A Selective Re-learning Mechanism for Hyperspectral Fusion Imaging

Supplementary Material

6. More Implementation Details

6.1. Network Architecture

Embedding layers. Given the LRHSI X and MSI Y, we employ a 3×3 convolution as the Embedding layer. This layer adjusts the channel dimension of the combined MSI and LRHSI, yielding the initial feature representation.

$$F_0 = f_{emb}(Concat(X, Y)), \tag{11}$$

where $f_{emb}(\cdot)$ denotes the function of the Embedding layer.

Preliminary Fusion Module. Each SS2D block is preceded by a LayerNorm layer and includes a residual connection. Let F represent the input, then the process can be formulated as:

$$O = SS2D(LN(F)) + F, \tag{12}$$

where O is the output of the block, and $LN(\cdot)$ represents the LayerNorm layer. The state size in SS2D is set to 15. Additionally, each downsampling operation is implemented with two strided convolutions. The first layer reduces the spatial dimensions by a factor of 4, using a 6×6 kernel with a stride of 4. The second layer further reduces the dimensions by a factor of 2, using a 4×4 kernel with a stride of 2. Before each PixelShuffle (PS) operation, a 1×1 convolution is applied to upsample the channels.

Selective Re-learning Module. Based on the preliminary fusion feature Z', we uses a LayerNorm operation to normalize Z'. Next, we generate pseudo-MSI Y' using the spatial response of the observation model. Finally, we calculate the SSIM score for each feature point in the pseudo-MSI:

$$M_{spa} = SSIM_p(Y', Y), \tag{13}$$

where $SSIM_p(\cdot)$ represents the calculation of the SSIM for each feature point. We then select feature points with low SSIM scores from $Y + Y'$ based on the ratio r and feed them into the Re-learning block. Both the Re-learning blocks for spatial refinement and spectral refinement use the Spectral Transformer structure, as shown in Fig. 10. Let $V_{YY'}$ denote the input of Re-learning block. the process is formulated as:

$$\begin{aligned} V_{temp} &= MSA_{spe}(LN(V_{YY'})) + V_{YY'}, \\ V &= FFN(LN(V_{temp})) + V_{temp}, \end{aligned} \tag{14}$$

where $MSA_{spe}(\cdot)$ represents the Spectral Multi-Self-Attention block. $FFN(\cdot)$ consists of two 3×3 convolutions and a ReLU activation function. Subsequently, by introducing the residual of $V_{YY'}$, we obtain the result of spatial re-learning, denoted as $Z^{'}_{spa}$. The re-learning process

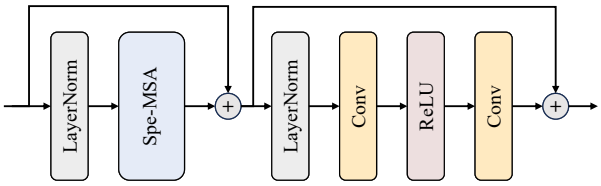


Figure 10. The architecture of the Re-learning Block. Spe-MSA is the Spectral Multi-Self-Attention block.

for spectrally distorted feature points follows a similar approach, resulting in the spectral re-learning output Z_{spe}. Finally, RF block fuses the spatial re-learning result and spectral re-learning result as:

$$\begin{aligned} Z^{'}_{spe\_up} &= PS(Z^{'}_{spe}), \\ Z_{SRL} &= f_{RF}(Z^{'}_{spe\_up}, Z^{'}_{spa}), \end{aligned} \tag{15}$$

where $f_{RF}(\cdot)$ represents the RF block. Z_{SRL} denotes the output of the Selective Re-learning Module.

6.2. Implementation Details

We implement our method using PyTorch 1.13 and train it on a single NVIDIA 3090 with an Intel i9-10920X CPU. We adopt the Adam optimizer to train the network for 2000 epochs. The initial learning rate is set to $4e-4$ and adjusted using a cosine annealing strategy. The L_1 loss is used as the optimization objective.

6.3. Dataset Setup

For the CAVE and Harvard datasets, we evaluate the performance of the method on the last 12 and 20 images, respectively, while using the remaining images as training data. We use the spectral response of the Nikon 700 to obtain the MSI and an 8×8 Gaussian blur kernel with a variance of 3 to obtain the LRHSI.

7. More Experiments

7.1. Proof for Overfitting Mitigation

As shown in Table 5, the training-testing loss gap without the selective re-learning mechanism is $0.00235 - 0.00115 = 0.0012$, which decreases to $0.00225 - 0.00119 = 0.00106$ with it—a 11% reduction—achieving better PSNR. Moreover, our method has lower FLOPs and better performance compared to other methods. We will add this analysis.

Table 5. Cross-dataset evaluation for deep learning-based methods.

Model	Train Loss	Test Loss	Difference
W/O selection	0.00235	0.00115	0.00120
W/ selection	0.00225	0.00119	0.00106

Table 6. Cross-dataset evaluation for deep learning-based methods. The best result is marked in bold font.

Method	PSNR	SAM	UIQI	SSIM
DHIF-Net	45.7367	3.1949	0.8848	0.9829
DSPNet	45.8346	3.1725	0.8763	0.9819
MIMO-SST	46.2921	3.0410	0.8764	0.9825
Mog-DCN	45.3561	3.1216	0.8783	0.9818
LRTN	44.9675	3.2694	0.8699	0.9813
Ours	**46.4822**	**2.9855**	**0.8816**	**0.9832**

7.2. FLOPs Analysis

When calculating the FLOPs of the network, we primarily calculate the FLOPs of convolutional layers, linear layers, and Mamba while ignoring the computation of the attention matrix in the Transformer. The FLOPs of Mamba are measured according to the method in the link https://github.com/state-spaces/mamba/issues/110.

7.3. Cross-dataset evaluation

To further assess generalization capability, we conduct cross-dataset evaluation. We evaluate the model trained on CAVE using the Harvard dataset, as shown in Table 6. The results demonstrate that the proposed approach achieves the best performance across all metrics, showcasing superior generalization capability.

7.4. More Visualization Results

We provide additional visual results of other test data from the CAVE and Harvard datasets, as shown in Fig. 11 and Fig. 12.

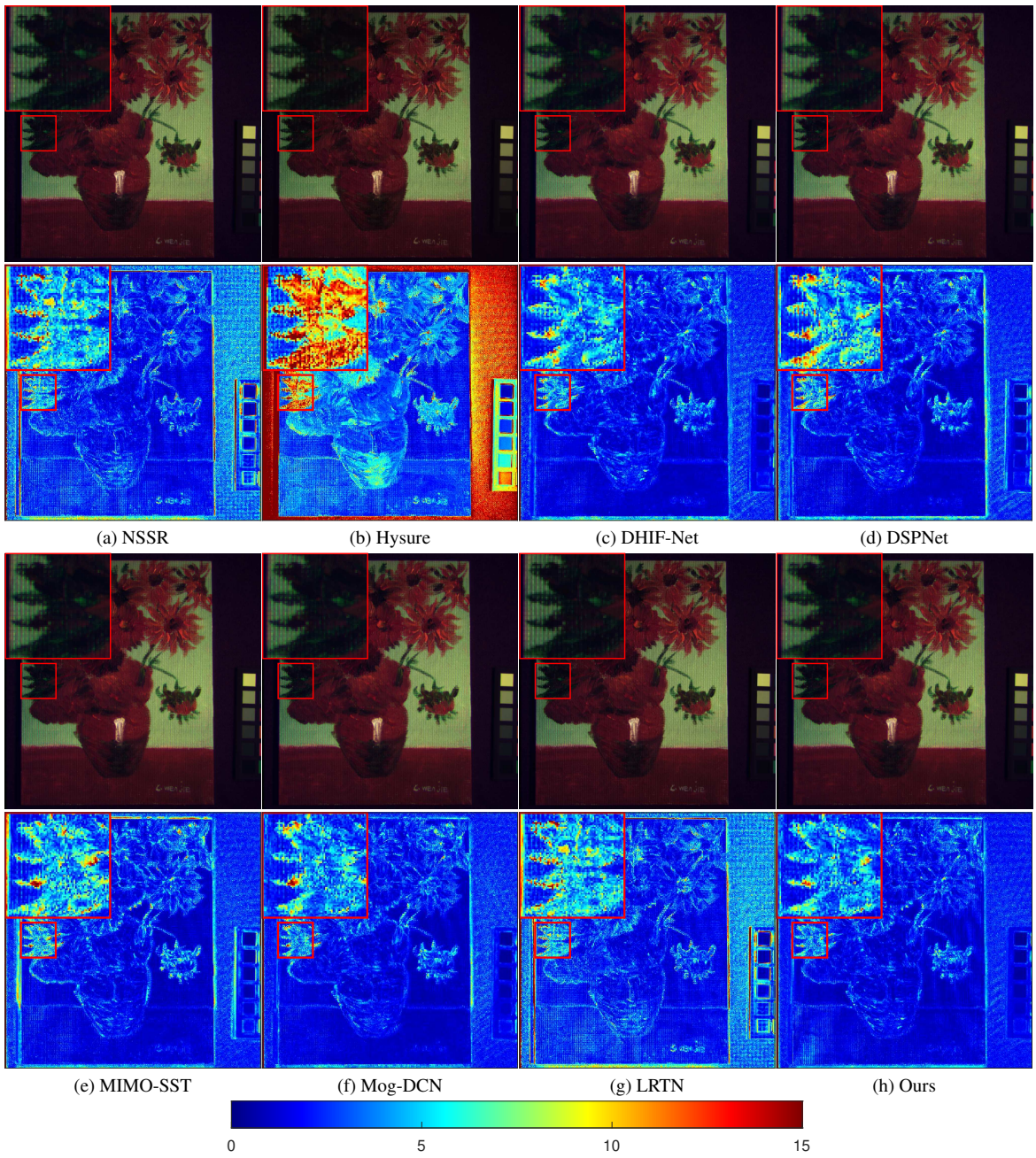
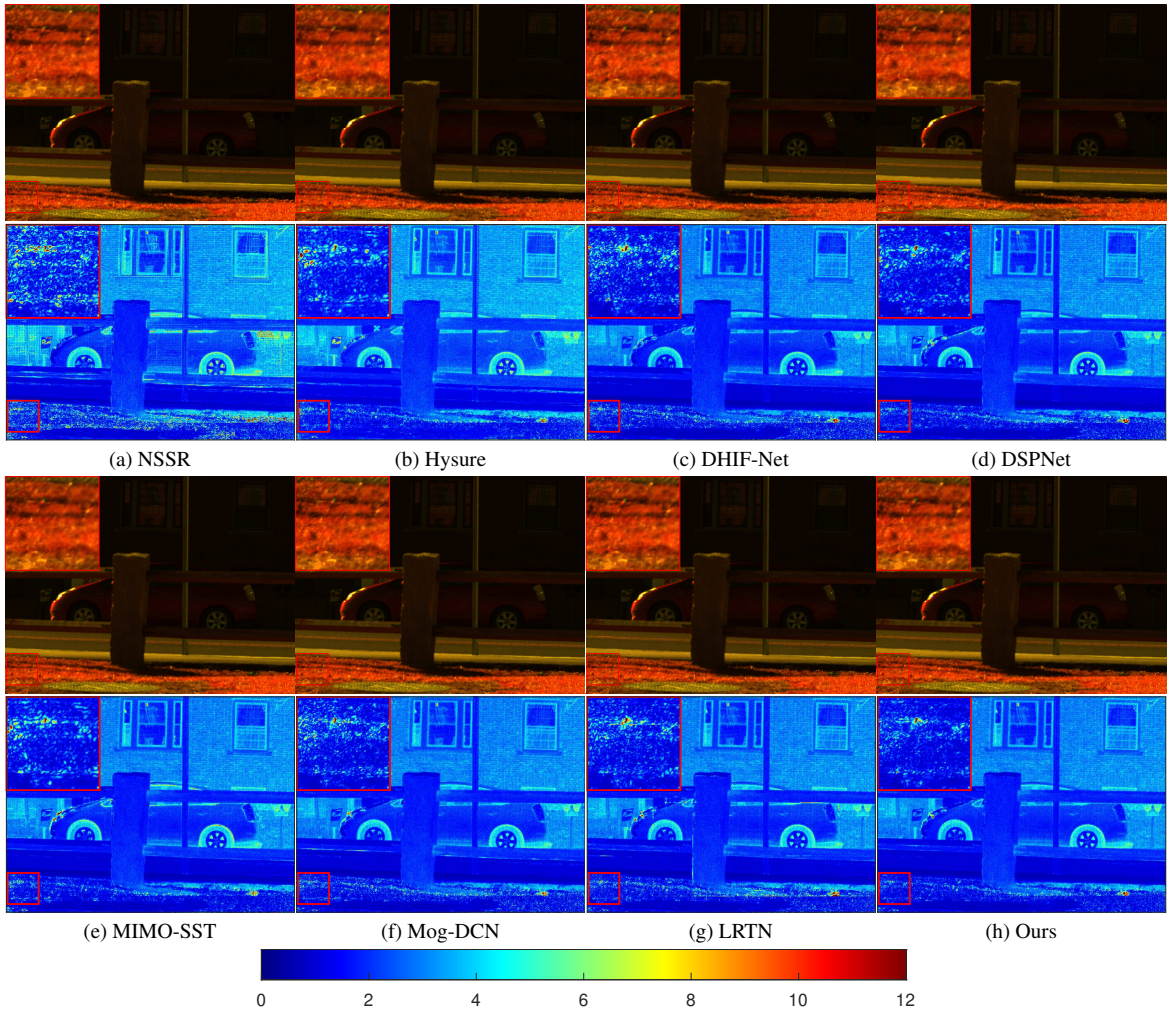


Figure 11. The pseudo-color images and spectral angle error maps of reconstructed oil_painting_ms (a test image of the CAVE dataset) by different methods.



(a) NSSR (b) Hysure (c) DHIF-Net (d) DSPNet

(e) MIMO-SST (f) Mog-DCN (g) LRTN (h) Ours

Figure 12. The pseudo-color images spectral angle error maps of reconstructed imgf1 (a test image of the Harvard dataset) by different methods.