Computer Vision report

Title: Restoration and enhancement of low light images

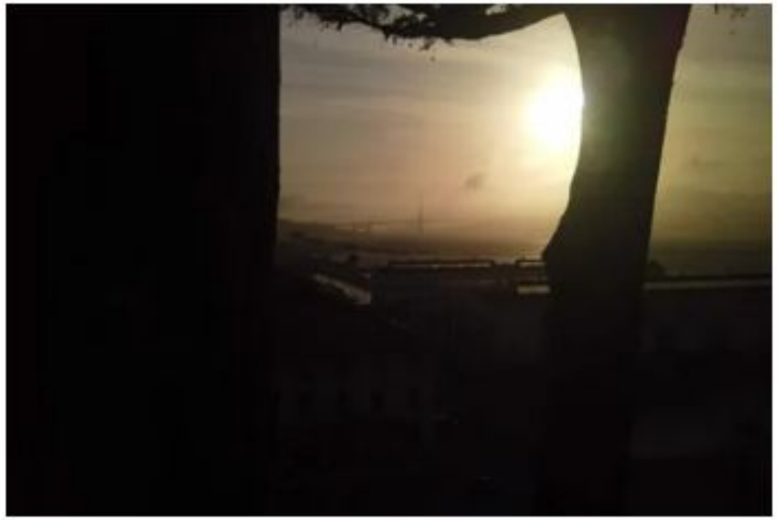
Aritra Bhaduri
2101AI40

Problem description

- Low light images are mainly caused due to under exposure, wrong lighting conditions or camera defects that make the image difficult to see.

Insufficient Lighting

Casted Shadow Backlight

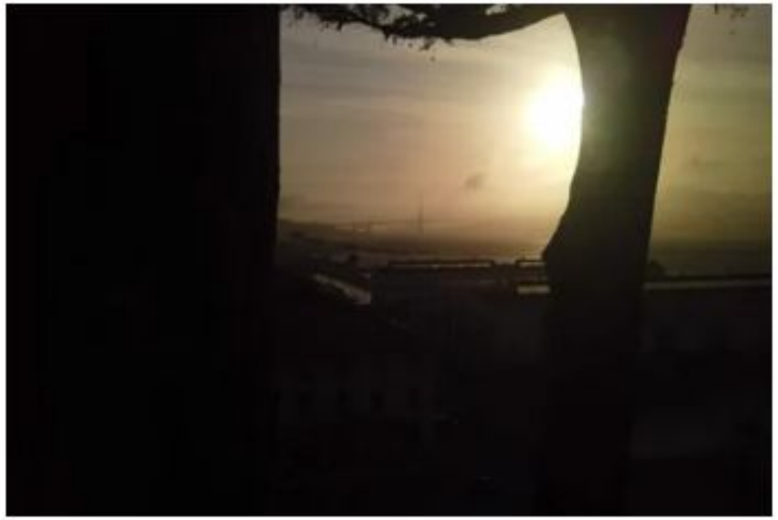


Underexposed

Types of low light images

Insufficient Lighting

Casted Shadow Backlight



Underexposed

Solution approach

- Normal filtering or computational based techniques cannot generate new information, so we use **generative methods** that add new information into the image.
- Possible approaches include GANs, Autoencoders, stable diffusion, Retinex models etc.
- These are **GPU intensive** solutions, so we need to reduce image dimensionality or model parameters to make it run on platforms like Google Colab.

Classical methods

- Histogram equalization
- Gamma transformation
- Logarithmic transformation

Original image

Histogram equalized image

Dataset

- We will be using the dataset called **LOL (LOw-Light dataset).**
- Dataset paper: Wei, Chen, Wenjing Wang, Wenhan Yang, and Jiaying Liu. "Deep retinex decomposition for low-light enhancement." *arXiv preprint arXiv:1808.04560* (2018).
- The dataset contains pairs of image and its low light counterpart, which is suitable for a deep learning-based framework.
- We scale the images down to **256x256** to work in the current hardware available.

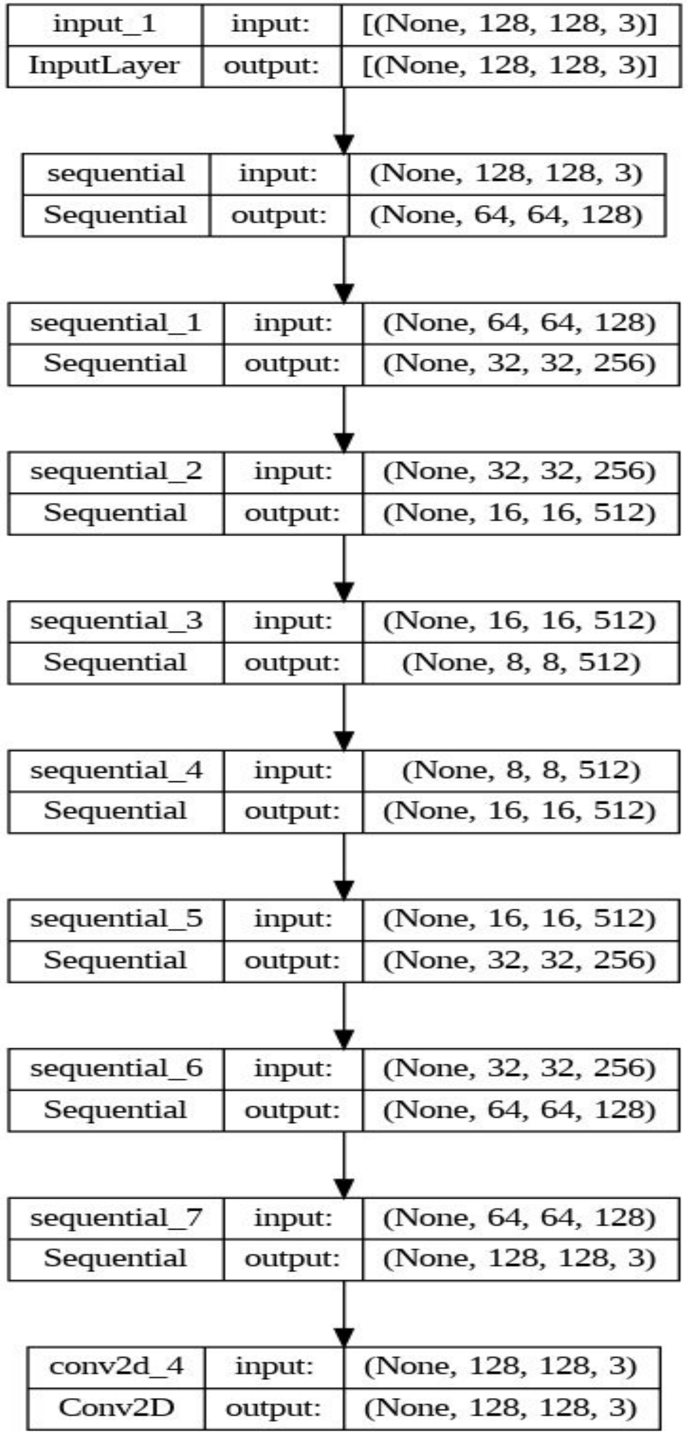
Baseline network architecture

We use the simple encoder-decoder architecture for our baseline model with following hyperparameters

- **Input**: 128x128x3
- **Output**: 128x128x3
- **Batch size**: 32
- **Dropout rate**: 0.1
- **Activation**: LeakyReLU
- **Learning Rate**: 0.001
- **Epochs**: 80

Layers:

- 4 downsampling CNN layers
- 4 upsampling CNN layers



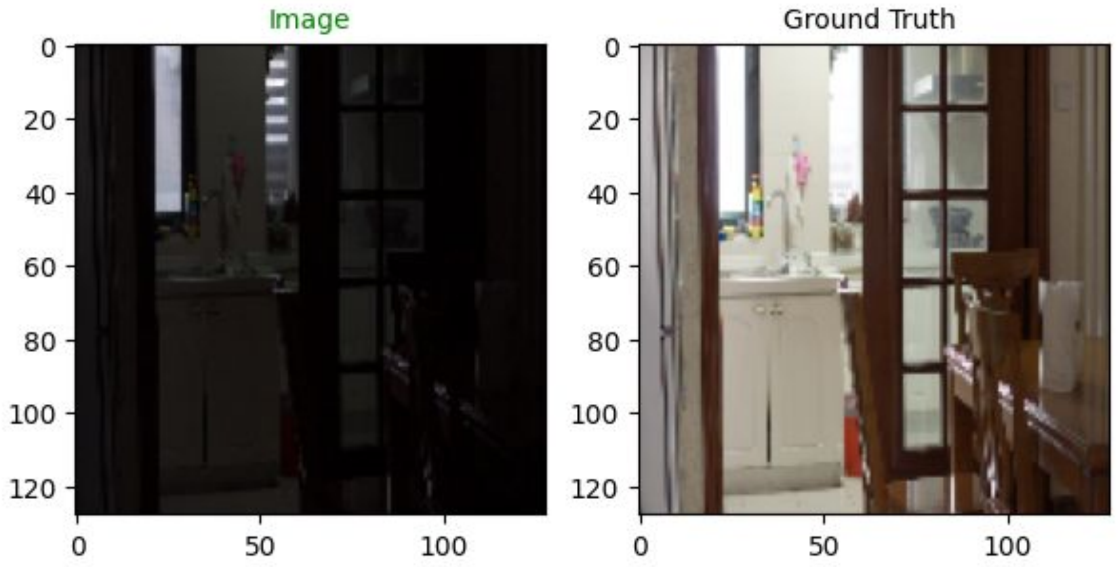


Image
Ground Truth
0
20
40
60
80
100
120
0
50
100

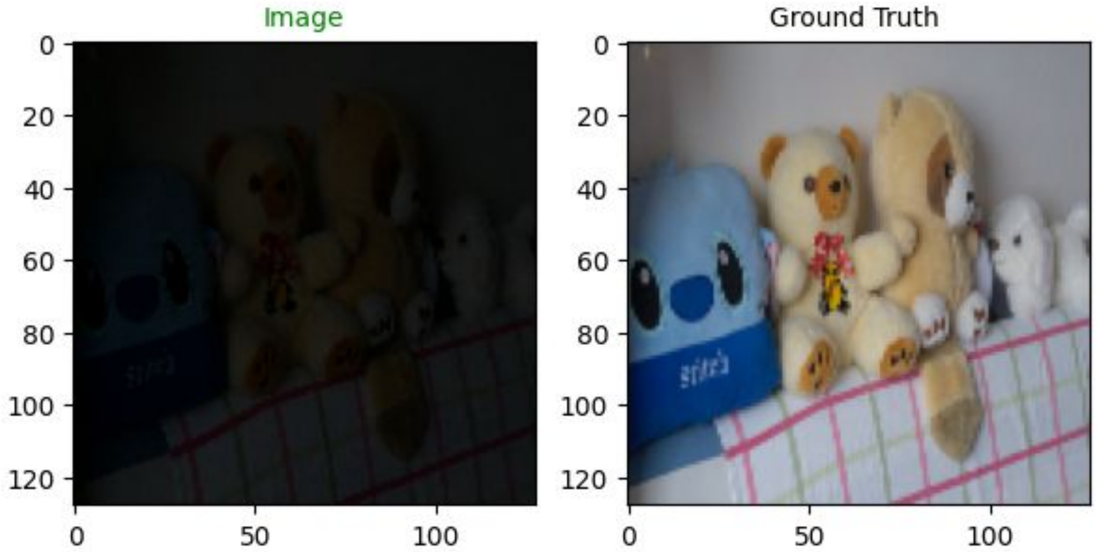


Image
Ground Truth
0
20
40
60
80
100
120
0
50
100

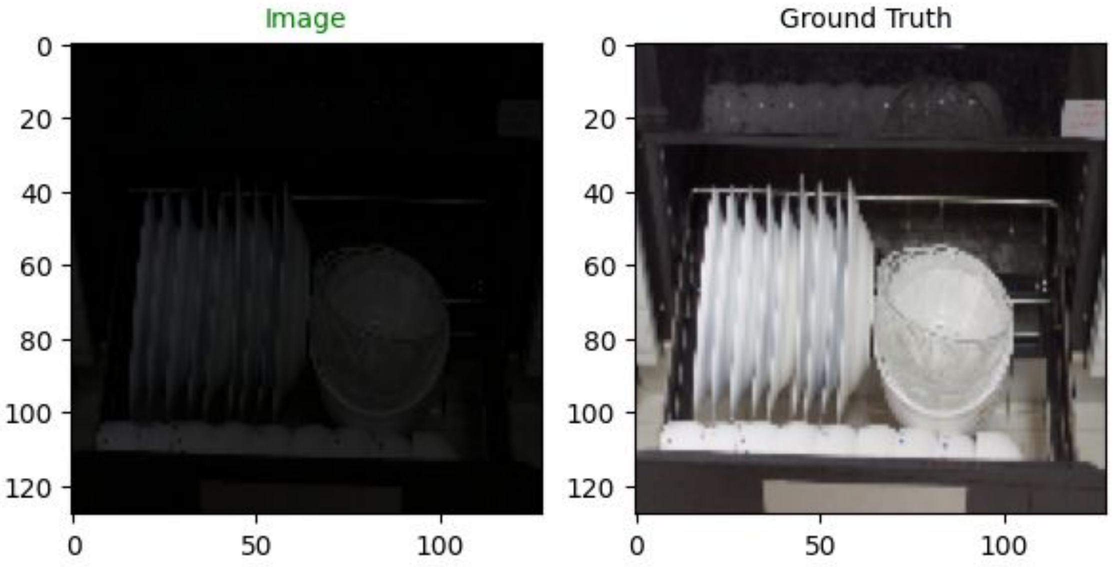
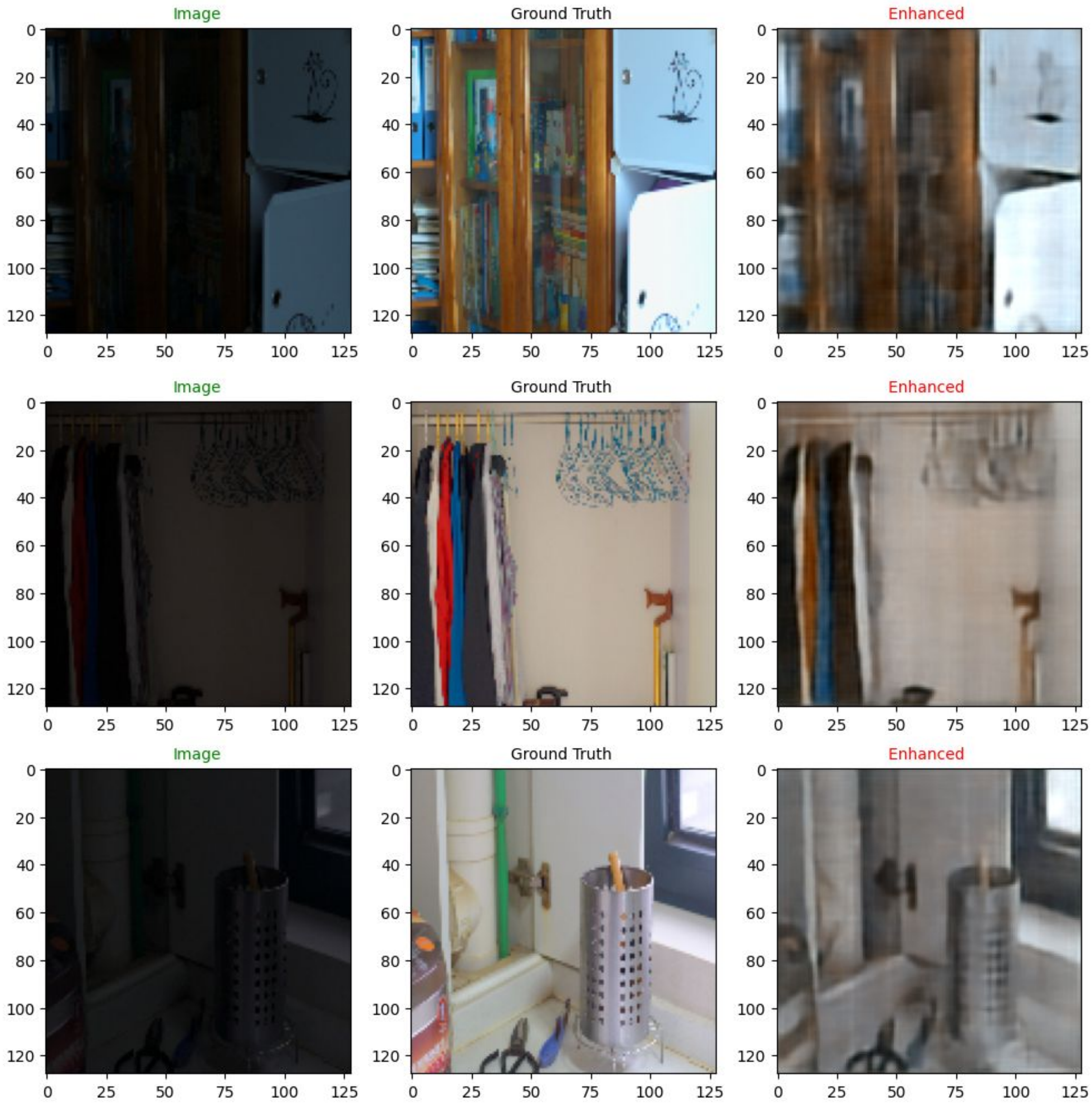


Image
Ground Truth
0
20
40
60
80
100
120
0
50
100

Preliminary results

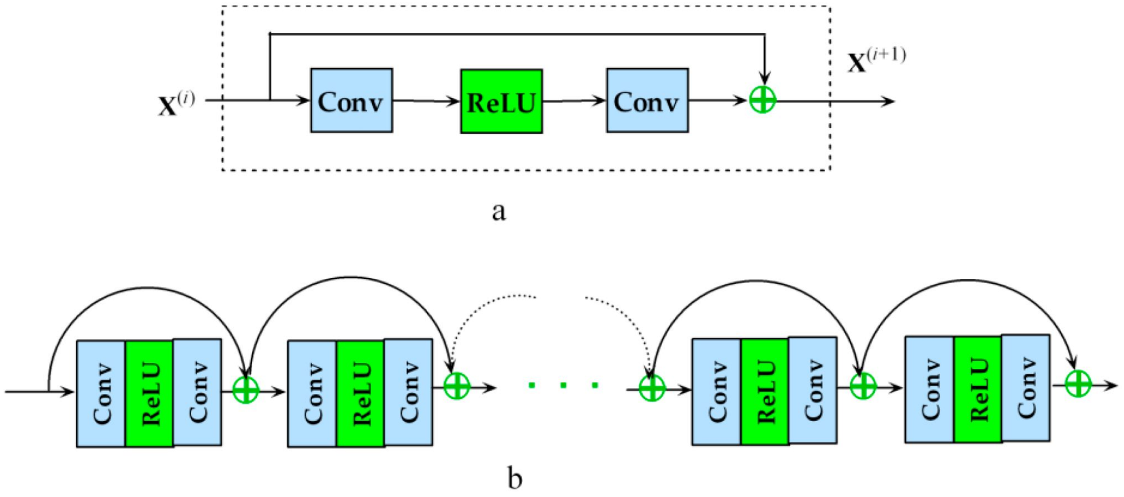


Advanced approaches

1. Autoencoder with residual connections

Residual deep CNN has been proposed to tackle the issue of accuracy degradation in deeper learning networks. Residual connection provides another path for data to reach latter parts of the neural network by skipping some layers.

It helps to solve issues like vanishing and exploding gradients, also it acts as a mechanism to retain some of the information that the network lost during forward pass.

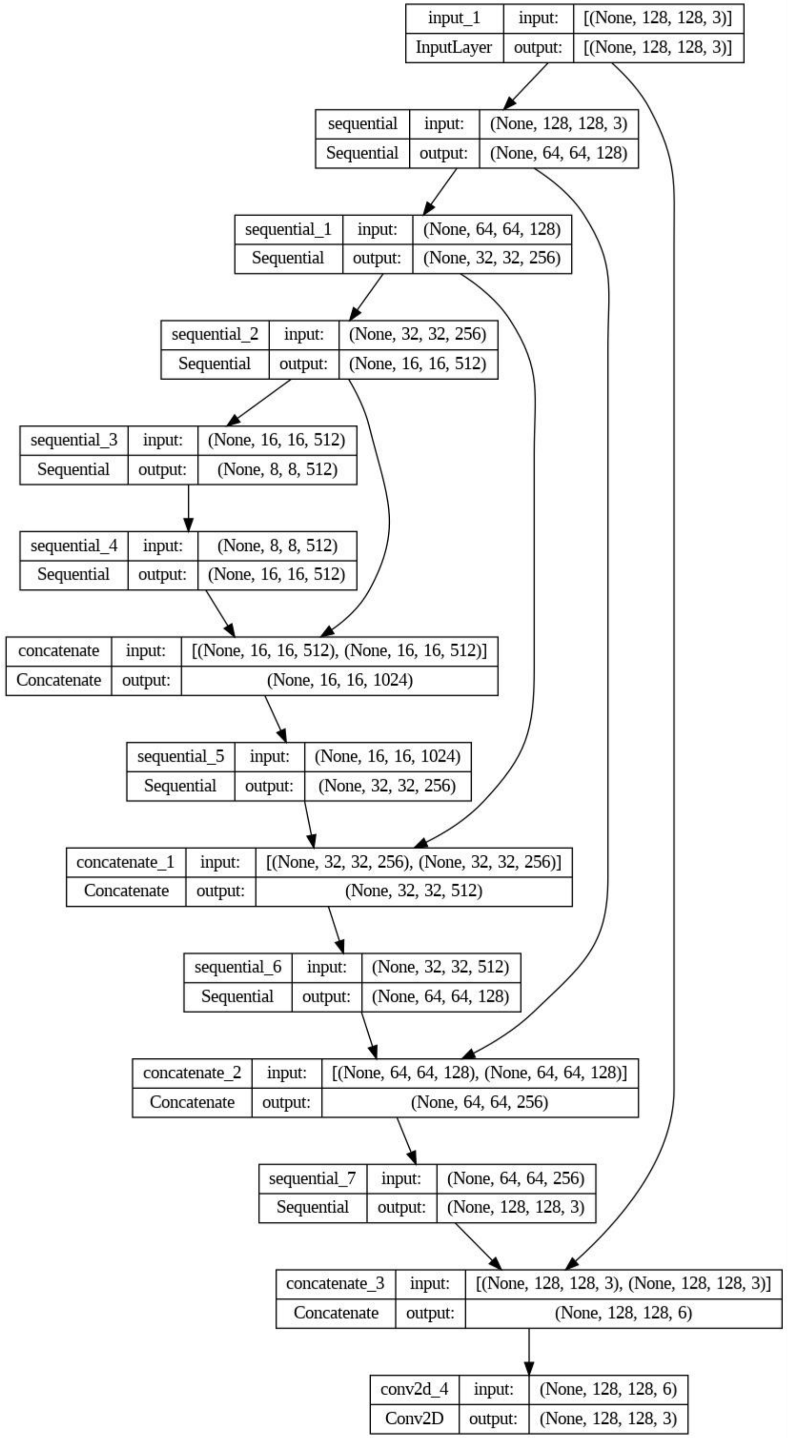


Note: Representative image

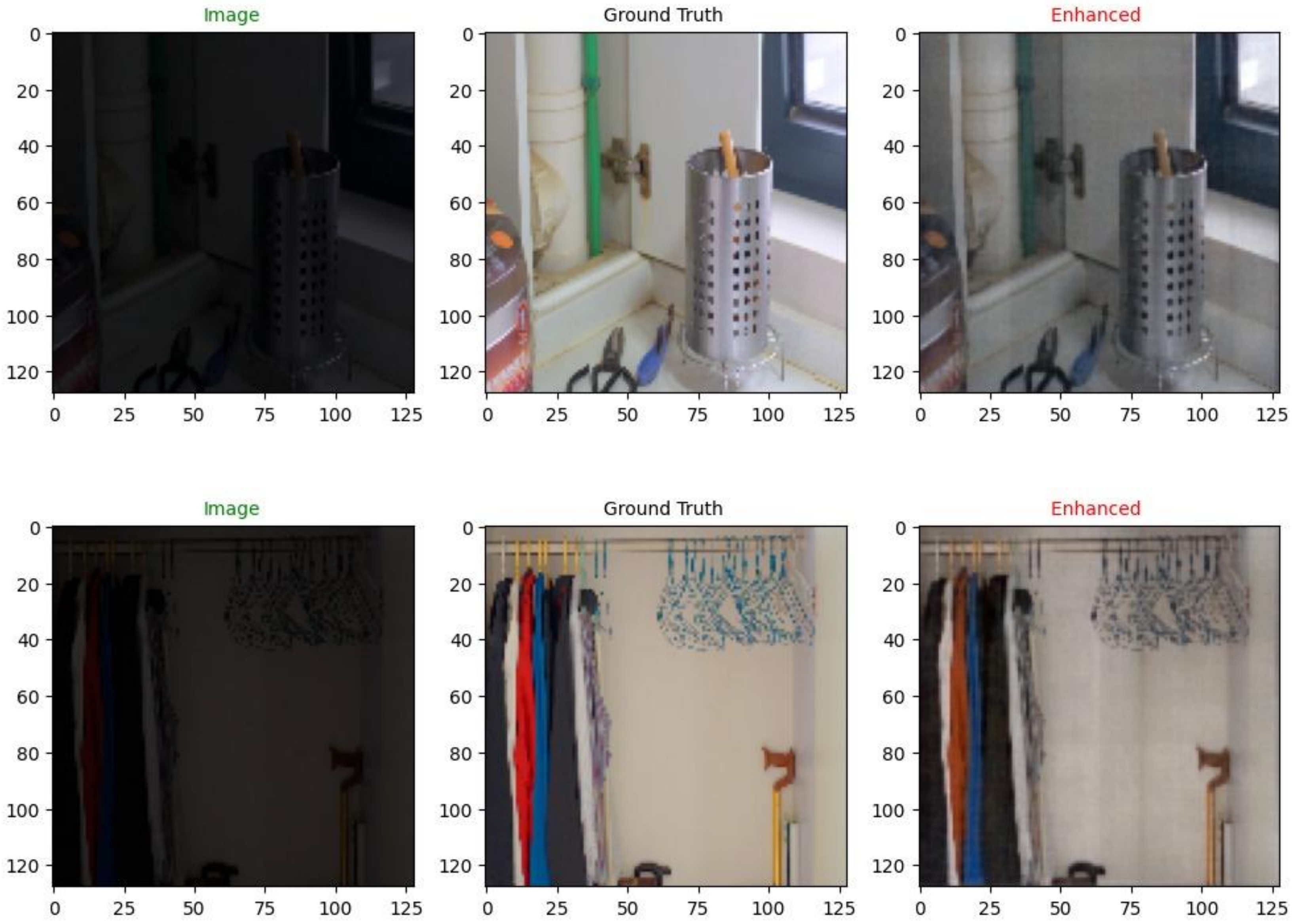
Model Summary

```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                Output Shape                 Param #   Connected to
==================================================================================================
 input_1 (InputLayer)        [(None, 128, 128, 3)]        0         []

 sequential (Sequential)     (None, 64, 64, 128)          9728      ['input_1[0][0]']

 sequential_1 (Sequential)   (None, 32, 32, 256)          819456    ['sequential[0][0]']

 sequential_2 (Sequential)   (None, 16, 16, 512)          3279360   ['sequential_1[0][0]']

 sequential_3 (Sequential)   (None, 8, 8, 512)            6556160   ['sequential_2[0][0]']

 sequential_4 (Sequential)   (None, 16, 16, 512)          6554112   ['sequential_3[0][0]']

 concatenate (Concatenate)   (None, 16, 16, 1024)         0         ['sequential_4[0][0]',
                                                                     'sequential_2[0][0]']

 sequential_5 (Sequential)   (None, 32, 32, 256)          6553856   ['concatenate[0][0]']

 concatenate_1 (Concatenate  (None, 32, 32, 512)          0         ['sequential_5[0][0]',
 )                                                                   'sequential_1[0][0]']

 sequential_6 (Sequential)   (None, 64, 64, 128)          1638528   ['concatenate_1[0][0]']

 concatenate_2 (Concatenate  (None, 64, 64, 256)          0         ['sequential_6[0][0]',
 )                                                                   'sequential[0][0]']

 sequential_7 (Sequential)   (None, 128, 128, 3)          19203     ['concatenate_2[0][0]']

 concatenate_3 (Concatenate  (None, 128, 128, 6)          0         ['sequential_7[0][0]',
 )                                                                   'input_1[0][0]']

 conv2d_4 (Conv2D)           (None, 128, 128, 3)          75        ['concatenate_3[0][0]']

==================================================================================================
Total params: 25430478 (97.01 MB)
Trainable params: 25428430 (97.00 MB)
Non-trainable params: 2048 (8.00 KB)
__________________________________________________________________________________________________
```

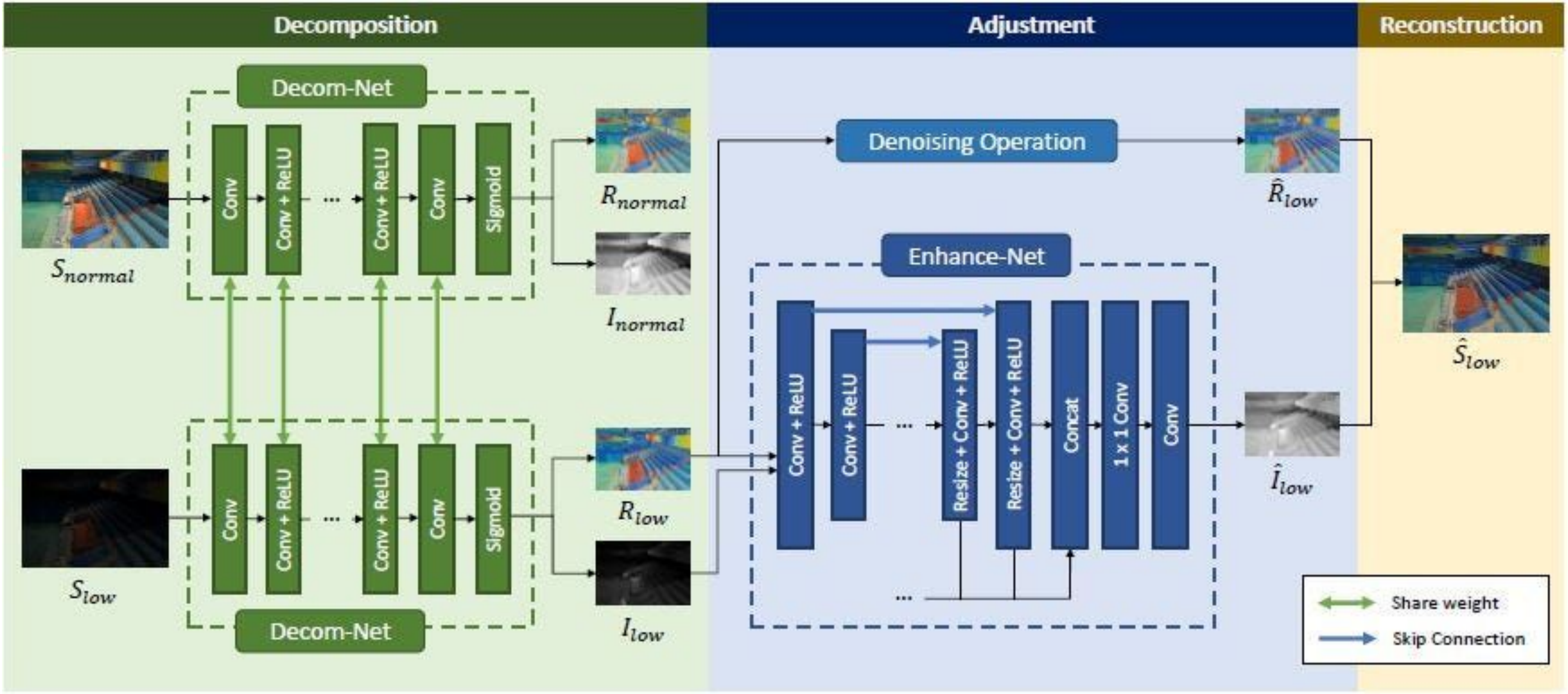
Network schematic



Autoencoder Results



2. Retinex-Net Model



Wei, Chen, et al. "Deep retinex decomposition for low-light enhancement." arXiv preprint arXiv:1808.04560 (2018).

How Retinex-Net works

- The enhancement process is divided into three steps: **decomposition**, **adjustment**, and **reconstruction**.
- In the decomposition step, a subnetwork Decom-Net decomposes the input image into reflectance and illumination.
- In the following adjustment step, an encoder-decoder based Enhance-Net brightens up the illumination.
- Multi-scale concatenation is introduced to adjust the illumination from multi-scale perspectives.
- Noise on the reflectance is also removed at this step.
- Finally, we reconstruct the adjusted illumination and reflectance to get the enhanced result.

Reflectance and illumination

The Decom-Net is learned with two constraints:

- First, low/normal-light images share the same reflectance.
- Second, the illumination map should be smooth but retain main structures, which is obtained by a structure-aware total variation loss.

(a) Input (b) Reflectance (c) Illumination

Retinex-Net Results

Original low light

Original bright image

Enhanced image

Original low light

Original bright image

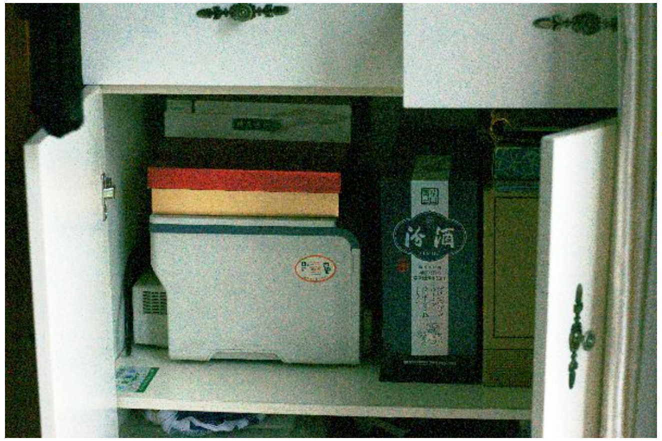
Enhanced image

Comparison between approaches

Ground truth

Autoencoder

PSNR: 12.934
SSIM: 0.708



Histogram Equalization

PSNR: 14.269
SSIM: 0.379

Retinex Net

PSNR: 11.927
SSIM: 0.316

PSNR: higher is better
SSIM: higher(closer to 1) is better

Comparison between approaches

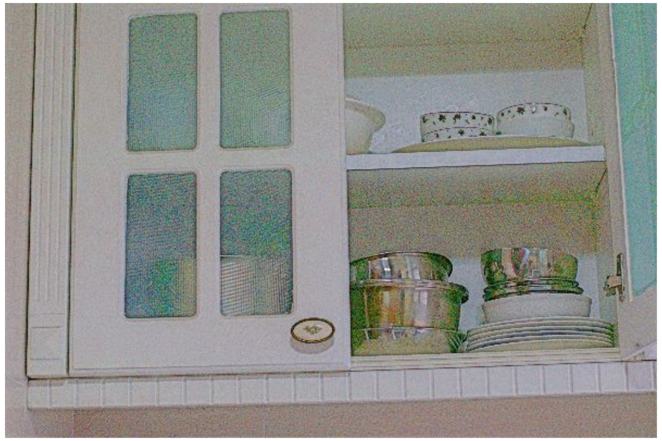
Ground truth

Autoencoder

PSNR: 21.315
SSIM: 0.836

Histogram Equalization

PSNR: 15.164
SSIM: 0.445



Retinex Net

PSNR: 17.779
SSIM: 0.492

PSNR: higher is better
SSIM: higher(closer to 1) is better

Comparison between approaches

Ground truth

Autoencoder

PSNR: 14.919
SSIM: 0.505

Histogram Equalization

PSNR: 12.31
SSIM: 0.416

Retinex Net

PSNR: 18.649
SSIM: 0.413

PSNR: higher is better
SSIM: higher(closer to 1) is better

Comparison between approaches

Ground truth

Autoencoder

PSNR: 19.719
SSIM: 0.766

Histogram Equalization

PSNR: 14.195
SSIM: 0.361

Retinex Net

PSNR: 17.219
SSIM: 0.365

PSNR: higher is better
SSIM: higher(closer to 1) is better

Final Results

Total 15 test image samples were tested with Histogram equalization, Autoencoder and Retinex-Net models(after training with a training set with 485 images).

- Histogram Equalization:
  - Average PSNR score: 14.5414
  - Average SSIM score: 0.3866
- Autoencoder Model:
  - Average PSNR score: 17.8019
  - Average SSIM score: 0.7149
- Retinex Model:
  - Average PSNR score: 16.7740
  - Average SSIM score: 0.4249

PSNR: higher is better
SSIM: higher(closer to 1) is better