Deep Learning Project

Image Denoising

Abstract

Deep convolutional networks have become a popular tool for image generation and restoration. Generally, their excellent performance is attributed to their ability to learn realistic image priors from many examples. In this paper, we show that, on the contrary, the structure of a generator network is sufficient to capture a great deal of low-level image statistics before any learning. To do so, we show that a randomly initialized neural network can be used as a handcrafted prior with excellent results in standard inverse problems such as denoising, super-resolution, and inpainting. Furthermore, the same prior can be used to invert deep neural representations, diagnose them, and restore images based on flash-no flash input pairs. Besides its diverse applications, our approach highlights the inductive bias captured by standard generator network architectures. It also bridges the gap between two popular families of image restoration methods: learning-based methods using deep convolutional networks and learning-free methods based on handcrafted image priors such as self-similarity.

Introduction

Denoising images remains a longstanding and pivotal challenge in image processing, captivating researchers over several decades. Historically, conventional methodologies involved filters to alleviate image noise, yielding satisfactory outcomes for images with modest noise levels. Unfortunately, using these filters had a downside - it caused noticeable blurring in the processed images. This compromise became particularly pronounced when addressing images with heightened noise, resulting in output so excessively blurred that critical details were inevitably compromised.

In response to these challenges, the field of image denoising has witnessed a transformative shift with the advent of deep learning. This report delves into the innovation, examining two distinct deep learning architectures that are substantial advancements in image denoising. In contrast to the limitations of traditional denoising filters, these state-of-the-art models promise superior noise reduction and the preservation of intricate image details.

The following report presents a meticulously crafted case study detailing the journey from problem formulation to realizing sophisticated deep learning models. This report is intended to provide a formal and comprehensive overview of the research conducted, including the methods and procedures employed and the results obtained. By presenting the findings clearly and concisely, this report aims to contribute to the broader body of knowledge in deep learning. We have implemented two solutions - one is the model without Skip Connections which is referred to as **Model 1** and the other is the model with Skip Connections which is referred to as **Model 2**.

Reference Image

Noisy Image

Denoised Image

Literature Review

Recognizing the limitations of these conventional approaches, the advent of deep learning has revolutionized image-denoising techniques. At the heart of this innovation lies the utilization of **Convolutional Neural Networks (CNNs)** and, in particular, the powerful concept of autoencoders. **Autoencoders**, a subclass of neural networks, are designed to learn efficient representations of input data by encoding and subsequently decoding it. Leveraging the inherent ability of autoencoders to capture intricate features, researchers have explored their application in image-denoising tasks. Furthermore, the **Integration of skip connections,** facilitating the direct flow of information between layers, enhances the efficiency of information transfer within the network, allowing for the preservation of fine details during denoising.

Traditional filters, such as Gaussian and median filters, were commonly employed to mitigate image noise. While these methods are straightforward and computationally efficient, they exhibit limitations when confronted with higher noise levels, often compromising image clarity due to the inherent blurring introduced during denoising. The Gaussian filters have an isotropic smoothing effect, meaning they treat all directions equally. In some cases, this may not be ideal, especially when dealing with images containing structures that are not isotropic. It is also ineffective in removing impulse noise (salt-and-pepper noise), which is common in images.

The performance of a median filter is highly sensitive to the choice of filter size. Choosing an excessively small size may not effectively remove noise, while choosing a large size may result in over-smoothing and loss of details. Both filters involve a trade-off between reducing noise and preserving image details. Striking the right balance depends on the specific characteristics of the image and the noise present.

Related Works

Attacks

(1) Fast Gradient Sign Method (FGSM)): This is one of the first attack methods that was introduced by Szegedy *et al*. Such methods craft adversarial examples by L-BFGS. Given a loss function L(X + ρ; θ), where θ denotes the parameters of the network, the goal is to maximize the loss function.
The fast gradient sign method (FGSM) is a one-step attack method and aims to find the adversarial perturbation by moving in the opposite direction to the gradient of the loss function L(X + ρ; θ), thus leading to the adversarial example Xadv:

$$Xadv=X+\varepsilon \cdot sign(\nabla L(X+\rho;\theta))$$

(2) Projected Gradient Descent (PGD): This is an iterative variant of FGSM. Thus, PGD is highly similar to FGSM. It replaces ε with α to take a small step in each iteration.PGD performs the update as follows:

$$Xi+1 = clip\varepsilon(Xi + \alpha \cdot sign(\nabla L(Xi + \rho;\theta)))$$

(3) Momentum Projected Gradient Descent: This method is similar to PGD with an additional momentum term to stabilize the direction of the gradient and escape local maxima. Momentum PGD is defined as follows:

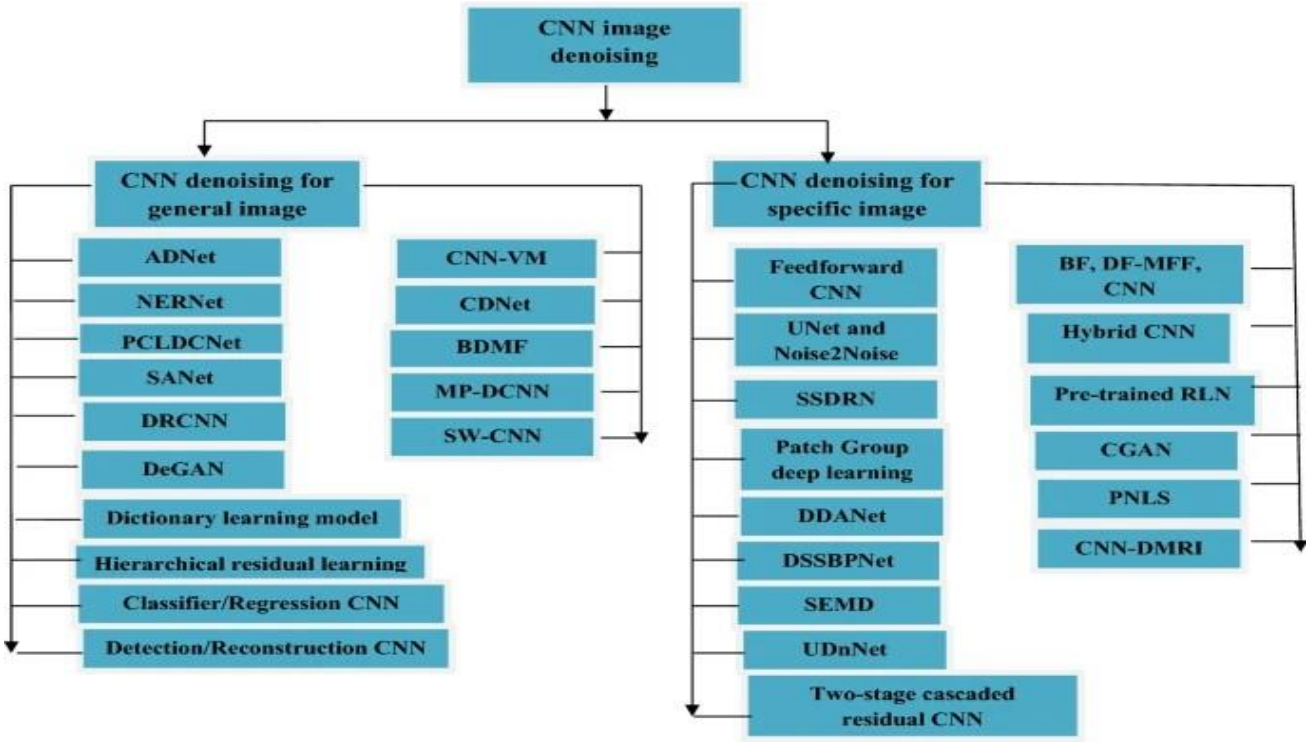
$$Gi+1=\mu \cdot Gi+||\nabla L(X+\rho;\theta)||,$$

Defenses

Our proposed method is intrinsically linked to image restoration and synthesis techniques based on learnable Convolutional Neural Networks (ConvNets), as discussed earlier. Simultaneously, it shares significant associations with an alternative category of restoration methods that bypass training on a hold-out set. This category includes approaches centered on joint modeling of groups of similar patches within a corrupted image, particularly useful for addressing

complex and highly variable corruption processes like spatially varying blur. Additionally, within this group are methods employing dictionary fitting to patches and convolutional sparse coding. The latter can also accommodate statistical models similar to shallow ConvNets in the reconstructed image. Previous research explores a model combining ConvNets with self-similarity-based denoising, bridging these two methodological groups but still requiring training on a hold-out set.

The prior imposed by deep ConvNets, as explored in our work, exhibits a profound connection to self-similarity-based and dictionary-based priors. The shared weights of convolutional filters across the entire spatial extent of the image inherently induce a level of self-similarity among individual patches, a characteristic generative ConvNets can potentially exploit. The association between ConvNets and convolutional sparse coding is further explored in the context of recognition networks and, more recently, in a single-layer convolutional sparse coding proposed for reconstruction tasks. Notably, our approach, when compared to similar methods, suggests that leveraging deep ConvNet architectures prevalent in modern deep learning-based methods may yield more accurate restoration results in certain circumstances.



Methodology

Overview

Image noise refers to the random variations in brightness or color that we observe in captured images. It's essentially a kind of interference introduced during the process of capturing the image from external sources. Mathematically, noise can be represented as ***A(x,y) = B(x,y) + H(x,y)***

Where,

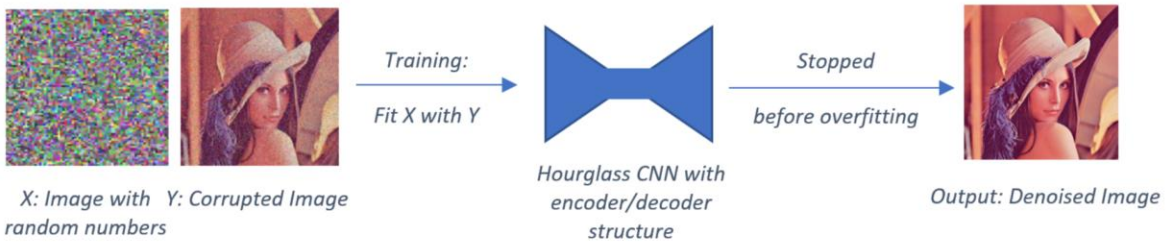
A(x,y) = function of noisy image

B(x,y) = function of original image

C(x,y) = function of noise

The extension of this methodology involves initializing a neural network and its subsequent training exclusively with the corrupted image. The network is set up to iteratively generate an image that closely resembles the initially corrupted input. The training process is carefully monitored to prevent overfitting, ensuring that the resulting image becomes progressively closer to the corrupted input but devoid of undesired artifacts, grains, and jagged edges. The key lies in strategically controlling the network's structure, including the number of hidden layers and units, as well as tuning hyperparameters, such as the learning rate and stopping criteria.

By halting the training at a strategic point, we achieve a balance where the network captures the original image's essential features while eliminating the undesired noise and artifacts. It's crucial to find this sweet spot during training to prevent overfitting, which could result in the network simply reproducing the exact corrupted image. This approach offers a unique advantage in noise removal, as it harnesses the inherent capabilities of neural networks without extensive learning requirements. Moreover, the controlled structure of the network allows for its versatile application, extending beyond noise removal to address issues like jagged edges and even smaller regions in need of inpainting.



To study this effect quantitatively, we consider the most basic reconstruction problem: given a target image x0, we want to find the value of the parameters $\theta*$ that reproduce that image. This can be set as the optimization of using a data term comparing the generated image to x0:

$$E(x;x0) = || x - x0 ||^2$$

Plugging this leads us to the optimization problem:

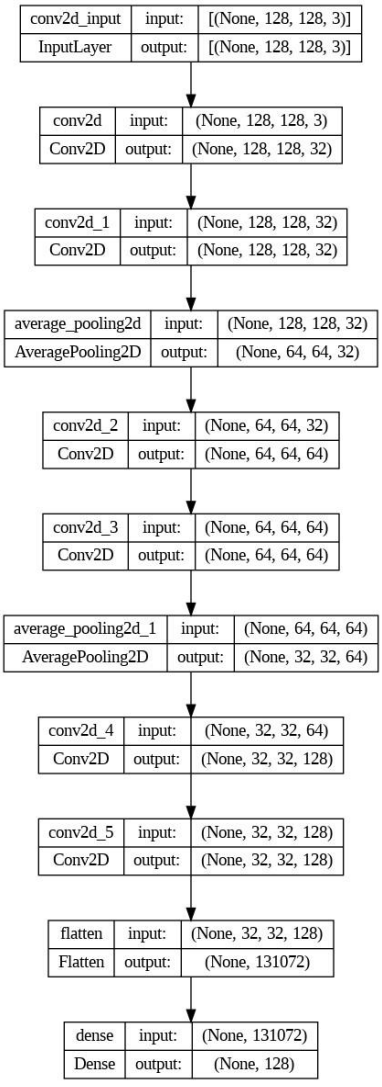
$$min_\theta || f_\theta(z) - x0 ||^2$$

Why Does It Work?

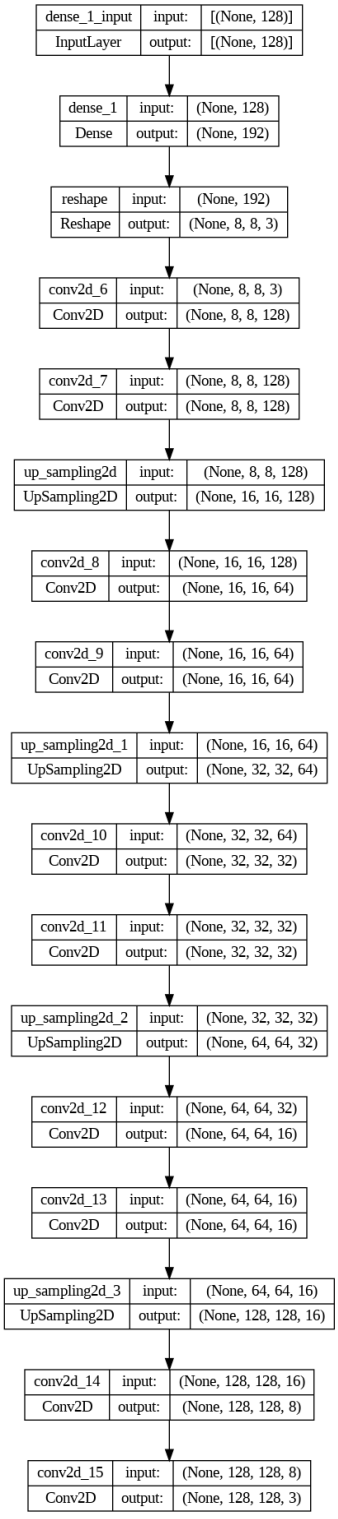
The network undergoes a unique process where it is compelled to transform a set of random numbers into an output closely resembling the provided corrupted image. This intentional constraint prevents the network from early exposure to the finer details of the noise. Instead, it prioritizes capturing the more significant features of the image, which facilitates effective learning without being hindered by the complexities introduced by noise. This approach prevents the network from fixating on noise-related patterns, which often have a high impedance compared to the more informative aspects of the image.

AutoEncoder without Skip Connections

Unlike autoencoders with skip connections, which facilitate the direct flow of information between encoder and decoder layers, the traditional autoencoder relies solely on the encoding and decoding process. The encoder compresses the input image into a lower-dimensional representation, and the decoder then attempts to reconstruct the original image from this compressed representation. During the training process, the autoencoder learns to capture essential features and patterns in the data, ultimately enabling it to generate denoised images when presented with noisy inputs.



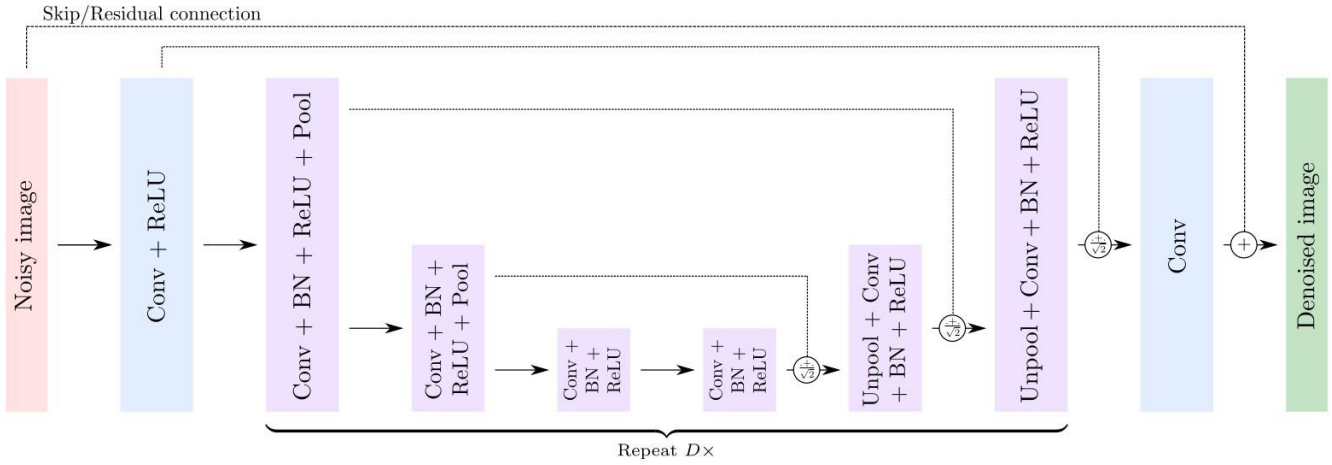
Encoder



Decoder

UNET Architecture with Skip Connections

The unit architecture, structured hierarchically, serves as a fundamental framework for feature extraction. This hierarchical approach enables the network to discern low-level and high-level features within the input image, fostering a nuanced understanding of its content. Introducing skip connections further amplifies the network's efficacy by establishing direct connections between layers. These connections act as conduits for the unimpeded flow of information, facilitating the seamless exchange of details. This strategic integration proves especially beneficial in our denoising task, as it aids in preserving finer image elements, preventing the loss of crucial information during the denoising process. The network structure can be referred to here.



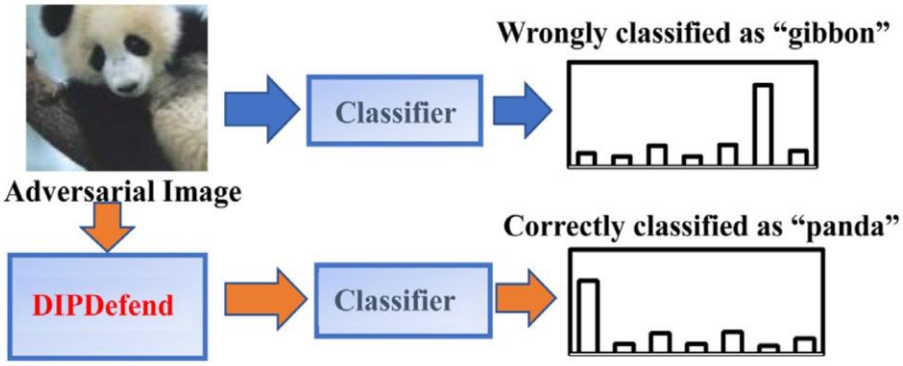
sequential_input	input:	[(None, 128, 128, 3)]
InputLayer	output:	[(None, 128, 128, 3)]

sequential	input:	(None, 128, 128, 3)
Sequential	output:	(None, 128)

sequential_1	input:	(None, 128)
Sequential	output:	(None, 128, 128, 3)

Deep Image Prior

The deep image prior (DIP) has demonstrated remarkable effectiveness, surpassing general external image priors, particularly in diverse image restoration tasks. Motivated by its widespread success, our investigation focuses on leveraging DIP as a defense mechanism against adversarial examples. DIP exhibits a unique characteristic of gradually reconstructing image structures, with the reconstructed images being accurately classified early in the reconstruction process. This distinctive attribute positions DIP as a promising tool for adversarial perturbation removal while concurrently rebuilding the original image structures. Our method assumes that the image reconstruction process in DIP follows a two-stage learning strategy: first, an early stage emphasizes robust feature learning for perturbation-insensitive patterns, and subsequently, a later stage focuses on non-robust feature learning for perturbation-sensitive patterns. This strategic two-stage learning approach forms the foundation of our method, contributing to its potential effectiveness in defending against adversarial attacks.



Experimental Setup

Code Implementation:

Programming language- Python.

Python Libraries- NumPy, Pandas, math, OpenCv (cv2), matplotlib.pyplot, Ipython.display, and Keras from TensorFlow.

Environment- Jupyter Notebook. It is an interactive web-based computing environment that allows users to create and share documents containing live code, equations, visualizations, and narrative text. Jupyter notebooks are organized into cells, each of which can contain code, text, or multimedia elements. This cell-based structure facilitates the step-by-step development of code and its immediate execution.

Platform Used: We have used **Google Colab** which is a cloud-based platform that provides free access to GPUs (Graphics Processing Units) and TPUs (Tensor Processing Units). It allows you to run Python and R code in a Jupyter Notebook-like environment.

Machine Configurations:

Runtime type: Python, *Hardware Accelerator:* T4 GPU

Resources used from Google Collab T4 GPU-

System RAM: 0.9 / 12.7 GB

GPU RAM : 1.6 / 15.0 GB

Disk : 26.2 / 78.2 GB

Specifications of T4 GPU-

processor : 0 vendor_id
: GenuineIntel cpu family:
6
model : 79
model name: Intel(R) Xeon(R) CPU @ 2.20GHz
stepping : 0
microcode : 0xffffffff
cpu MHz : 2199.998
cache size : 56320 KB

Memory Information-
MemTota : 13290480 kB
MemFree : 6627020 kB
MemAvailable:11185372 kB
Buffers :342096 kB Cached
:4413692 kB

Datasets:
We have created and used a dataset combining the **MNIST** (Modified National Institute of Standards and Technology) dataset, sample images from the **SSID** (Smartphone Image Denoising Dataset) Dataset, and some samples from **set14** dataset. The image shape in our dataset is 128*128*3.

MNIST (Modified National Institute of Standards and Technology) is a dataset of handwritten digits commonly used for training various image processing systems. It comprises 28x28 pixel grayscale images of handwritten digits (0 through 9) and corresponding labels. The MNIST dataset is often used for tasks related to digit recognition, but the noise is artificially added to it for denoising with Gaussian noise sigma 0.25.

SSID, also known as Smartphone Image Denoising Dataset, is enough for denoising experiments because Smartphone images often capture real-world scenes in diverse conditions, including low light, various environments, and different noise levels. It includes a wide variety of noise types commonly found in smartphone images (such as sensor noise, compression artifacts, etc.), which is beneficial. We have used only some sample images from this dataset as it is large enough and requires much processing time and space.**The main reason behind choosing the Set14 dataset is that** it consists of 14 images commonly used for testing the performance of Image Super-Resolution models.

Although we can do a train/test split, we haven't done it because We are using Autoencoders here with No Skip and Skip Connection (UNET Architecture). Both are the CNN architectures for specific image Denoising. We have experimented with these models on our small created dataset.

Results

Performance metrics-
We have chosen **PSNR (Peak Signal-to-Noise Ratio)** and **SSIM (Structural Similarity Index)** performance metrics to evaluate the quality of reconstructed or processed images.

PSNR is a measure of the quality of a reconstructed or processed signal in terms of the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. In the context of images, it is often used to measure the quality of a reconstructed image compared to the original image.
PSNR is calculated using the mean squared error (MSE) between the original and reconstructed images. The formula follows:

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$

MAX is the maximum possible pixel value (usually 255 for 8-bit images) and MSE is the mean squared error.
Higher PSNR values indicate better image quality. A higher PSNR implies that the reconstructed image has less distortion and is closer to the original.

The reason for choosing PSNR is that it is easy to compute, widely used, and provides a simple and intuitive measure of image quality. It is computationally efficient and can be quickly calculated

SSIM is a metric that quantifies the similarity between two images by considering three components: luminance, contrast, and structure. It is designed to mimic human perception and is sensitive to changes in structural information. The SSIM formula is based on three comparison measurements between the samples of x and y: luminance (l), contrast (c), and structure (s).

$$l(x,y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2}$$

$$s(x,y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3}$$

$$\mathrm{SSIM}(x,y) = l(x,y)^\alpha \cdot c(x,y)^\beta \cdot s(x,y)^\gamma$$

Setting the weights α, β, γ to 1, the formula can be reduced to the form shown above.

SSIM values range from -1 to 1, where 1 indicates perfect similarity. Higher SSIM values signify greater similarity between the original and reconstructed images. **The reason SSIM is chosen is that** it takes into account structural information, contrast, and luminance, making it more aligned with human perception. It provides a more comprehensive assessment of image quality.

Results Obtained from the Experiments -

1. For the Snail Image (100 epochs, 20 iterations):

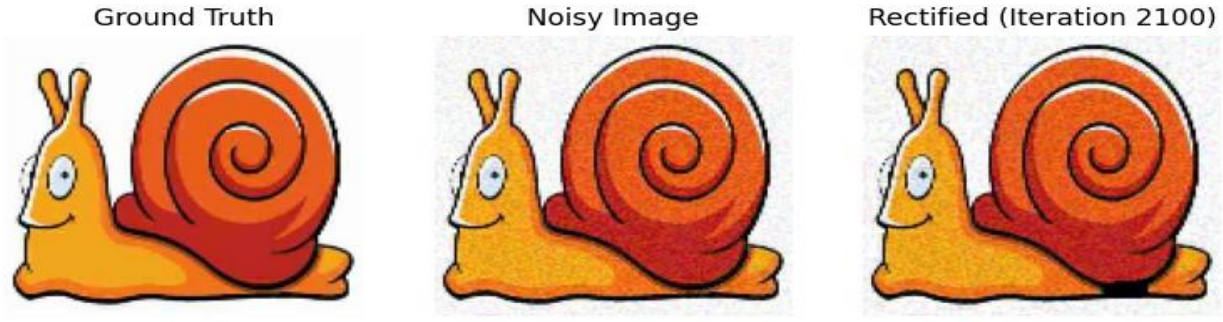
AutoEncoder with No Skip Connections (Model 1)		**Autoencoder with Skip Connections (UNET Architecture) (Model 2)**	
Actual Vs Noised Image	**Actual Vs. Denoised Image**	**Actual Vs. Noised Image**	**Actual Vs. Denoised Image**
PSNR value is 29.58 dB	**PSNR** value is 30.640 dB **SSIM** value is 0.8029	**PSNR** value is 29.59379 dB	**PSNR** value is 30.45 dB **SSIM** value is 0.8031

However, Model 2(Skip) should have performed better than Model 1. It is observed that the PSNR value of Model 1 is higher than Model 2, So The results suggest that the network with no skip connections (Model 1) is better at removing noise from the image. A network with no skip connections outperforms the skip

connections because our Model 2 suffers from overfitting. But with hyperparameter tunning Model 2 will definitely outperform Model 1.

The SSIM value of Model 2 should be better than Model 1 because the skip connections allow the network to bypass the encoder and decoder layers, which can help preserve the high-frequency components of the image that are important for preserving the structure. But here they both have almost similar values because the training data used for both models was relatively small or lacked significant structural variations, both models might not have learned significant differences in preserving image structure through skip connections

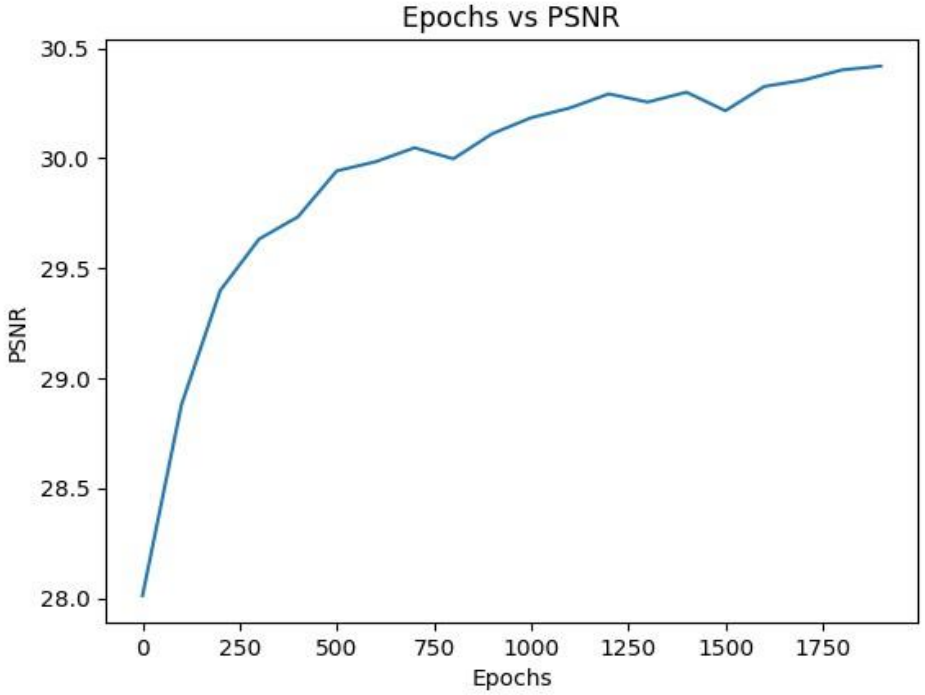
No skip(model 1)



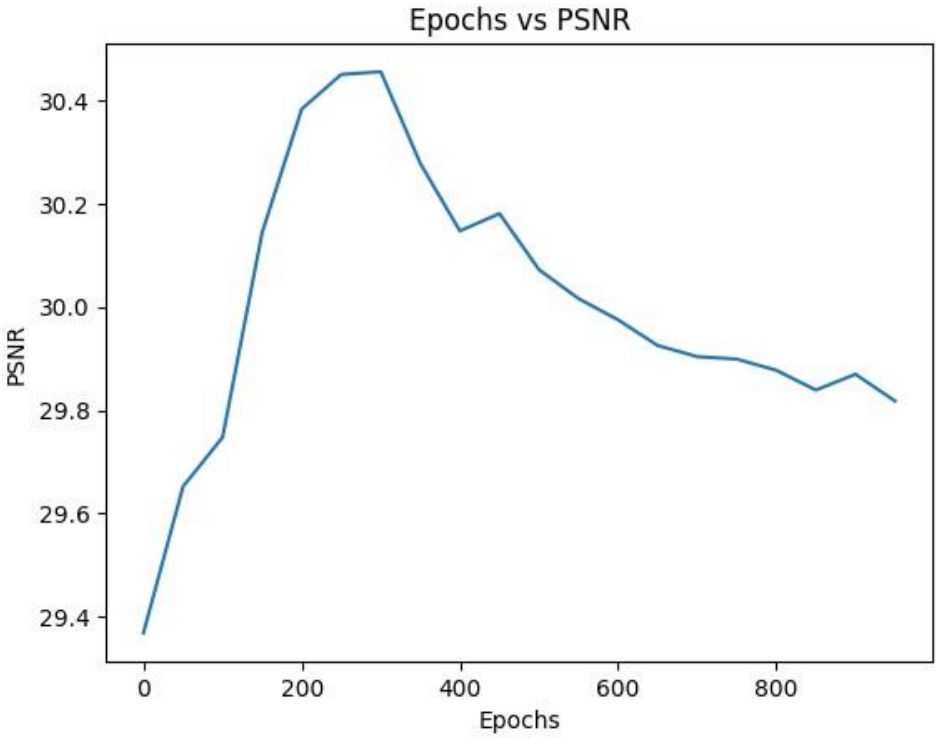
Skip (Model 2)

2. PSNR values For Different Images

AutoEncoder with No Skip Connections			Autoencoder with Skip Connections (UNIT Architecture)	
IDs	**Actual Vs. Noised Image**	**Actual Vs. Denoised Image**	**Actual Vs Noised Image**	**Actual Vs Denoised Image**
20056	28.8043 dB	30.20316 dB	28.7973 dB	29.8169 dB
20057	28.8626 dB	29.9506 dB	28.8711 dB	29.8796 dB
20061	28.9565 dB	30.4107 dB	28.9533 dB	29.6172 dB
20088	29.009 dB	29.5198 dB	29.0233 dB	29.6351 dB
GT SRGB_ 0101	28.8742 dB	29.9762 dB	28.8308 dB	29.3318 dB



Model 1 (No Skip Connections)



Model 2 (Skip Connections)

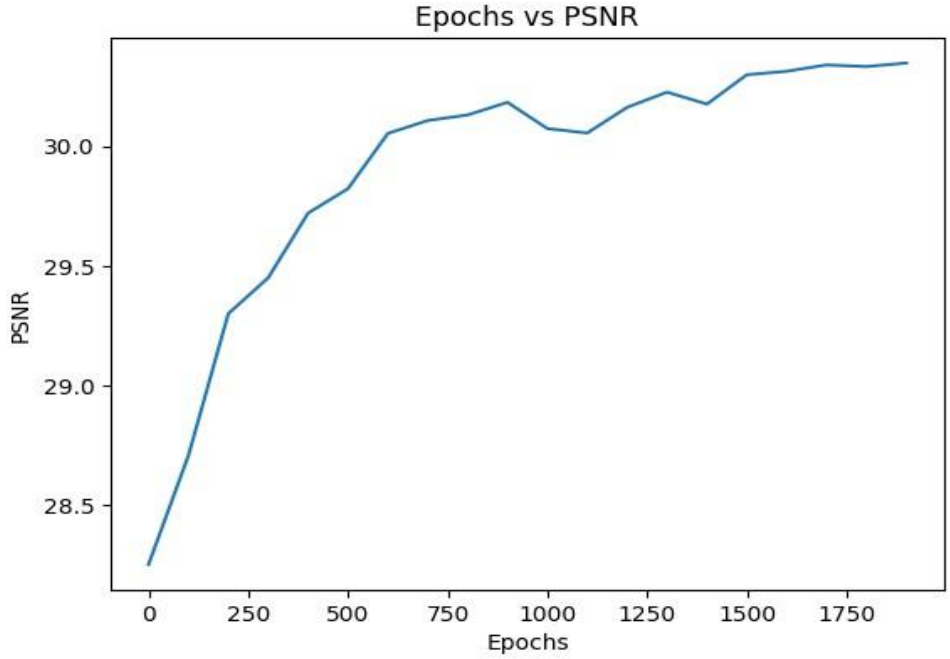
Ablation Studies

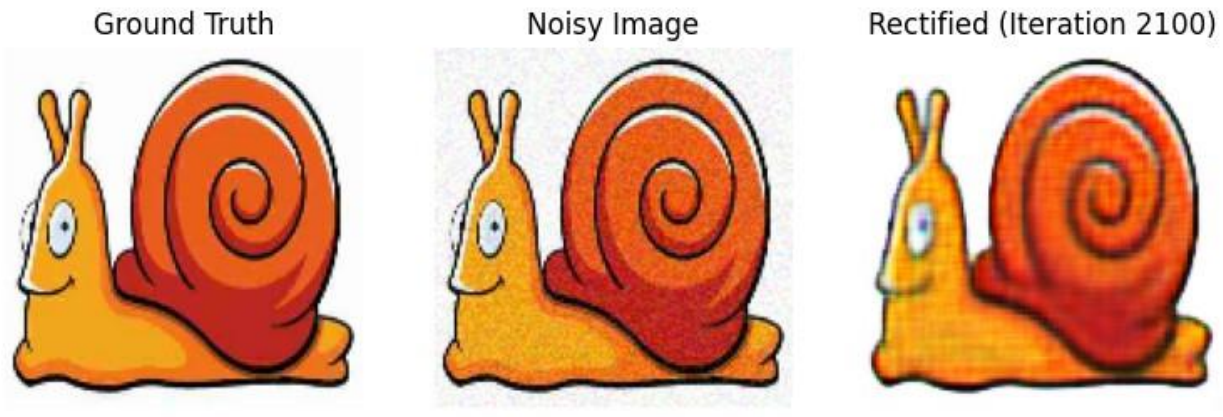
We have constructed two models for our project.
Model 1: Autoencoder Network
Model 2: UNET-based Autoencoder Network with Skip Connections.

Ablation Study on Model 1-

Stacked another layer of Encoder Decoder

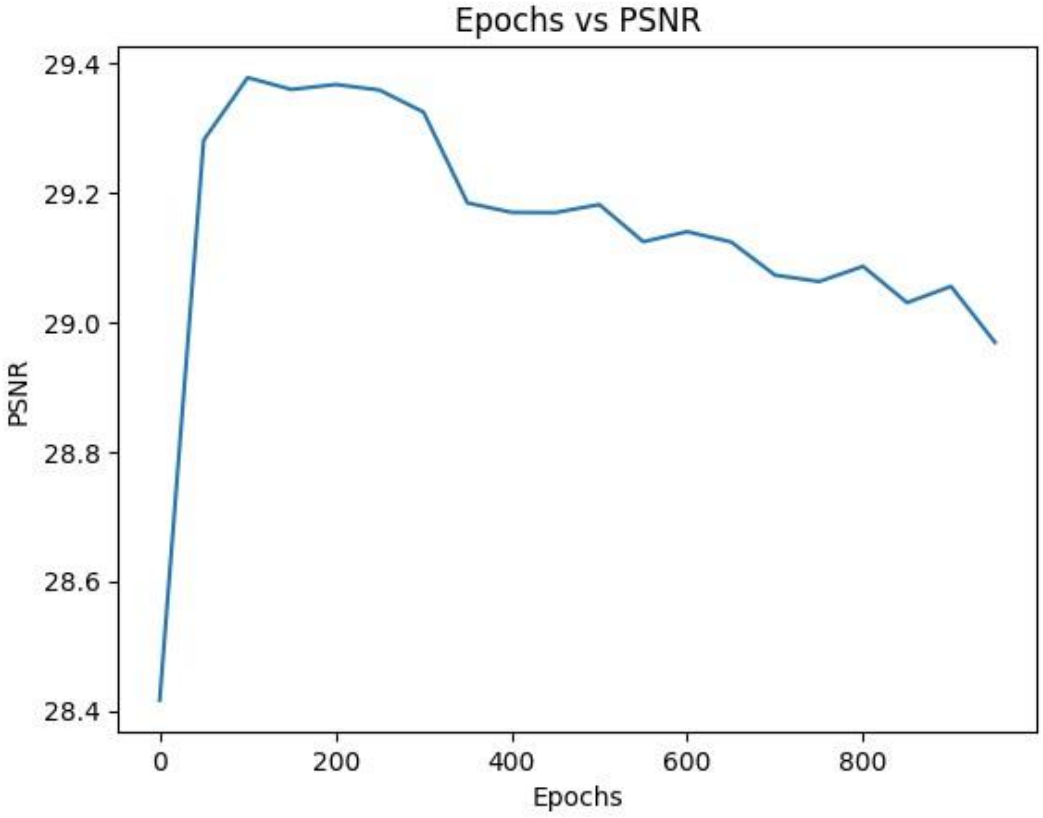
In the conducted ablation study, an additional layer of the encoder-decoder network was introduced to Model 1, aimed at enhancing its denoising capabilities. The resulting PSNR value of 30.35 dB serves as a quantitative measure of the model's performance, indicating a substantial improvement in the quality of denoised images. The decision to stack another layer was driven by the desire to explore the impact of increased model complexity on denoising efficacy. However, as with any architectural modification, careful consideration of potential overfitting and generalization issues is crucial. The study also emphasizes the importance of balancing model complexity and computational efficiency. Future iterations may involve fine-tuning hyperparameters and further experimentation to optimize the performance of the enhanced Model 1.

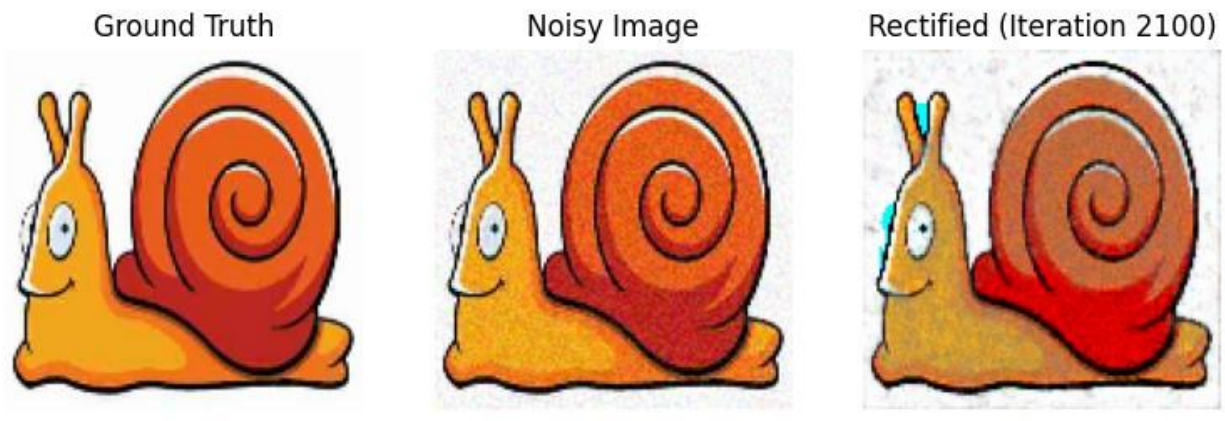




Ablation study for Model 2-

1. Changed the loss function to cosine similarity



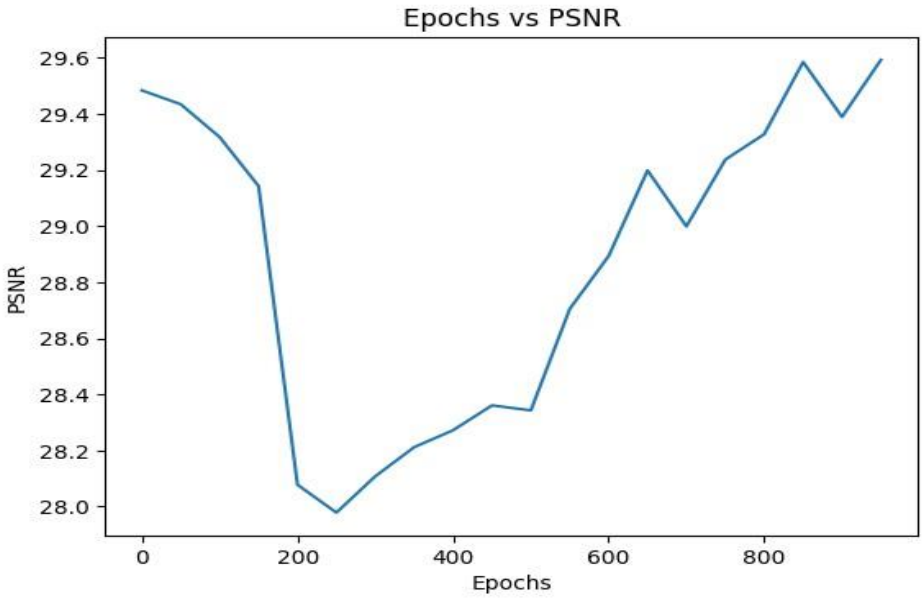


The observed change in the PSNR value from 30.45 dB to 29.55 dB in the ablation study on Model 2, where the loss function was switched to Cosine Similarity, is a significant finding. The transition in the loss function can have a notable impact on the training dynamics and the model's ability to minimize the discrepancy between the predicted and ground truth images. The decrease in PSNR suggests that, under the new loss function, the denoising performance may have been affected in comparison to the original loss function. It's crucial to interpret these results in the context of the specific characteristics of your denoising task and the behavior of the Cosine Similarity loss in capturing the similarity between predicted and target images.

Consider exploring the reasons behind this change by conducting a detailed analysis of denoised images, looking for visual artifacts or perceptual differences introduced by the altered loss function. Additionally, compare the performance of the model on different types of noise or under varying levels of noise to gain insights into the robustness of the model under the new loss regime.

This adjustment in the loss function serves as a reminder of the sensitivity of neural networks to the choice of objective functions, and it highlights the importance of carefully selecting and fine-tuning loss functions based on the specific characteristics of the denoising task at hand. Further experimentation, potentially incorporating a range of loss functions or hybrid loss combinations, could provide a more comprehensive understanding of the model's behavior and aid in refining the denoising process.

2. Changed the number of epochs to 10 per iteration



The alteration in the number of epochs from the original 20 to 10 in the ablation study on Model 2 has resulted in a change in the PSNR value from 30.45 dB to 29.5 dB. This shift indicates a potential impact of the training duration on the denoising performance of the model. The decrease in PSNR suggests that the model may not have converged fully or achieved its optimal denoising capability within the reduced number of epochs.

It's essential to interpret these findings considering the trade-off between computational resources and model convergence. A shorter training duration may lead to underfitting, where the model fails to capture the intricate patterns and

features required for effective denoising. Consider experimenting with intermediate values for the number of epochs to strike a balance between computational efficiency and model convergence.

Additionally, inspect the training dynamics by inspecting the training and validation loss curves. Understanding how the loss evolves over epochs can provide insights into whether the model has reached a stable denoising capability or if further training is warranted.

Iterative experimentation, accompanied by thorough analysis, will help fine-tune the training parameters for Model 2. This process may involve adjusting the learning rate, exploring different optimization algorithms, or considering learning rate schedules to enhance the convergence of the model within the specified training duration. By iteratively refining these training parameters, you can achieve a better understanding of the optimal conditions for denoising with Model

Discussions

Pre-Training

The approach of not requiring pre-training is a distinctive feature of the Deep Image Prior method. By utilizing corrupted images as the sole input, the network dynamically adjusts its weights during the training process. This implies a form of unsupervised learning, where the network learns to reconstruct images solely from the corrupted versions, showcasing the adaptability and flexibility of the method.

Variety of Applications

The versatility of the Deep Image Prior method is a noteworthy aspect. Extensive testing has demonstrated its efficacy across various image processing tasks, including denoising, super-resolution, inpainting, natural pre-image, flash-no-flash reconstruction, and more. This wide applicability positions the method as a powerful tool for diverse computer vision challenges.

Limitations of the Project

Despite the promising results, this project has certain limitations. The models were trained and evaluated on a specific dataset, and their generalizability to diverse real-world scenarios is subject to validation on more extensive datasets. The scope of noise types and levels considered in the project may not cover all potential real-world scenarios, limiting the models' adaptability to various noise characteristics.

Biggest Risks and Mitigation

The most significant risk that could lead to project failure is the potential lack of robustness of the models in handling unforeseen or complex noise patterns. Continuous model evaluation on diverse datasets and noise scenarios is essential to address this risk. Regular updates to the training data with more diverse samples can enhance model robustness. Additionally, incorporating data augmentation techniques and exploring advanced noise modeling approaches can help mitigate the risk of model failure due to inadequate training data.

Processing Time

A potential trade-off associated with the Deep Image Prior method is the computational cost. Processing time is expected to be relatively high since the network needs to be executed individually for each image. Both forward propagation and computationally expensive backpropagation are required for reproducing each image. This characteristic contrasts with supervised networks that only necessitate forward propagation in production environments. Efficient implementation strategies and optimization techniques would be crucial for practical deployment.

The Project's Future Scope:

The project opens avenues for future exploration and enhancement. Further refinement of Model 1 through additional ablation studies or hyperparameter tuning could yield even better denoising results. Exploring hybrid architectures that combine the strengths of Model 1 and Model 2 may lead to a more comprehensive solution, leveraging the benefits of increased model complexity and skip connections.

One Ring to Rule Them All?

The discussion around whether a single network architecture can be universally effective for various images raises essential considerations. While the methodology provides compelling results, the practicality of handcrafting a network for each image type in production environments is questioned. The challenge lies in achieving a balance between a generic architecture that works well across diverse datasets and the nuanced customization often required for optimal performance. Further research could explore the development of adaptable architectures or meta-learning approaches to address this challenge, allowing for more generalized deployment without sacrificing performance.

New Applications

The denoising models developed in this project have broader applications beyond the current scope. They can be applied to medical imaging for noise reduction in various modalities, such as MRI or X-ray images. Furthermore, the models can be integrated into real-time systems for video denoising, benefiting applications like video surveillance and autonomous vehicles. Additionally, the

denoising techniques could find utility in enhancing the quality of images captured in low-light conditions, thereby extending the applicability of the project to diverse domains.

Conclusions

In the landscape of image denoising, where traditional methods often rely on abundant training data, the Deep Image Prior method emerges as a pioneering approach that challenges the conventional paradigm. Unlike models dependent on extensive pre-training datasets, Deep Image Prior demonstrates a remarkable capability to operate with zero pre-training. While the computational cost of de-corrupting images is notable, even on GPU platforms, the method surpasses existing techniques that demand no training, underscoring its efficacy in capturing intricate image statistics through network architecture.

The significance of Deep Image Prior extends beyond being a mere alternative in scenarios where datasets are scarce. It not only presents a compelling solution when no pre-trained models are available but also lays the foundation for a broader exploration of unsupervised deep learning approaches. The method's ability to outperform existing methods, coupled with its adaptability to scenarios with minimal or no training data, accentuates the potential for further advancements in unsupervised learning.

Despite the successes observed, it's essential to acknowledge the limitations of the project. The models were trained and evaluated on a specific dataset, and their generalizability to diverse real-world scenarios needs further validation on more extensive datasets. The scope of noise types and levels considered may not cover all potential scenarios, prompting the need for continuous model evaluation and refinement.

The Deep Image Prior method, explored in the context of pre-training and its diverse applications, brings a unique perspective to the discussion. The adaptability demonstrated by the method without needing pre-training is a remarkable feature, albeit at the cost of increased processing time. The potential challenge of handcrafting a network for each image type in production

environments prompts further consideration of how to achieve a balance between generality and specificity in network architectures.

In summary, this project has contributed valuable insights to the field of image denoising, showcasing the complexities involved in model development and the nuanced considerations required for real-world deployment. As we move forward, continued research and development will be essential for addressing challenges, expanding the scope of applications, and refining the models for practical implementation in diverse settings and the insights gained from both the ablation studies on traditional models and the exploration of innovative approaches like Deep Image Prior contribute to a deeper understanding of the complexities and possibilities within the realm of image denoising. Looking forward, the success of Deep Image Prior sparks curiosity and enthusiasm for future research endeavors, paving the way for developing more efficient, adaptive, and unsupervised deep learning methods in computer vision.

References

[1] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in European Conference on Computer Vision, Springer, 2016, pp. 630–645

[2] A. Kurakin, I. Goodfellow, S. Bengio, Adversarial examples in the physical world, in International Conference on Learning Representations, 2017.

[3] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, in International Conference on Learning Representations, 2014.

[4] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, J. Li, Boosting adversarial attacks with momentum, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 9185–9193.

[5] S.-M. Moosavi-Dezfooli, A. Fawzi, P. Frossard, Deepfool: a simple and accurate method to fool deep neural networks, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2574–2582.

[6] W.D. Carlini N, Towards evaluating the robustness of neural networks, Security and Privacy, 2017.