AI6126: Advanced Computer Vision Project 2

Blind Face Super-Resolution

Adnan Azmat (G2303265K)

Challenge Description

The goal of this mini-challenge was to generate high-quality (HQ) face images from the corrupted low-quality (LQ) ones (see Figure 1). The data for this task came from the Flickr-Faces-HQ (FFHQ). For this challenge, a mini dataset was provided, which consists of 5000 HQ images for training and 400 LQ-HQ image pairs for validation. Note that the LQ images in the training set was not provided. During the training, the corresponding LQ images was needed to be generated on the fly by corrupting HQ images using the random second-order degradation pipeline (see Figure 2). This pipeline contains 4 types of degradations: Gaussian blur, Downsampling, Noise, and Compression. The code of each degradation function as well as an example of the degradation config for reference was provided.

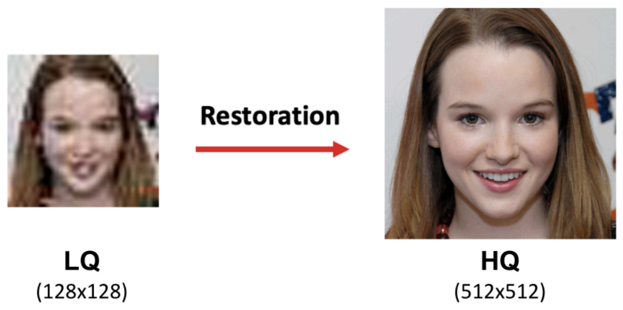


Figure 1: Illustration of blind face restoration

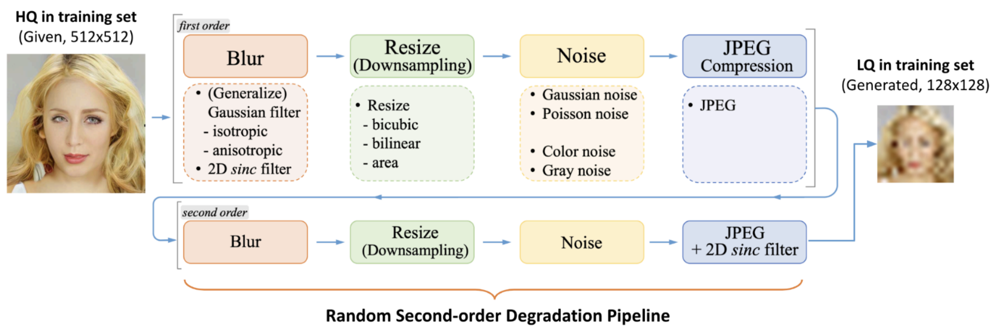


Figure 2: Illustration of second-order degradation pipeline during training

Experiment Details

The experiment utilized the simple network **SRResNet** that was provided in the sample Real-ESRGAN configuration file in the project handout. To increase the complexity of the model I increased the num_block (number of residual blocks) from 16 to 24. This change keeps the model parameters within the allowed limits for the challenge.

During training, the Adam optimizer was used with a learning rate of 2e-4 and a total of

300,000 iterations were performed. The L1 loss served as the pixel-wise loss function to guide the training process. A CosineAnnealingRestartLR scheduler was employed to adjust the learning rate with two restart periods of 150,000 iterations each.

Model Used

The **SRResNet** architecture, inspired by ResNet, is designed for super-resolution tasks. It comprises key elements: an Input Layer for low-res images, Convolutional Layers for feature extraction with batch normalization and ParametricReLU activation, Residual Blocks as the model's core with two convolutions and skip connections to address gradient issues, Upsampling Blocks to enhance feature map resolution using convolution and PixelShuffle, and an Output Layer for generating the high-res image. This structured approach enables effective super-resolution image enhancement.

Loss Function

L1 loss (often referred to as L1_pix loss) serves as a metric to assess the discrepancy between a generated high-resolution image and the corresponding ground truth (original high-resolution) image.

Mathematical Formulation: Let I_{gt} denote the ground truth image (a tensor representing pixel values) and I_{gen} denote the generated image (a tensor representing pixel values). Let N represent the total number of pixels in the image. The L1 loss can be expressed as:

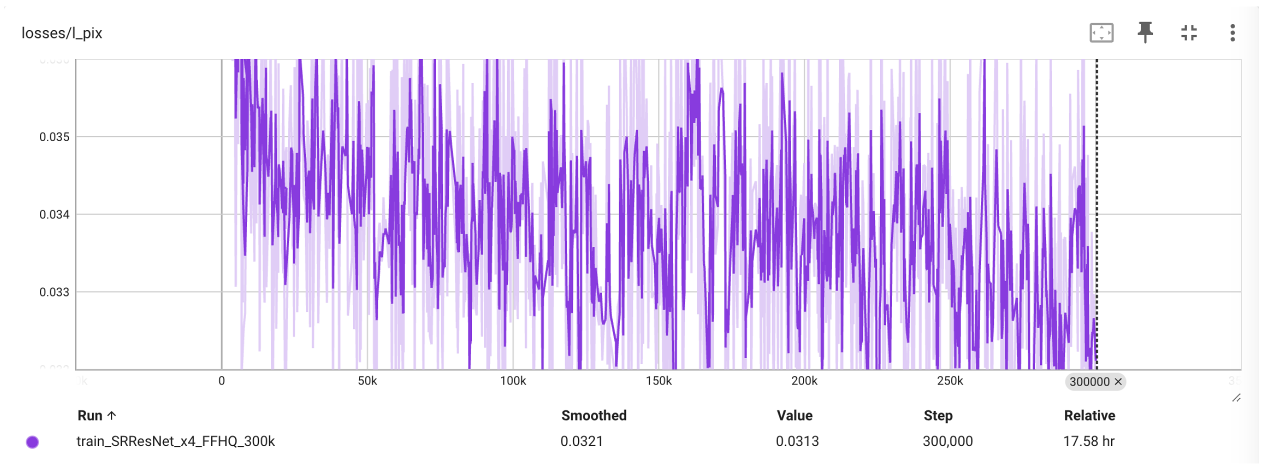
$$L1_{loss} = \frac{1}{N} \sum_{i=1}^{N} |I_{gt}(i) - I_{gen}(i)| \tag{1}$$

Due to its simplicity and focus on large errors (less affected by outliers), L1 loss remains a popular choice for image super-resolution tasks, but it can lead to overly smooth, potentially blurry reconstructions.

Training Plots

Training Loss

Training curve (loss) over 300 iterations.

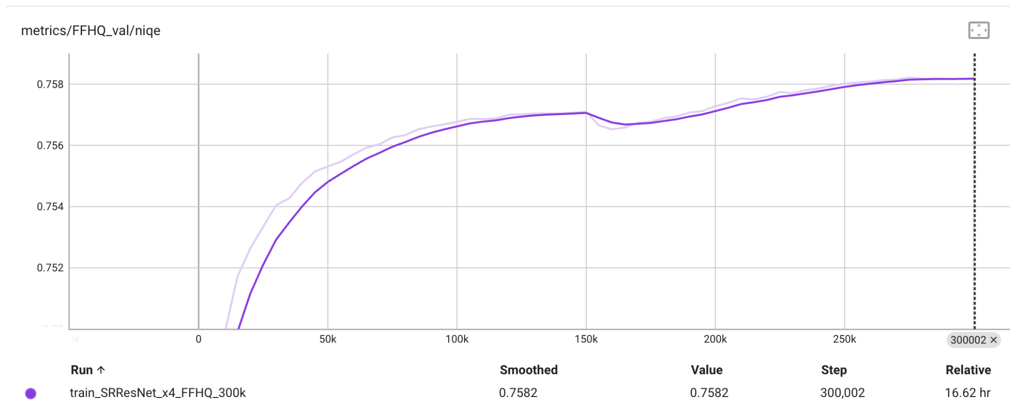


Run ↑	Smoothed	Value	Step	Relative
● train_SRResNet_x4_FFHQ_300k	0.0321	0.0313	300,000	17.58 hr

Figure 3: Training Loss

Metrics on validation set

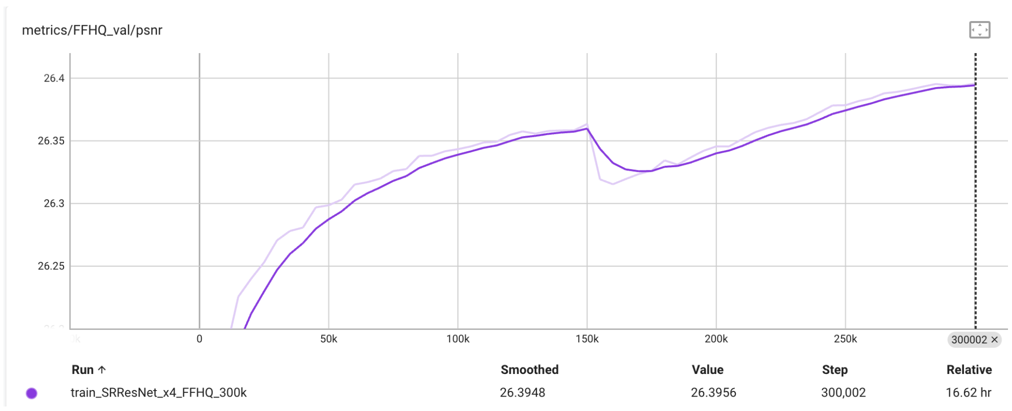
- Natural image quality evaluator (NIQE)



Run ↑	Smoothed	Value	Step	Relative
● train_SRResNet_x4_FFHQ_300k	0.7582	0.7582	300,002	16.62 hr

Figure 4: NIQE

- Peak signal-to-noise ratio (PSNR)



Run ↑	Smoothed	Value	Step	Relative
● train_SRResNet_x4_FFHQ_300k	26.3948	26.3956	300,002	16.62 hr

Figure 5: PSNR

- PSNR values using evaluate.py at saved model checkpoints

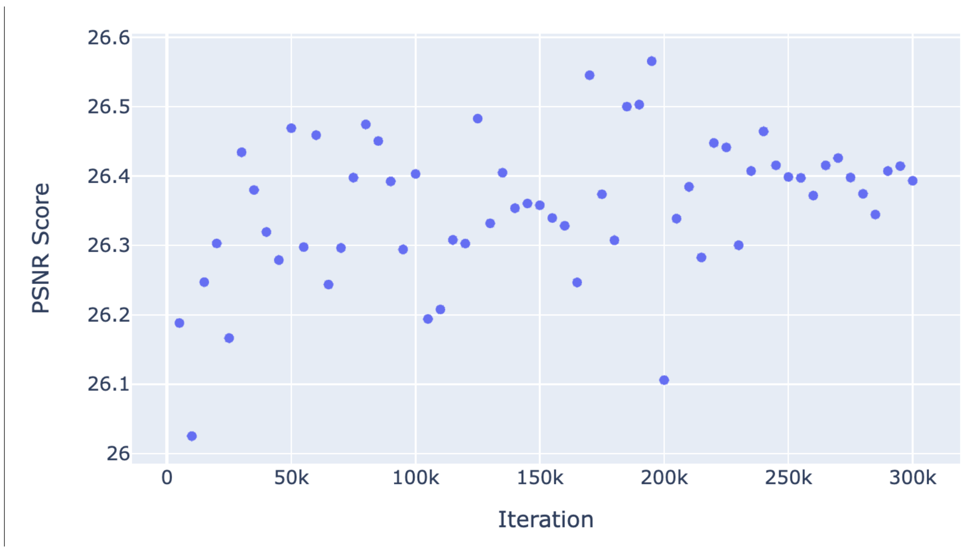


Figure 6: PSNR values of different checkpoints on validation set using evaluate.py

The 195,000 th iteration scored the highest on the validation set with PSNR of **26.5656**. This model checkpoint was used to evaluate on the test set. This model achieved PSNR of **26.65289** on codalabs scoring, securing 7th place on the leaderboard.

Number of parameters

Total parameters of the model: **2,108,419**

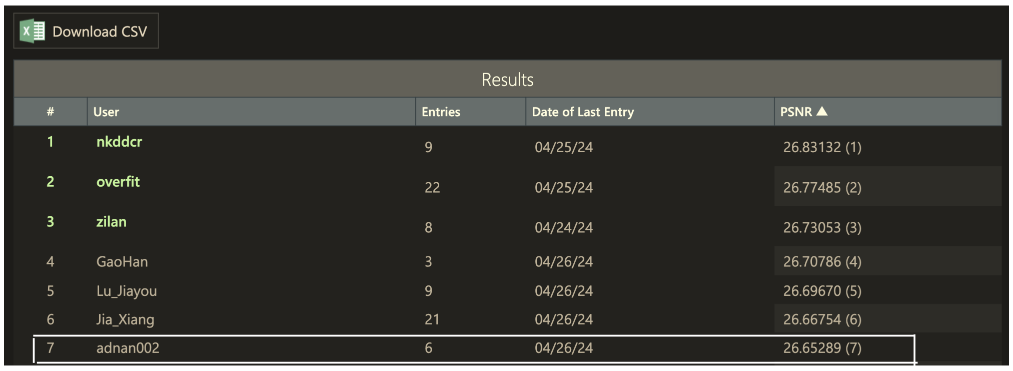
Specs of training machine

I used four **NVIDIA A100-SXM4-40GB GPU** on NSCC ASPIRE2A System. To speed up the trainig process I trained the model on 4 GPUs. Below our the statistics of the run on 300K iterations for the above experiment.

```
Execution Nodes Used: (x1000c0s1b0n1:ncpus=64:mem=461373440kb:ngpus=4)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
GPU Duration: 12.69hrs
GPU Energy Consumed: 288.96W
GPU Max GPU Memory Used: 61.02GB
Memory Throughput Rate (Avg): x1000c0s1b0n1:(gpu1:39%+gpu3:39%+gpu0:39%+gpu2:39%)
Memory Throughput Rate (Max): x1000c0s1b0n1:(gpu1:58%+gpu3:58%+gpu0:58%+gpu2:58%)
Memory Throughput Rate (Min): x1000c0s1b0n1:(gpu1:0%+gpu3:0%+gpu0:0%+gpu2:0%)
GPU SM Utilization (Avg): x1000c0s1b0n1:(gpu1:87%+gpu3:87%+gpu0:78%+gpu2:87%)
GPU SM Utilization (Max): x1000c0s1b0n1:(gpu1:100%+gpu3:100%+gpu0:100%+gpu2:100%)
GPU SM Utilization (Min): x1000c0s1b0n1:(gpu1:0%+gpu3:0%+gpu0:0%+gpu2:0%)
```

Codalab Screenshot

My position on leaderboard and best submission can be seen in the figure below:



Results

#	User	Entries	Date of Last Entry	PSNR ▲
1	nkddcr	9	04/25/24	26.83132 (1)
2	overfit	22	04/25/24	26.77485 (2)
3	zilan	8	04/24/24	26.73053 (3)
4	GaoHan	3	04/26/24	26.70786 (4)
5	Lu_Jiayou	9	04/26/24	26.69670 (5)
6	Jia_Xiang	21	04/26/24	26.66754 (6)
7	adnan002	6	04/26/24	26.65289 (7)

Figure 7: Codalab submission