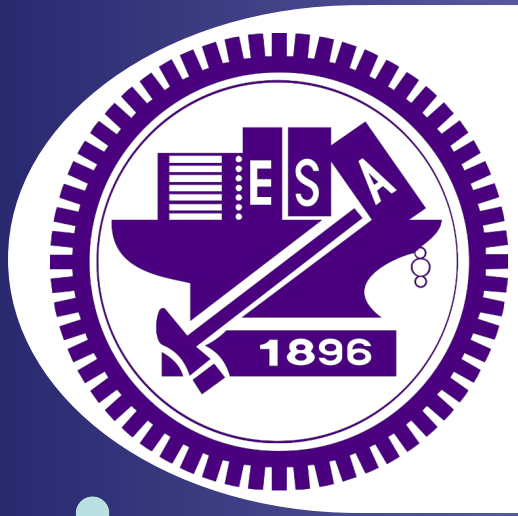


Visual Recognition using Deep Learning
2025 Spring, Homework 4

Release Date: 2025/05/07 12:00

Homework 4

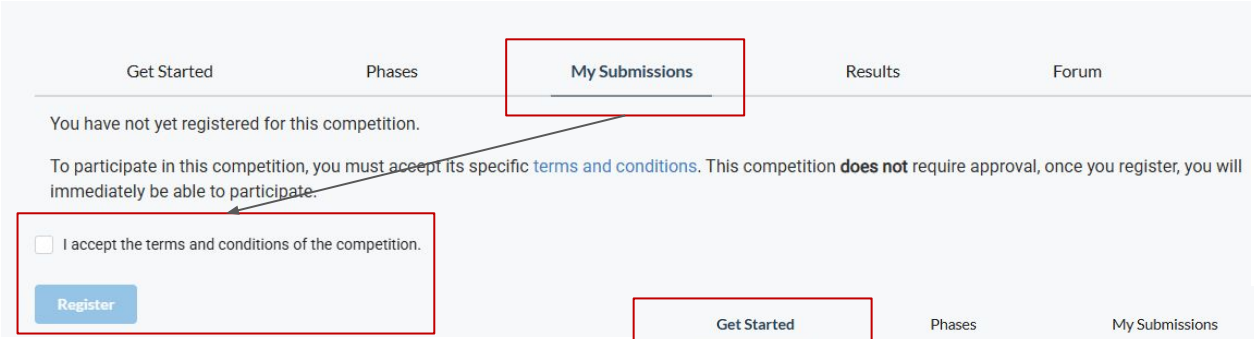
- Deadline: 23:59, 05/28 (Wed), 2025
- **Participate the competition** (80%): Image Restoration
  - Participant the competition on the CodaBench and get the highest score as possible. (70%)
  - Code reliability: GitHub (10%)
- **Report and code** (20%): Document your method and findings.
  - Report
    - In PDF format and written in English. (5pt penalty)
    - Introduction to your method (e.g., data pre-processing, model architecture, hyper-parameters)
    - Conduct additional experiments to further improve the model and analyze their results.
  - Code
    - Zip your code (.py) alone with report - Submit to E3.
    - You should also put your code on your GitHub repository and provide the link in the report.

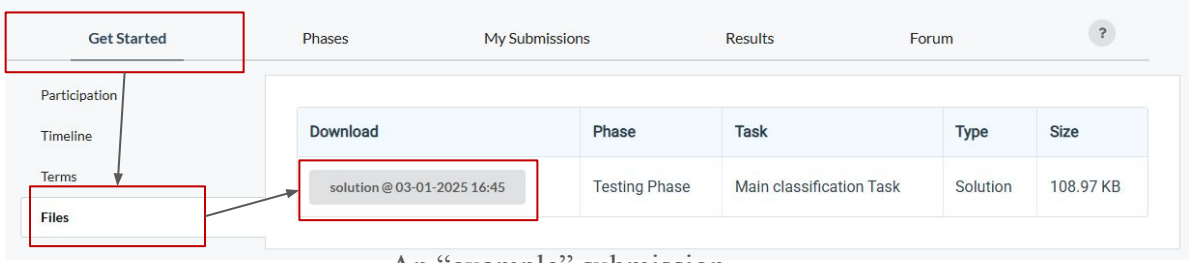
Links

- Link to the dataset and sample code
- Link to the competition

How to participate the competition and do submission

1. Register an account on CodaBench
   a. When registering the account, please use your studentID as the UserName
2. After you click the competition link, go to My Submissions, and join the competition



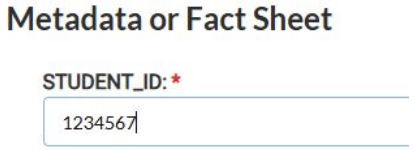


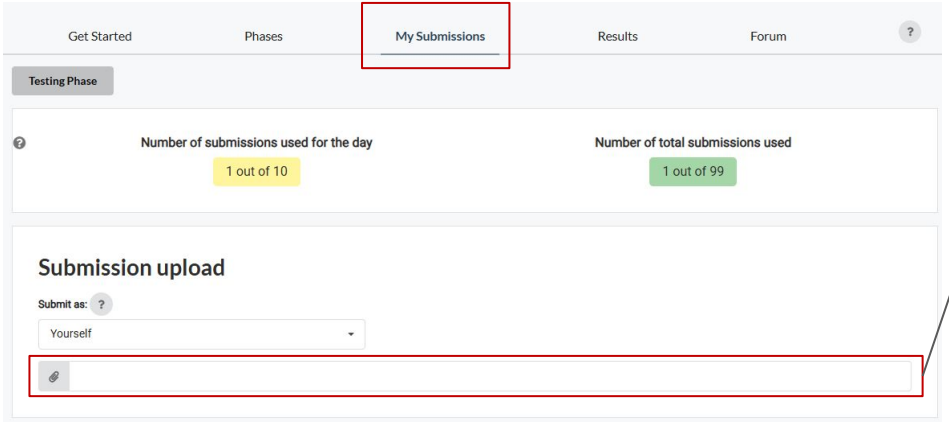
An “example” submission

How to participate the competition and do submission

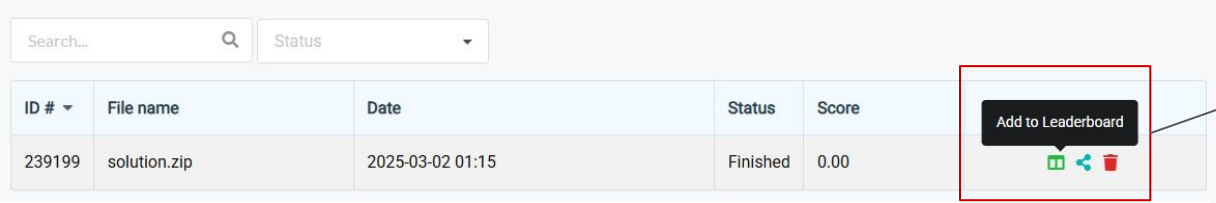
3. Submit your results and don't forget to "Add to Leaderboard"

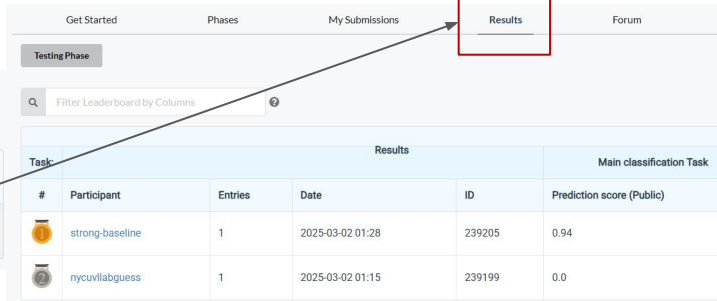
4. Don't forget to check your results can be found on the leaderboard





No limitation for the name of the "zip" file; however, inside the zip file, your **result files must be named "pred.npz"**





Coding Environment

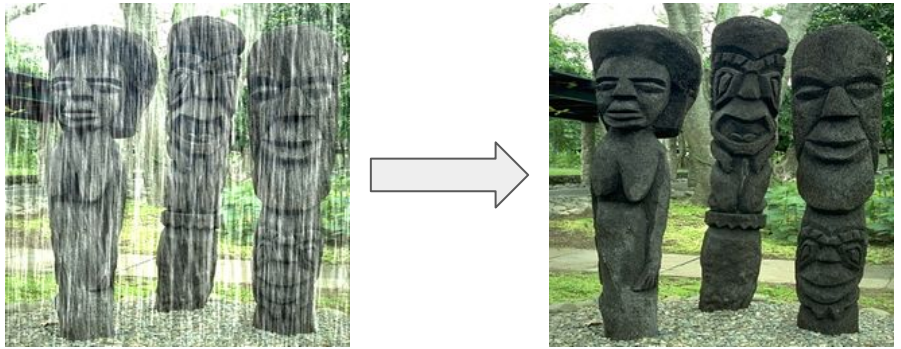
- Recommnedation: Python 3.9 or higher
- Tips
  - We recommend you to use **virtual environments** when implementing your homework assignments.
  - Here are some popular virtual environment management tools
    - Poetry
    - Conda
    - Virtualenv

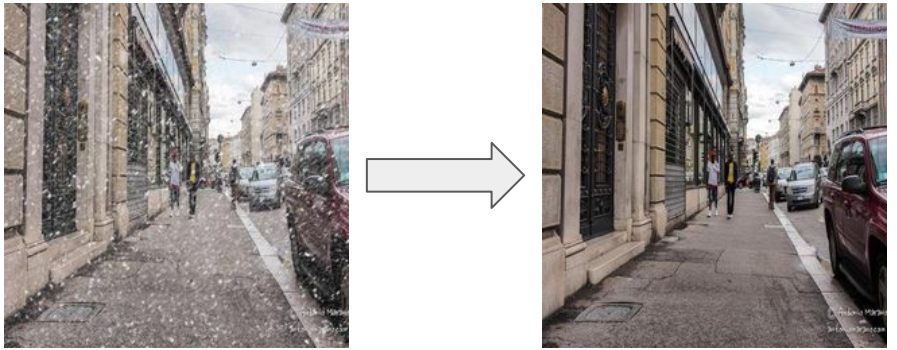
Numpy & PyTorch

- Numpy Tutorial: Link
- PyTorch Tutorial: Link
    - Free to use any modules and functions

Task and Dataset

- Task type: Image Restoration
- Dataset
  - Degraded images (2 types: Rain and Snow)
  - Training / Validation:
    - 1600 degraded images per type
    - 1600 corresponding clean images per type
  - Test:
    - 50 degraded images per type
- Target
  - Clean images corresponding to each degraded image
- Evaluation
  - PSNR (Peak Signal-to-Noise Ratio)





Task Requirement and Limitations

- **Requirement**: Train a single model capable of restoring both types of degraded images (Rain and Snow).
- **Limitations**
  - No external data (i.e., data from other sources) is allowed.
  - Only pure vision-based models are permitted (vision-language models are not allowed).
  - You may modify components/modules of PromptIR to improve performance. In your report, you must:
    - Elaborate on the key design or contribution of PromptIR.
    - Explain the modifications you made to improve the model performance.
    - Cite the paper properly.
- **Note**: Pretrained weights are not allowed — the model must be trained from scratch.

Dataset Inspection

train/
- degraded
  - rain-1.png
  - …
  - rain-1600.png
  - snow-1.png
  - …
  - snow-1600.png
- clean
  - rain_clean-1.png
  - …
  - rain_clean-1600.png
  - snow_clean-1.png
  - …
  - snow_clean-1600.png

test/
- degraded
  - 0.png
  - …
  - 99.png

The test images are named generically (e.g., 0.png to 99.png) and do not reveal the type of degradation.

Output Format Example for Image Restoration

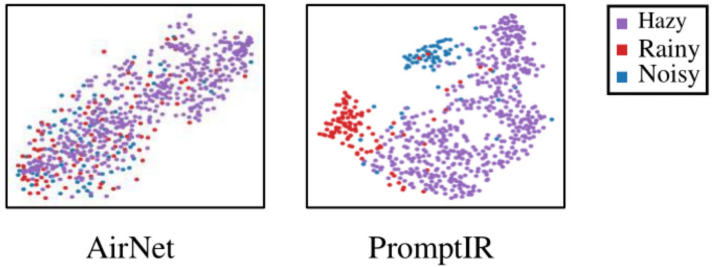
You are required to submit a file named pred.npz. This file should contain a set of images from your dataset, stored in a dictionary-like format using NumPy.

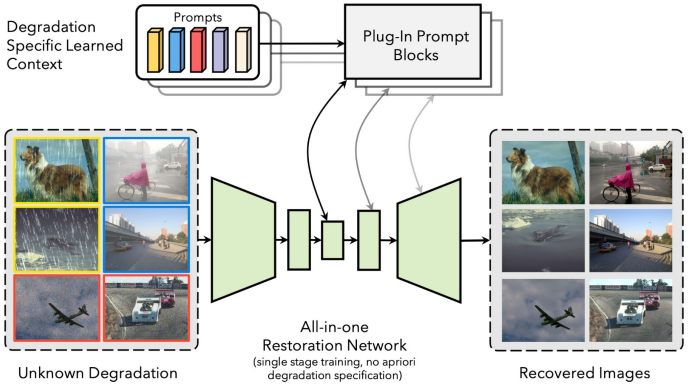
- Keys are the filenames of the original images (e.g., '0.png', '1.png', …)
- Values are NumPy arrays of shape (3, H, W) representing the RGB image data
    - Shape explanation:
        - 3 channels (Red, Green, Blue)
        - Height and Width match the original image size
        - Values can be uint8 (0–255)
- Submission may take some time—please plan accordingly and submit on time.

```python
import os
import numpy as np
from PIL import Image

# Set your image folder path
folder_path = '/path/to/your/folder'
output_npz = 'pred.npz'

# Initialize dictionary to hold image arrays
images_dict = {}

# Loop through all files in the folder
for filename in os.listdir(folder_path):
    if filename.lower().endswith(('.png', '.jpg', '.jpeg')):
        file_path = os.path.join(folder_path, filename)

        # Load image and convert to RGB
        image = Image.open(file_path).convert('RGB')
        img_array = np.array(image)

        # Rearrange to (3, H, W)
        img_array = np.transpose(img_array, (2, 0, 1))

        # Add to dictionary
        images_dict[filename] = img_array

# Save to .npz file
np.savez(output_npz, **images_dict)

print(f"Saved {len(images_dict)} images to {output_npz}")
```

PromptIR: Prompting for All-in-One Blind Image Restoration

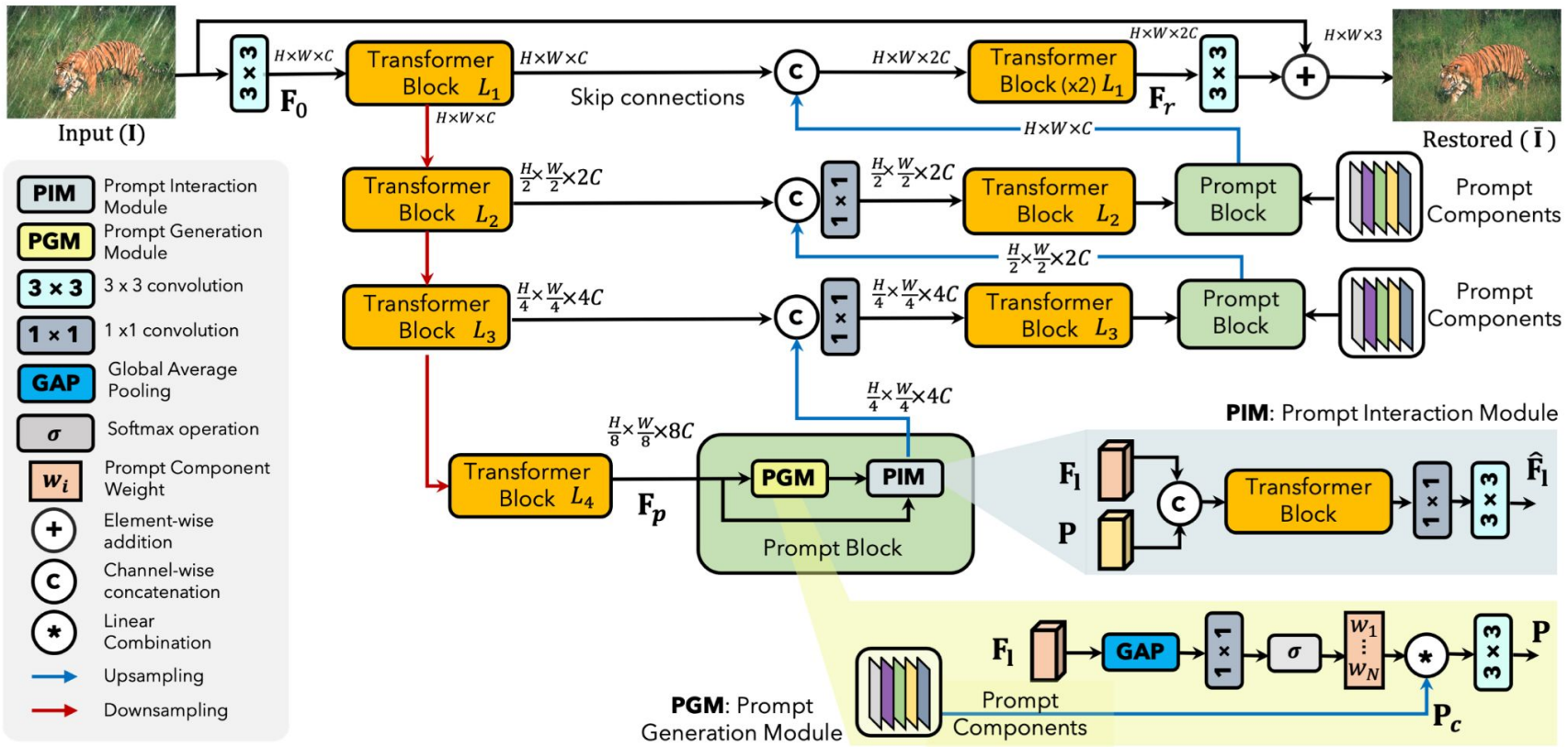
- Deep learning models are often designed for specific tasks (e.g., denoising, deblurring).
- Real-world images may suffer from multiple and unknown degradations.
- AirNet uses contrastive learning to distinguish degradation types.
  - Relies on a two-stage training process and an additional encoder, increasing computational cost.
  - Struggles to produce fully disentangled representations of corruptions.
- PromptIR utilizes prompts, which are a set of tunable parameters that encode crucial discriminative information about various types of image degradation.





[Paper link]

PromptIR: Prompting for All-in-One Blind Image Restoration



[Paper link]

Grading Policy - Report (20%)

- Format: PDF, written in English. (-5pts if not followed)
- Make sure to place your GitHub link at the beginning of the report.
- Sections that you should include

15pts

  - **Introduction** to the task and core idea of your method
  - **Method:** Descbible how you pre-process the data; what is your model architecture, and hyperparameters, etc.
    - You need to elaborate on your modificaion.
  - **Results**: Describe your findings and list/plot your model performance (e.g., training curve, confusion matrix, etc.)
    - Visualize predicted clean images.
  - **References**: Your method references (Paper / Github sources, must include if you use any.)

We encourage you to stand on the shoulders of giants - only clone repo and run it is not enough.

5pts

- Among various architectures, why do you choose this one as your module? What are the pros and cons?
- Visualizations of model output under different parameter settings.
- **Additional experiments** to explore better performance
  - Simply tuning the hyper-parameters doesn't count (e.g., batch-size, LR, different optimizers)
  - Hint: Try to add/remove some layers, use different design, use different loss functions, etc.
- You should 1) include your hypothesis (why you do this), 2) How this may (or may not) work, and 3) The experiment results and their implications.

Grading Policy - Code Reliability (10%)

Python Coding Style Guide Reference

1. PEP8
2. Google Python Style

1. Please follow the PEP8 instructions and lint your code.
2. Push your code to the GitHub
   - It should contains a README.md to introduce this work (And your StudentID)
   - Runable codes

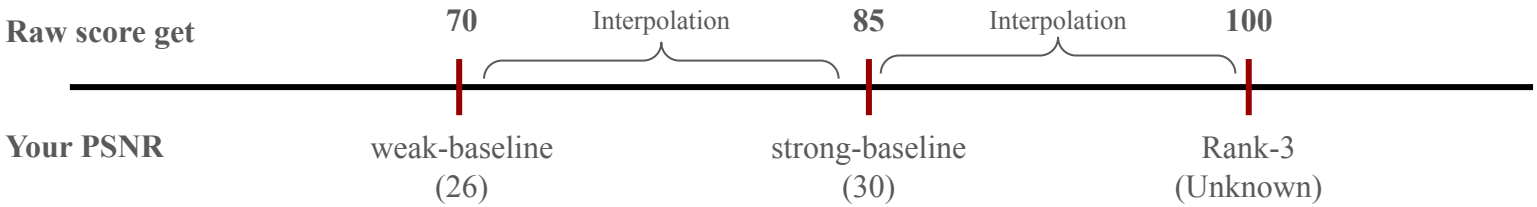
An example: README.md

Grading Policy (70%)

We will use **private (hidden) leaderboard** to evaluate the performance (the distribution is similar for data in public and private set.) The public leaderboard is for you as reference.

Your score (competinion):

- Less than weak-baseline (PSNR < w.baseline): S = 0
- Between weak-baseline and strong baseline (PSNR >= w.baseline & PSNR < s.baseline): (70 + (PSNR - w.baseline) / (s.baseline - w.baseline) * (85 - 70)) * 0.70
- Between strong-baseline and Rank3: (85 + (PSNR - s.baseline) / (PSNR.rank3 - s.baseline) * (100 - 85)) * 0.70
- Rank1,2,3 = 100 * 0.70



Submission

- Compress your **code** and **report** into a **.zip file** and submit it to E3.
  - **Don’t forget to push your code to GitHub. And your GitHub link should be written in the report.**
- Report should be written in English.
- <STUDENT ID>_HW4.zip
  - codes (.py, folders, etc)
  - <STUDENT ID>_HW4.**pdf** (NO .doc, .docx or others format)

- Don’t put the data (e.g. x.jpg / train.csv / test.csv) and model checkpoints into submission file (-5 if not followed)

Other rules

- **Late Policy**: A penalty of **20 points** per additional late day. (-20pt / delayed.day)
  - For example, If you get 90 points but delay for two days, your will get only 50 points!

- **No Plagiarism**: You should complete the assignment by yourself. Students engaged in plagiarism will be penalized heavily. Super serious penalty.
  - e.g. -100pt for the assignment or failed this course, etc
  - Report to academic integrity office

FAQs

- Can I use any library/package/framework from GitHub or other resources?
  - **Yes**, we encourage you to learn how to leverage existing knowledge on your own task
    - e.g., Github of published works and model zoo from Torchvision
    - Focus on how to step forward from them - That's why part of scores comes from your competition ranks
  - You **should not copy-and-paste from your classmates** (Plagiarism)
- How to handle the GPU Out-of-Memory (OOM) issue?
  - Easy answer - Make your batch size smaller or make your model smaller.
  - Advanced methods: Try to figure it out by yourself. (Many online resources and AI-assistance)

FAQs

- If I don’t have my own GPU – Use Google Colab
  - It should be 12 hours, please check this discussion in the stackoverflow
  - And some tricks here may make it longer.

- If you have other questions, ask on **E3 forum** first! We will reply as soon as possible.

It’s your turn! Have Fun!