codebase Maintenance

## Version Control (GIT)

The code is available in git at the following path: <https://github.com/OmriCohen15/Denoising_project-23-2-R-10/tree/main/part_b/noise2noise>

For version control the developer can open a new branch, work on the code, and make a pull request which we will approve. Alternatively, the developer can download the code to a local environment and make the desired changes in it (without updating in git).

Dependency Management

Python libraries tend to change and be updated over time, therefore it is useful to check every period of time whether an update or improvement has been made in one of the functions of the libraries used in the model and update accordingly. Both for the purpose of improving the model and for a necessary dependency adjustment.

Feature Datasets generation

To create a dataset and add noise, use one of the two scripts according to user guide/README file:  
urban\_sound\_noise\_dataset\_generator.py – random noises  
white\_noise\_dataset\_generator.py – only white noise

The folders Datasets/trainset\_clean and Datasets/testset\_clean must be filled before running with the desired recordings to add the noise.

Feature Updates and Implementation

## Adding New Features

Adding a feature or changing the model should be done as follows.

* Write documentation that will explain the manner and nature of the change.
* Design the implementation of the change.
* Implementation
* Testing

# Contact Information

Free to contact us with any problem or suggestion.

Omri Cohen: Omri.cohen2@e.braude.ac.il

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