Converting UNet to a 1-D Denoiser for Signal Noise Reduction on Hyperspectral Dataset

Rahul Chavali
Washington University in St. Louis
Email: c.rahul@wustl.edu

***Abstract*—This paper explores the modification of the UNet architecture into a 1-dimensional denoiser for signal noise reduction. A variety of noise levels are considered, and the effectiveness of the approach is evaluated through quantitative metrics and visualizations.**

I. Introduction/Problem

Image denoising is a critical problem in fields ranging from machine learning to medicine. Diving deeper into the topic, and the subject of my research, is cleaning image data that has a larger number of spectral bands or wider wavelength than a regular image. This is where hyperspectral imaging comes in. Hyperspectral image denoising has applications in forensics, remote sensing, satellite imaging, and any data that has more input than the visible eye. The typical hyperspectral data is 3D, with length, width, and height. Normally, this type of problem is solved using a 2D or 3D denoiser, however, these models take high computational complexity. For the purpose of this research, which is to use this model as a prior for spectral super resolution, I wanted to see if we could garner high quality data with a 1D denoiser on hyperspectral imaging data for lower computational complexity. My goal was to train the spectra of the dataset and develop high-quality results leading to a significant change in SNR (Signal-to-noise-ratio) once running through the trained model.

In this specific problem, I am using a hyperspectral dataset of Indian Pine trees which has a size of (145 x 145 x 220). My method was successful on this dataset and showed a significant change in SNR.

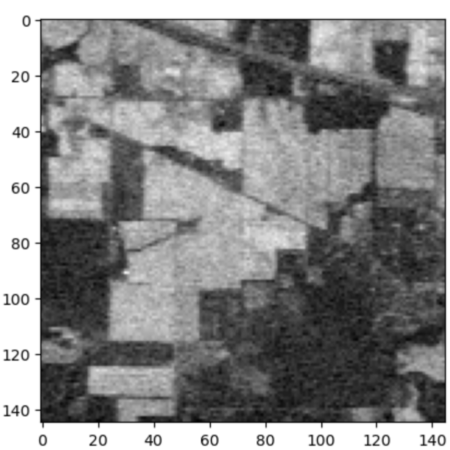

Fig. 1. Dataset image.

II. Method

The UNet architecture, typically used for image segmentation, is adapted to process 1-dimensional signals. This involves modifying the input, convolutional layers, and upsampling operations to operate along a single dimension. The training process uses a dataset with synthetic noise applied to clean signals.

A. Model Architecture

To start this process, I first began by modifying a 2D UNet design into 1D. This involved ensuring the input data was in 1D format, modifying the number of channels, and changing the shape of the data from (height, width, channels) to (channels, height, width). I then reshape the data to be 1D by converting it to (1, channels, height) which represents a 1D signal. This model consists of a downsampling path where the number of channels increases and spatial dimensions are reduced, a bottleneck where the most feature maps are apparent, and an upsampling path where the original spatial dimensions are restored and depth is reduced. The initial model had 1 channel and took in a noisy signal and output a denoised signal, however, the final model has 2 channels and takes in a noisy signal and noise level and outputs a denoised signal.

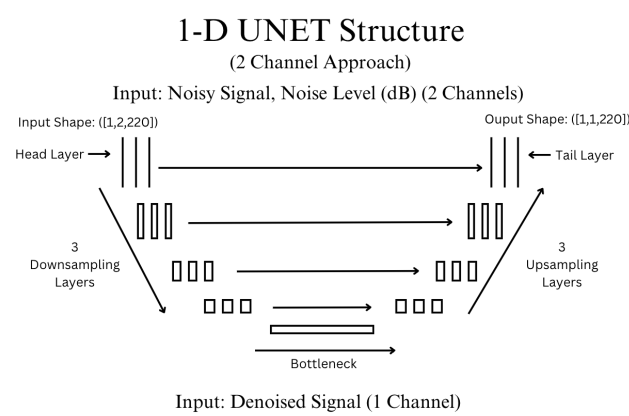

Fig. 2. Modified UNet architecture for 1D signal denoising.

B. Initial Training

To develop a competent model that could identify noise, I first set out to train a model on a 0.05 noise level. To do this, I first split the dataset into a 113x113 sized square to train, and a 32x32 sized square to test. Now, on the NVIDIA GeForce GTX 1080 Ti GPU, I looped through the training data and artificially added 0.05 noise to every signal and trained the model to denoise this data. Then, in the testing loop, over 100 epochs, I looped over the testing data and trained the model on those signals to see how well the UNET model worked on unseen data. To explain this further, I first added noise to a single signal and then I input that noisy signal into the 1-channel model. Then the model outputs a denoised signal. I calculated the SNR with the formula snr = 20 * torch.log10(torch.norm(x) / (torch.norm(x - xhat) + 1e-10)) over each sample in the batch across the dataset. I kept track of average loss to see how well the UNET model fit the data. At the end of this loop, I calculated the average of both of these values at each epoch. Finally, I saved the model, SNR values, and the average loss values.

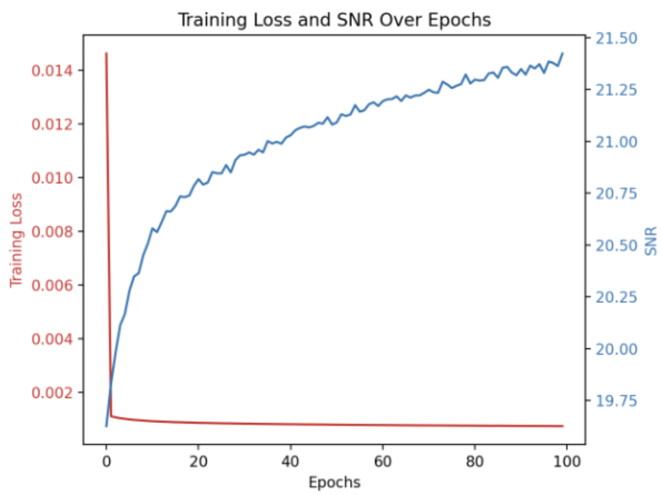

Fig. 3. Results for original model.

As you can see here, the UNET design showed promised as the average SNR drastically increased showing that the model was denoising, and the average loss was decreasing showing that the UNET model was fitting to the data.

C. Final Training

Although this model was working, I realized it could be improved. First, the model only worked on one noise level, that being 0.05. To build a proper denoising model, we want it to work on a wider range of noise levels. Second, the SNR seemed like it was still rising, signifying that we could run the model on more epochs resulting in better results. To implement these changes, I first had to change the model architecture to take in 2 channels to accommodate the input of both the noisy signal and the varying noise level, sigma, that the model would train on. The output of the model is still one signal as we only want the denoised signal to output. To explain further, the 2-channel model takes in the noisy signal and the noise level of the signal with input shape [1, 2, 220] as the input and then outputs the denoised signal of shape [1,1,220]. In the train loop I randomly generated a noise level from 0 to 0.5 to artificially add to the signals. To test this data, I selected a 0.25 noise level in the test loop to ensure the SNR was still increasing and the average loss was still decreasing. Lastly, I increased the epoch number from 100 to 500 to see the improved data in SNR.

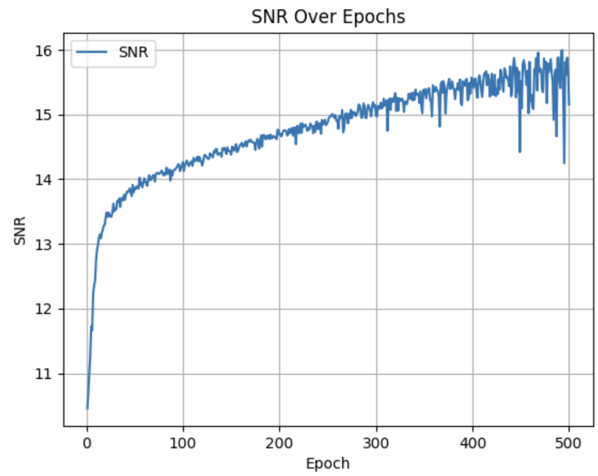

Fig. 4. SNR over Epochs for new model(0.25 noise level).

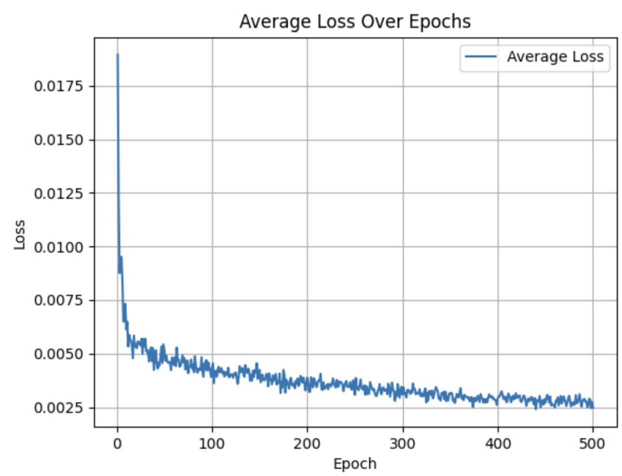

Fig. 5. Average Loss over Epochs for new model.

From these graphs, we can see that the model was successful on a 0.25 noise level, as the SNR increases and the average loss decreases, signifying that the model fit to the training data and was successful on the test data.

III. Results

The model's performance is evaluated using Signal-to-Noise Ratio (SNR).

A. Qualitative Results

Visualizations of denoised signals show significant improvements in quality.

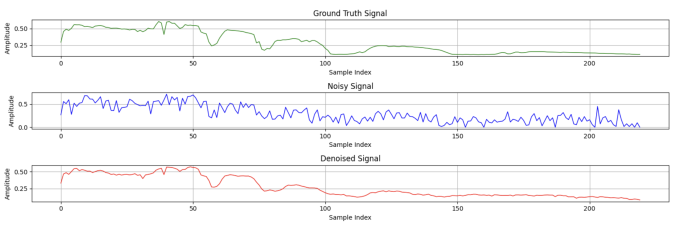

Fig. 6. Sample signals before and after denoising (Noise = 0.1).

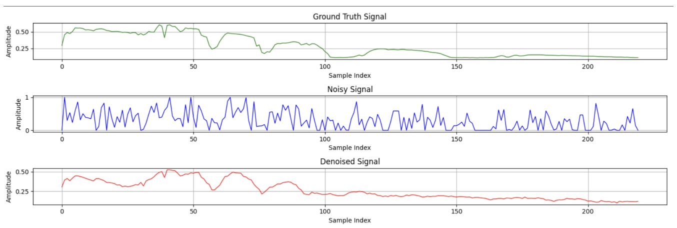

Fig. 7. Sample signals before and after denoising (Noise = 0.3).

These graphs show a before and after view of a selected ground truth signal of trees[1,1,:]. The graph of the noisy and denoised signals is shown, meaning that the model can denoise spectra on multiple different noise levels.

B. Quantitative Results

Tables and graphs summarize performance metrics in different noise scenarios.

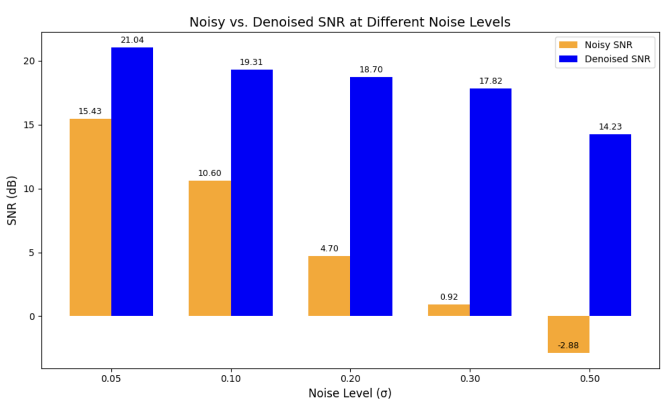

Fig. 8. Bar Plot of Noisy SNR vs. Denoised SNR at various noise levels

This graph also shows that with varying noise levels, the SNR drastically increased from the noisy signal to the denoised signal. In fact, at the 0.05 level there was a growth of 5.61 dB, at the 0.1 level there was a growth of 8.71 dB,

Noise Level (σ)	Noisy SNR (dB)	Denoised SNR (dB)
0.05	15.43	21.04
0.1	10.6	19.31
0.2	4.7	18.7
0.3	0.92	17.82
0.5	-2.88	14.25

Fig. 9. Table of Noisy SNR vs. Denoised SNR at various noise levels

at the 0.2 level there was a level there was a growth of 14.00 dB, at the 0.3 level there was a level there was a growth of 16.90 dB, and at the 0.5 level there was a growth of 17.11 dB. The average growth of 12.47 dB on 1D signals. This shows that our denoiser works and shows promise.

We can conclude that our model works as a 1D denoiser on hyperspectral data with various noise levels, and it can be used in plug-and-play techniques as a prior for spectral super resolution. There is a growth in SNR and the UNET denoising model is working, however, how can we assess how good the model is comparatively to other models?

IV. Conclusion and Further Work

A. Conclusion

As we can see from our data, we have a successful UNET 1D denoiser for hyperspectral data. However, how can we quantify this against other methods of denoising and changes in SNR? Well, I compared the UNET model to a well-known image denoising model called BM3D. This model uses block matching to search for similar blocks and transform the image. My implementation of BM3D on the Indian Pines dataset was a 2D denoising method. I took the test data, a 32 x 32 square, and added noise to the region. Then I called the BM3D model to denoise each band, where the region was 2D, written in this format: [:, :, band]. Next, I looped over all the bands in the test data and denoised each specific band. To show the denoising effect, I used the slice at band 10.

To compare this to the UNET denoiser, I created a 2D image using the finalized UNET model. I did this by looping through the test data and adding a noise level to a signal, then entering a noisy signal and noise level into the model. Next, I added all ground truth signals, noisy signals, and denoised signals to three lists. Finally, I joined the signals and reshaped the data to the initial height, width, and channel of the test data. I reshaped the data from (1024, 1, 220), which was (32x32, 1, 220), to (32,32,220). I then took a specific slice of length 10 out of 220 to get a 2D image, the same slice used in the denoised image using the BM3D model.

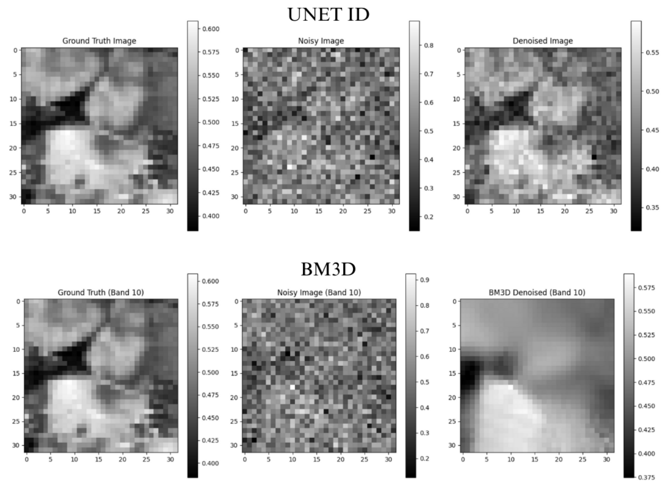

Fig. 10. Reconstructed UNET Image Vs.Reconstructed BM3D Image at 0.1 Noise Level

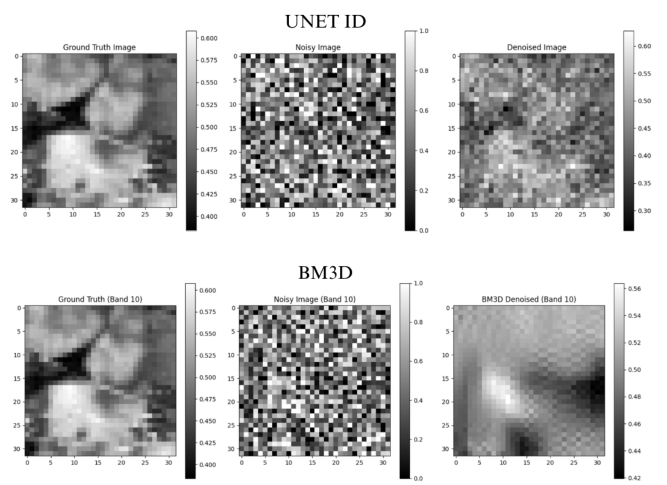

Fig. 11. Reconstructed UNET Image Vs.Reconstructed BM3D Image at 0.3 Noise Level

I computed the SNR growth of the noisy to denoised image using a 2D snr calculation. I took the average SNR of all 2D regions over the 220 bands for the final SNR values. The 0.1 noise level saw a growth in SNR from 7.43 db to 22.30 db. The 0.3 noise level saw a growth from -0.76 db to 12.47 db.

Noise Level (σ)	Noisy SNR (dB)	Denoised SNR (dB)	Change in SNR (dB)
0.1	7.43	22.3	14.87
0.2	2.17	16.63	14.46
0.3	-0.76	12.47	13.23
0.4	-2.55	9.77	12.32
0.5	-3.73	7.85	11.58

Fig. 12. SNR computed on bands and averaged across all bands (BM3D)

I also computed the SNR growth of the noisy to denoised image using a 2D SNR calculation of the concatenated image by computing SNR of the 2D ground truth image verses the 2D noisy and denoised images respectively. The 0.1 noise level saw a growth in SNR from 10.16 db to 18.93 db. The 0.3 noise level saw a growth from 1.99 db to 14.59 db.

Noise Level (σ)	Noisy SNR (dB)	Denoised SNR (dB)	Change in SNR (dB)
0.1	10.16	18.93	8.77
0.2	4.86	15.94	11.08
0.3	1.99	14.59	12.6
0.4	0.18	13.17	12.99
0.5	-0.96	11.83	12.79

Fig. 13. SNR computed on 2D ground truth region vs. noisy and denoised 2D regions respectively

Now that we have the values, let us compare them with a table. I used change in SNR as the metric to compare rather than the higher denoised SNR for a noise level because the way I added noise for both models was different so the initial noisy SNR varied.

Noise Level (σ)	BM3D Change in SNR (dB	1D Denoiser Change in SNR (dB)
0.1	12.21	14.87
0.2	13.8	14.46
0.3	12.81	13.23
0.4	11.67	12.32
0.5	10.5	11.58

Fig. 14. UNET and BM3D SNR values compared

To more specifically compare the UNET 1D model to the BM3D 2D approach to reconstructing the image, I compared the 2D SNR values of both with two graphs, one showing the denoised SNR between models and noise levels, and one showing the change in SNR from noisy to denoised between models and noise levels. As stated before, the change in SNR graph (Figure 16) is more relevant due to the variance in the SNR of the noisy images.

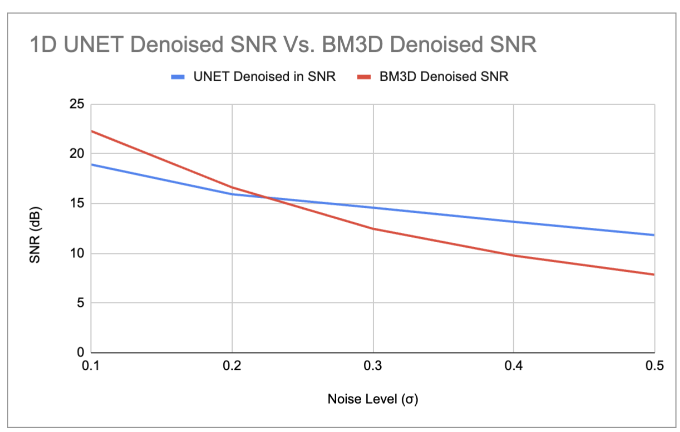

Fig. 15. BM3D Denoised SNR Vs. UNET 1D Denoiser Denoised SNR

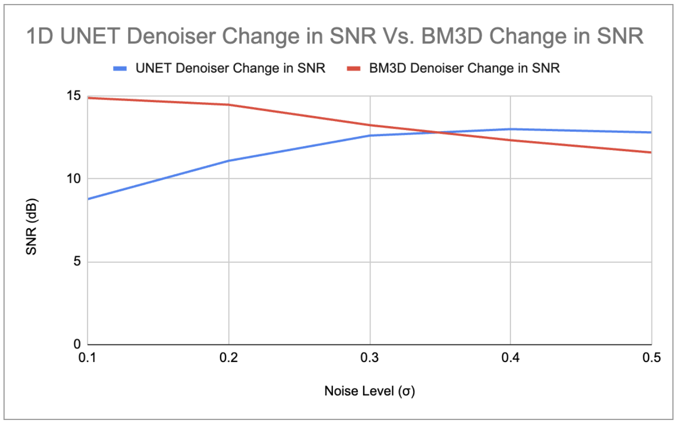

Fig. 16. BM3D Change in SNR Vs. UNET 1D Denoiser Change in SNR

Figure 15 shows that before around a 0.22 noise level the BM3D model has a higher denoised SNR that the UNET. However, after that noise level the UNET 1D model has a better denoised SNR. In Figure 16, we can see that before around a 0.35 noise level the BM3D model has a larger change in SNR that the UNET. However, after that noise level the UNET 1D model has a greater change in SNR. The UNET model's average change in 2D SNR is 11.65 dB, and the BM3D 2D model has an average change of 13.29 dB, a comparable value. The UNET model's average SNR on 1D signals, 12.47 dB, is even closer. This shows that the UNET model works nearly as well as a recognizable 2D model, even performing better at higher noise levels. It should also be noted that the UNET denoiser improved it's change in SNR over higher noise levels while the BM3D model decreased it's change in SNR over noise levels. Since the UNET denoiser is based in 1D data, it may work better on spectra than the BM3D model, however, it should be highlighted that it is similar to BM3D, which is designed for 2D and 3D data as block matching works better, on 2D data.

I also compared the SNR values over a 3D dataset by setting the ground truth to the test data and comparing with the noisy and denoised arrays over all bands for both model. The results were comparable to the 2D comparison.

Noise Level (σ)	Noisy SNR (dB)	Denoised SNR (dB)	Change in SNR (dB)
0.1	9.19	14.7	5.51
0.2	4.56	13.64	9.08
0.3	1.85	13.01	11.16
0.4	0.1	12.16	12.06
0.5	-1.02	11.17	12.19

Fig. 17. UNET 3D SNR values calculated by comparing the test data to the noisy and denoised dataset

Noise Level (σ)	Noisy SNR (dB)	Denoised SNR (dB)	Change in SNR (dB)
0.1	10.2	22.47	12.27
0.2	4.87	18.66	13.79
0.3	1.97	14.8	12.83
0.4	0.18	11.69	11.51
0.5	-1	9.51	10.51

Fig. 18. BM3D 3D SNR values calculated by comparing the test data to the noisy and denoised dataset

Noise Level (σ)	UNET Change in SNR	BM3D Change in SNR
0.1	5.51	12.27
0.2	9.08	13.79
0.3	11.16	12.83
0.4	12.06	11.51
0.5	12.19	10.51

Fig. 19. 3D changes in SNR amongst models

I once again graphed the same graphs as the 2D comparison to get more detailed results for 3D comparison over the entire dataset. For the 3D dataset the noisy SNR's were a lot less varied than the 2D data, this can be seen from the tables and graphs.

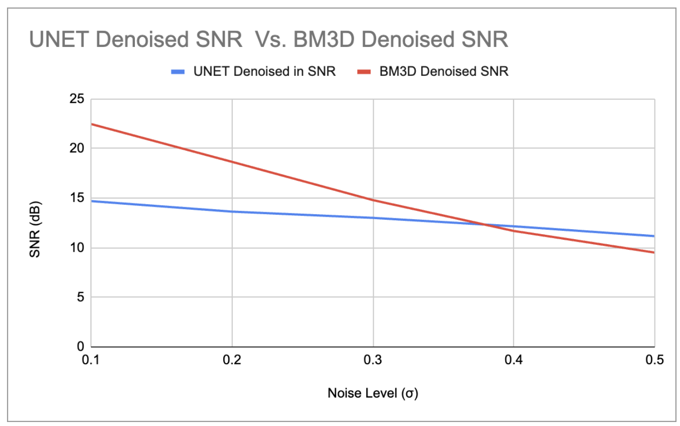

Fig. 20. BM3D Denoised SNR Vs. UNET 1D Denoised SNR

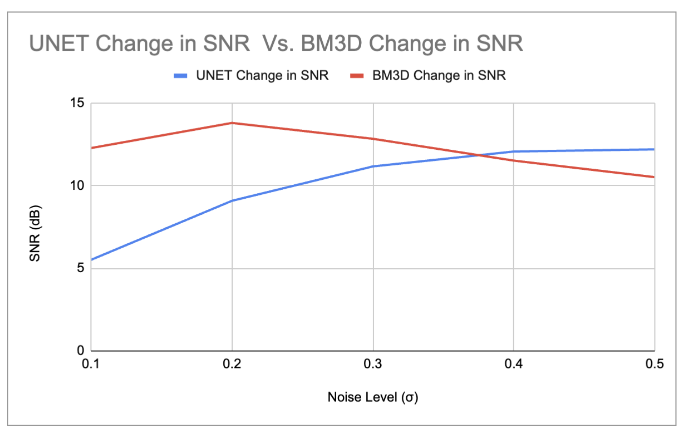

Fig. 21. BM3D Change in SNR Vs. UNET 1D Denoiser Change in SNR

From Figure 20 and 21 it can be seen that the trend of the UNET overtaking the BM3D at higher noise levels is consistent on 3D in both denoised SNR and change in SNR. The average change in SNR in 3D is 10 dB for the UNET and 12.18 dB for the BM3D, again similar values. The UNET design is able to stay relatively near the BM3D model in perfomance over the entire dataset while decreasing computational complexity.

B. Further Work

The goal of this research was to train a UNET 1D denoiser on hyperspectral data to be used as a prior for spectral super resolution, a technique to obtain hyperspectral images from RGB images. To do this, I needed to develop a model to add Gaussian blur to a 1D signal. I did this by using a 1D convolutional layer and defining the Gaussian kernel and sigma to add blur. Then I graphed the ground truth and blurred signals with trees[1,1,:]. In my further work, I can use the blur model and denoiser for spectral super resolution.

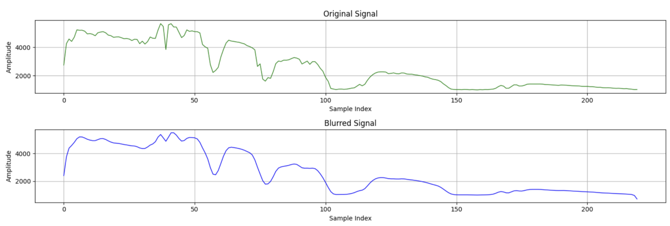

Fig. 22. Ground Truth Signal vs. Blurred Signal

References

- Jonas, E. M. (n.d.). pybm3d: A Python implementation of BM3D denoising algorithm. GitHub repository. Retrieved December 10, 2024, from https://github.com/ericmjonas/pybm3d
- Hendriksen, A. (n.d.). pybindbm3d: Python bindings for BM3D. Retrieved December 10, 2024, from https://ahendriksen.github.io/pybindbm3d/
- Y. Liu, Q. Tang, and M. Wang, “Hyperspectral Signal Noise Reduction Using Adaptive Thresholds,” *Photonics*, vol. 10, no. 6, p. 674, 2023. [Online]. Available: https://www.mdpi.com/2304-6732/10/6/674
- PyTorch, “Training a Classifier Tutorial,” 2024. [Online]. Available: https://pytorch.org/tutorials/beginner/introyt/trainingyt.html
- PyTorch, “torch.nn.MSELoss,” 2024. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.MSELoss.html
- Kaggle, “Indian Pines Hyperspectral Dataset,” [Online]. Available: https://www.kaggle.com/datasets/abhijeetgo/indian-pines-hyperspectral-dataset
- B. Biggio *et al.*, “Adversarial Examples for Hyperspectral Image Classification,” 2020. [Online]. Available: https://arxiv.org/pdf/2008.13751