VLG SUMMER 2024 PROJECT

Report on Low-Light Image Denoising

Anirudha Thakur - 23112016

INTRODUCTION:

This report presents the development and evaluation of the PixelWarriorNetwork, an advanced deep learning model designed for low-light image enhancement. Low-light conditions pose significant challenges for image quality, often resulting in poor visibility, noise, and loss of detail. Enhancing these images is crucial for applications in photography, security, autonomous driving, and more. Our model combines the strengths of the UNet architecture and a series of Multi-Layer Perceptron (MLP) blocks to improve the visibility and quality of low-light images.

DATASET:

The dataset used for training consists of low-light images paired with their enhanced versions. This dataset is organised as follows:

- **Training set:** Contains low-light images and their corresponding enhanced versions.

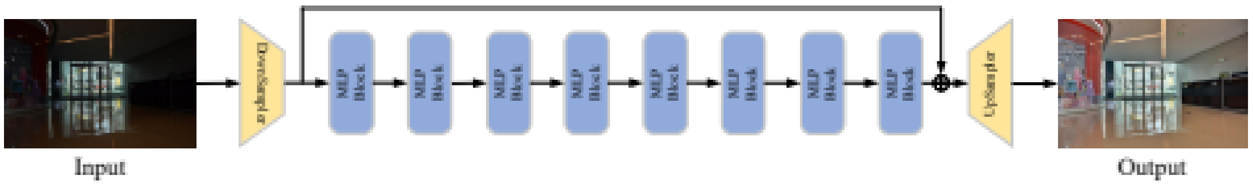
The images are loaded and transformed into tensors using the PyTorch library. The directories for the datasets are:

- Training low-light images: `/kaggle/input/dataset00/augmented_Train/augmented/low`
- Training high-light images: `/kaggle/input/dataset00/augmented_Train/augmented/high`

The dataset loader ensures that each low-light image is paired with its corresponding enhanced image, applying the necessary transformations to prepare the data for training.

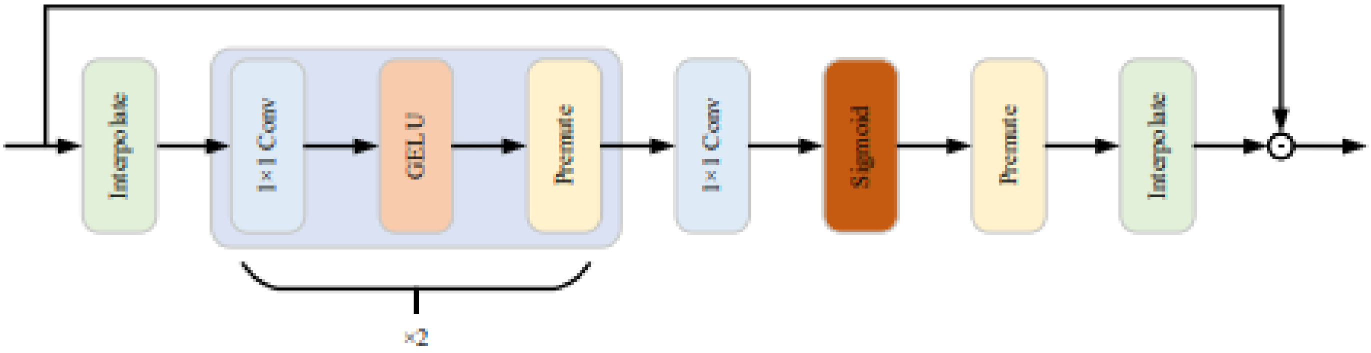
PAPER IMPLEMENTED: https://arxiv.org/pdf/2404.14248

MODEL ARCHITECTURE:



The PixelWarriorNetwork is composed of several key components:

1. **UNet Architecture:**
   - The UNet is the backbone of the network, providing a symmetric encoder-decoder structure with skip connections to capture both spatial and contextual information.
   - The encoder (downsampling path) extracts features at multiple levels, while the decoder (upsampling path) reconstructs the image using these features, integrating fine details from the skip connections.
2. **MLP Blocks:**
   - These blocks are inserted between the downsampler and upsampler to enhance feature representation.
   - Each MLP block consists of a convolutional layer followed by a GELU activation, another convolutional layer, and a Sigmoid activation. A residual connection is used to add the output of the MLP block to its input, facilitating the learning of refined features.



3. **Downsampler and Upsampler:**
   - The downsampler reduces the spatial dimensions of the input image, facilitating the extraction of high-level features.
   - The upsampler restores the spatial dimensions, applying the extracted features to the original resolution to enhance the image quality.

TRAINING:

The model was trained using Kaggle's GPU P100's environment with over 100 epochs. The model is trained using the following procedure

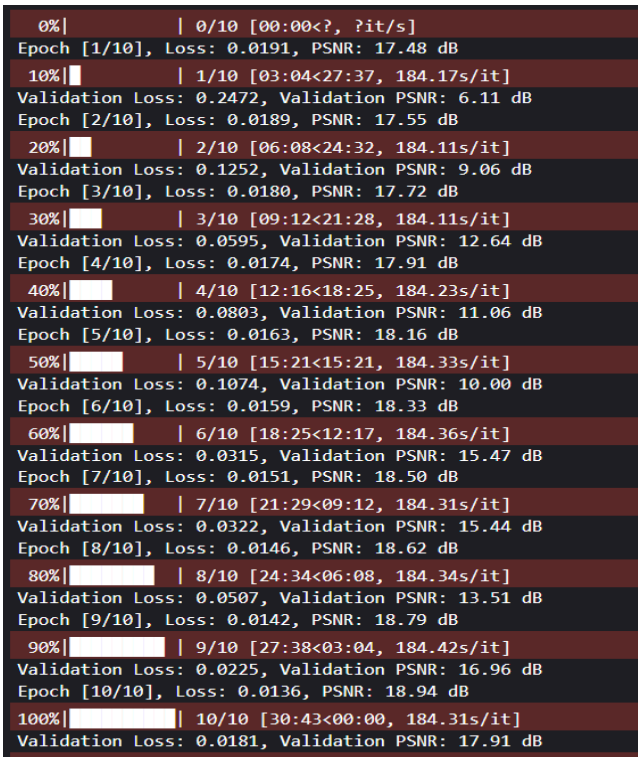
1. **Loss Function:**
   - Mean Squared Error (MSE) loss is used to measure the difference between the enhanced image and the ground truth. This loss function penalises large errors more significantly, driving the model to minimise discrepancies between the predicted and target images.
2. **Optimizer:**
   - The Adam optimizer is employed with an initial learning rate of 0.0001. Adam combines the advantages of the AdaGrad and RMSProp algorithms, providing efficient and robust optimization.
3. **Learning Rate Scheduler:**
   - A CosineAnnealingLR scheduler adjusts the learning rate to promote efficient convergence, gradually reducing the learning rate to prevent overshooting and improve fine-tuning in the later stages of training.
4. **Mixed Precision Training:**
   - The training loop utilizes automatic mixed precision to accelerate training while maintaining accuracy. This is achieved using PyTorch's `autocast` and `GradScaler`, allowing for faster computations and reduced memory usage.

SUMMARY:

The PixelWarriorNetwork successfully enhances low-light images, providing high-quality outputs with improved visibility. The combination of UNet architecture and MLP blocks allows the model to capture intricate details and enhance images effectively.

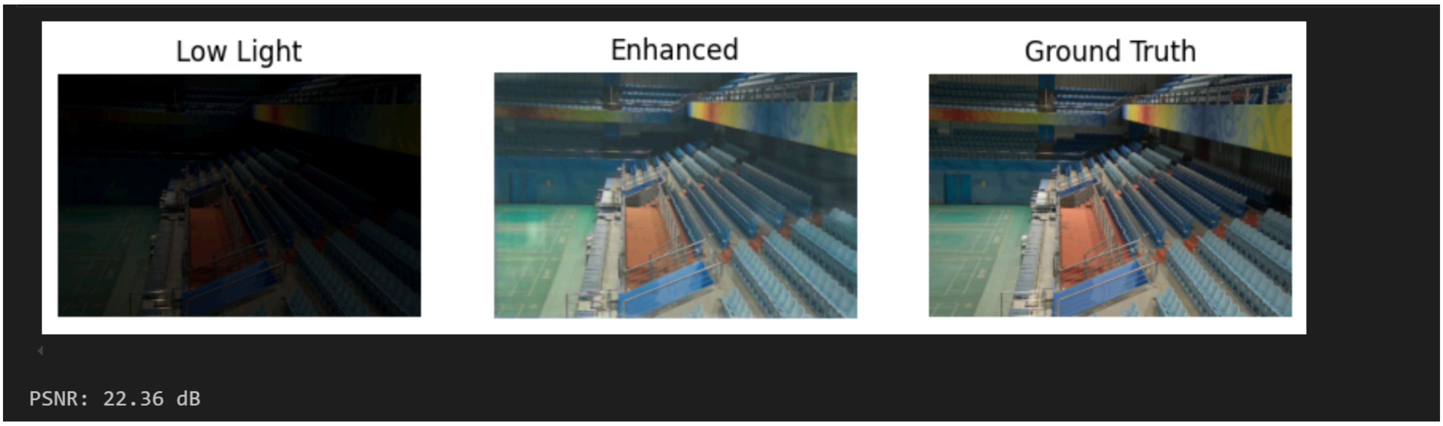
RESULTS:

PSNR Scored -

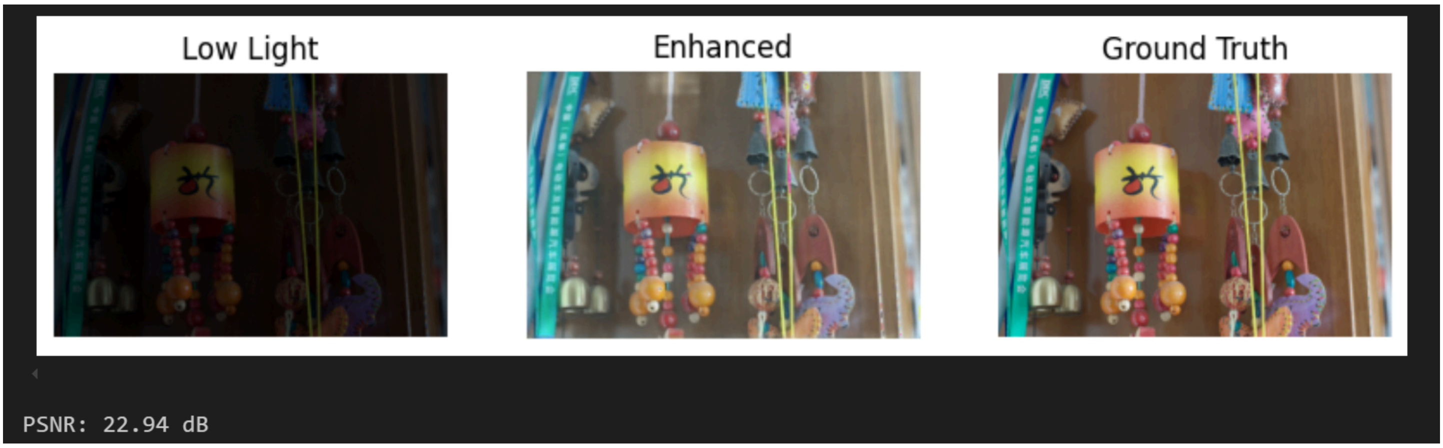


```
  0%|          | 0/10 [00:00<?, ?it/s]
Epoch [1/10], Loss: 0.0191, PSNR: 17.48 dB
 10%|█         | 1/10 [03:04<27:37, 184.17s/it]
Validation Loss: 0.2472, Validation PSNR: 6.11 dB
Epoch [2/10], Loss: 0.0189, PSNR: 17.55 dB
 20%|██        | 2/10 [06:08<24:32, 184.11s/it]
Validation Loss: 0.1252, Validation PSNR: 9.06 dB
Epoch [3/10], Loss: 0.0180, PSNR: 17.72 dB
 30%|███       | 3/10 [09:12<21:28, 184.11s/it]
Validation Loss: 0.0595, Validation PSNR: 12.64 dB
Epoch [4/10], Loss: 0.0174, PSNR: 17.91 dB
 40%|████      | 4/10 [12:16<18:25, 184.23s/it]
Validation Loss: 0.0803, Validation PSNR: 11.06 dB
Epoch [5/10], Loss: 0.0163, PSNR: 18.16 dB
 50%|█████     | 5/10 [15:21<15:21, 184.33s/it]
Validation Loss: 0.1074, Validation PSNR: 10.00 dB
Epoch [6/10], Loss: 0.0159, PSNR: 18.33 dB
 60%|██████    | 6/10 [18:25<12:17, 184.36s/it]
Validation Loss: 0.0315, Validation PSNR: 15.47 dB
Epoch [7/10], Loss: 0.0151, PSNR: 18.50 dB
 70%|███████   | 7/10 [21:29<09:12, 184.31s/it]
Validation Loss: 0.0322, Validation PSNR: 15.44 dB
Epoch [8/10], Loss: 0.0146, PSNR: 18.62 dB
 80%|████████  | 8/10 [24:34<06:08, 184.34s/it]
Validation Loss: 0.0507, Validation PSNR: 13.51 dB
Epoch [9/10], Loss: 0.0142, PSNR: 18.79 dB
 90%|█████████ | 9/10 [27:38<03:04, 184.42s/it]
Validation Loss: 0.0225, Validation PSNR: 16.96 dB
Epoch [10/10], Loss: 0.0136, PSNR: 18.94 dB
100%|██████████| 10/10 [30:43<00:00, 184.31s/it]
Validation Loss: 0.0181, Validation PSNR: 17.91 dB
```

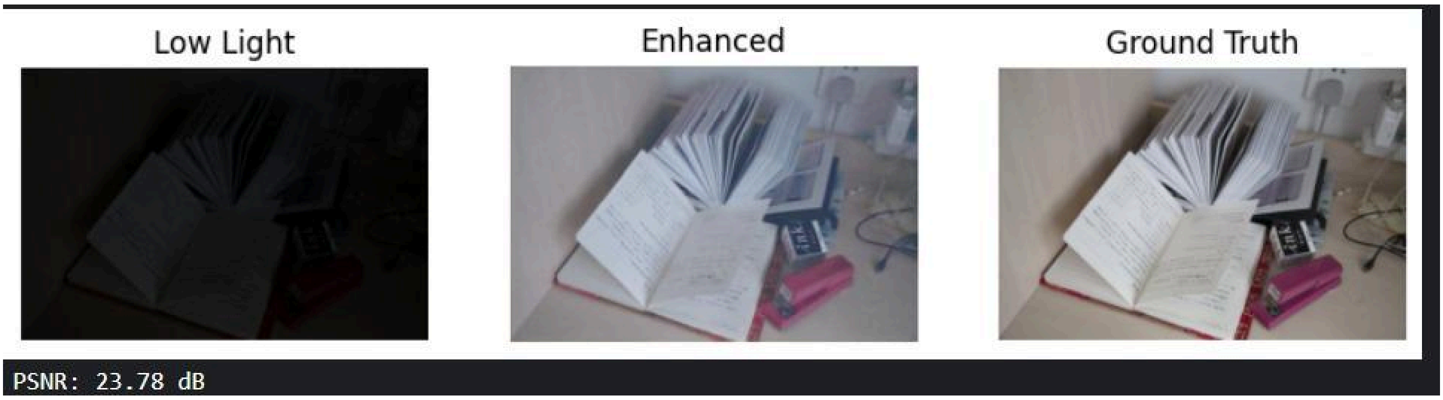
IMAGES -



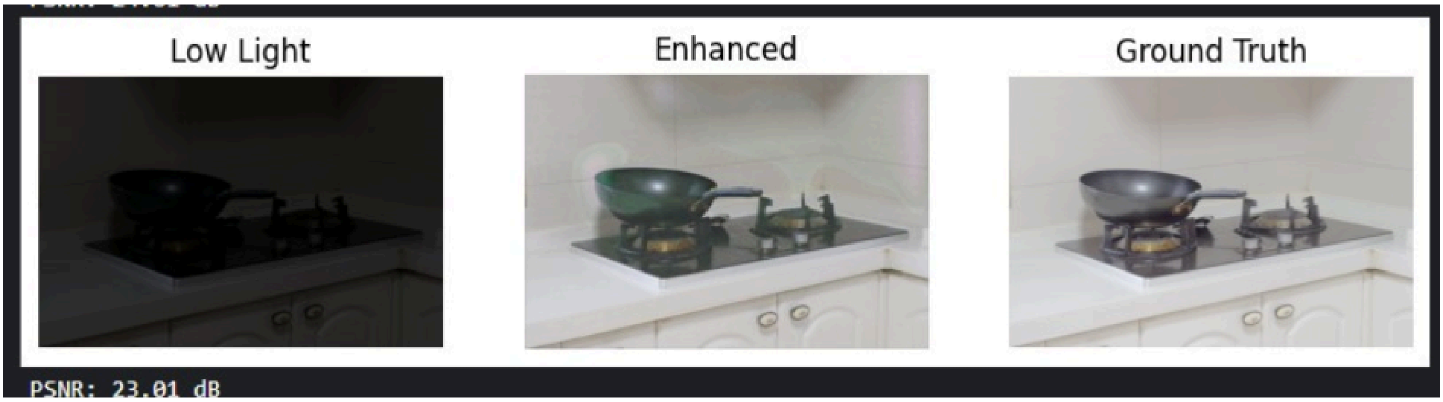
```
PSNR: 22.36 dB
```



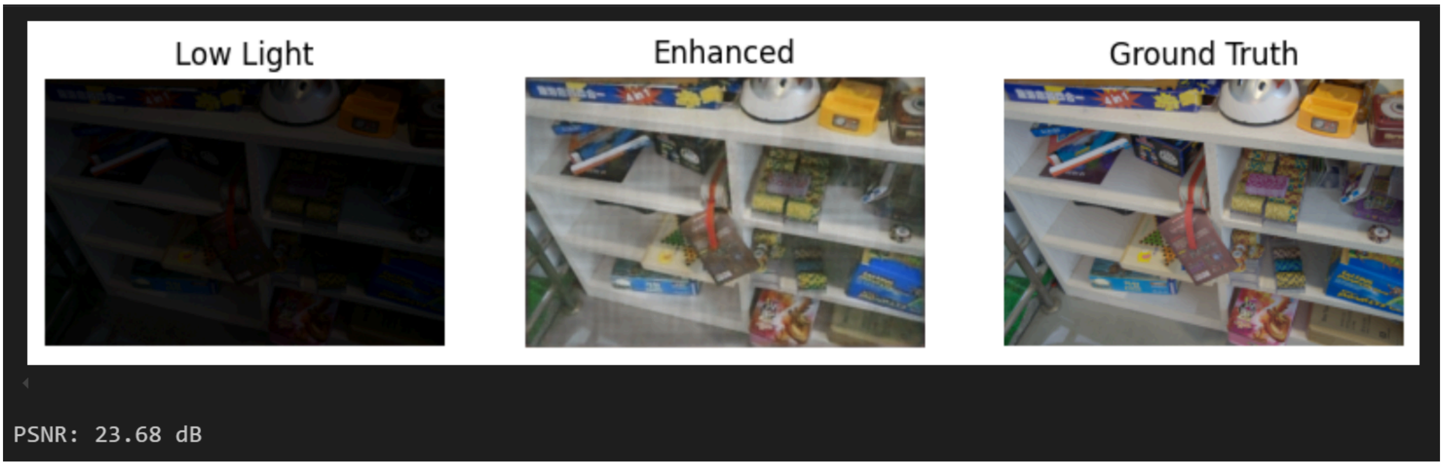
```
PSNR: 22.94 dB
```



```
PSNR: 23.78 dB
```



```
PSNR: 23.01 dB
```



```
PSNR: 23.68 dB
```