Selected Topics in Visual Recognition using Deep Learning, Homework 4

111550061, 邱冠崴

GitHub: HW4_LINK

Part. 1, Introduction

In this assignment, the objective is to perform image restoration for two types of degradations: rain and snow, using purely vision-based models such as PromptIR [1]. After training, the predictions are converted into .npz files for submission to Codabench, where model performance is evaluated based on PSNR (Peak Signal-to-Noise Ratio).

I selected PromptIR for fine-tuning. The original implementation employed only L1 loss and test-time augmentation (TTA) [2], resulting in a PSNR of just 32.02 on the public test set. To enhance performance, I conducted additional experiments by modifying the loss function to incorporate a combination of L1 loss, SSIM (Structural Similarity Index) [3], and perceptual loss [4] using VGG16 [5]. Furthermore, I applied ensemble methods. These improvements significantly boosted the PSNR to 32.64 and 32.73, respectively.

Part. 2, Method

(1) Data Pre-processed:

I use the RandomCrop function from torchvision to extract a random patch of size patch_size × patch_size from both the clean and degraded images. This not only introduces spatial diversity but also helps reduce memory consumption during training. Additionally, I apply a series of data augmentation techniques—including random horizontal flipping, random rotation (±15°), color jittering, Gaussian blur, and random erasing—to simulate real-world variations and enhance the model's generalization ability.

(2) Model architecture and hyperparameters:

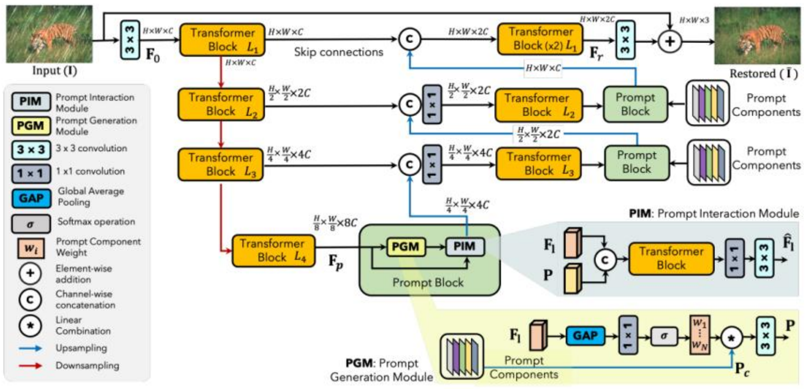
Model	PromptIR
Optimizer	Adamw [6]
Learning rate	$2*10^{-4}$
Batch Size	2
Patch Size	224
Scheduler	LinearWarmupCosineAnnealingLR [7] (warmup_epochs=15)

Model architecture (PromptIR):

I chose PromptIR as my model because of its strong generalization ability, unified multi-task design, and competitive performance across various types of image degradation. As an all-in-one blind restoration framework, PromptIR can handle multiple degradation types without requiring task-specific supervision, making it more practical and generalizable compared to conventional models that are typically tailored for a single task.

The main advantages of PromptIR include its task-agnostic capability, high restoration quality, and modular prompt design, which make it easy to extend or adapt to new scenarios. However, there are also some limitations. Due to its Transformer-based backbone and the integration of multi-scale prompt modules, PromptIR tends to have higher memory consumption and longer training times compared to lightweight CNN-based architectures.

The figure of PromptIR architecture:



PromptIR can be divided into several main components, including an encoder, a decoder, a Transformer backbone, and a set of prompt modules that guide the restoration process.

(a) Encoder:

The encoder is built in a hierarchical fashion with four levels. At each level, the input is downsampled, and features are extracted using Transformer blocks. These blocks consist of attention and feed-forward layers that effectively capture both local and global information. The encoder captures multi-scale representations that are essential for restoring fine details and structures in degraded images.

(b) Latent Transformer Blocks:

At the deepest level of the encoder, multiple Transformer blocks are used to refine the encoded features before passing them to the decoder. This latent stage helps the model understand the global context of the image.

(c) Decoder:

The decoder mirrors the encoder structure. It progressively upsamples the features to reconstruct the restored image, using skip connections from the encoder to retain spatial information. Transformer blocks are also used in the decoder to enhance feature transformation and fusion.

(d) Prompt modules

The most distinctive part of PromptIR is its prompt modules, which are inserted at multiple levels of the decoder. Each prompt module generates content-aware prompts based on the input features and modulates the decoding process accordingly. These prompts act as dynamic guidance, allowing the model to adapt to different degradation types during inference, even without explicit task labels.

Train strategy:

In the original implementation provided on the PromptIR GitHub repository [8], the authors trained the model using only the L1 loss. When I replicated this setup and incorporated test-time augmentation (TTA) during inference, the model achieved a PSNR of only 32.02. TTA is a technique

used to improve the robustness and accuracy of model predictions by applying various augmentations—such as horizontal flips, vertical flips, or rotations—to the input image at inference time. The model generates predictions for each augmented version, and the final output is obtained by averaging the results after reversing the augmentations. However, I believe that L1 loss only captures pixel-wise differences and fails to account for perceptual and structural information, which are critical in image restoration tasks. To address this limitation, I adopted a combined loss function consisting of L1 loss, SSIM (Structural Similarity Index), and perceptual loss based on VGG16 features to provide more comprehensive supervision. Further details are discussed in the Additional Experiments section.
In addition, I found that the default patch size of 128 was too small, which may cause the model to miss important contextual information and fine details, making image restoration more difficult. Therefore, I increased the patch size to 224. To prevent out-of-memory issues due to the larger input size, I also reduced the batch size from 8 to 2.

Part. 3, Results

(1) Training curve

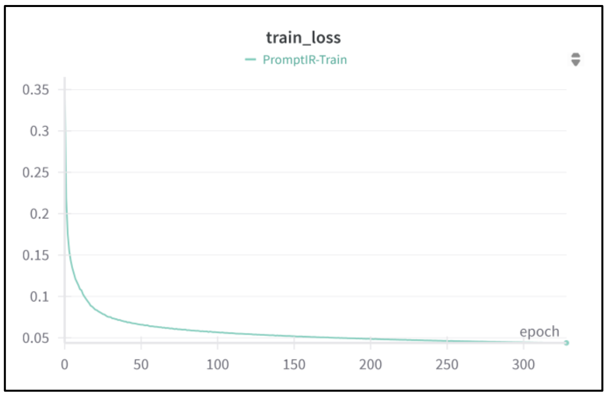


Figure 1

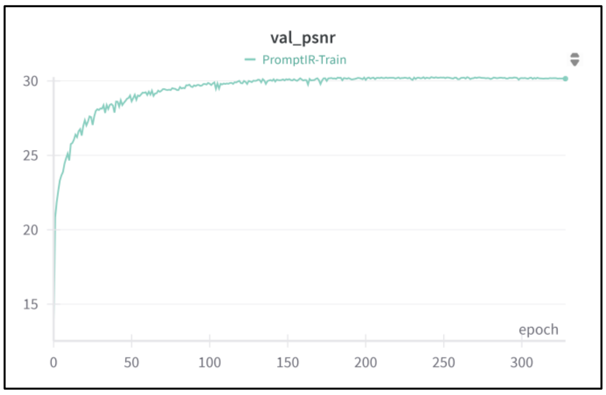


Figure 2

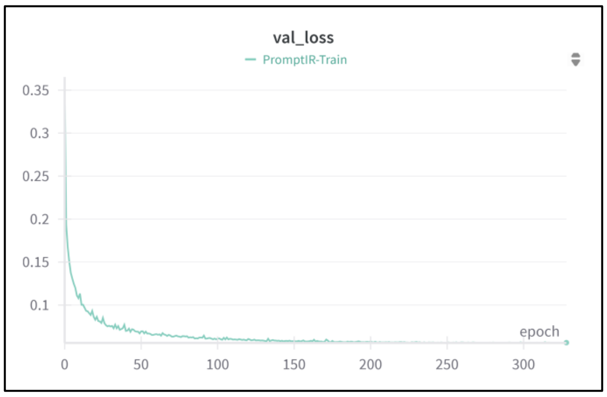
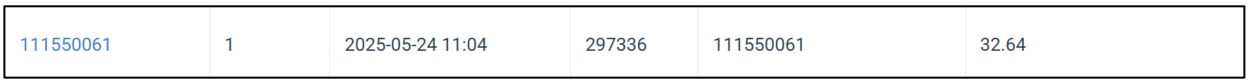


Figure 3

The learning curves of the PromptIR training process demonstrate stable and effective model convergence. As shown in Figure 1, the training loss steadily decreases from approximately 0.36 to around 0.043 over the course of 300 epochs, indicating consistent learning without signs of instability. Similarly, the validation loss in Figure 3 exhibits a downward trend that closely mirrors the training loss, suggesting good generalization to unseen data and no evident overfitting. The validation PSNR curve in Figure 2 rises sharply during the initial training phase, particularly within the first 50 epochs, and gradually plateaus around a value of 30 after approximately 100 epochs. Beyond that point, it continues to increase at a slower pace, ultimately reaching a peak PSNR of 32.2 on the validation set. This trend suggests that most performance gains occur early in training, with diminishing improvements in later epochs.

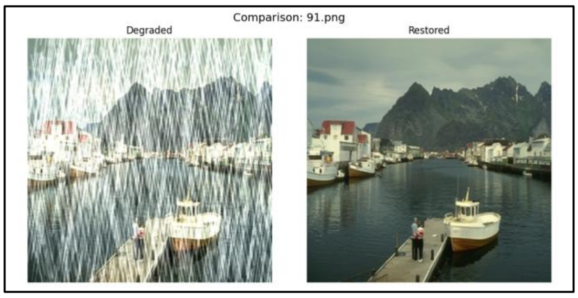
Based on the insights from the three figures, I eventually chose the model weight obtained after fine-tuning for 265 epochs. This checkpoint achieved a PSNR of 32.64 on the public test set.



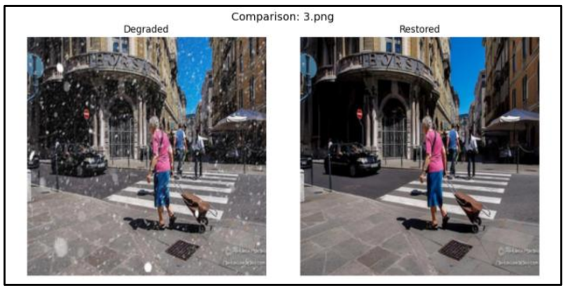
111550061	1	2025-05-24 11:04	297336	111550061	32.64

(2) Predicted clean images visualization

Rain:



Snow:



In the first left comparison image (91.png), the left image shows a scene heavily obscured by synthetic rain streaks, which significantly reduce visibility and distort the overall appearance of the landscape. The restored image on the right demonstrates a successful removal of the rain, revealing clear details. The colors appear natural, and the scene regains its original clarity, highlighting the model's capability in handling rain degradation.

In the second right comparison image (3.png), the degraded image on the left is affected by synthetic snow, with white blotches scattered across the frame. These artifacts obscure key elements such as the pedestrian, crosswalk, and vehicles, disrupting the visual coherence of the urban scene. The restored image on the right effectively eliminates the snow artifacts, restoring sharpness and detail across the entire image.

Part. 4, References

[1] V. Potlapalli, S. W. Zamir, S. Khan, and F. S. Khan, "PromptIR: Prompting for all-in-one blind image restoration," in Proc. 37th Int. Conf. Neural Information Processing Systems (NeurIPS), New Orleans, LA, USA, Dec. 2023, Art. no. 3121, pp. 71275–71293.

[2] D. Shanmugam, D. Blalock, G. Balakrishnan and J. Guttag, "Better Aggregation in Test-Time Augmentation," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 2021, pp. 1194-1203, doi: 10.1109/ICCV48922.2021.00125.

[3] Zhou Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," in IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600-612, April 2004, doi: 10.1109/TIP.2003.819861.

[4] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 694–711.

[5] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), Kuala Lumpur, Malaysia, 2015, pp. 730-734, doi: 10.1109/ACPR.2015.7486599.

[6] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[7] I. Loshchilov and F. Hutter, "SGDR: Stochastic Gradient Descent with Warm Restarts," *arXiv preprint arXiv:1608.03983*, 2016.

[8] V. Potlapalli, S. W. Zamir, S. Khan, and F. S. Khan , "PromptIR: Prompting for All-in-One Blind Image Restoration," GitHub repository, https://github.com/va1shn9v/PromptIR, 2023.

Part. 5, Additional experiments

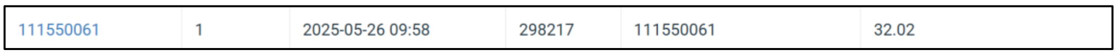
(1) Experiment 1:

After experimenting with L1 loss as the sole loss function, I observed that it consistently failed to achieve satisfactory PSNR scores. In my view, this is because L1 loss primarily captures pixel-wise differences and does not account for perceptual and structural aspects that are essential in image restoration tasks. To address this limitation, I incorporated SSIM and perceptual loss based on VGG16 into the loss function. The SSIM component helps preserve structural information such as edges, textures, and contrast by modeling human visual perception. Meanwhile, the perceptual loss captures high-level semantic and appearance differences by comparing intermediate feature representations of the restored and ground truth images extracted from a pretrained VGG16 network. As a result, the final loss function was defined as:

L1 loss + 0.2 × (1 − SSIM) + 0.05 × perceptual loss.

As a result, the PSNR on the public test dataset improved significantly—from 32.02 to 32.64—after modifying the loss function. This substantial performance gain supports my hypothesis that incorporating SSIM and perceptual loss can effectively enhance the model's ability to capture structural and perceptual information, leading to improved image restoration quality.

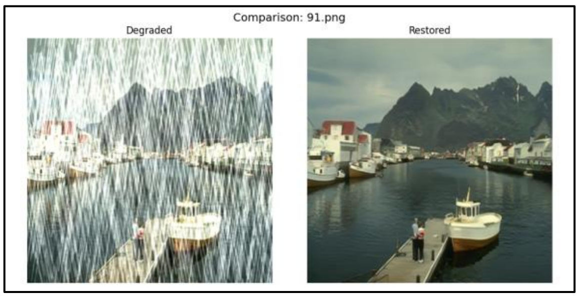
PromptIR model with only L1 loss:



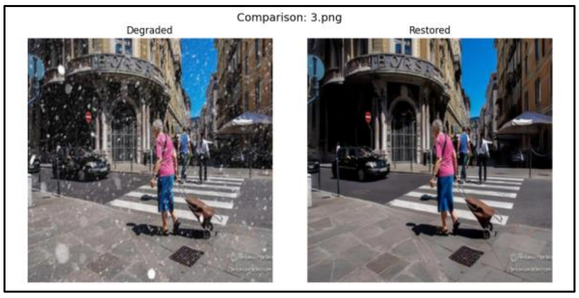
111550061	1	2025-05-26 09:58	298217	111550061	32.02

Visualization:

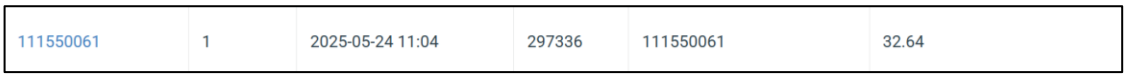
Rain:



Snow:



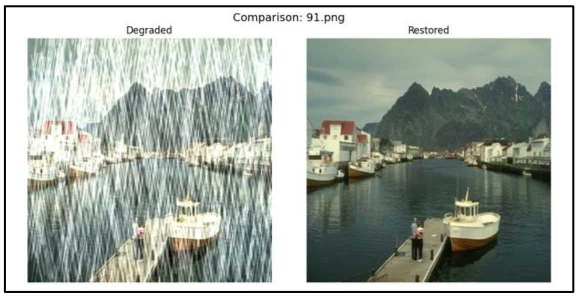
PromptIR model with combination of L1 loss, SSIM, and perceptual loss:



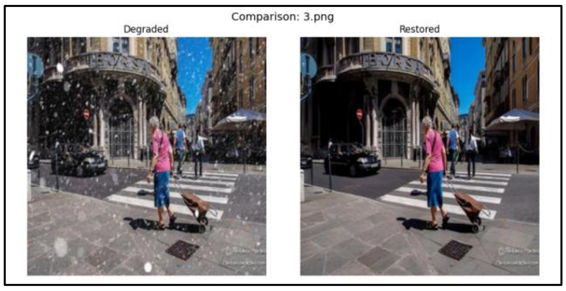
111550061	1	2025-05-24 11:04	297336	111550061	32.64

Visualization:

Rain:



Snow:



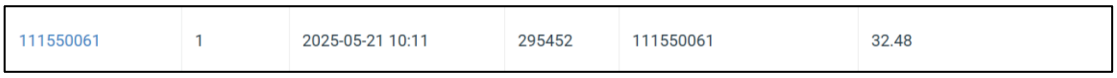
(2) Experiment 2:

In addition to modifying the loss function, I applied an ensemble method to further enhance the performance of the image restoration task. Specifically, I combined the outputs of two restored images by averaging them pixel-wise. One image was generated by a model trained with a batch size of 4 and a patch size of 160, while the other was produced by a model trained with a batch size of 2 and a patch size of 224.

My hypothesis was that ensembling could help reduce noise and artifacts present in individual outputs, leading to a more accurate and visually consistent restoration result. Since each model might capture slightly different features or details, averaging the outputs can smooth out inconsistencies and reinforce the commonly restored structures.

The original PSNR scores of the two models on the public test set were 32.48 and 32.64, respectively. After applying the ensemble method, the PSNR increased slightly to 32.73. Although the improvement was modest, it confirmed that even simple ensemble techniques can contribute to performance gains in image restoration tasks and also supports my hypothesis.

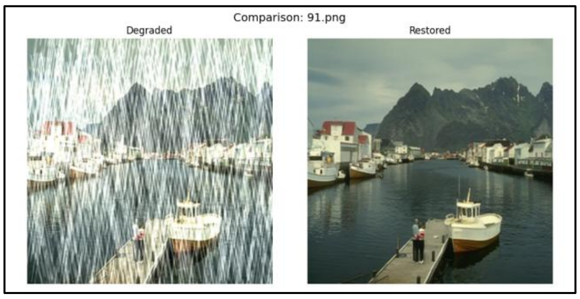
PromptIR model with batch size of 4 and patch size of 160:



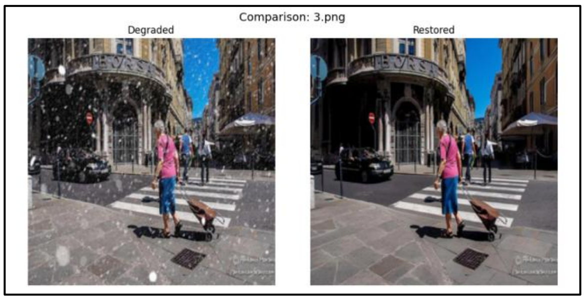
111550061	1	2025-05-21 10:11	295452	111550061	32.48

Visualization:

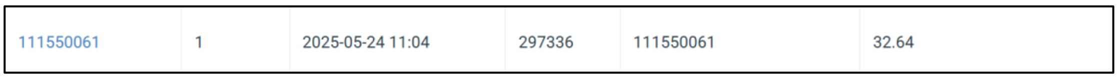
Rain:



Snow:



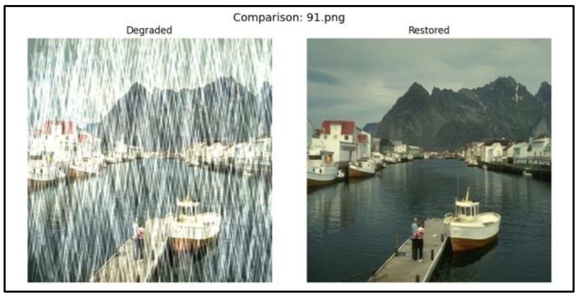
PromptIR model with batch size of 2 and patch size of 224:



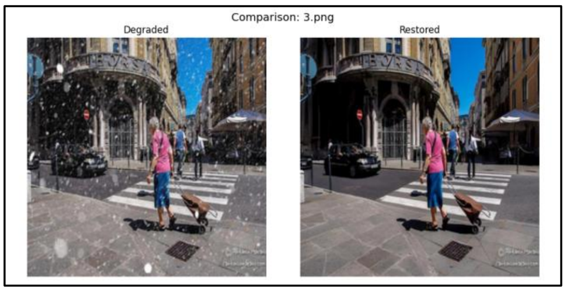
111550061	1	2025-05-24 11:04	297336	111550061	32.64

Visualization:

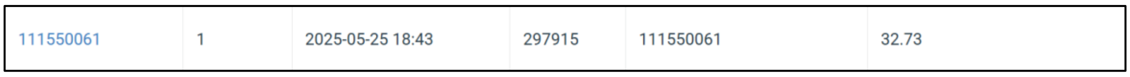
Rain:



Snow:



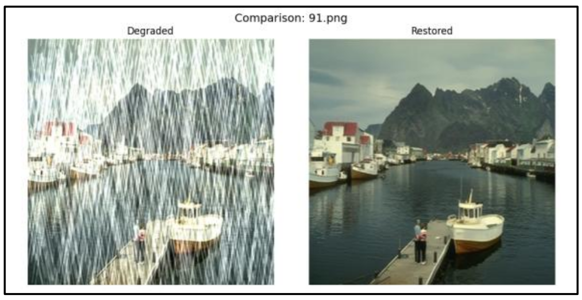
Ensemble method:



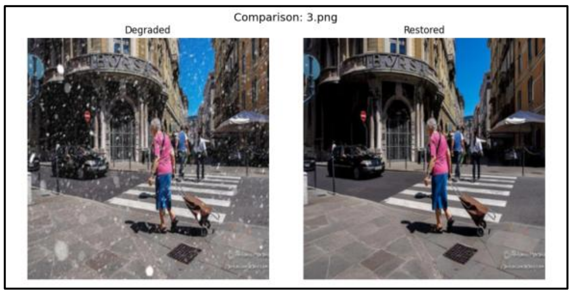
111550061	1	2025-05-25 18:43	297915	111550061	32.73

Visualization:

Rain:

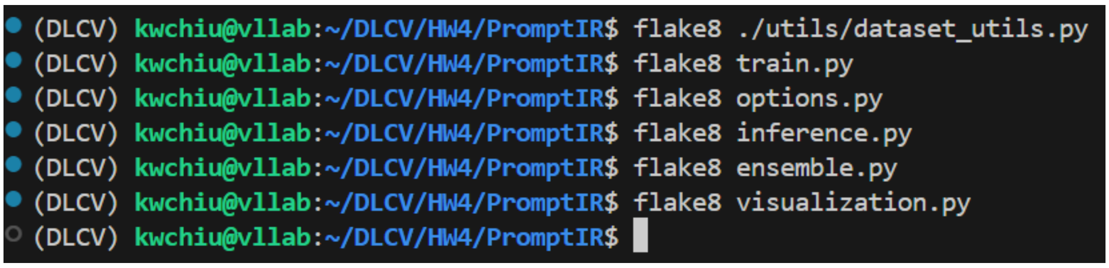


Snow:



Part. 6, Code Reliability

Lint the code:



```
● (DLCV) kwchiu@vllab:~/DLCV/HW4/PromptIR$ flake8 ./utils/dataset_utils.py
● (DLCV) kwchiu@vllab:~/DLCV/HW4/PromptIR$ flake8 train.py
● (DLCV) kwchiu@vllab:~/DLCV/HW4/PromptIR$ flake8 options.py
● (DLCV) kwchiu@vllab:~/DLCV/HW4/PromptIR$ flake8 inference.py
● (DLCV) kwchiu@vllab:~/DLCV/HW4/PromptIR$ flake8 ensemble.py
● (DLCV) kwchiu@vllab:~/DLCV/HW4/PromptIR$ flake8 visualization.py
○ (DLCV) kwchiu@vllab:~/DLCV/HW4/PromptIR$
```