Extreme Low-light image denoising Challenge

Li Qihang

May 5, 2024

1 Network Architecture

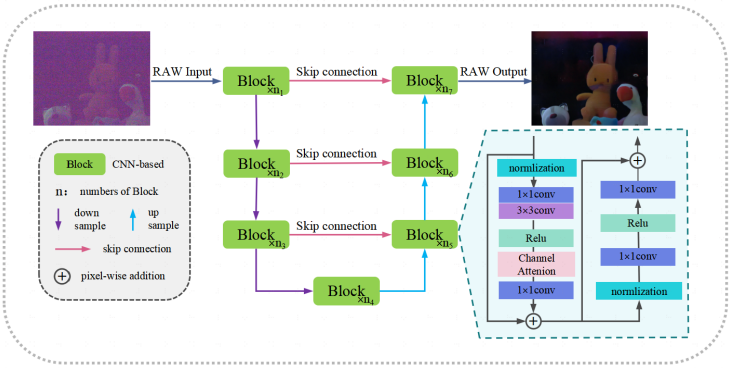

Figure 1: The overall network architecture of our method

To reduce the inter-block complexity, we adopt the classic single-stage U-shaped architecture with skip-connections, as shown in Figure 1. Neural Networks are stacked by blocks.We start from a plain block with the most common components, i.e. convolution, ReLU, and shortcut.Besides, we realize the vanilla channel attention meets the requirements: computational efficiency and brings global information to the feature map. In addition, the effectiveness of channel attention has been verified in the image restoration task, thus we add the channel attention to the plain block.

For the downsample layer, we use the convolution with a kernel size of 2 and a stride of 2. For the upsample layer, we double the channel width by a pointwise convolution first, and then follows a pixel shuffle module.There are skip connections from the encoder block to the decoder block, we simply element-wise add the encoder and decoder features as the feature fusion approach.The default width and number of blocks are 64 and 36, our network architecture consists of a total of 5 layers. The encoder has 4 layers with block quantities of 2, 2, 4, and 8 respectively. The intermediate connection layer has 12 blocks. The decoder

also has 4 layers with block quantities of 2 for each layer.

2 Training strategy

Similar to most denoising methods, we utilize the L_1 loss function as the training objective.We employ the same data preprocessing and optimization strategy as ELD during pre-training. The raw images with long exposure time in the SID train subset are utilized for noise synthesis. Concerning data preprocessing, we pack the Bayer images into 4 channels, followed by cropping the long exposure data with a patch size of 512×512, non-overlapping, step 256.We train the models for 300 epochs using the Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for optimization, without applying weight decay. The initial learning rate is set to 10^{-4} and is halved at the 150th epoch before being further reduced to 10^{-5} at the 220th epoch.

The inference code and the pre-trained models are released at here.