NYCU Visual Recognition using Deep Learning
Homework 4

[312551185], [郭晏涵]

GitHub Link:

https://github.com/slovengel/312551185_HW4

Introduction:

This project aims to tackle an image restoration problem focused on removing rain and snow degradations. The dataset consists of RGB images, with 1600 degraded-clean image pairs per weather type for training and validation, and 50 degraded images per type for testing. The objective is to restore each degraded image to its corresponding clean version.

To enhance restoration performance, I employed several key strategies including custom data augmentation, adaptive learning rate scheduling, and combining loss functions. The core of my method is built upon the PromptIR framework, trained from scratch without relying on pretrained weights, and adapted specifically for rain and snow removal tasks.

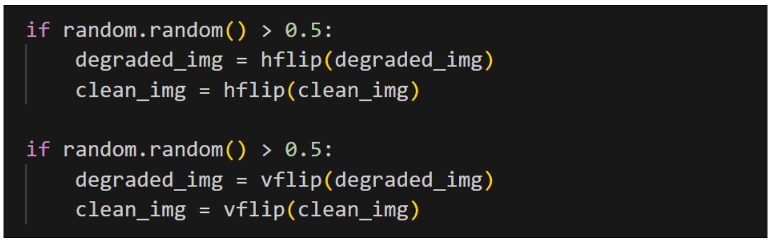
The proposed model achieved a PSNR of 29.74 in the CodaBench competition (on public leaderboard), demonstrating its effectiveness in handling the image restoration task.

Method:

1. Data pre-processing

The entire dataset of degraded and clean image pairs is converted to tensors. Unlike classification tasks that often resize inputs to a fixed resolution, all images in this pipeline are kept at their original resolution to preserve spatial fidelity.

Unlike single-image inputs, image restoration requires that both the degraded and the clean images undergo identical augmentations to maintain pixel-level alignment. Since torchvision transforms (RandomHorizontalFlip and RandomVerticalFlip) apply changes independently, I implemented manual augmentation to apply the same flipping transformation to both images, as shown below:



```python
if random.random() > 0.5:
    degraded_img = hflip(degraded_img)
    clean_img = hflip(clean_img)

if random.random() > 0.5:
    degraded_img = vflip(degraded_img)
    clean_img = vflip(clean_img)
```

More complex transformations (e.g., rotation or color jitter) were excluded from the final training pipeline to prevent introducing inconsistencies between the degraded and ground truth images, which could confuse the restoration model during training.

2. Model architecture

The model is built upon the PromptIR architecture, proposed in the paper "PromptIR: Prompting for All-in-One Blind Image Restoration" [1]. It adopts a UNet-style encoder–decoder design enhanced with prompt-driven transformer blocks for degradation-specific feature adaptation. It consists of three main components: an encoder with hierarchical feature extraction, prompt blocks, and a decoder that reconstructs the restored image.

- Encoder: The encoder follows a UNet-style structure with transformer blocks embedded at multiple levels. These layers extract multi-scale hierarchical features from the degraded input image, capturing both local textures and global contextual information essential for rain and snow removal.
- Prompt Block: The prompt block is the key innovation in PromptIR and consists of two submodules:
  - **Prompt Generation Module:** This module generates degradation-aware prompts based on the input features and predefined prompt components. The prompts are conditioned on the degradation characteristics present in the input image.
  - **Prompt Interaction Module:** This module fuses the generated prompts with the input features through transformer attention mechanisms, enabling effective adaptation of the features for the restoration task.
- Decoder: The decoder reconstructs the clean image using the enriched multi-scale features from the encoder and prompt interactions. It refines the outputs progressively, allowing the model to restore fine details lost due to rain and snow degradation.

3. Hyperparameters and Training Details

The models were trained using a **batch size of 1** for **25 epochs**. The **AdamW** optimizer was used with a **learning rate of 2e-4** and a weight decay of 1e-2 to prevent overfitting. Only parameters requiring gradients were updated during training.

CosineAnnealingLR learning rate scheduler was applied to dynamically adjust the learning rate over time. It creates a smooth and continuous decay of the learning rate without interruptions. The learning rate starts at **2e-4** in the first epoch and gradually decreases to **1e-6** by the final epoch.

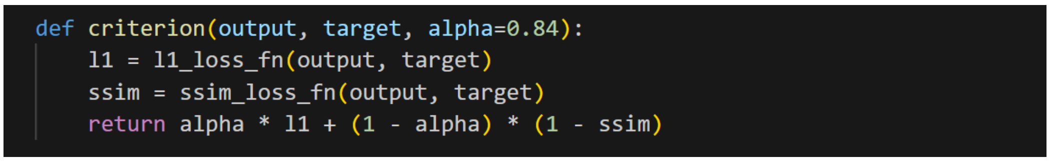
To support mixed-precision training and improve computational efficiency, a**utomatic mixed precision (AMP)** was enabled using torch.amp.GradScaler().

Additional experiments - Loss Function Comparison:

This experiment investigates the impact of different loss functions on the performance of the PromptIR model for image restoration. Specifically, I compare L1 Loss, MSE Loss, and a combined L1 + SSIM Loss to evaluate which formulation better optimizes restoration quality under rain and snow degradation.

The detailed implementation is as follows:

1. Train three PromptIR models independently using the same training pipeline, differing only in the loss function used during optimization.
2. The first model uses standard L1 Loss (nn.L1Loss()), which measures the mean absolute error between the restored and clean images.
3. The second model uses a combined L1 + SSIM Loss, inspired by the formulation in the paper "Loss Functions for Image Restoration With Neural Networks" [2]. This loss balances pixel-wise accuracy (L1) with perceptual similarity (SSIM), and is implemented as:

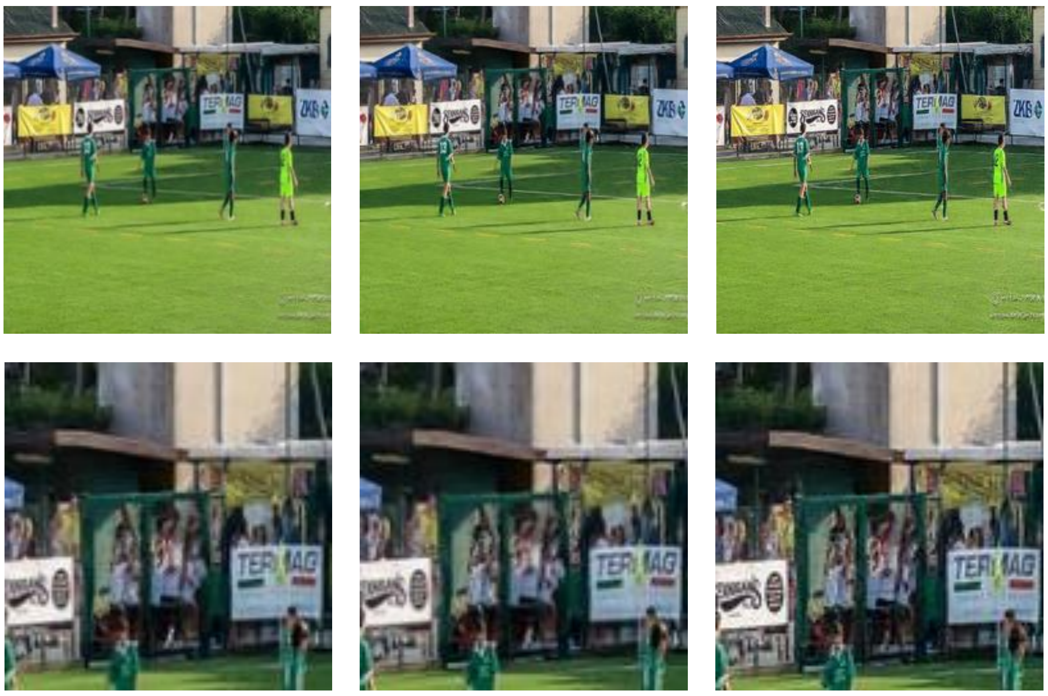


```
def criterion(output, target, alpha=0.84):
    l1 = l1_loss_fn(output, target)
    ssim = ssim_loss_fn(output, target)
    return alpha * l1 + (1 - alpha) * (1 - ssim)
```

   where alpha determines the trade-off between pixel-wise fidelity and structural similarity.

4. The third model is trained using MSE Loss (nn.MSELoss()), which emphasizes squared pixel-wise differences, penalizing larger errors more heavily.

Visualizations of model output using different loss functions:



(a) L1 Loss (b) L1 + SSIM Loss (c) MSE Loss

The following are the experiment results:

Model Name	**Best PSNR**
L1 Loss	28.88
L1 + SSIM Loss	29.34
MSE Loss	29.74

Analysis:

Among the three evaluated losses, MSE Loss achieved the highest PSNR, indicating strong pixel-level restoration capability. Its squared-error formulation helps suppress noise more aggressively, particularly beneficial in challenging degradations like rain streaks and snow particles.

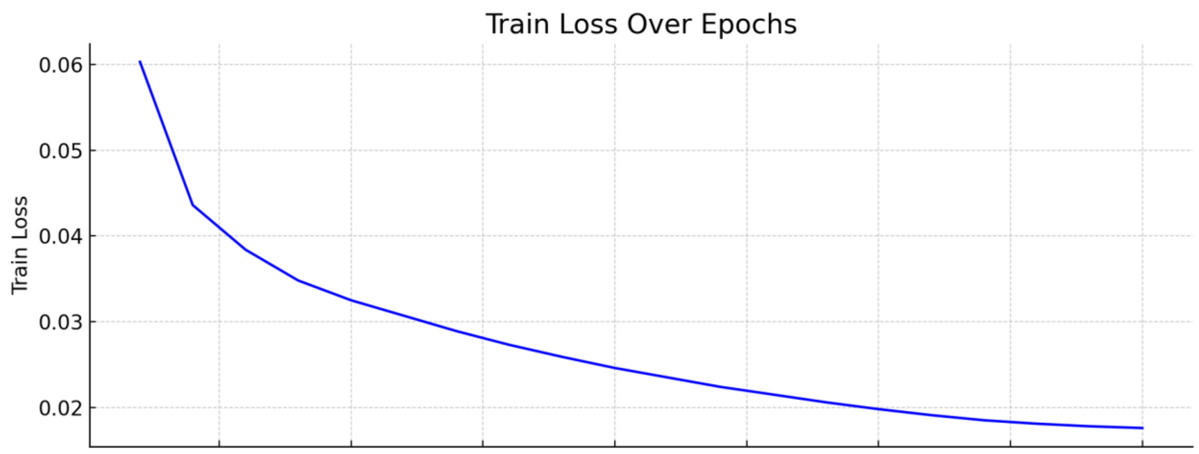
The L1 + SSIM Loss, influenced by perceptual image quality metrics, produced competitive results. By jointly optimizing for structural similarity and pixel accuracy, it preserves textures and edges better than L1 alone. However, tuning the weight parameter alpha is important for balancing both objectives.

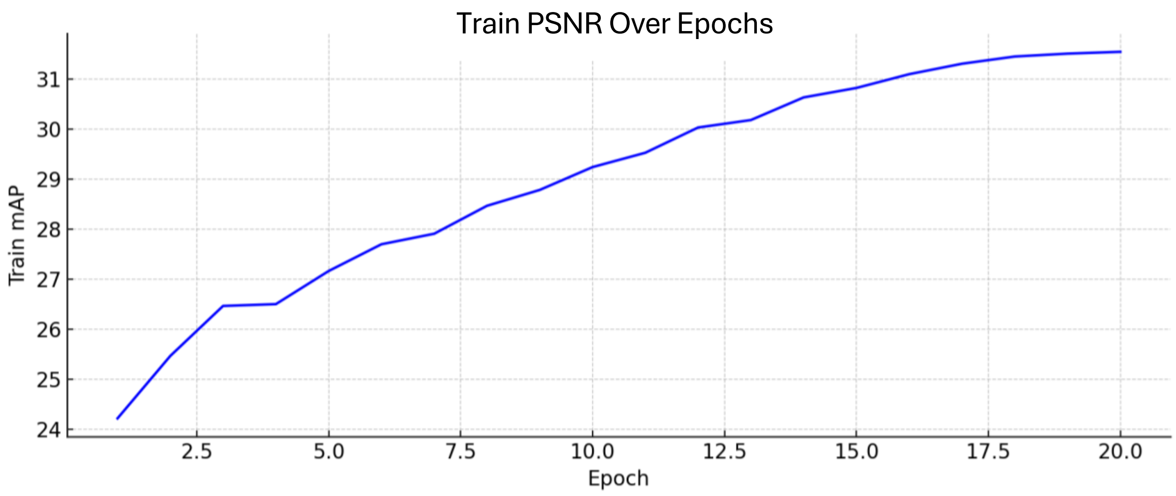
Pure L1 Loss, though widely used for its robustness to outliers and stable gradients, underperformed slightly in terms of PSNR. This result highlights the benefit of combining perceptual cues like SSIM in restoration objectives.

These results suggest that MSE Loss is most effective for maximizing numerical restoration quality under PSNR. However, when structural perceptual quality is important, the L1 + SSIM approach provides a valuable trade-off.

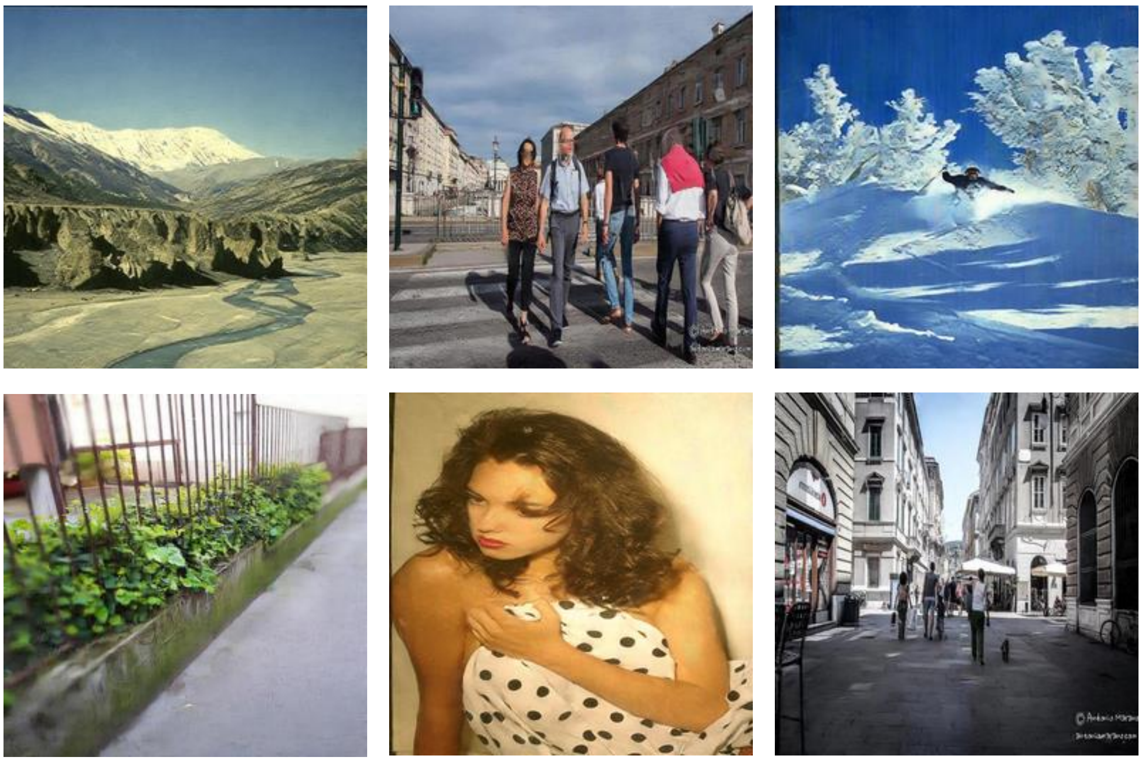
Results:

Based on the experimental results, neither L1 loss nor L1 + SSIM loss provided significant improvements. Since the combination of the PromptIR model with MSE loss achieved the best performance, it was selected as the final model for inference. The following are the learning curves for training the proposed model.





Visualization of predicted clean images:



Code Reliability:

I lint the code using Flake8 to follow the PEP8 instructions.

References:

[1] PromptIR: Prompting for All-in-One Blind Image Restoration
https://arxiv.org/abs/2306.13090
https://github.com/va1shn9v/PromptIR

[2] Loss Functions for Image Restoration With Neural Networks
https://arxiv.org/abs/1511.08861