

FM and OFDM Signal Denoising Using Deep Learning Techniques

Team 5

Authors:

Satya Srinivas Paladugu: BL.AI.U4AID 24063

Prithvi S:: BL.AI.U4AID 24076

Ekansh Khullar: BL.AI.U4AID 24075

Akshita Dindukurthi: BL.AI.U4AID 24004

Abstract:

This paper explores two critical aspects of modern Software Defined Communications (SDC): the enhancement of Frequency Modulation (FM) signals using Deep Learning and the practical implementation of Orthogonal Frequency Division Multiplexing (OFDM). We propose and implement two novel architectures: a 1D U-Net operating directly on time-domain waveforms and a 2D U-Net applied to Short-Time Fourier Transform (STFT) spectrograms. Our system achieves real-time denoising capabilities through multi-threaded architecture, processing audio chunks with minimal latency. The FM denoising pipeline demonstrates robust performance across varying Signal-to-Noise Ratio (SNR) conditions (5-20 dB), while the OFDM implementation showcases practical transmission and reception using Software Defined Radio (SDR) hardware.

Introduction:

FM and OFDM Signals are widely used in the modern world. FM radio stations are popular and still used while OFDM signals are the pinnacle of modern digital communication. Regardless, both modulation techniques are susceptible to noise. Noise has always been a persistent challenge and the root problem for signal degradation in wireless communications.

1. FM RADIO:

While Classical Signal Denoising based filters like Weiner filter or Wavelet Thresholding filters are functional, most of them struggle at patterns. Deep Learning Techniques promise a much superior pattern recognition which thereby has proven to be useful in terms of filtering things too. As a result, we have implemented 1DUNet and 2D U-Net

applied to STFT spectrograms. We have taken 2 types of datasets, Speech data from LibriSpeech and Music and trained two different models and one general model trained on both.

2. OFDM signals: Orthogonal Frequency Division Multiplexing is a modern technique of modulation, widely used for wireless communication and WIFI. Regardless of its advantages over FM or any other modulation technique, it is susceptible to noise.

Related Work

The application of Deep Learning (DL) to the Physical (PHY) layer of wireless communications has gained significant traction as a method to overcome the limitations of traditional expert-based systems.

A. Deep Learning for Interference Suppression Oyedare et al. [12] provide a comprehensive survey of DL-based interference suppression, highlighting the shift from "avoidance" techniques to active "suppression." Their analysis of Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Autoencoders demonstrate that DL approaches consistently outperform traditional methods like zeroing or ramp filtering, particularly in complex, non-linear decision boundaries. However, they identify a critical "Open Challenge": the stochasticity of the wireless channel often leads to a distribution mismatch between training data (often simulated) and real-world testing data.

B. Spectrogram-Based Denoising The efficacy of treating radio signals as images is explored by Almazrouei et al. [13], who proposed Convolutional Denoising AutoEncoders (CDAE) to clean IEEE 802.11 WLAN preambles. By converting time-series signals into Short-Time Fourier Transform (STFT) spectrograms, they leveraged the spatial locality of Convolutional Networks to achieve reconstruction accuracies exceeding 85% under Rayleigh fading channel conditions. This validates our project's **2D U-Net approach**.

C. U-Net Architectures in Signal Processing The specific use of the U-Net architecture for signal restoration is supported by Du et al. [14] in their work on "DNCNet" for radar signals. They demonstrated that a U-Net-based denoising subnetwork could effectively restore signals even when the test dataset has a significantly lower SNR than the training set. Their "divide-and-conquer" strategy—denoising the signal before passing it to a classifier—mirrors our objective of restoring audio intelligibility before human perception. Their findings confirm that U-Net skip connections are essential for preserving high-frequency details that are often lost in standard bottleneck architectures.

FM Radio Denoising

A. Dataset Construction

1. Clean Audio Corpus

Speech Dataset:

We utilize the LibriSpeech ASR corpus, a 1000-hour dataset of read English speech derived from audiobooks. Characteristics:

- **Format:** FLAC (Free Lossless Audio Codec)
- **Sample Rate:** 16 kHz (telephony bandwidth)
- **Speakers:** 2,484 speakers with diverse accents and prosody
- **Content:** Narrative text spanning fiction, non-fiction, history
- **Selection:** 2,700 files from "train-clean-100" subset (high SNR recordings)

Music Dataset:

Curated collection of 400 diverse music files:

- **Genres:** Classical, jazz, rock, electronic, ambient
- **Formats:** FLAC, WAV (uncompressed)
- **Sample Rate:** 44.1 kHz (CD quality)
- **Duration:** 30-300 seconds per file
- **Criteria:** Minimal compression artifacts, wide dynamic range

Rationale for Dual Corpus:

Speech and music exhibit fundamentally different spectrotemporal characteristics:

- **Speech:** Sparse spectral content concentrated in formant frequencies (200-8000 Hz), high temporal variability
- **Music:** Dense harmonic structures, sustained tones, broader frequency range (20-20,000 Hz)

Training on both modalities ensures the model learns generalizable denoising strategies rather than overfitting to speech-specific patterns.

2. How we got Noise

We scoured the entire spectrum/space for different types of noise. Scan FM broadcast band (88-108 MHz) to identify unoccupied frequencies (no carrier signal)

Dynamic "Mix-on-the-Fly" Augmentation:

- Rather than saving static pairs of noisy/clean files, the training loop dynamically generates samples.
- For every iteration, a random clean segment is mixed with a random noise segment at a uniformly sampled SNR between **5 dB (heavy interference)** and **20 dB (light static)**. This prevents the model from overfitting to specific noise patterns.

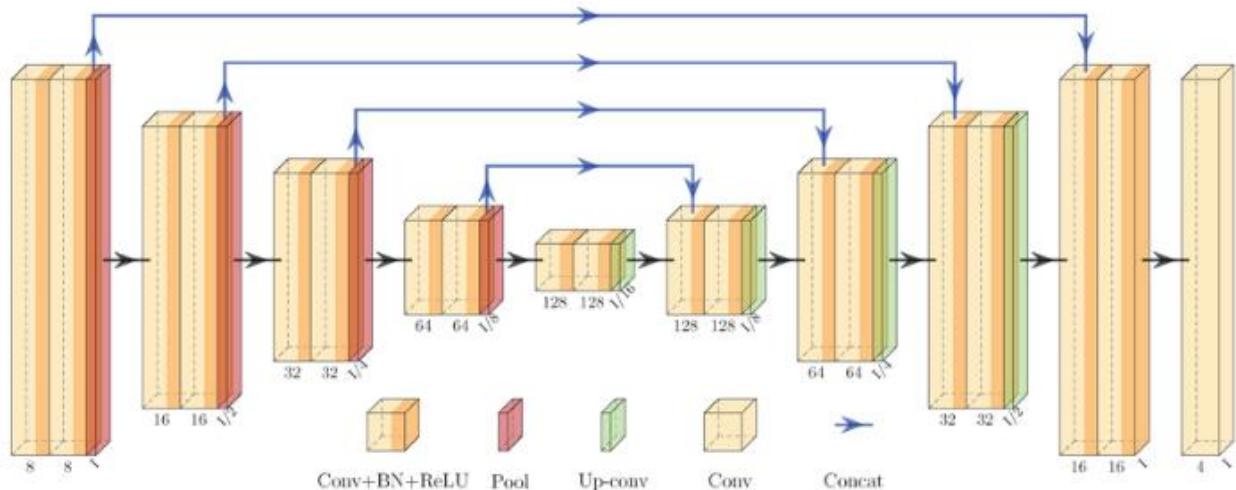
Advantages of On-The-Fly Augmentation:

- **Infinite Training Diversity:** Each epoch presents unique noisy examples (no memorization)
- **Storage Efficiency:** Avoids pre-computing and storing millions of noisy variants
- **SNR Curriculum:** Model exposed to full range of noise conditions, improving generalization
- **Computational Balance:** Mixing overhead negligible compared to forward/backward pass

B. Models selected and training

We implemented two variations of the U-Net, a fully convolutional encoder-decoder architecture with skip connections.

The architecture of the 1D UNet. It is an encoder–decoder network with four layers in both the encoder and decoder. The initial input feature number is four, and the feature channel is doubled while the feature size is halved after passing one layer of the encoder. In contrast, after passing one layer of the decoder, the feature channel is halved while the feature size is doubled. Meanwhile, a concatenation operation is employed to merge features from different scales. Finally, a convolutional layer is applied to produce the outputs with the same channel number as the inputs



U-Net's main advantage is its **encoder-decoder symmetric architecture** with [skip connections](#), which makes it highly effective for precise segmentation, especially on smaller datasets like medical images, because it preserves spatial details. Other architectures like [Vision Transformers \(ViT\)](#) break images into patches and use self-attention, excelling at complex patterns and large datasets but requiring more data.

1D Time-Domain U-Net

- **Input:** Raw audio samples
- **Architecture:** Consists of a contracting path (encoder) that captures context and an expanding path (decoder) that enables precise localization.
- **Mechanism:** Downsampling is achieved via strided 1D convolutions (acting as pooling), while upsampling uses transposed convolutions.
- **Skip Connections:** Critical for audio, these concatenate features from the encoder directly to the decoder. This preserves high-frequency phase information that would otherwise be lost in the bottleneck, ensuring the output is crisp rather than muffled.
- **Use Case:** Optimized for **low-latency speech** applications where converting to a spectrogram is computationally expensive.

Short-time Fourier transform (STFT)

The short time [Fourier transform](#) is introduced to overcome the problems of the FFT. It is usually used for the extraction of narrow-band frequency content in non-stationary or [noisy signals](#). The basic idea of STFT is to develop the initial signal into small time windows and employ the FT to each time segment for expressing the variation in signal frequency content over time that lived in that segment

The 2D U-Net approach in your project fundamentally treats the audio denoising task as an **image-to-image translation problem**. Instead of processing the raw 1D audio waveform directly, the signal is first transformed into a visual representation—a spectrogram. The U-Net then "sees" this spectrogram, learns to identify the visual patterns of noise versus signal, and generates a clean version.

2D Frequency-Domain U-Net

- **Input:** Magnitude spectrogram of the Short-Time Fourier Transform (STFT).

- **Processing:** The model treats the spectrogram as a single-channel image. It predicts a "ratio mask" which, when element-wise multiplied with the noisy magnitude, suppresses noise bins.
- **Reconstruction:** The clean STFT magnitude is estimated by applying the predicted mask to the noisy input magnitude. The waveform is then reconstructed by combining this clean magnitude with the original noisy phase) and performing an Inverse STFT:

$$\widehat{X_{STFT}} = M_{pred} \cdot |Y_{STFT}| \times e^{j\angle Y_{STFT}}$$

The noisy phase angle is reused for reconstruction because the human ear is relatively insensitive to phase errors compared to magnitude errors, and estimating clean phase is computationally expensive.

-
- **Use Case:** Superior for **music**, where preserving harmonic structure in the frequency domain is more critical than temporal precision.

C. Real Time inferencing Pipeline

For the real time inferencing, we are using a software called VB cable. This software changes the logical routing of the speaker input/output of an application from physical to virtual.

Using VB cable, we have redirected the output of SDRSharp into a python program that uses 1DUNET and STFT+CNN implementation in pytorch to denoise actively.

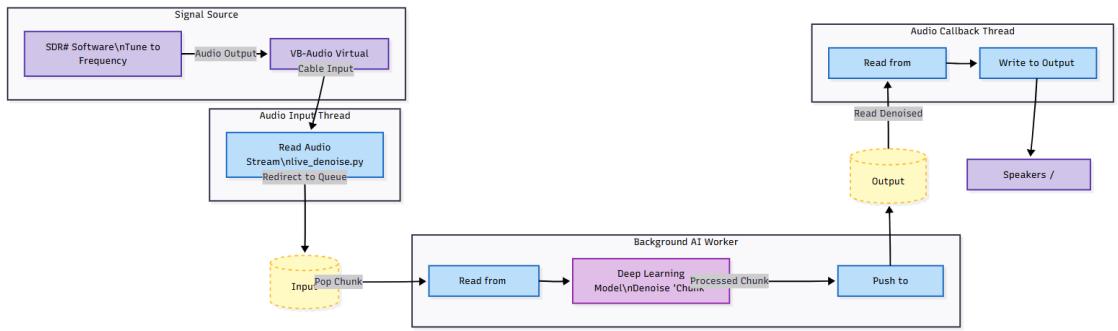
The audio received is broken down into chunks. These chunks are then pushed into an `in_queue`. Each chunk is then taken out one by one and processed and denoised and then pushed into an `out_queue`. This `out_queue` holds the denoised chunk. This can be made much faster by introducing the concept of multi threading.

Why chunks? **Real-time audio:** You can't wait for the whole audio to finish before denoising; you must process and play small pieces as they arrive (like a live radio or call)

Chunk size is a balance between latency and efficiency.

- Smaller chunk = lower latency, but more frequent processing (higher CPU load, more context switches).
- Larger chunk = higher latency, but more efficient for the model and system.

4096 samples at 44.1 kHz \approx 93 ms per chunk. This is a common value in audio apps: it's short enough for real-time, but long enough for efficient neural net inference.



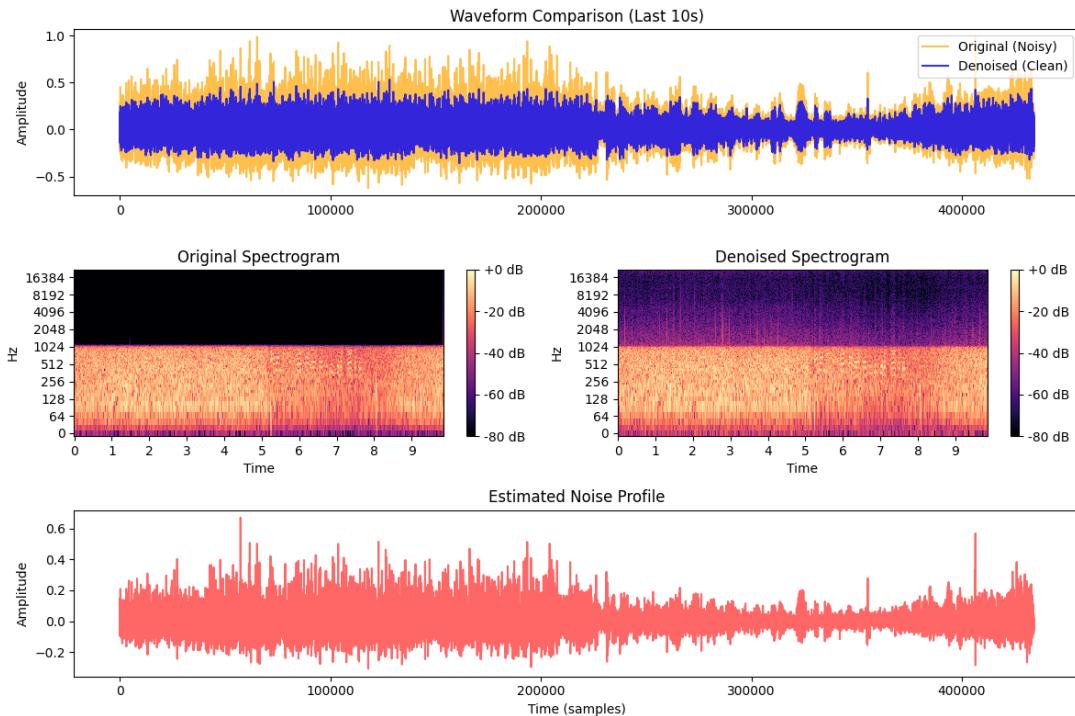
RESULTS:

We have tested all three models of both architecture. Speech and 1D-UNet, Music and 1D-UNet, General and 1D-UNet, Speech and STFT, Music and STFT, General and STFT.

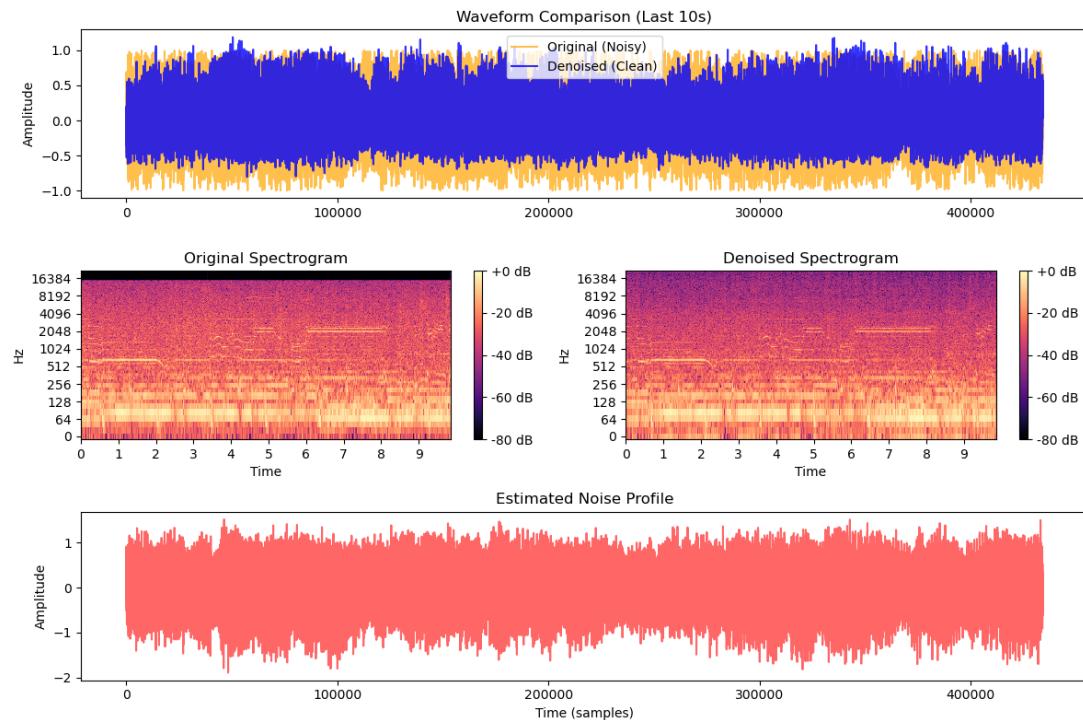
Out of all this, 1D-UNet With speech has performed the best. Reason being is not the model's features. I mean it is one of the reasons, but main reason is because Speech data has patterns that can be observed and learnt better than any other. 1D-Unet is really good at recognizing the patterns due to its architecture.

But when it comes to music, we need more data. Music has a lot of patterns that are hard to be recognized and understood. 1D-Unet and STFT+CNN would not excel at it unless trained at a much larger dataset.

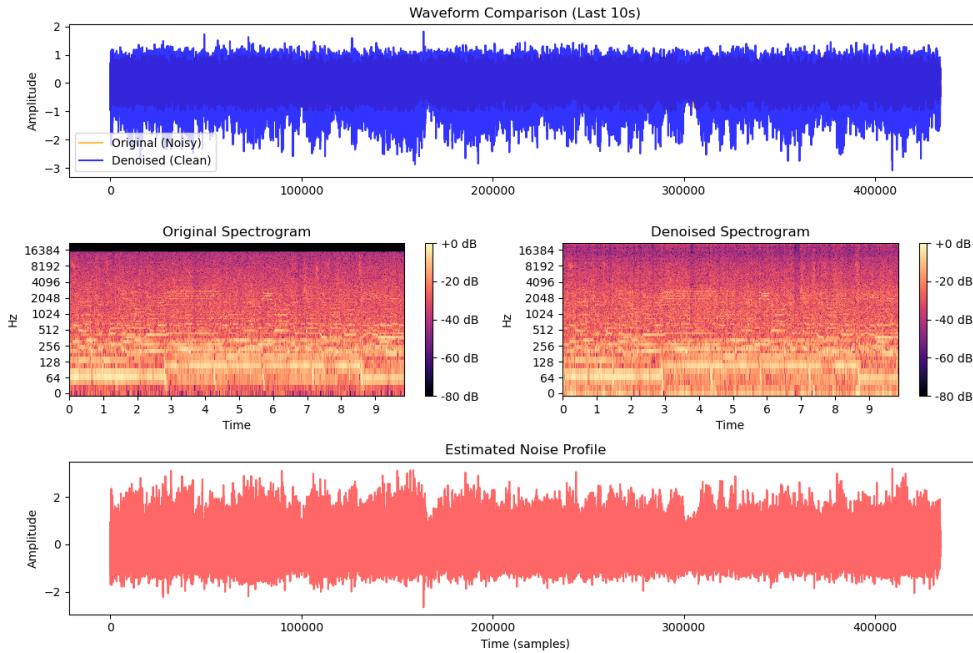
General models performed bad at both music and speech audio.



Speech dataset



General (Mediocre filtering)

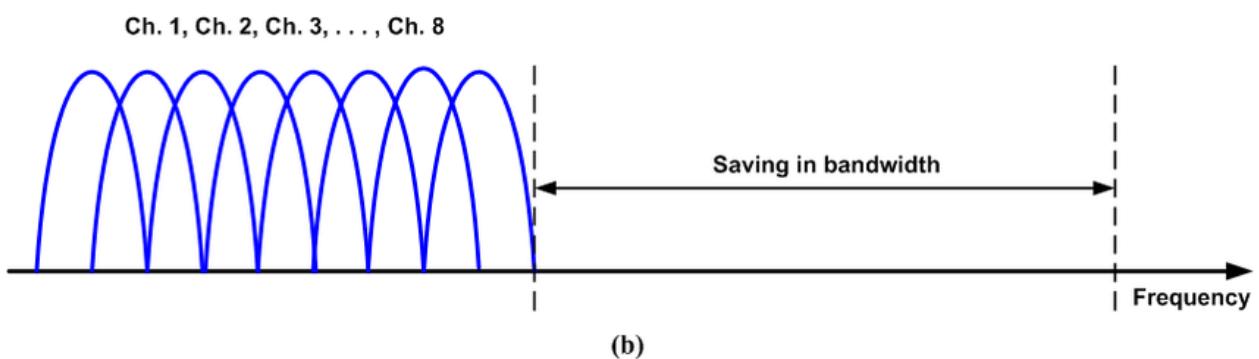
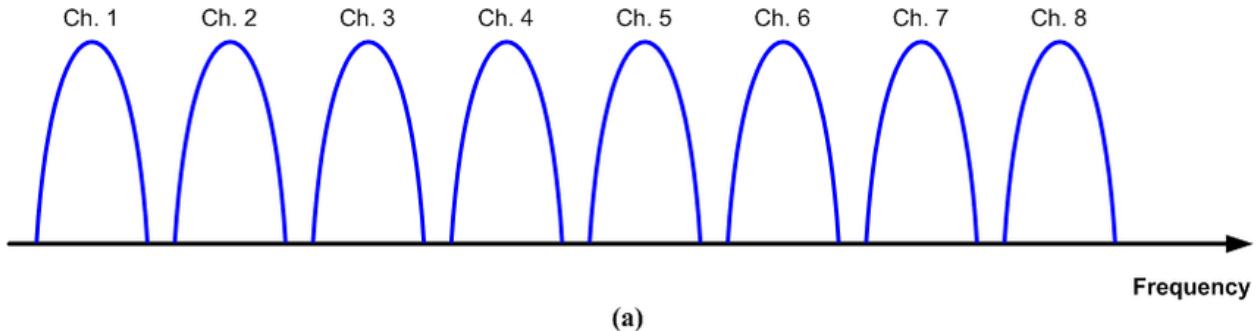


music (the original waveforms behind the denoised. Performed really bad)

Orthogonal Frequency Division Multiplexing

Orthogonal Frequency Division Multiplexing (OFDM) is another widely used modulation method used to achieve high data rates and spectral efficiency. It is known as a multicarrier modulation method as many carriers are used instead of just one. OFDM takes the serial data to be transmitted and divides it up into many slower serial streams each of which is modulated onto one of multiple subcarriers. There may be as few as 40 subcarriers or many thousands. The subcarriers are spaced from one another by an amount that makes them orthogonal to one another. This means that despite their close adjacent spacing from one another they will not interfere with one another.

Figure 63.7 shows the spectrum of an OFDM signal. The entire set of carriers is transmitted simultaneously within the assigned bandwidth. The modulation on each carrier is usually BPSK, QPSK, or some form of QAM. This makes OFDM one of the most spectrally efficient of all modulation methods. Furthermore, it is highly resistant to noise and various propagation effects that afflict the transmission channel. OFDM is found in most modern wired and wireless systems today.

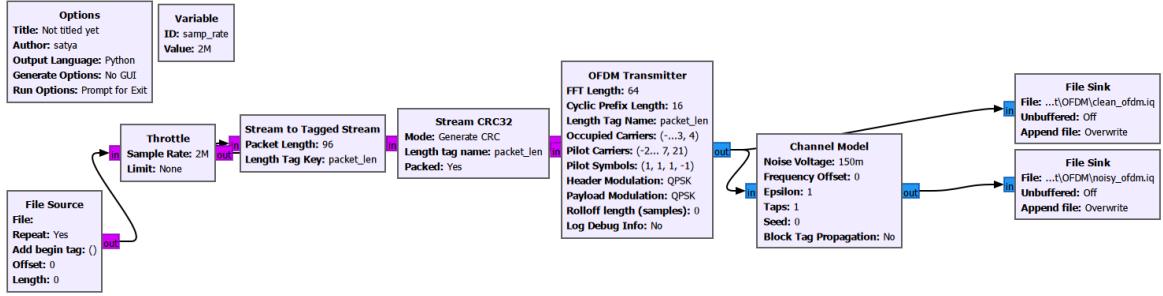


Working of OFDM:

- Take data and make into N samples
- Apply inverse Fourier transform to make it not interfering into each one. It is applied to every sample
- Then apply cyclic prefixes so that they don't mess up the orthogonality and interfere
- Receiver will remove the prefix and apply FFT to get back to Frequency domain

A. Dataset generation

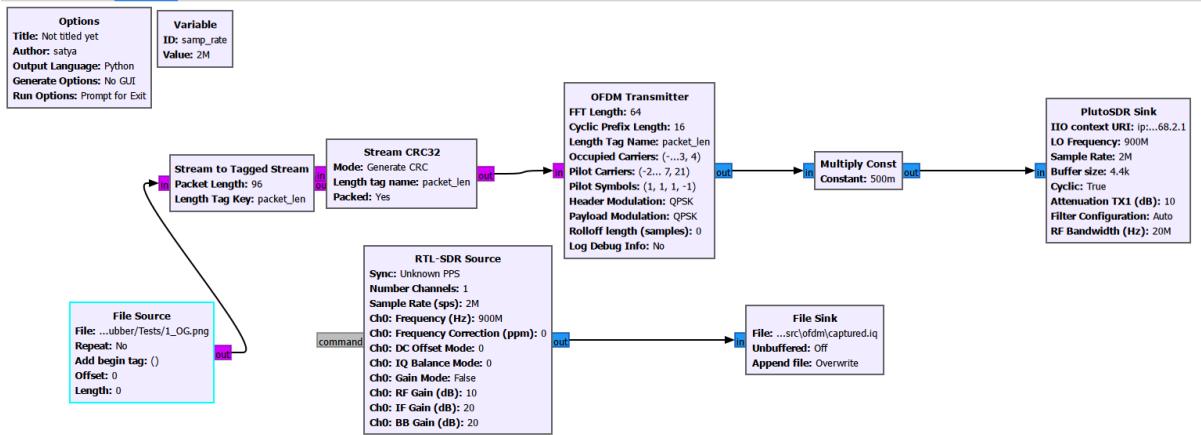
We have generated our dataset using GNU-RADIO. The source is not set to random source. Instead we have taken a different approach by taking all the images in the MNIST dataset and concatenated each one to make one bin file. This enables the model to learn about how each image looks like and about the meta data. The modulation technique is QPSK.



The flowgraph was run for 70 seconds in total, generating almost 1.98 Gigabytes huge dataset. We have trained a 1D-UNet model on this dataset considering its architecture that gives it the advantage and upper hand compared to other autoencoders.

B. Transmission and Receiving

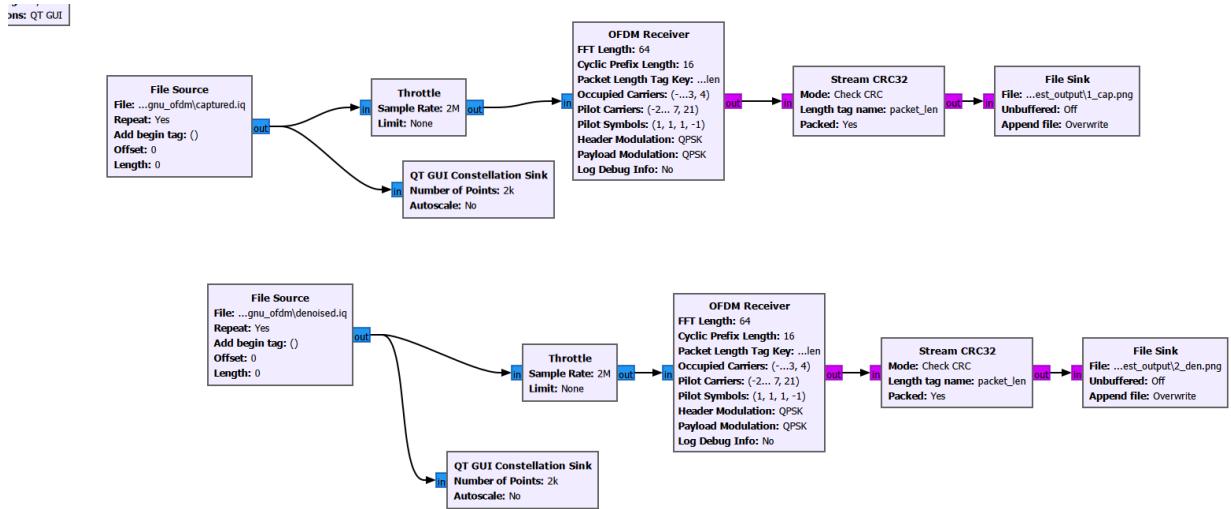
We have used an PlutoSDR to send data and an RTLSDR to receive data from.



The received data is stored in its raw Inphase and Quadrature format.

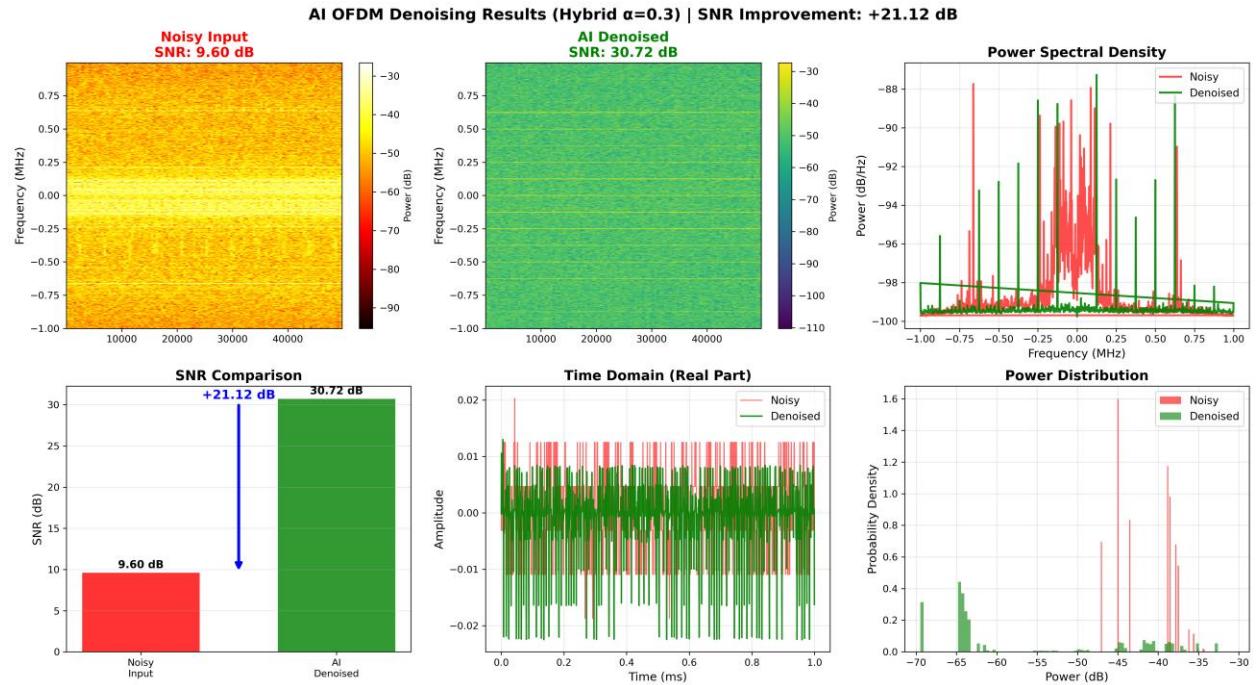
C. Denoising and comparison

The captured data is passed through the DL model and denoised. After denoising you get another file named denoised.iq which contains the denoised IQ format.



At the end, you are supposed to receive both files back with or without some distortions.

Results and discussions:



About plots:

Spectrogram: - Shows the signal's time/frequency structure.

- Noisy = everything is yellow/orange, structure buried.
- Denoised = OFDM subcarriers visible, clean bands.

PSD (Power Spectral Density):

- Shows energy at each frequency.

- OFDM peaks visible after denoising, noise floor drops
- proves protocol is preserved and noise suppressed.

SNR Bar Chart:

- Shows how much signal quality (bit recovery) is improved.
- Large gain in SNR means better error rate and reliability.

Variance Plots:

- Show temporal stability of the denoised signal
- low, flat variance after denoising is best.

Images:

Before transmitting:

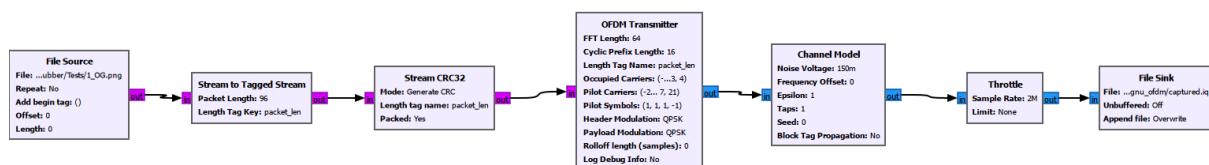


After capturing(RAW) and Denoised: NO IMAGE

This is when we used RTL-SDR
To make sure our workflow was
simulation using GNU RADIO at
RTLSRD and PLUTOSDR, we
with addition of noise using

and PLUTOSDR.

not wrong, we have performed a
the sametime. Instead of using
have taken a direct connection
channel modulator.



IMAGES:

Before transmitting:



After capturing RAW



After denoising



Conclusion:

From this, we have to conclude a few things:

- When generating dataset, we have to not only take Gaussian noise into consideration but even other factors like echo and even the antenna length.
- This problem cannot be an OFDM denoising problem. Rather it should be a **QPSK denoising problem.**
- Bit errors directly result from QPSK points being perturbed across decision boundaries.
 - Even slight constellation "noise" causes symbol flips, not just cosmetic changes.
- OFDM is just the transport mechanism. The real challenge is noise corrupting the **modulation symbols** (QPSK), and any denoising must preserve constellation cluster integrity.

- AI denoisers fail for OFDM. Autoencoders work well with continuous data. They do not work with QPSK symbols or other symbols as they fail to understand the meaning of metadata. They do not understand which symbols are needed and not needed for an image.
- OFDM signals work with pilots and preambles which Autoencoders fail to understand the relations about.
 - **The Noise IS Part of the Received Signal.** Due to this, the denoising model doesn't know which symbol is noise or not. Because it simply does not understand the difference between a noisy qpsk and clean healthy qpsk symbol.
- Python does not work well with RTL SDR and PLUTOSDR. You have to use gnuradio based python scripts only to work with them due to its robust nature in controlling and managing gain and power of the signals and SDR.

RESOURCES

- [The architecture of the 1D UNet. It is an encoder-decoder network with... | Download Scientific Diagram](#)
- [**Orthogonal Frequency Division Multiplexing - an overview | ScienceDirect Topics**](#)
- [Spectral efficiency of OFDM compared to conventional FDM\[2\] . | Download Scientific Diagram](#)
- T. Oyedare, V. K. Shah, D. J. Jakubisin, and J. H. Reed, "Interference Suppression Using Deep Learning: Current Approaches and Open Challenges," *IEEE Access*, vol. 10, 2022.
- E. Almazrouei, G. Gianini, N. Almoosa, and E. Damiani, "A Deep Learning Approach to Radio Signal Denoising," *Emirates ICT Innovation Center (EBTIC)*.
 - M. Du, P. Zhong, X. Cai, and D. Bi, "DNCNet: Deep Radar Signal Denoising and Recognition," *Journal of Latex Class Files*.
[DNCNet Deep Radar Signal Denoising and Recognition.pdf](#)