MobileIE: An Extremely Lightweight and Effective ConvNet for Real-Time Image Enhancement on Mobile Devices

Supplementary Material

1. Motivation

Principle of Simplicity [41]:

Usability: A simple architecture without complex operators, making optimization straightforward.

Uniformity: Few core modules for reduced complexity and faster deployment.

Effectiveness: Strong model performance.

Efficiency: Minimal parameters and computation with balanced accuracy.

Generally, popular works have developed more efficient CNNs from three perspectives: 1) Convolutional blocks [7–9, 16, 33], 2) Attention modules [3, 34, 40, 41], and 3) Training strategies [1, 2, 25, 32]. We revisit the design of the efficient model through these components.

For Convolutional Block: By employing a reparameterization paradigm, we combine the training advantages of multi-branch models with the inference advantages of single-path models, enhancing training performance while accelerating inference and saving memory.

For Attention Module: We design lightweight and efficient attention modules that achieve an optimal balance between speed and performance through adaptive weighting and structural optimization.

For Training Strategy: We tackle efficiency from a different perspective, optimizing training strategies instead of adding components. Most methods [4, 45] rely on complex backbones, making model size minimization challenging. Instead, optimizing training strategies and loss boosts performance and efficiency without added complexity.

Balancing performance and efficiency is crucial, especially for resource-limited mobile devices. This paper introduces MobileIE, a lightweight CNN for image enhancement. Designed for efficiency, deployability, and strong performance, MobileIE follows the principles of Simplicity and Effectiveness. By utilizing fundamental CNN building blocks, it delivers impressive results with a minimalist structure. To validate MobileIE's practicality, we conducted extensive tests on various commercial smartphones. Results show that MobileIE, with just 4K parameters and 0.924 GFLOPs, enhances images in real time, exceeding 100 FPS on mobile devices. This makes it a promising solution for efficient edge computing and mobile deployment.

2. Network Architectures

The detailed architecture of MobileIE is summarized in Table 1, comprising four key components: shallow feature extraction, deep feature extraction, feature transform, and an attention mechanism.

Description	Training Stage		Inference Stage
	MBRConv5 × 5 (Input=3, output=12)		
Shallow Feature Extraction	Name	Details	Conv5 × 5, C(3, 12, 5, 1, 2)
	Conv5 × 5	C(3, 48, 5, 1, 2)	
	Conv3 × 3	C(3, 48, 3, 1, 1)	
	Conv3 × 1	C(3, 48, 3, 1, (1, 0))	
	Conv1 × 3	C(3, 48, 3, 1, (0, 1))	
	Conv1 × 1	C(3, 48, 1, 1, 0)	
	Channel-Wise Concatenation		
	Conv1 × 1	C(480, 12, 1, 1, 0)	
	PReLU(Channel = 12)		PReLU(12)
Two-Deep Feature Extraction	MBRConv3 × 3 (Input=12, output=12)		Conv3 × 3, C(12, 12, 3, 1, 1)
	Name	Details	
	Conv3 × 3	C(12, 48, 3, 1, 1)	
	Conv3 × 1	C(12, 48, 3, 1, (1, 0))	
	Conv1 × 3	C(12, 48, 3, 1, (0, 1))	
	Conv1 × 1	C(12, 48, 1, 1, 0)	
	Channel-Wise Concatenation		
	Conv1 × 1	C(384, 12, 1, 1, 0)	
	Feature Self-Transform		Feature Self-Transform
Hierarchical Dual-Path Attention Mechanism	AdaptiveAvgPool(output size=1)		AdaptiveAvgPool(1)
	MBRConv1 × 1 (Input=12, output=12)		Conv1 × 1, C(12, 12, 1, 1, 0)
	Name	Details	
	Conv1 × 1	C(12, 48, 1, 1, 0)	
	Channel-Wise Concatenation		
	Conv1 × 1	C(96, 12, 1, 1, 0)	
	Sigmoid		Sigmoid
	MBRConv1 × 1, (Input=1, output=12)		Conv1 × 1, C(1, 12, 1, 0, 1)
	Sigmoid		Sigmoid
Output Result	MBRConv3 × 3, (Input=12, output=3)		Conv3 × 3, C(12, 3, 3, 1, 1)

Table 1. MobileIE detailed architecture specifications.

Shallow feature extraction uses an MBRConv5 × 5 layer with a PReLU activation function to capture low-level features such as textures and edges, which primarily depend on local spatial information. Deep feature extraction employs two cascaded MBRConv3 × 3 layers to extract high-level semantic features, further enhanced by the feature self-transform module for improved feature representation. A hierarchical dual-path attention mechanism guides the model to focus on salient image regions. Finally, an MBRConv3 × 3 layer processes the refined feature maps into the desired output format. During inference, all MBRConv layers are simplified into standard convolutions, significantly reducing computational overhead and model size.

The primary goal of shallow feature extraction is to capture basic, low-level features that rely on local information and straightforward nonlinear mappings. These features, being relatively simple, do not benefit from complex trans-

formations. In contrast, feature self-transform is designed for deep feature extraction, where high-level semantic features require more sophisticated transformations to enhance their expressiveness. Using feature self-transform in shallow feature extraction would be unnecessarily complex and less effective for capturing basic features. The use of two cascaded MBRConv3 $\times$ 3 layers in the deep feature extraction module strikes a balance between performance and computational efficiency.

2.1. Design of MBRConv.

1. Kernel Size Selection: The choice of convolution kernel size has a substantial impact on both the performance and latency of CNNs. Larger kernels enhance feature extraction [10] but increase computational complexity and memory overhead, rendering them less suitable for deployment on mobile devices. Additionally, mainstream compilers and libraries often optimize 3×3 kernels more effectively than larger ones [9]. To balance performance and efficiency, especially for mobile inference, our design is inspired by the architectural principles of MobileNetV3 [16]. Specifically, MBRConv5 $\times$ 5 is utilized in the shallow feature extraction, whereas the computationally lighter and more efficient MBRConv3 $\times$ 3 is adopted for the subsequent modules.

2. Re-parameterization of MBRConv: MBRConv incorporates three types of kernels: 5×5, 3×3, and 1×1. For instance, in MBRConv3 $\times$ 3, let the input and output channels be denoted as in_C and out_C, respectively. Within the multi-branch structure, the largest kernel determines the equivalent kernel obtained after re-parameterization. During execution, the input channels are first transformed into intermediate channels (mid_C) through parallel branches. The features generated from these branches are then concatenated to effectively capture multi-scale information. Finally, a 1×1 convolution integrates these features and reduces the channel dimensions to out_C.

Re-parameterization is an equivalent transformation that consolidates multi-branch convolution parameters into a single standard convolution, ensuring no degradation in model performance. Specifically, a sequence comprising a standard convolution and a batch normalization layer can be equivalently transformed into a single convolution operation. The operation of a standard convolution can be expressed as:

$$F_{\text{out}} = Conv\left(F_{in}\right) = F_{in} * K + B \tag{1}$$

$$BN\left(F_{in}\right) = \gamma \cdot \frac{\left(F_{in} - mean\right)}{\sqrt{var}} + \beta. \tag{2}$$

Here, $*$ denotes the convolution operation. F_{in}, F_{out} represent the input and output feature maps, respectively. The convolution kernel, $K \in \mathbb{R}^{out_C \times in_C \times k \times k}$, defines the weights, while $B \in \mathbb{R}^{out_C}$ is the bias term. The $mean$ and var are computed for normalization, and γ, β are learnable parameters for scaling and shifting, respectively.

Substituting F_{out} into the Batch Normalization process yields:

$$\begin{aligned} &BN\left(Conv\left(F_{in}\right)\right) = \gamma \cdot \frac{\left(F_{in} * K + B - mean\right)}{\sqrt{var}} + \beta \\ &= F_{in} * \left(\frac{\gamma \cdot K}{\sqrt{var}}\right) + \frac{\gamma}{\sqrt{var}} \cdot (B - mean) + \beta \end{aligned} \tag{3}$$

Thus, the new standardized convolution kernel and bias are given as:

$$\begin{cases} Knew = \frac{\gamma \cdot K}{\sqrt{var}} \\ B_{new} = \frac{\gamma}{\sqrt{var}} \cdot (B - mean) + \beta \end{cases} \tag{4}$$

Multi-branch Convolution Fusion: By merging BN parameters into the convolution kernel and bias, the original BN-augmented structure is simplified into one without BN layers. The convolution kernels of all branches are then concatenated to form a unified kernel.

For a multi-branch setup, the parameters of 3×3, 1×1, 3×1, and 1×3 convolutions are denoted as $\{K_{3\times3}, B_{3\times3}\}$, $\{K_{1\times1}, B_{1\times1}\}$, $\{K_{3\times1}, B_{3\times1}\}$ and $\{K_{1\times3}, B_{1\times3}\}$ respectively. Using $K_{3\times3}$ as a reference, other kernels are zero-padded to match its size. All kernels and biases are then concatenated along the channel dimension to obtain $\{K_{mid}, B_{mid}\}$, which are finally fused with the output's 1×1 convolution:

$$\begin{aligned} &F_{out} = K_{out} * ((K_{mid} * F_{in} + B_{mid}) + B_{out} \\ &= (K_{out} * K_{mid}) * F_{in} + (K_{out} * B_{mid} + B_{out}) \end{aligned} \tag{5}$$

where K_{out}, B_{out} represent the parameters of the output 1×1 convolution. K_{final} and B_{final} represent the equivalent parameters of the MBRConv3 $\times$ 3 converted into a single standard 3×3 convolution. These are defined as:

$$\begin{cases} K_{final} = K_{out} * K_{mid}, \\ B_{final} = K_{out} * B_{mid} + B_{out} \end{cases} \tag{6}$$

2.2. Design of Feature Self-Transform (FST).

Unlike traditional linear weighting methods, FST achieves second-order feature interaction by performing element-wise multiplication on the weighted features. Specifically, the FST derived from MBRConv3 $\times$ 3 can be described as:

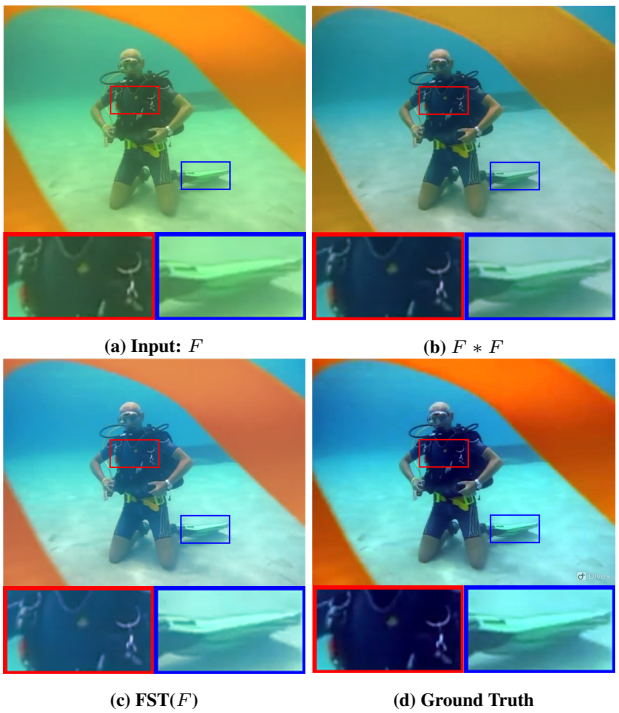
$$\begin{aligned} &FST(F_{in}) = Scale \cdot (K * F_{in} + B)^2 + Bias \\ &= (Scale \cdot K^2) {F_{in}}^2 + Scale \cdot (2KB * F_{in} + B^2) + Bias \end{aligned} \tag{7}$$

where F_{in} represents the input features, and K and B denote the parameters of the MBRConv3 $\times$ 3. $Scale$ and $Bias$ are learnable scaling and offset factors. Features constructed through second-order interactions preserve their linear components while effectively capturing nonlinear characteristics. This approach enriches the network with

complex nonlinear structures, significantly enhancing its representation capability, especially in scenarios demanding high-dimensional feature extraction.

FST offers rich feature representations through simple weighting and multiplication operations, with minimal computational overhead. Compared to adding extra convolutional layers or employing complex nonlinear functions, this method is simpler, more efficient, and highly suitable for lightweight models or tasks prioritizing inference speed.

Learnable weight and bias in feature self-transform dynamically adjust feature distribution and mitigate optimization imbalance from feature range differences. Visual results (Figure 1) demonstrate significant improvements in detail restoration and color reconstruction.



(a) Input: F **(b)** $F * F$

(c) FST(F) **(d) Ground Truth**

Figure 1. Visualization of feature self-transform (FST) ablation.

2.3. Design of Hierarchical Dual-Path Attention.

Inspired by the human visual system's focus on salient regions, attention mechanisms enhance feature utilization in CNNs by prioritizing critical input areas. Traditional attention models demand substantial computational resources, particularly for high-channel, high-resolution features, which restricts their applicability on resource-constrained devices.

To overcome these limitations, we introduce a simplified Hierarchical Dual-Path Attention (HDPA) mechanism tailored for MobileIE. HDPA integrates hierarchical and dual-path designs to balance efficiency and performance. Global average pooling enhances global feature representation, whereas local max pooling captures fine-grained details, enabling effective global-local integration. By leveraging lightweight convolutions, HDPA minimizes computational overhead, rendering it particularly suitable for mobile and embedded systems. Moreover, its dual-path structure facilitates mutual optimization during backpropagation, thereby improving feature precision.

The mutual optimization process of the dual paths during backpropagation can be formalized as follows:

$$\frac{\partial \mathrm{HDPA}(F)}{\partial F} = \frac{\partial A_g(F)}{\partial F} \cdot A_l(F) \cdot F + A_g(F) \cdot \frac{\partial A_l(F)}{\partial F} \cdot F + A_g(F) \cdot A_l(F) \tag{8}$$

Where F represents the input feature map, $A_g(F)$ the global attention weights, and $A_l(F)$ the local attention weights. The formula shows how global and local paths dynamically influence each other during gradient propagation, ensuring efficient and precise feature refinement.

2.4. Analysis of LVW loss.

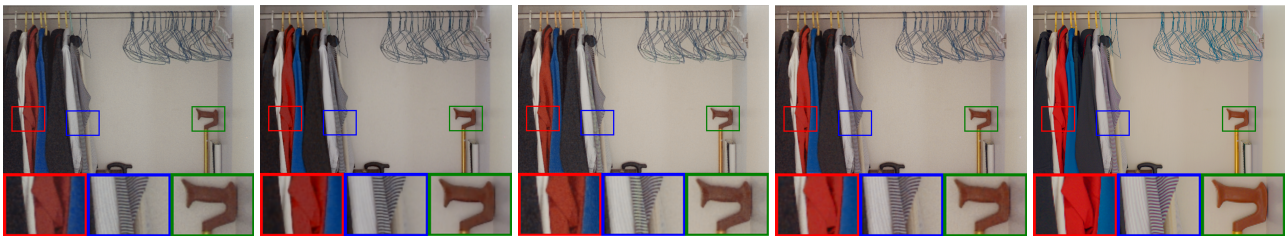
The visualized results (Figure 2) show that LVW loss outperforms other loss functions in restoring texture details and achieving more accurate color correction.

Comparing the results of different loss functions, we observe that other methods struggle to fully recover the intricate structures in challenging regions, such as the striped fabric (blue area), the clothing folds (red area), and the wooden handle (green area). These losses tend to produce slight blurring or color distortions, failing to fully reconstruct the details present in the ground truth.

In contrast, LVW loss effectively enhances structural clarity and reduces pixel-level artifacts by adaptively constraining outlier pixels. The results show that LVW preserves sharper edge contrast in the striped fabric, maintains more natural and well-defined clothing folds without excessive smoothing, and achieves more accurate color reproduction in the wooden handle, outperforming other methods.

3. Mobile Deployment and Efficiency Analysis.

We evaluated the performance of various methods across four commercial smartphones. Table 2 details the specifications of the smartphones used in the benchmark tests, including their model, release year, SoC, and GPU configurations. Notably, the Honor 30 (#4, released in 2020) was included in the tests to evaluate the performance limits of older hardware, providing insights into the model's adaptability to legacy devices. All evaluations were conducted using the AI Benchmark application [17] on Android devices. Before deployment, the model was converted step-by-step from PyTorch to TFLite format (PT $\rightarrow$ONNX $\rightarrow$ TF $\rightarrow$ TFLite) to ensure compatibility. Figure 3 illustrates



(a) Charbonnier loss **(b) Robust loss** **(c) L1 loss** **(d) Ours** **(e) Ground Truth**

Figure 2. Visualization of different loss functions.

the overall evaluation workflow using the AI Benchmark application.

Phone Model	Launch	System on Chip (SoC)	GPU
(# 1) Xiaomi 14	10/2023	Qualcomm Snapdragon 8 Gen 3	Adreno 750
(# 2) Redmi K70 Ultra	07/2024	MediaTek Dimensity 9300	Immortalis-G720 MC12
(# 3) Nubia Z50	12/2022	Qualcomm Snapdragon 8 Gen 2	Adreno 740
(# 4) Honor 30	04/2020	Kirin 985	Mali-G77

Table 2. Description of the selected commercial smartphone devices. In the main text, (# 1) Xiaomi 14 is selected as the mobile device for testing purposes.

As illustrated in Tables 3, 4, and 5, the proposed MobileIE exhibits exceptional performance across various commercial mobile devices. Remarkably, even on older hardware (# 4: Honor 30, released in 2020), it achieves inference times below 100 ms per image. We adopted the $SCORE$ [20] to comprehensively evaluate the trade-off between model performance and efficiency. The $SCORE$ is formally defined as follows:

$$SCORE = \frac{2^{2 \cdot PSNR}}{C \cdot latency} \tag{9}$$

where $PSNR$ reflects the improvement in image quality, while $latency$ measures the inference speed. The constant C serves as a normalization factor.

MobileIE strikes an impressive balance between performance and speed, demonstrating its effectiveness not only on advanced devices but also on older hardware. This highlights its remarkable compatibility across diverse hardware platforms and underscores its practicality and superiority for commercial applications.

Figure 3. AI Benchmark [17] performance evaluation on LLE.

4. More Visual Results

In this section, we provide more visual comparisons: LLE (Figure 4 and 5), UIE (Figure 6), and ISP (Figure 7).

5. Limitations

CNN-based methods achieve superior performance by dynamically learning enhancement strategies, but their computational complexity increases quadratically with image resolution. In contrast, 3DLUT-based methods rely on efficient lookup and interpolation operations, making their inference time less sensitive to image size. This results in superior efficiency for large images but limits their adaptability to diverse tasks, as LUTs must be specifically designed for each enhancement scenario. Moreover, 3DLUTs rely on fixed mappings, often resulting in weaker capability.

Future work could explore hybrid approaches that combine CNN adaptability with 3DLUT efficiency, using techniques like pruning, quantization, and knowledge distillation to enhance speed without compromising versatility.

Method	Venue	PSNR↑	PSNR↑	Latancy↓ (ms, size: 600×400)								
Name	Year	(LOLv1 [36])	(LOLv2 [38])	Phone # 2↓	$SCORE_{v1}$↑	$SCORE_{v2}$↑	Phone # 3↓	$SCORE_{v1}$↑	$SCORE_{v2}$↑	Phone # 4↓	$SCORE_{v1}$↑	$SCORE_{v2}$↑
Kind++ [43]	IJCV'21	17.75	17.66	-	-	-	-	-	-	-	-	-
DDNet [31]	IEEE TITS'24	21.82	23.02	-	-	-	-	-	-	-	-	-
PairLIE [12]	CVPR'23	19.51	19.88	-	-	-	-	-	-	-	-	-
IAT [5]	BMVC'22	23.38	25.46	386.1	3.09	55.18	282.3	4.22	75.46	-	-	-
Zero-DCE [14]	CVPR'20	14.86	18.06	135.4	-	0.01	98.8	-	0.01	-	-	-
3DLUT [39]	IEEE TPAMI'20	17.59	19.68	/	/	/	/	/	/	/	/	/
Zero-DCE++ [24]	IEEE TPAMI'21	14.68	17.23	78.6	-	-	74.0	-	-	490.6	-	-
SCI [27]	CVPR'22	14.90	17.30	184.3	-	-	131.3	-	-	-	-	-
SGZ [47]	WACV'22	15.28	17.34	84.5	-	-	72.5	-	-	486.4	-	-
RUAS [25]	CVPR'21	16.40	15.33	173.2	-	-	128.1	-	-	-	-	-
SYELLE [13]	ICCV'23	21.03	21.26	17.5	2.62	3.6	9.8	4.68	6.44	88.8	0.52	0.71
Adv-LIE [35]	MMM'24	23.02	21.95	-	-	-	-	-	-	-	-	-
Ours	/	23.62	25.08	13.9	119.58	905.0	7.8	213.09	1612.76	72.6	22.89	173.27

Table 3. Comprehensive efficiency comparison of low-light enhancement methods on LOLv1, v2 Datasets. "-" donates $latency > 500$ ms and $SCORE < 0.01$. The top results are marked: best in red and second in blue.

Method	Venue	PSNR↑	Latancy↓ (ms, size: 640x480)					
Name	Year	UIEB [23]	Phone # 2↓	SCORE↑	Phone # 3↓	SCORE↑	Phone # 4↓	SCORE↑
FUnIE-GAN [21]	IEEE RA-L'20	19.72	109.2	0.07	95.7	0.08	404.5	0.02
Shallow-UWNet [29]	AAAI'21	16.69	-	-	499.1	-	-	-
PUIE [11]	ECCV'22	21.25	-	-	-	-	-	-
UIE-WD [28]	ICASSP'22	20.92	-	-	474.5	0.08	-	-
U-Shape [30]	IEEE TIP'23	21.25	-	-	-	-	-	-
FiveA+ [22]	BMVC'23	22.51	-	-	-	-	-	-
Boths [26]	IEEE GRSL'23	22.23	85.2	2.84	62.1	3.90	293.4	0.82
SFGNet [46]	ICASSP'24	21.66	-	-	-	-	-	-
LiteEnhanceNet [42]	ESWA'24	21.44	257.3	0.31	172.3	0.47	-	-
LSNet [48]	Arxiv'24	19.24	-	-	-	-	-	-
Ours	/	22.81	25.1	21.54	11.4	47.43	93.6	5.78

Table 4. Comprehensive efficiency comparison of underwater image enhancement methods on UIEB Dataset. "-" donates $latency > 500$ ms and $SCORE < 0.01$. The top results are marked: best in red and second in blue.

Figure 4. Visualization comparison of different Low-Light Image Enhancement models on LOLv1 [36] dataset.

Method	Venue	PSNR	Latancy (ms, size: 448×448)					
Name	Year	ZRR [19]	Phone # 2↓	SCORE↑	Phone # 3↓	SCORE↑	Phone # 4↓	SCORE↑
PyNet [19]	CVPRW'20	21.19	-	-	-	-	-	-
AWNet(raw) [6]	ECCVW'20	21.42	-	-	-	-	-	-
MW-ISP [18]	ECCVW'20	21.16	/	/	/	/	/	/
LiteISP [44]	ICCV'21	21.28	-	-	-	-	-	-
NAFNet [3]	ECCV'22	21.12	98.6	0.53	89.8	0.58	196.5	0.26
SYEISP [13]	ICCV'23	20.84	19.3	1.83	17.7	1.99	51.2	0.69
FourierISP [15]	AAAI'24	21.65	-	-	-	-	-	-
Ours	/	21.43	17.5	4.56	17.2	4.64	45.1	1.77

Table 5. Comprehensive efficiency comparison of image signal processing Methods on ZRR Dataset. "-" donates $latency > 500$ ms and $SCORE < 0.01$. The top results are marked: best in red and second in blue.

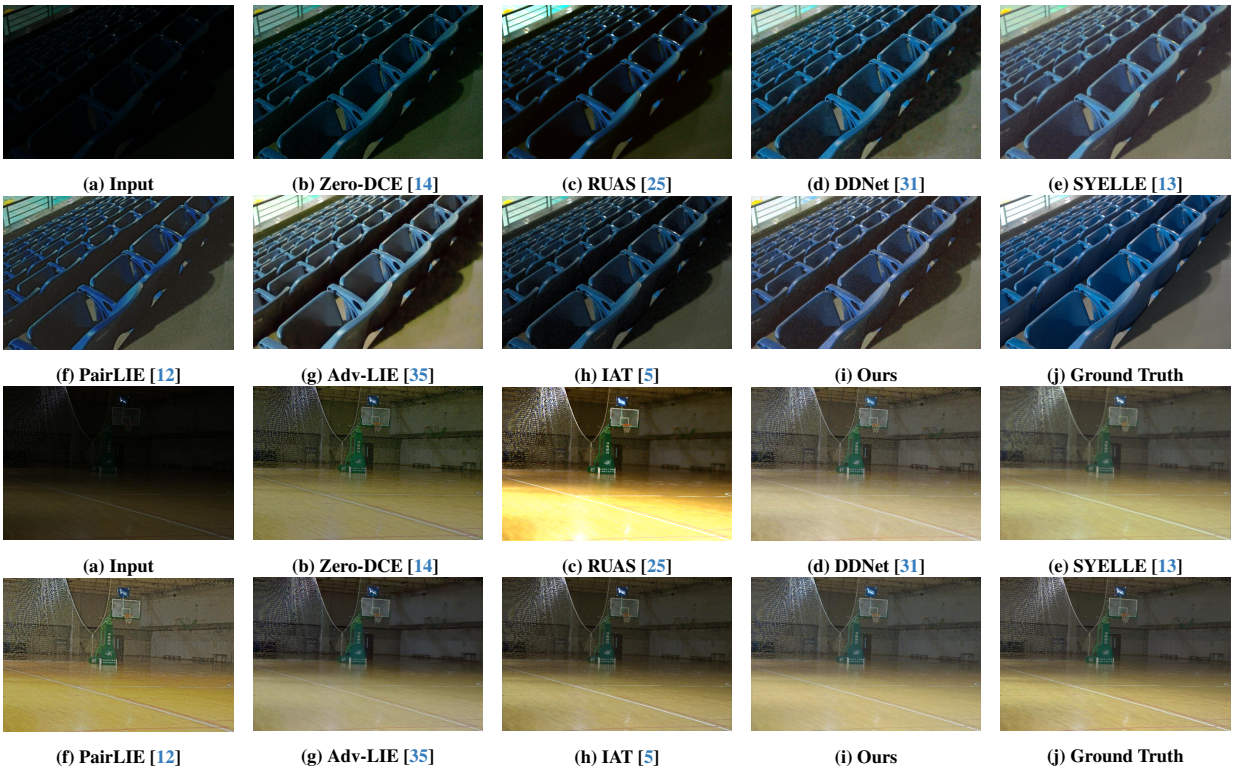


Figure 5. Visualization comparison of different Low-Light Image Enhancement models on LOLv2 [38] dataset.

References

[1] Lusine Abrahamyan, Valentin Ziatchin, Yiming Chen, and Nikos Deligiannis. Bias loss for mobile neural networks. In *ICCV*, pages 6556–6566, 2021. 1

[2] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *CVPR*, pages 12299–12310, 2021. 1

[3] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. In *ECCV*, pages 17–33. Springer, 2022. 1, 6

[4] Haram Choi, Cheolwoong Na, Jihyeon Oh, Seungjae Lee, Jinseop Kim, Subeen Choe, Jeongmin Lee, Taehoon Kim, and Jihoon Yang. Reciprocal attention mixing transformer for lightweight image restoration. In *CVPR*, pages 5992–6002, 2024. 1

[5] Ziteng Cui, Kunchang Li, Lin Gu, Shenghan Su, Peng Gao, Zhengkai Jiang, Yu Qiao, and Tatsuya Harada. You only need 90k parameters to adapt light: a light weight transformer for image enhancement and exposure correction. In *BMVC*, 2022. 5, 6, 7

[6] Linhui Dai, Xiaohong Liu, Chengqi Li, and Jun Chen. Awnet: Attentive wavelet network for image isp. In *ECCVW*, pages 185–201. Springer, 2020. 6, 8

[7] Xiaohan Ding, Yuchen Guo, Guiguang Ding, and Jungong Han. Acnet: Strengthening the kernel skeletons for power-

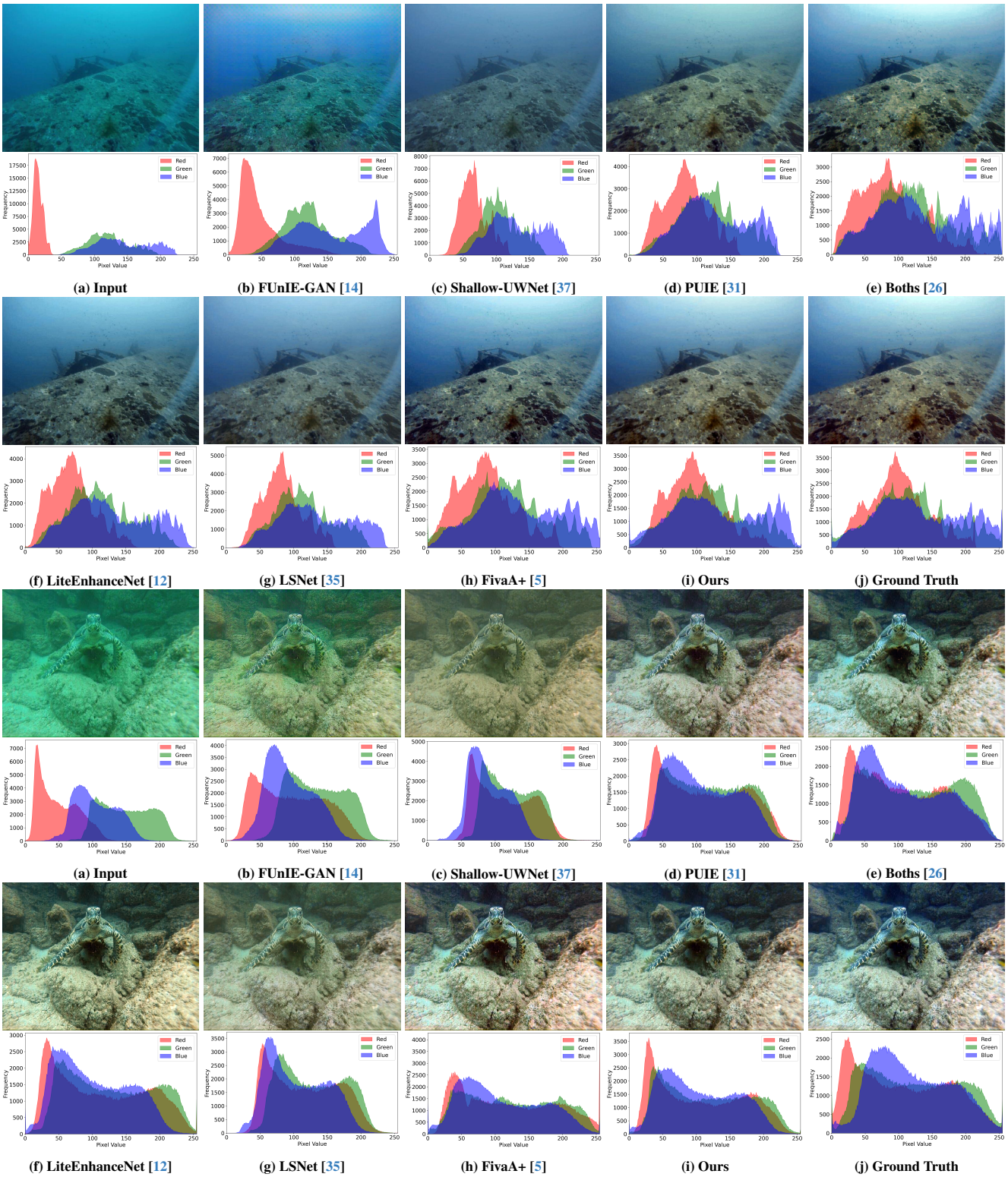


Figure 6. Visualization comparison of different Underwater Image Enhancement models on UIEB [23] dataset.

ful cnn via asymmetric convolution blocks. In *ICCV*, pages 1911–1920, 2019. 1

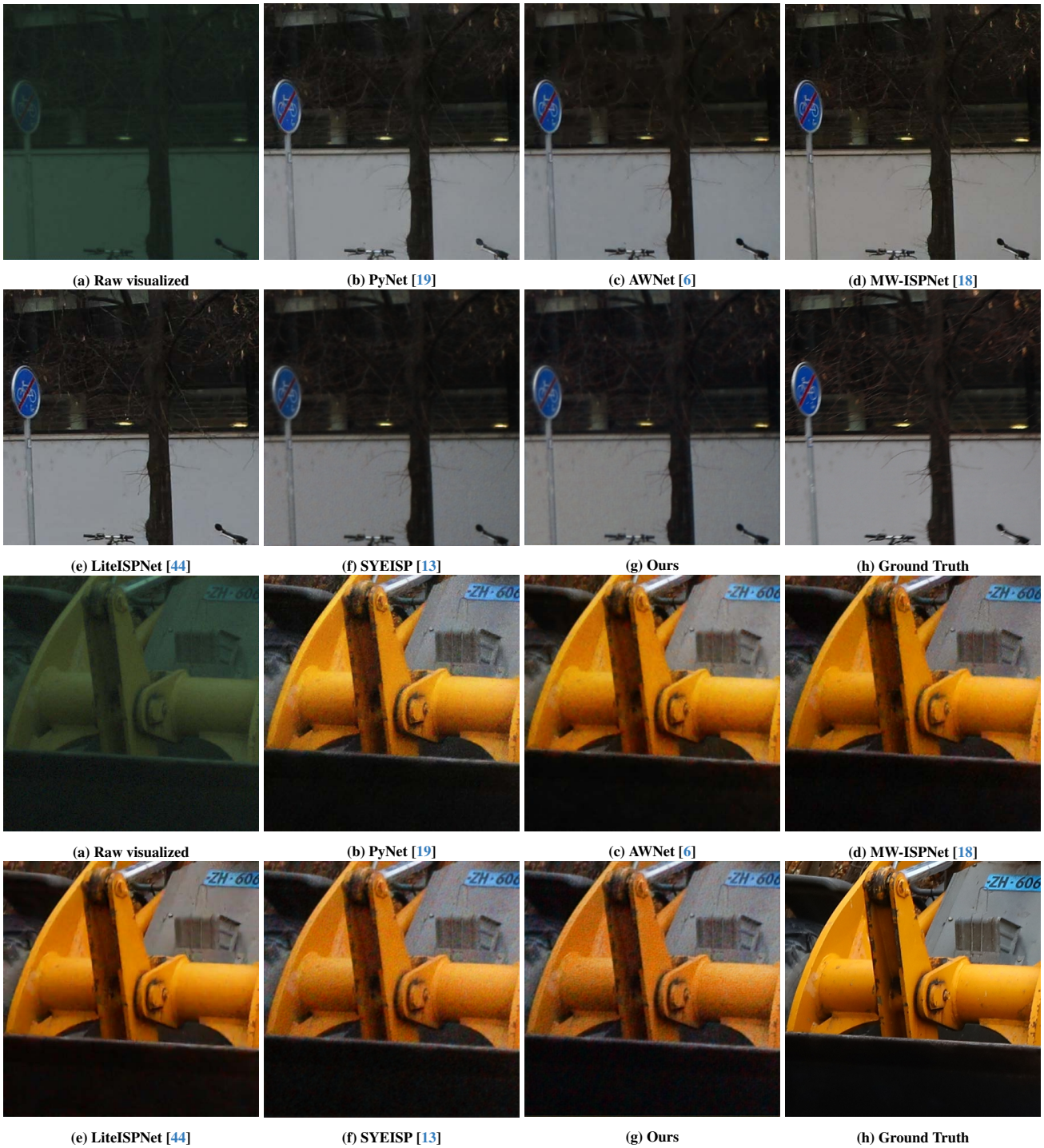


Figure 7. Visualization comparison of different Image Signal Processing models on ZRR [19] dataset.

[8] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Diverse branch block: Building a convolution as an inception-like unit. In *CVPR*, pages 10886–10895, 2021.

[9] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *CVPR*, pages 13733–13742, 2021. 1, 2

[10] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *CVPR*, pages 11963–11975, 2022.

2

[11] Zhenqi Fu, Wu Wang, Yue Huang, Xinghao Ding, and Kai-Kuang Ma. Uncertainty inspired underwater image enhancement. In *ECCV*, pages 465–482. Springer, 2022. 5

[12] Zhenqi Fu, Yan Yang, Xiaotong Tu, Yue Huang, Xinghao Ding, and Kai-Kuang Ma. Learning a simple low-light image enhancer from paired low-light instances. In *CVPR*, pages 22252–22261, 2023. 5, 6, 7

[13] Weiran Gou, Ziyao Yi, Yan Xiang, Shaoqing Li, Zibin Liu, Dehui Kong, and Ke Xu. Syenet: A simple yet effective network for multiple low-level vision tasks with real-time performance on mobile device. In *ICCV*, pages 12182–12195, 2023. 5, 6, 8

[14] Chunle Guo, Chongyi Li, Jichang Guo, Chen Change Loy, Junhui Hou, Sam Kwong, and Runmin Cong. Zero-reference deep curve estimation for low-light image enhancement. In *CVPR*, pages 1780–1789, 2020. 5, 6, 7

[15] Xuanhua He, Tao Hu, Guoli Wang, Zejin Wang, Run Wang, Qian Zhang, Keyu Yan, Ziyi Chen, Rui Li, Chengjun Xie, et al. Enhancing raw-to-srgb with decoupled style structure in fourier domain. In *AAAI*, pages 2130–2138, 2024. 6

[16] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *ICCV*, pages 1314–1324, 2019. 1, 2

[17] Andrey Ignatov, Radu Timofte, William Chou, Ke Wang, Max Wu, Tim Hartley, and Luc Van Gool. Ai benchmark: Running deep neural networks on android smartphones. In *ECCVW*, pages 0–0, 2018. 3, 4

[18] Andrey Ignatov, Radu Timofte, Zhilu Zhang, Ming Liu, Haolin Wang, Wangmeng Zuo, Jiawei Zhang, Ruimao Zhang, Zhanglin Peng, Sijie Ren, et al. Aim 2020 challenge on learned image signal processing pipeline. In *ECCVW*, pages 152–170. Springer, 2020. 6, 8

[19] Andrey Ignatov, Luc Van Gool, and Radu Timofte. Replacing mobile camera isp with a single deep learning model. In *CVPRW*, pages 536–537, 2020. 6, 8

[20] Andrey Ignatov, Radu Timofte, Shuai Liu, Chaoyu Feng, Furui Bai, Xiaotao Wang, Lei Lei, Ziyao Yi, Yan Xiang, Zibin Liu, et al. Learned smartphone isp on mobile gpus with deep learning, mobile ai & aim 2022 challenge: report. In *ECCVW*, pages 44–70. Springer, 2022. 4

[21] Md Jahidul Islam, Youya Xia, and Junaed Sattar. Fast underwater image enhancement for improved visual perception. *IEEE Robotics and Automation Letters*, 5(2):3227–3234, 2020. 5

[22] Jingxia Jiang, Tian Ye, Jinbin Bai, Sixiang Chen, Wenhao Chai, Shi Jun, Yun Liu, and Erkang Chen. Five a+ network: You only need 9k parameters for underwater image enhancement. In *BMVC*, 2023. 5

[23] Chongyi Li, Chunle Guo, Wenqi Ren, Runmin Cong, Junhui Hou, Sam Kwong, and Dacheng Tao. An underwater image enhancement benchmark dataset and beyond. *IEEE TIP*, 29: 4376–4389, 2019. 5, 7

[24] Chongyi Li, Chunle Guo, and Chen Change Loy. Learning to enhance low-light image via zero-reference deep curve estimation. *IEEE TPAMI*, 44(8):4225–4238, 2021. 5

[25] Risheng Liu, Long Ma, Jiaao Zhang, Xin Fan, and Zhongxuan Luo. Retinex-inspired unrolling with cooperative prior architecture search for low-light image enhancement. In *CVPR*, pages 10561–10570, 2021. 1, 5, 6

[26] Xu Liu, Sen Lin, Kaichen Chi, Zhiyong Tao, and Yang Zhao. Boths: Super lightweight network-enabled underwater image enhancement. *IEEE Geoscience and Remote Sensing Letters*, 20:1–5, 2023. 5, 7

[27] Long Ma, Tengyu Ma, Risheng Liu, Xin Fan, and Zhongxuan Luo. Toward fast, flexible, and robust low-light image enhancement. In *CVPR*, pages 5637–5646, 2022. 5

[28] Ziyin Ma and Changjae Oh. A wavelet-based dual-stream network for underwater image enhancement. In *ICASSP*, pages 2769–2773, 2022. 5

[29] Ankita Naik, Apurva Swarnakar, and Kartik Mittal. Shallow-uwnet: Compressed model for underwater image enhancement (student abstract). In *AAAI*, pages 15853–15854, 2021. 5

[30] Lintao Peng, Chunli Zhu, and Liheng Bian. U-shape transformer for underwater image enhancement. *IEEE TIP*, 32: 3066–3079, 2023. 5

[31] Jingxiang Qu, Ryan Wen Liu, Yuan Gao, Yu Guo, Fenghua Zhu, and Fei-Yue Wang. Double domain guided real-time low-light image enhancement for ultra-high-definition transportation surveillance. *IEEE Transactions on Intelligent Transportation Systems*, 2024. 5, 6, 7

[32] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Gautier Izacard, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, et al. Resmlp: Feedforward networks for image classification with data-efficient training. *IEEE TPAMI*, 45(4):5314–5321, 2022. 1

[33] Pavan Kumar Anasosalu Vasu, James Gabriel, Jeff Zhu, Oncel Tuzel, and Anurag Ranjan. Mobileone: An improved one millisecond mobile backbone. In *CVPR*, pages 7907–7917, 2023. 1

[34] Qilong Wang, Banggu Wu, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. Eca-net: Efficient channel attention for deep convolutional neural networks. In *CVPR*, pages 11534–11542, 2020. 1

[35] William Y Wang, Lisa Liu, and Pingping Cai. Adversarially regularized low-light image enhancement. In *International Conference on Multimedia Modeling*, pages 230–243. Springer, 2024. 5, 6, 7

[36] Chen Wei, Wenjing Wang, Wenhan Yang, and Jiaying Liu. Deep retinex decomposition for low-light enhancement. In *BMVC*, 2018. 5

[37] Wenhui Wu, Jian Weng, Pingping Zhang, Xu Wang, Wenhan Yang, and Jianmin Jiang. Uretinex-net: Retinex-based deep unfolding network for low-light image enhancement. In *CVPR*, pages 5901–5910, 2022. 7

[38] Wenhan Yang, Shiqi Wang, Yuming Fang, Yue Wang, and Jiaying Liu. From fidelity to perceptual quality: A semi-supervised approach for low-light image enhancement. In *CVPR*, pages 3063–3072, 2020. 5, 6

[39] Hui Zeng, Jianrui Cai, Lida Li, Zisheng Cao, and Lei Zhang. Learning image-adaptive 3d lookup tables for high perfor-

mance photo enhancement in real-time. *IEEE TPAMI*, 44 (4):2058–2073, 2020. 5

[40] Hu Zhang, Keke Zu, Jian Lu, Yuru Zou, and Deyu Meng. Epsanet: An efficient pyramid squeeze attention block on convolutional neural network. In *ACCV*, pages 1161–1177, 2022. 1

[41] Jiangning Zhang, Xiangtai Li, Jian Li, Liang Liu, Zhucun Xue, Boshen Zhang, Zhengkai Jiang, Tianxin Huang, Yabiao Wang, and Chengjie Wang. Rethinking mobile block for efficient attention-based models. In *ICCV*, pages 1389–1400, 2023. 1

[42] Song Zhang, Shili Zhao, Dong An, Daoliang Li, and Ran Zhao. Liteenhancenet: A lightweight network for real-time single underwater image enhancement. *Expert Systems with Applications*, 240:122546, 2024. 5

[43] Yonghua Zhang, Xiaojie Guo, Jiayi Ma, Wei Liu, and Jiawan Zhang. Beyond brightening low-light images. *IJCV*, 129: 1013–1037, 2021. 5

[44] Zhilu Zhang, Haolin Wang, Ming Liu, Ruohao Wang, Jiawei Zhang, and Wangmeng Zuo. Learning raw-to-srgb mappings with inaccurately aligned supervision. In *ICCV*, pages 4348–4358, 2021. 6, 8

[45] Chen Zhao, Weiling Cai, Chenyu Dong, and Chengwei Hu. Wavelet-based fourier information interaction with frequency diffusion adjustment for underwater image restoration. In *CVPR*, pages 8281–8291, 2024. 1

[46] Chen Zhao, Weiling Cai, Chenyu Dong, and Ziqi Zeng. Toward sufficient spatial-frequency interaction for gradient-aware underwater image enhancement. In *ICASSP*, pages 3220–3224, 2024. 5

[47] Shen Zheng and Gaurav Gupta. Semantic-guided zero-shot learning for low-light image/video enhancement. In *WACV*, pages 581–590, 2022. 5

[48] Fuheng Zhou, Dikai Wei, Ye Fan, Yulong Huang, and Yonggang Zhang. A 7k parameter model for underwater image enhancement based on transmission map prior. *arXiv preprint arXiv:2405.16197*, 2024. 5