[Github link]

1. Introduction

Task Overview

This assignment focuses on the challenging task of **blind image restoration**, where the goal is to recover clean images from degraded inputs affected by **rain** or **snow**. Unlike traditional restoration tasks that handle only a single type of corruption, this task requires designing a single model capable of restoring multiple degradation types without prior knowledge of the specific corruption present in each input. The dataset includes paired degraded and clean images for both rain and snow conditions, and model performance is evaluated using **Peak Signal-to-Noise Ratio (PSNR)**. As external data and pretrained weights are not allowed, we adopt and improve the **PromptIR** architecture, a recent model that introduces tunable prompts to encode degradation-specific features.

Method Overview

I use PromptIR as the backbone model. PromptIR is a novel transformer-based framework for blind image restoration, where the model doesn't rely on explicit knowledge of degradation types. Instead, it learns to adaptively guide restoration via learned prompts. These prompts act like conditioning signals generated on-the-fly to steer the model in different stages of the restoration process. It builds on hierarchical transformers and introduces Prompt Blocks (Prompt Generation Module PGM and Prompt Interaction Module PIM) that adaptively generate and inject restoration cues at different stages of decoding.

To improve the adaptability and expressiveness of prompt features in PromptIR, I introduced two key modifications: adjusting the prompt length and enhancing the Prompt Generation Block (PromptGenBlock) with multi-scale feature processing.

2. Data Preprocessing

Data Augmentation

In PromptIR, the data augmentation strategy is simple yet effective, aimed at enhancing generalization for blind image restoration. The authors apply random horizontal flipping with a 50% probability and random rotations in 90-degree increments (0°, 90°, 180°, or 270°) to introduce spatial diversity. Additionally, random cropping is performed to extract fixed-size patches (e.g., 128×128) from both the input image and its corresponding ground truth, promoting robustness to scale and positional variance. These augmentations are applied consistently to both the degraded input and the clean target to preserve alignment, ensuring that the model learns meaningful mappings under varied spatial transformations.

Loading Images

In my preprocessing pipeline for the dataset, each degraded image from the degraded/ folder is paired with its corresponding clean image in the clean/ folder by matching filenames

with modified prefixes (e.g., 'rain-' → 'rain_clean-'). The degradation type is inferred from the filename and encoded as an integer label (0 for rain, 1 for snow), which is later used for conditioning the model. During training, paired images undergo synchronized data augmentations including random horizontal flipping and random cropping to a fixed patch size (e.g., 128×128), ensuring spatial alignment. Images are then normalized to the [0, 1] range and converted to PyTorch tensors. The dataset returns a tuple containing the clean image name, degradation type ID, degraded tensor, and clean tensor, enabling the model to learn degradation-aware restoration.

3. Model Architecture & Training Strategy

Model Summary

PromptIR adopts a hierarchical encoder-decoder architecture based on transformer blocks, designed for blind image restoration. The model begins with an overlapping patch embedding layer, followed by a series of transformer blocks across four encoding stages, each increasing in depth and receptive field via downsampling. After processing through a latent transformer bottleneck, the decoder mirrors this hierarchy through upsampling and transformer-based reconstruction, supported by skip connections. A key innovation is the Prompt Generation Module (PGM), which generates spatial prompts from a set of learnable tokens. These prompts are dynamically selected based on the global image context and injected into the decoder to guide restoration.

To improve the model's adaptability to diverse degradations, I introduced two enhancements to the prompt mechanism. First, I varied the number of prompt tokens (prompt_len)—experimenting with values of 2, 5, and 10—to explore how the token set size affects performance under different conditions. Second, I modified the PromptGenBlock by incorporating multi-scale convolutions (3×3, 5×5, 7×7) followed by a 1×1 convolution for adaptive fusion. This enables the model to aggregate contextual features across varying spatial scales, improving its ability to handle structurally diverse degradations like rain and snow.

Training Setup

Optimizer	`optim.AdamW(filter(lambda p: p.requires_grad, model.parameters()), lr=2e-4, weight_decay=1e-4)`
Scheduler	`LinearWarmupCosineAnnealingLR( optimizer=optimizer,warmup_epochs=15, max_epochs=150)`
Loss	`nn.L1Loss()`
LR	`2e-4`
Batch size	`16`

Evaluation

Use PSNR for performance evaluation. PSNR (Peak Signal-to-Noise Ratio) is a widely used metric in image and video processing to measure the quality of a reconstructed or compressed image compared to the original (reference) image. It's especially common in tasks like denoising, compression, and image restoration. PSNR is calculated as followed:

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

Higher PSNR = better image quality (less distortion) and lower PSNR = more error/noise. A common intuition:

- > 40 dB – almost indistinguishable from the original.
- 30–40 dB – good quality.
- 20–30 dB – moderate quality (visible artifacts).
- < 20 dB – poor quality.

4. Experiment

More Training Epochs

Based on my observations of the PromptIR training process, I found that the model typically requires more than 200 epochs to achieve optimal performance. Notably, the model demonstrates a resistance to overfitting throughout training. The corresponding training curve is presented below:

I train the model for 500 epochs and submit the result of 100, 200, 300, 500 for testing result. The score is shown as followed:

Epochs	CodaBench PSNR
100	27.22
200	28.89
300	30.89
500	31.41

Prompt Length

In the PromptIR framework, the parameter prompt_len defines the number of learnable prompt tokens used within the Prompt Generation Module (PGM). These prompts act as a set of dynamic guidance features that are adaptively combined based on the global context of the input image. Specifically, prompt_len controls how many distinct prompt templates the model can choose from when synthesizing the final prompt used to guide the

restoration process.

The original prompt_len = 5 and I additionally test prompt_len = 2, prompt_len = 10, try to find a better prompt_len fit for the degradation scenario. (epoch = 300)

prompt_len	CodaBench PSNR
2	30.30
5	30.89
10	30.87

PromptGenBlock

To enhance the model's ability to capture diverse contextual patterns from the generated prompts—particularly important since raindrops and snowflakes differ in scale and structure, I incorporate multi-scale convolutions (3×3, 5×5, 7×7) followed by a 1×1 Convolution for adaptive fusion, inspired by the design principles of multi-scale feature aggregation in image restoration networks [Zamir et al., 2021] and channel-wise attention fusion mechanisms [Hu et al., 2018].

Specifically, I apply three separate convolutional layers with different kernel sizes: 3×3, 5×5, and 7×7 (conv3x3, conv5x5, conv7x7). These layers extract local, mid-range, and broad receptive field information, respectively, from the interpolated prompt. In contrast, the original implementation uses only a single 3×3 convolution.

After extracting multi-scale features, I concatenate their outputs along the channel dimension and fuse them using a 1×1 convolution (attn). This serves as a lightweight attention mechanism that adaptively weighs and integrates the multi-scale information into a unified prompt representation.

Version	CodaBench PSNR
Original (500 ep)	31.41
Mine (500 ep)	31.71

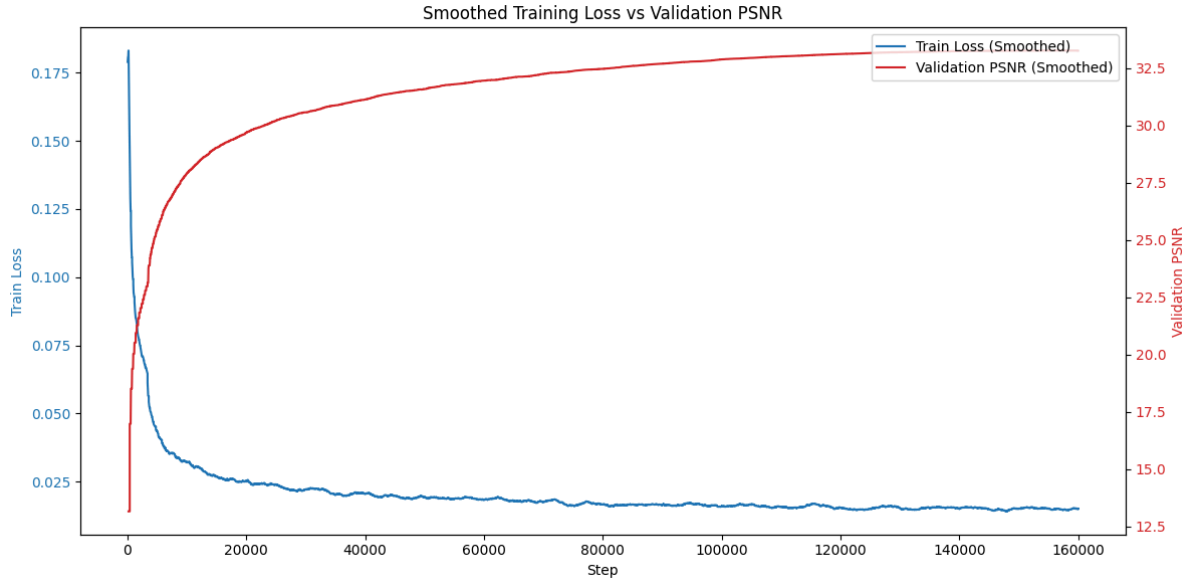
Restoration Result

- Original test images:

- PSNR 30 results:

Training Curve

Since the training process does not exhibit signs of overfitting, the training PSNR is used directly as a proxy for validation PSNR.



5. Conclusion

In this assignment, I explored the challenging task of blind image restoration under rain and snow degradations using the PromptIR framework. By analyzing and extending the original PromptIR design, I introduced two key architectural enhancements: adjusting the prompt length and redesigning the PromptGenBlock with multi-scale convolutional fusion. Experimental results demonstrate that increasing training epochs significantly improves performance, and that prompt length plays a subtle but important role in adapting to different degradation patterns. Notably, my multi-scale PromptGenBlock modification improving PSNR from 31.41 dB to 31.71 dB on the CodaBench leaderboard. These results highlight the effectiveness of leveraging degradation-aware prompts and multi-scale contextual cues t0 enhance restoration quality. Overall, the work confirms PromptIR's flexibility as a strong baseline for blind restoration and shows that targeted architectural improvements can only yield a little performance benefits.

6. References

[1] Potlapalli, V., Zamir, S. W., Khan, S. H., & Shahbaz Khan, F. (2023). Promptir: Prompting for all-in-one image restoration. Advances in Neural Information Processing Systems, 36, 71275-71293.

[2] Zamir, S. W., Arora, A., Khan, S., Hayat, M., Khan, F. S., Yang, M.-H., & Shao, L. (2021). Multi-stage progressive image restoration. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 14821–14831).

[3] Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 7132–7141).