Low-Light Image Enhancement with Wavelet-based Diffusion Models

Anushree Chaudhary

1 Implementation Details

The low-light image enhancement model is implemented using PyTorch and trained on the LOLv1 dataset. The model was trained for 500 epochs with a batch size of 8. The Adam optimizer was employed with an initial learning rate of 1×10^{-4}, decaying by 0.8 every 50 steps. Validation was performed every 1000 iterations, and checkpoints were saved every 50 epochs. Epoch times and losses were logged every 10 steps for analysis. The model architecture is based on U-Net. A linear beta schedule is applied during the diffusion process, with 200 timesteps.

1.1 Hardware Setup

- **CPU**: 13th Gen Intel(R) Core(TM) i7-1335U - Training on 3 datapoints with 5-10 epochs took 5-8 minutes due to slow processing.
- **GPU**: NVIDIA GeForce RTX 3090 - Training on the GPU significantly sped up the process compared to the CPU, allowing efficient training over 500 epochs, with regular checkpoints and logging of epoch times and losses. The average training time for an epoch was 28.99 seconds.

1.2 Issues in Requirement Installations and Modifications

While setting up the environment, I encountered issues with installation of packages such as `nori2` and `refile`, which were listed in the requirements.txt file of the official GitHub repository but could not be installed using `pip`. After investigating these packages and their purpose in the code, it was determined that they were not necessary for running the main model, and were removed from the requirements. Additionally, new packages were added to support other operations as needed during the course of the project. To combat version compatibility issues, the versions of some requirements were changed as well.

2 Dataset Description

The model was trained and evaluated on the LOLv1 dataset, consisting of 498 real-world low-light/normal-light image pairs.

3 Results

The noise, photo and frequency losses in the training steps are plotted in the Figure 1. It can be seen that the losses are deceasing as the number of iterations increase.

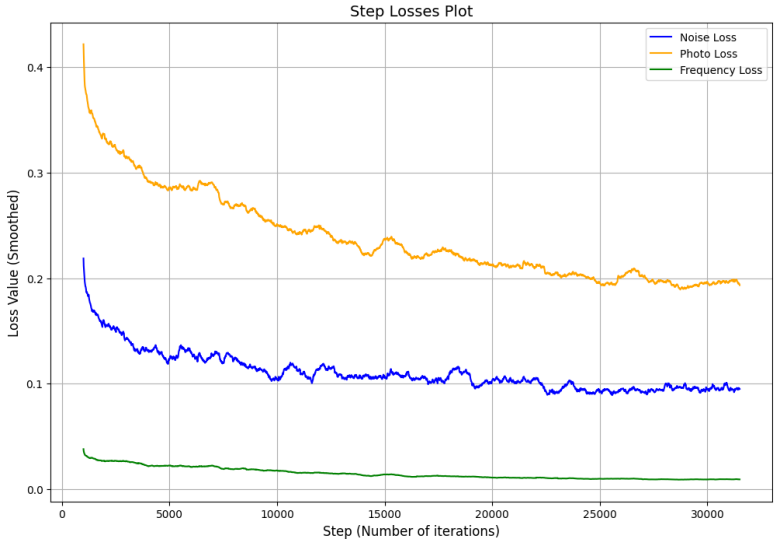


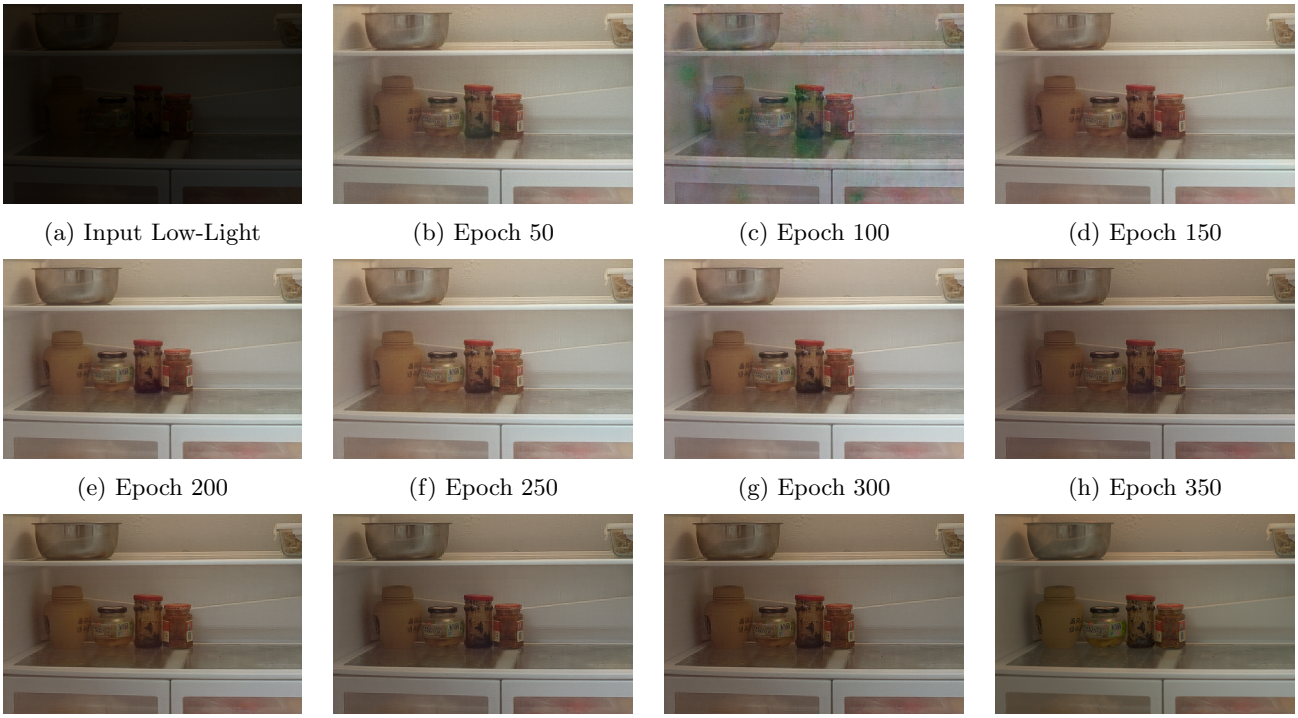
Figure 1: Loss reduction over training steps (smoothed over a rolling window of size = 100)

The PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index) metrics were used to compare the outputs of the models at regular checkpoint after every 50 epochs (training data as input) with the corresponding ground truth. On an average, these metrics increased with the increase in number of training epochs as can be seen in Table 1.

Epoch	**PSNR (dB)**		**SSIM**	
	Mean	**Median**	**Mean**	**Median**
50	28.35	28.01	0.82	0.84
100	28.08	27.92	0.79	0.82
150	28.71	28.35	0.85	0.87
200	29.00	28.55	0.86	0.88
250	29.17	28.72	0.86	0.88
300	29.35	29.14	0.87	0.88
350	29.33	28.88	0.87	0.88
400	29.53	29.16	0.87	0.89
450	29.58	29.28	0.87	0.89
500	29.87	29.47	0.87	0.89

Table 1: Mean and Median PSNR and SSIM values for model outputs at regular checkpoints

A sample result is shown in Figure 2. A low light image is given as input to the model and the model outputs at regular checkpoints after every 50 epochs have been shown. The last image is of the corresponding ground truth.



(a) Input Low-Light (b) Epoch 50 (c) Epoch 100 (d) Epoch 150

(e) Epoch 200 (f) Epoch 250 (g) Epoch 300 (h) Epoch 350

(i) Epoch 400 (j) Epoch 450 (k) Epoch 500 (l) Ground Truth

Figure 2: Comparison of model outputs at the regular checkpoints after every 50 epochs