

LogicIR: Logic Gate Networks for Image Restoration

Hongjae Lee[✉], Myungjun Son[✉], Jaeseong Yu[✉], and Seung-Won Jung^{*✉}

Korea University, Seoul, Korea
{jimmy9704, sonbill, jsyu624, swjung83}@korea.ac.kr

Abstract. Image restoration aims to reconstruct high-quality images from degraded low-quality inputs. As the computational demands of image restoration models continue to rise, there is growing interest in lightweight architectures optimized for fast and efficient inference. Logic gate networks (LGNs), which operate using fundamental logic operations such as NAND and XOR, have recently emerged as a promising direction for achieving highly efficient computation. However, their potential remains largely untapped in the domain of image restoration. In this work, we introduce LogicIR, the first LGN specifically designed for image restoration tasks. LogicIR incorporates a UNet-inspired architecture composed entirely of logic gates. In addition, we propose a differentiable bit decoding layer and an index shuffling mechanism that improves information propagation across logic gates. Experimental results across multiple image restoration benchmarks demonstrate that LogicIR achieves strong performance with significantly reduced computational cost, establishing LogicIR as a viable and efficient alternative for image restoration. The source code is available at our project page <https://github.com/jimmy9704/LogicIR>.

Keywords: logic gate network · image restoration · binary neural network

1 Introduction

Image restoration tasks, such as denoising, deblocking, and deblurring, aim to reconstruct high-quality images from degraded low-quality inputs. Although deep neural networks (DNNs) [18, 35, 37] have led to significant improvements in image restoration, conventional DNNs typically incur substantial computational demands. Such high computational costs constrain their applicability in practical, resource-constrained environments, such as smartphones, wearable devices, and embedded systems. Consequently, extensive research has been conducted to develop computationally efficient image restoration models. Existing approaches include the use of lookup tables (LUTs) [9, 12, 16], network pruning [30, 42], quantization [7, 14], and binary neural networks (BNNs) [11, 32, 33, 40].

* Corresponding author.

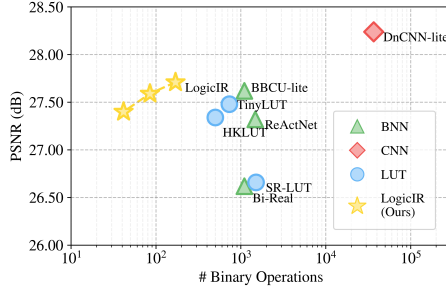


Fig. 1: Trade-off between denoising performance and binary operation count on BSD68 ($\sigma = 25$). Binary operation count is shown on a logarithmic scale.

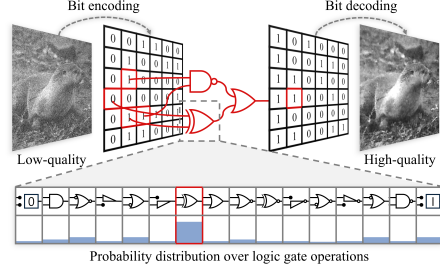


Fig. 2: Overview of LogicIR using logic gates for image restoration. Each gate softly selects among 16 logic operations during training and applies the most probable one at inference.

In particular, BNNs utilize binary weights and activations, substantially reducing memory and computational demands by replacing multiplication operations with efficient binary operations. Recently, significant efforts have been directed toward improving BNNs for image restoration. For instance, BAM [33] introduces a bit accumulation BNNs mechanism that iteratively refines binary weights and activations to approximate full-precision convolutions, while BBCU [32] proposes a residual alignment strategy to mitigate the value-range mismatch between binary and full-precision features. In parallel, LUT-based methods [9, 12, 15, 23] accelerate inference by training a compact network with a small receptive field and encoding its input–output mappings into a precomputed LUT. However, as shown in Fig. 1, BNN-based methods still rely on full-precision operations in several layers and involve a large number of trainable parameters, leading to high computational cost, whereas LUT-based approaches suffer from the exponential growth of LUT size, which restricts receptive field expansion and limits restoration performance.

To overcome these limitations, we explore logic gate networks (LGNs) [1, 24, 25, 34], which have recently emerged as a promising paradigm for efficient computation, and extend them to image restoration for the first time. LGNs perform inference exclusively using discrete logic operations, such as NAND and XOR, thereby eliminating the need for trainable weights. By operating directly at the logic-gate level, LGNs achieve exceptionally high computational efficiency and inference speed, especially when deployed on hardware platforms optimized for logic operations, such as FPGAs and ASICs. Furthermore, the explicit logical structure of LGNs improves the interpretability of their decision-making processes. However, LGNs were mainly developed for classification tasks, and thus lack the spatial modeling and hierarchical representation capabilities required for image restoration.

To this end, we propose LogicIR, the first LGN specifically designed for image restoration. As shown in Fig. 1, LogicIR significantly reduces computa-

tional complexity, measured by binary operation count, compared to existing lightweight approaches, while maintaining competitive performance. Unlike conventional methods that rely on arithmetic operations, LogicIR performs image restoration solely through logic operations, as depicted in Fig. 2. To enable this, we design a UNet-inspired architecture composed entirely of logic gates, allowing hierarchical feature representation. Furthermore, we introduce a differentiable bit decoding layer that facilitates effective training of the logic gate-based image restoration models. Additionally, we introduce a novel index shuffling method to enhance information flow across logic gates. As a result, our approach achieves 27.71 dB in PSNR using only 169.3 G binary operations for denoising, demonstrating outstanding computational efficiency and competitive performance.

2 Related Work

Binarized neural networks for image restoration. Recent studies have explored the binarization of neural networks to enable efficient image restoration [11, 32, 33, 40]. BTM [11] proposed a BNN for super-resolution by removing batch normalization. Building upon this, BBCU [32] extended the approach to broader restoration tasks, such as denoising and deblocking, by introducing a basic binary convolution unit. FRBC [40] enhanced the effectiveness of binarized networks through a distillation-guided training strategy that aligns the representations of the binary and full-precision models. However, these approaches still require storage for weight parameters and rely on full-precision adders within residual blocks, as well as non-binarized layers at the input and output, which hinder deployment on resource-constrained hardware platforms, such as FPGAs and ASICs.

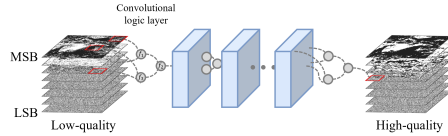
Look-up tables for image restoration. Recent research has investigated the use of look-up table (LUT)-based methods to accelerate image restoration [9, 12, 15–17, 23]. SR-LUT [12] proposed a practical approach for image super-resolution by training a deep network with a small receptive field and encoding its input-output mappings into an LUT for efficient inference. MuLUT [16] extended this approach by leveraging multiple small LUTs to effectively increase the receptive field and generalized the method to various image restoration tasks. HKLUT [9] further reduced the LUT size by allocating different LUT resolutions based on the significance of bit planes in 8-bit input images. Despite these advancements, LUT-based methods suffer from a fundamental limitation: the size of the LUT grows exponentially with the receptive field. This imposes a practical constraint on the receptive field size, making it difficult for these methods to capture global spatial patterns and long-range dependencies.

3 Background

Logic gate networks. LGNs [1, 24, 25, 34] are composed of binary logic gates such as AND, NAND, and XOR, operating at the lowest abstraction level of

Table 1: Logic gate network-based methods and other approaches on BSD68 ($\sigma = 25$). BOPs: binary operations.

Method	BOPs	RF size	PSNR
DnCNN-lite [37]	36 562.6 G	35×35	28.24
BBCU-lite [32]	1 097.2 G	35×35	27.62
HKLUT [9]	499.4 G	5×5	27.34
StackedCLGN	39.7 G	25×25	17.19
LogicIR-S	41.4 G	47×47	27.40
LogicIR-S-4RT	169.3 G	47×47	27.71

**Fig. 3:** Overview of StackedCLGN, which stacks multiple convolutional logic layers. Each gray node represents a logic gate within a layer. The model processes 8-bit planes as input and output.

digital circuits. One of the key advantages of LGNs is their ability to perform inference using only simple logic operations, without relying on complex floating-point arithmetic, making them highly efficient for deployment on hardware platforms such as FPGAs and ASICs. Training LGNs involves a combinatorial optimization problem, as it requires selecting a logic gate for each node and defining the overall connection pattern. However, their non-differentiable nature precludes the use of gradient-based optimization methods. To overcome this limitation, a differentiable relaxation method has been recently introduced, enabling end-to-end training by learning probability distributions over logic gate selections [24]. After training, the most probable gate is selected at each node, resulting in a discretized logic circuit well-suited for efficient hardware implementation.

Convolutional logic gate networks. Convolutional LGNs (CLGNs) [25] address a critical limitation of conventional LGNs, which operate solely on flattened inputs and thus fail to capture spatial structure. To overcome this, CLGNs incorporate convolutional logic layers that employ hierarchical logic gate trees to extract local spatial correlations through localized logic operations. This design significantly enhances the network’s ability to exploit spatial patterns, leading to improvements in both accuracy and scalability for classification tasks. Each kernel in a convolutional logic layer is constructed as a complete binary tree of depth d , where each internal node performs a logic gate operation (*e.g.*, NAND, XOR), and each leaf node receives inputs from a predefined receptive field of size $s_c \times s_k \times s_k$, with s_c denoting the number of input channels and s_k the kernel size. These connections are randomly initialized and remain fixed throughout training and inference.

For example, a logic tree kernel of depth $d = 2$ requires three logic gates: two at the first layer and one at the second layer, as illustrated in Fig. 2. Given four input binary activations, a_1, a_2, a_3 , and a_4 , sampled from a local receptive field, the output activation $\hat{a}$ is computed as:

$$\hat{a} = f_3(f_1(a_1, a_2), f_2(a_3, a_4)), \quad (1)$$

where each logic gate f_i is selected from a softmax-based probability distribution over a predefined set of logic operations [24, 25]. This formulation allows the

CLGN to perform localized spatial reasoning similar to conventional convolutional neural networks (CNNs), while retaining the efficiency and interpretability of fully logic-based computation.

Limitations. While CLGNs can effectively extract local spatial patterns from images, they remain insufficient for image restoration tasks for several reasons. First, they lack a bit decoding mechanism required to reconstruct images from features extracted through logic operations. Second, simply stacking convolutional logic layers cannot effectively capture the hierarchical representations crucial for high-quality image restoration. Finally, image restoration tasks inherently require a strong locality inductive bias [3, 8], rendering randomly connected logic nodes with large receptive fields less effective. We address these limitations by introducing a UNet-based architecture, a differentiable bit decoding module, and an index shuffling mechanism, which together enable effective logic gate-based image restoration.

4 Logic Gate Networks for Image Restoration

Image restoration aims to reconstruct a high-quality image from a degraded low-quality observation. Recent advances in CNNs have led to significant improvements in image restoration, primarily through the use of convolutional layers to extract complex feature representations. A straightforward way to apply LGNs to image restoration is to build a network by serially stacking multiple convolutional logic layers, a baseline we refer to as StackedCLGN, as shown in Fig. 3. In this design, both the low-quality input and high-quality output images are converted into 8-channel binary representations corresponding to their bit planes. All convolutional logic kernels in StackedCLGN use a fixed tree depth of $d = 3$.

While StackedCLGN can extract local spatial features in a manner similar to CNNs and rely solely on logic operations for inference, it lacks key architectural components required for reconstructing high-quality images from degraded inputs. Specifically, it does not incorporate hierarchical feature representations, skip connections, and a dedicated bit decoding module for pixel value reconstruction, all of which are essential for effective image restoration. As shown in Tab. 1, StackedCLGN significantly underperforms compared to CNN-based models such as DnCNN-lite [37], and even BNN-based models like BBCU-lite [32], which still benefit from residual connections and hierarchical feature representations. These limitations motivate the development of LogicIR, an LGN architecture tailored for image restoration.

4.1 LogicIR Architecture

The overall architecture of LogicIR, illustrated in Fig. 4, consists of two main components: a UNet-style [4, 27] encoder-decoder composed of convolutional logic layers and a bit decoder that reconstructs the residual image from bit predictions. Specifically, the low-quality input image $\mathbf{I}_{LQ} \in \mathbb{R}^{H \times W}$ is first converted into an 8-channel binary representation, denoted as $\mathbf{B}_{LQ} \in \{0, 1\}^{8 \times H \times W}$.

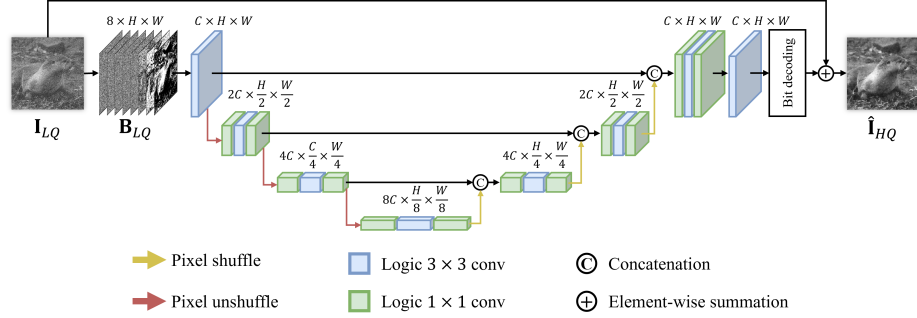


Fig. 4: LogicIR architecture featuring a UNet-style encoder-decoder with convolutional logic layers. The model receives an 8-channel binary input and reconstructs the residual image using a differentiable bit decoding module.

Each block in the encoder-decoder, except for the first and the last, comprises a 1×1 convolutional logic layer to capture inter-channel dependencies, followed by 3×3 and 1×1 convolutional logic layers to extract and refine spatial features. All convolutional logic layers throughout the network operate with a fixed tree depth of $d = 3$. Beyond simply replacing conventional convolutional layers with logic-based alternatives, LogicIR incorporates several architectural components specifically considered for LGN-based image restoration.

First, LogicIR avoids the use of addition operations commonly employed in conventional residual blocks [6, 20]. To preserve high-frequency details and enable deeper training, it uses skip connections between the encoder and decoder stages via channel-wise concatenation. Second, for spatial resolution changes, LogicIR replaces conventional max pooling and transposed convolution operations with pixel unshuffle and pixel shuffle, respectively. These operations are not only well-suited for logic-based implementations, but also more effective in preserving spatial information. For example, pixel unshuffle transforms a feature map of shape $C \times H \times W$ into $Cr^2 \times H/r \times W/r$, where the scale factor r is set to 2. Finally, following the common practice in image restoration [4, 37, 41], LogicIR is trained to predict a residual image $\mathbf{R} \in \mathbb{R}^{H \times W}$ rather than directly reconstructing the high-quality output image. The final output is then obtained by adding the predicted residual to the input low-quality image, i.e., $\hat{\mathbf{I}}_{HQ} = \mathbf{I}_{LQ} + \mathbf{R}$.

In addition to these design considerations, we introduce two novel components tailored to enhance image restoration with LGNs, which will be described in the following subsections.

4.2 Bit Decoding

Although LogicIR can be configured to directly produce bit-planes representing pixel values, we found that this output format significantly complicates the training process. In particular, the less significant bit planes are often highly noisy and lack semantic consistency, which hinders the learning of meaning-

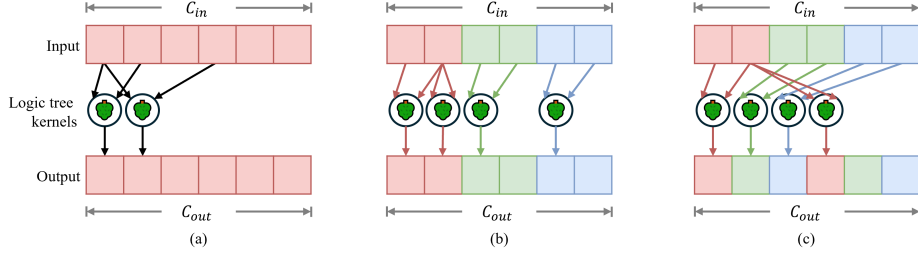


Fig. 5: Index shuffling for convolutional logic layers with logic tree depth $d = 1$. (a) Randomly connected without grouping. (b) Grouped logic layer with 3 groups, where each logic tree connects within a group. (c) Index shuffling cyclically reorders input indices.

ful gradients across all bits during pixel-value regression. To address this issue, we instead design LogicIR to produce a multi-channel binary output, which is subsequently transformed into a single-channel residual image using a bitcount operation defined as:

$$\bar{\mathbf{R}}(:, :) = \frac{\sum_{c=0}^{C-1} \mathbf{A}(c, :, :) - 0.5C}{0.5C}, \quad (2)$$

where $\mathbf{A} \in \{0, 1\}^{C \times H \times W}$ denotes the final output activation map produced by the UNet decoder, and $\bar{\mathbf{R}} \in [-1, 1]^{H \times W}$ is the resulting normalized residual image. This formulation allows the logic-based network to learn in a more noise-robust manner. However, restricting the residual values to a fixed range may limit the network’s representational capacity, potentially degrading performance. To mitigate this, we introduce a simple scaling mechanism that adapts the effective output range by obtaining the final residual image as $\mathbf{R} = \alpha \bar{\mathbf{R}}$, where α is a learnable scalar parameter.

4.3 Index Shuffling

A convolutional logic layer is composed of multiple logic tree kernels. Each logic tree receives its input from randomly selected positions in the input activation map, as illustrated in Fig. 5(a). CLGN [25], inspired by the grouped convolution [13], introduces a constraint on these random connections by limiting them within fixed groups (Fig. 5(b)). While this strategy enables a form of structured randomness, it also presents a limitation: group isolation restricts inter-group information flow, effectively forming independent sub-networks.

To address this issue, we draw inspiration from ShuffleNet [39] and introduce an index shuffling mechanism in LGNs. As shown in Fig. 5(c), our method achieves inter-group information exchange by reordering the input indices connected to each logic tree kernel. Specifically, let $\mathbf{A}_{in} \in \{0, 1\}^{C_{in} \times H \times W}$ denote an input activation map with C_{in} channels. The input channels are divided into

G groups, where $G = C_{in}/2^d$ and d denotes the depth of the logic tree kernel. This grouping can be expressed as:

$$\mathbf{A}_{in} = \{\mathbf{A}_{in}^0 \mid \mathbf{A}_{in}^1 \mid \cdots \mid \mathbf{A}_{in}^{G-1}\}, \quad \mathbf{A}_{in}^g \in \{0, 1\}^{2^d \times H \times W}, \quad (3)$$

where $\mid$ represents channel-wise concatenation. Let $\mathbf{A}_{out} \in \{0, 1\}^{C_{out} \times H \times W}$ denote an output activation map with C_{out} channels. Each output channel is computed by applying a logic tree kernel to a cyclically selected input group, as follows:

$$\mathbf{A}_{out}(n, :, :) = \mathcal{F}^{(n)}(\mathbf{A}_{in}^{g_{cyc}}), \quad \text{for } n = 0, 1, \dots, C_{out} - 1 \quad (4)$$

where $g_{cyc} = n \bmod G$, and $\mathcal{F}^{(n)}$ represents the logic tree kernel for the n -th output channel.

4.4 Training

Loss functions. Following prior works in image restoration [16, 37], we adopt the L2 loss, denoted as $\mathcal{L}_2$, as the training objective, which is defined as the pixel-wise difference between the reconstructed output and the ground-truth image:

$$\mathcal{L}_2 = \left\| \mathbf{I}_{HQ} - \hat{\mathbf{I}}_{HQ} \right\|_2, \quad (5)$$

where $\hat{\mathbf{I}}_{HQ}$ and $\mathbf{I}_{HQ}$ denote the predicted and ground-truth images, respectively, and $\|\cdot\|_2$ represents the L2 norm. This loss formulation encourages accurate reconstruction by penalizing large deviations in pixel intensity.

In addition to the primary reconstruction loss $\mathcal{L}_2$, we incorporate an auxiliary supervision that accounts for the varying significance of individual bit planes. Specifically, we reconstruct a reference image $\tilde{\mathbf{I}}_{HQ}$ using only the top four most significant bit (MSB) planes of the ground-truth image $\mathbf{I}_{HQ}$ and measure the MSB loss, denoted as $\mathcal{L}_{MSB}$, as follows:

$$\mathcal{L}_{MSB} = \left\| \tilde{\mathbf{I}}_{HQ} - \hat{\mathbf{I}}_{HQ} \right\|_2. \quad (6)$$

This auxiliary term encourages the model to prioritize information contained in the most informative bit planes, which are critical for preserving structural details. The final training objective is defined as a weighted sum of the two loss terms:

$$\mathcal{L}_{total} = \mathcal{L}_2 + \lambda \mathcal{L}_{MSB}, \quad (7)$$

where λ is a hyperparameter that controls the relative contribution of the MSB loss.

Fine-tuning using straight-through estimators. During training, each logic gate in the differentiable LGNs [24, 25] performs a soft selection over all 16 possible binary logic operations, with each operation weighted by its learned probability. In contrast, during inference, the gate applies a discrete selection,

choosing the operation with the highest probability. The computation of the logic gate in both training and inference phases can be expressed as:

$$\hat{a} = \begin{cases} \sum_{i=0}^{15} p_i \cdot f_i(a_1, a_2), & \text{Training,} \\ f_{i^*}(a_1, a_2), \quad \text{where } i^* = \arg \max_i p_i, & \text{Inference.} \end{cases} \quad (8)$$

Here, f_i denotes the i -th logic operation, and p_i is its corresponding learned probability. The soft formulation during training allows gradients to flow through all candidate logic operations.

However, this mismatch between soft logic selection during training and hard logic selection during inference can lead to a performance gap. To address this issue, we introduce an additional fine-tuning strategy based on the straight-through estimator (STE) [2], applied after the main training phase. Specifically, in the fine-tuning stage, hard logic operations are used in the forward pass, while gradients are computed as if the soft combination were still applied. This enables the network to better adapt to the discrete behavior used at inference time. In addition, the scaling parameter α , which modulates the range of the residual output, is also optimized during this fine-tuning stage.

Rotational ensemble. Due to the random connectivity in the logic gate network, each logic tree is exposed to a limited set of patterns within its receptive field. To address this limitation, we adopt a rotational ensemble strategy, in which the input image is rotated at multiple fixed angles, and the corresponding outputs are averaged to produce the final result. This approach increases the diversity of patterns seen by the network without modifying its structure. Formally, let $\mathcal{R}_\theta(\cdot)$ denote a rotation operation by angle θ , and let $\mathcal{G}(\cdot)$ be the LogicIR model. The final output $\hat{\mathbf{I}}$ is computed as:

$$\hat{\mathbf{I}}_{HQ} = \frac{1}{|\Theta|} \sum_{\theta \in \Theta} \mathcal{R}_{-\theta}(\mathcal{G}(\mathcal{R}_\theta(\mathbf{I}_{LQ}))), \quad (9)$$

where Θ is the set of rotation angles, and $\mathcal{R}_{-\theta}(\cdot)$ denotes the inverse rotation used to restore the original orientation. To enable this ensemble strategy, we perform additional rotation-aware training, initialized from a pre-trained model originally trained without rotation.

5 Experiments

5.1 Experiment Settings

Implementation details. Following DnCNN [37], we use 400 grayscale images from the BSD dataset [28] for training. The deraining models are trained on the Rain13K dataset [10]. When processing color images, LogicIR treats the R, G, and B channels independently and processes them in parallel. All models are trained for 8×10^4 iterations using the Adam optimizer with a learning rate of 10^{-2} . For tasks other than denoising, such as JPEG deblocking and deraining, we initialize the network using the model trained on the denoising task.

Table 2: Quantitative results for image denoising with a noise level of $\sigma = 25$ and JPEG deblocking with a quality factor of $q = 10$. BOPs (binary operation count) are measured on 1280×720 input images. BBCU-lite-fully denotes a network with all layers binarized. LogicIR-S-2RT and LogicIR-S-4RT denote models applying rotational ensembles with 2 and 4 rotations, respectively.

Method		Denoising				Deblocking		
		BOPs	BSD68	PSNR Set12	Urban100	BOPs	PSNR LIVE1	Classic5
FP	DnCNN-lite [37]	36.6 T	28.24	29.05	27.73	36.6 T	28.78	28.89
	DnCNN [37]	1 023.1 T	29.23	30.44	29.95	1 023.1 T	29.40	29.19
	SwinIR [18]	21 151.7 T	29.50	31.01	31.30	21 151.7 T	29.86	30.27
BNN	BBCU-lite-fully [32]	700.6 G	25.23	25.39	24.81	700.6 G	28.13	28.27
	Bi-Real [20]	1 095.3 G	26.62	26.96	26.11	1 095.3 G	28.31	28.39
	BBCU-lite [32]	1 097.2 G	27.62	28.22	26.84	1 097.2 G	28.43	28.57
	ReActNet [19]	1 471.4 G	27.32	27.91	26.53	1 471.4 G	28.40	28.51
	BBCU [32]	9 152.1 G	28.45	29.33	28.16	9 152.1 G	29.06	30.00
LUT	HKLUT [9]	499.4 G	27.34	27.94	26.30	502.7 G	28.54	28.65
	TinyLUT [15]	729.8 G	27.48	28.26	26.42	715.4 G	28.52	28.63
	SR-LUT [12]	1 505.5 G	26.66	27.10	25.94	1 693.6 G	28.51	28.61
Logic	LogicIR-S	41.4 G	27.40	27.83	26.57	45.2 G	28.48	28.52
	LogicIR-L	78.6 G	27.50	27.94	26.69	73.6 G	28.54	28.57
	LogicIR-S-2RT	84.7 G	27.59	28.07	26.75	92.3 G	28.57	28.61
	LogicIR-S-4RT	169.3 G	27.71	28.22	26.85	184.6 G	28.62	28.66

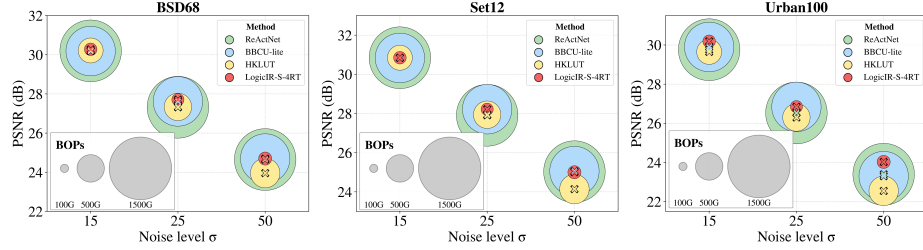


Fig. 6: Denoising performance versus complexity (BOPs) across noise levels on BSD68, Set12, and Urban100. Bubble area is proportional to BOPs, and the $\times$ marker denotes the center of each bubble.

Evaluation settings. Following BBCU [32], BBCU-lite and other BNN-based methods use DnCNN-lite as their baseline, a lightweight variant of DnCNN with 12 channels. To evaluate both image restoration quality and hardware efficiency, we adopt PSNR, CLIP-IQA [31], LPIPS [38], NIQE [21], and SSIM as image quality metrics. We report the binary operation count (BOPs) [1, 24, 25], measured on 1280×720 input images. Similar to FLOPs, it represents the hardware cost and resource usage in image processing. Details on BOPs counting are provided in prior work [1, 24, 25] and the supplementary material. In addition, Power, Area, and Runtime are measured to assess hardware efficiency.

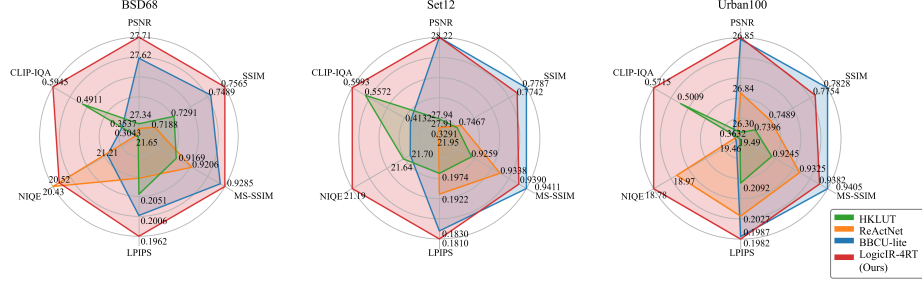


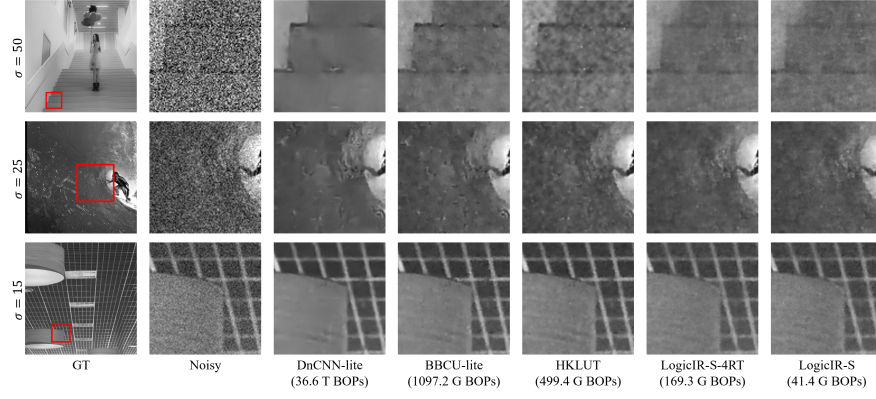
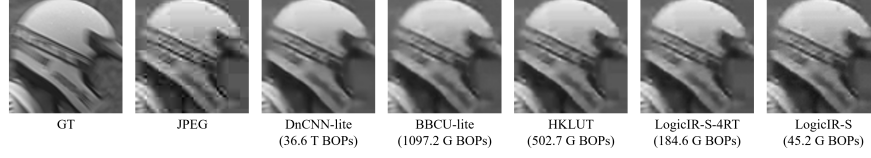
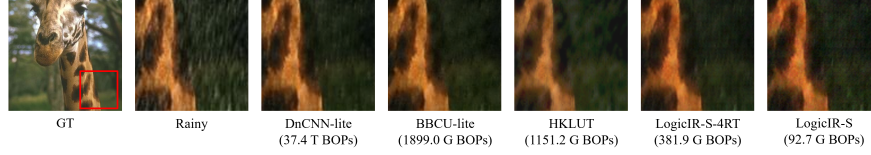
Fig. 7: Comparison of image quality metrics for various methods with a noise level of $\sigma = 25$.

5.2 Evaluation on Image Denoising

We compare LogicIR with various baseline methods, including full-precision models (DnCNN, SwinIR), BNNs (BBCU, Bi-Real, ReActNet), and LUT-based approaches (HKLUT, TinyLUT, SR-LUT). LogicIR-S and LogicIR-L refer to the small and large variants of our model, using $C = 2048$ and $C = 4096$ channels in Fig. 4, respectively.

Quantitative results. Tab. 2 presents quantitative grayscale image denoising results. LogicIR-S achieves comparable PSNR to ReActNet with only 2.8% of its BOPs, while LogicIR-L outperforms the LUT-based HKLUT using just 15.7% of its BOPs. Rotational ensembles further improve performance: LogicIR-S-4RT reaches 27.71 dB on BSD68 with 169.3 G BOPs, surpassing BBCU-lite (27.62 dB) with $6.5\times$ fewer operations. Unlike prior BNNs (e.g., Bi-Real, ReActNet, BBCU-lite) that keep the first and last layers in full precision, LogicIR is fully binarized, making it natively compatible with logic-based implementations; in contrast, fully binarized variants of those BNNs (e.g., BBCU-lite-fully) incur large performance drops. As shown in Fig. 6, across all three noise levels ($\sigma = 15, 25, 50$), LogicIR consistently achieves comparable PSNR while using the fewest BOPs among all methods.

Qualitative and perceptual evaluation. Fig. 7 compares LogicIR with other methods using distortion metrics (PSNR, SSIM, MS-SSIM) and perceptual metrics (LPIPS, NIQE, CLIP-IQA). LogicIR-S-4RT achieves superior overall quality with far fewer binary operations. LUT-based methods (e.g., HKLUT) show weaker perceptual quality due to limited receptive fields, whereas LogicIR’s hierarchical logic design effectively restores fine details with high efficiency. Qualitative examples in Fig. 8 further show that LogicIR yields visually comparable results to lightweight models such as BBCU-lite, DnCNN-lite, and HKLUT despite using fewer operations. Additional qualitative and quantitative results are provided in the supplementary material.

**Fig. 8:** Qualitative results on image denoising.**Fig. 9:** Qualitative results on JPEG deblocking.**Fig. 10:** Qualitative results on image deraining.

5.3 Evaluation on Image Deblocking

We further evaluate deblocking performance at a quality factor of $q = 10$, using the LIVE1 [29] and Classic5 [5] datasets, as presented in Tab. 2. LogicIR-S achieves competitive performance, outperforming existing BNN-based models such as ReActNet and BBCU-lite. The larger variant, LogicIR-S-4RT, incorporates four rotations, achieves a PSNR of 28.62 dB on LIVE1, outperforming the LUT-based HKLUT (28.54 dB) while using only 36.7% of its operations. These results highlight the high efficiency of LogicIR in the deblocking task. As shown in Fig. 9, LogicIR effectively reduces compression artifacts and preserves structural details, delivering visual quality on par with other methods while operating with significantly fewer operations.

5.4 Evaluation on Image Deraining

We further evaluate the performance of LogicIR on the image deraining task on the Test100 [36] dataset, as presented in Tab. 3. LogicIR-S achieves competitive

Table 3: Quantitative comparison for image deraining.

Method	BOPs	PSNR	SSIM
DnCNN-lite [37]	37.4 T	23.34	0.7828
PreNet [26]	15 025.3 T	24.81	0.8510
Bi-Real [20]	1 893.4 G	22.79	0.7410
BBCU-lite [32]	1 899.0 G	22.81	0.7647
ReActNet [19]	2 269.5 G	22.77	0.7562
HKLUT [9]	1 151.2 G	22.71	0.7422
LogicIR-S	92.7 G	22.75	0.7403
LogicIR-S-4RT	381.9 G	22.95	0.7651

Table 5: Ablation of color-processing strategies in LogicIR. Y-only uses Y in YCbCr. RGB-joint restores RGB jointly with a widened network. RGB-channel uses per-channel R/G/B with shared weights.

Method	BOPs	PSNR	SSIM
Y-only	35.9 G	22.63	0.7394
RGB-joint	169.1 G	22.79	0.7422
RGB-channel	92.7 G	22.75	0.7403

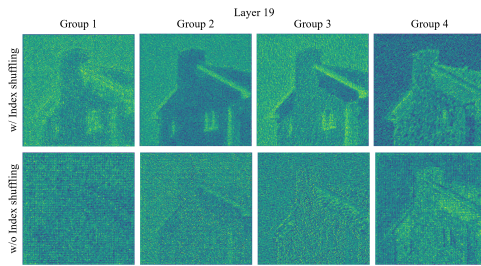
results compared to the existing LUT-based HKLUT. For instance, LogicIR-S achieves a PSNR of 22.75 dB, comparable to HKLUT (22.71 dB), while using only 8.1% of its operations. Considering an even larger variant of our model, LogicIR-S-4RT achieves a PSNR of 22.95 dB, surpassing all BNN-based methods while maintaining high computational efficiency, using only 16.8% to 20.2% of their operations. Qualitative deraining comparisons are shown in Fig. 10. We additionally provide results on another restoration task, low-light image enhancement, in the supplementary material.

5.5 Ablation Studies

Component ablation. Tab. 4 presents an ablation study on the key components of LogicIR-S. Starting from a basic stacked convolutional logic gate network, adding the bit decoding module yields a large PSNR gain by converting binary activations into continuous residuals. Incorporating the UNet backbone and index shuffling further enhances hierarchical and inter-group feature learning, while the MSB loss and STE fine-tuning provide additional improvements. In particular, Fig. 12 shows that fixed grouped logic layers quickly saturate because they repeatedly operate within the same channel groups, making the model behave like isolated sub-networks. In contrast, index shuffling redistributes fea-

Table 4: Ablation study of LogicIR components on image denoising.

Method	PSNR	bit decoding	UNet	index shuffling	MSB loss	fine-tuning
StackedCLGN	17.19	✗	✗	✗	✗	✗
+ Bit decoding	26.83	✓	✗	✗	✗	✗
+ UNet backbone	27.15	✓	✓	✗	✗	✗
+ Index shuffling	27.58	✓	✓	✓	✗	✗
+ MSB loss	27.72	✓	✓	✓	✓	✗
LogicIR-S	27.83	✓	✓	✓	✓	✓

**Fig. 11:** Group-wise averaged feature maps at layer 19 with and without index shuffling, showing enhanced representations.

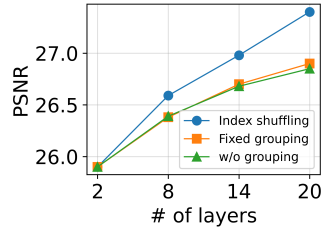


Fig. 12: Grouping ablation with varying numbers of logic layers.

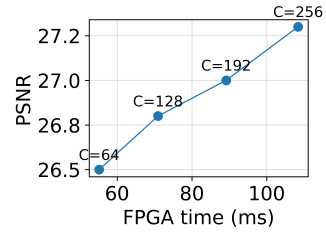


Fig. 13: FPGA runtime of LogicIR variants with different channel widths.

Table 6: Complexity analysis of the denoising models.

Method	BOPs	Energy (mJ)	Area (mm ²)	Runtime (ms)	PSNR
BBCU-lite [32]	1 097.2 G	0.69	2.44	571.6	27.62
HKLUT [9]	499.4 G	0.31	1.11	71.2	27.34
LogicIR-S	41.4 G	0.03	0.09	28.2	27.40
LogicIR-S-4RT	169.3 G	0.11	0.38	112.6	27.71

tures across groups between layers, enabling richer inter-group communication and consistent improvements as more logic layers are stacked. This is further supported by Fig. 11, where index shuffling promotes inter-group information exchange and helps each group learn more meaningful representations. Together, these components contribute complementary benefits, achieving the best overall denoising performance.

Color-processing ablation. Tab. 5 compares three color-processing strategies. Y-only restores only the luminance channel in the YCbCr domain, yielding the lowest cost (35.9G BOPs) but the lowest quality at 22.63 dB. RGB-joint widens the network from C to $3C$ to process RGB jointly, improving quality to 22.79 dB at a much higher cost of 169.1G BOPs. RGB-channel instead processes channels separately with shared parameters across R/G/B, retaining most of the gain at 22.75 dB with far fewer BOPs (92.7G). We therefore use RGB-channel by default for the best performance–efficiency trade-off.

5.6 Complexity and Hardware Deployment

Complexity analysis. To further evaluate the practical efficiency of LogicIR, we analyze not only the operation count but also latency, energy, and hardware area. As shown in Tab. 6, LogicIR-S achieves a runtime of 28.2 ms, which is $20.3\times$ faster than BBCU-lite (571.6 ms) and $2.5\times$ faster than HKLUT (71.2 ms). Even with the rotational ensemble (LogicIR-S-4RT), the runtime remains competitive at 112.6 ms, achieving higher PSNR than the BNN-based baseline. The logic-simulated runtime is measured on an RTX 4090 GPU using the Dif-Logic library [24]. Following the methodology of [16, 22], LogicIR-S consumes only 0.03 mJ, compared with 0.69 mJ for BBCU-lite and 0.31 mJ for HKLUT, demonstrating its superior energy efficiency. In terms of chip area based on the

TSMC N5 process technology, LogicIR-S requires only 0.09 mm^2 , far smaller than HKLUT (1.11 mm^2) and BBCU-lite (2.44 mm^2), supporting its practicality for edge deployment.

Practical FPGA deployment. We implement LogicIR variants of different sizes on real FPGA devices, following the FPGA implementation protocol of DiffLogic. Since LogicIR is scalable along the channel dimension, we vary the main UNet backbone width C from 64 to 256, while keeping the final layer before bit counting fixed at 2048 channels. For FPGA deployment, all variants are evaluated with fixed-size 24×24 input patches. Full images can be handled via sliding-window tiling and stitching. We verify these variants on an Intel Cyclone V C9 FPGA under the 4RT inference setting, where four rotated inputs are processed for rotational ensembling. As shown in Fig. 13, the resource usage increases gradually as C increases, indicating that channel width provides a simple and effective knob for adapting LogicIR to different hardware budgets.

6 Conclusion

In this work, we introduced LogicIR, the first logic gate network specifically designed for image restoration. Unlike conventional neural networks that rely on arithmetic operations and floating-point representations, LogicIR operates with discrete logic gates, enabling highly efficient inference. Our architecture incorporates several key components, including a UNet backbone composed of convolutional logic layers, a differentiable bit decoding module, index shuffling for inter-group communication, and MSB-plane-guided supervision. Through extensive experiments on image denoising, deblocking, and deraining tasks, LogicIR demonstrated superior performance compared to existing lightweight approaches such as BNNs and LUT-based baselines, while significantly reducing the required number of operations. These results highlight the potential of LogicIR for edge and resource-constrained deployment under tight resource budgets.

Limitations. While LogicIR demonstrates strong performance across several image restoration benchmarks, it also presents certain limitations that require further investigation. This study focuses solely on restoration tasks in which the input and output share the same spatial resolution. However, tasks such as image super-resolution involve explicit upsampling, which poses additional architectural and algorithmic challenges. Designing high-quality and scalable upsampling mechanisms within a logic gate framework remains an open and important direction for future research.

Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2025-16068196), and Samsung Electronics Company Ltd., under Grant IO230313-05434-01.

References

1. Benamira, A., Peyrin, T., Yap, T., Guérard, T., Hooi, B.: Truth table net: Scalable, compact & verifiable neural networks with a dual convolutional small boolean circuit networks form. In: IJCAI. pp. 13–21 (2024)
2. Bengio, Y., Léonard, N., Courville, A.: Estimating or propagating gradients through stochastic neurons for conditional computation. arXiv preprint arXiv:1308.3432 (2013)
3. Chen, X., Wang, X., Zhang, W., Kong, X., Qiao, Y., Zhou, J., Dong, C.: HAT: Hybrid attention transformer for image restoration. arXiv preprint arXiv:2309.05239 (2023)
4. Cho, S.J., Ji, S.W., Hong, J.P., Jung, S.W., Ko, S.J.: Rethinking coarse-to-fine approach in single image deblurring. In: ICCV. pp. 4641–4650 (2021)
5. Foi, A., Katkovnik, V., Egiazarian, K.: Pointwise shape-adaptive dct for high-quality denoising and deblocking of grayscale and color images. IEEE TIP **16**(5), 1395–1411 (2007)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
7. Hong, C., Lee, K.M.: AdaBM: on-the-fly adaptive bit mapping for image super-resolution. In: CVPR. pp. 2641–2650 (2024)
8. Hu, J., Gu, J., Yu, S., Yu, F., Li, Z., You, Z., Lu, C., Dong, C.: Interpreting low-level vision models with causal effect maps. IEEE TPAMI pp. 1–14 (2025)
9. Huang, B., Li, J.C.L., Ran, J., Li, B., Zhou, J., Yu, D., Wong, N.: Hundred-kilobyte lookup tables for efficient single-image super-resolution. In: IJCAI. pp. 857–865 (2024)
10. Jiang, K., Wang, Z., Yi, P., Chen, C., Huang, B., Luo, Y., Ma, J., Jiang, J.: Multi-scale progressive fusion network for single image deraining. In: CVPR. pp. 8346–8355 (2020)
11. Jiang, X., Wang, N., Xin, J., Li, K., Yang, X., Gao, X.: Training binary neural network without batch normalization for image super-resolution. In: AAAI. pp. 1700–1707 (2021)
12. Jo, Y., Kim, S.J.: Practical single-image super-resolution using look-up table. In: CVPR. pp. 691–700 (2021)
13. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NeurIPS (2012)
14. Lee, H., Yoo, J.S., Jung, S.W.: RefQSR: Reference-based quantization for image super-resolution networks. IEEE TIP **33**, 2823–2834 (2024)
15. Li, H., Guan, J., Rui, L., Ma, S., Gu, L.: TinyLUT: Tiny look-up table for efficient image restoration at the edge. In: NeurIPS (2024)
16. Li, J., Chen, C., Cheng, Z., Xiong, Z.: Toward dnn of luts: Learning efficient image restoration with multiple look-up tables. IEEE TPAMI **46**(12), 8284–8301 (2024)
17. Li, Y., Li, J., Xiong, Z.: Look-up table compression for efficient image restoration. In: CVPR. pp. 26016–26025 (2024)
18. Liang, J., Cao, J., Sun, G., Zhang, K., Van Gool, L., Timofte, R.: SwinIR: Image restoration using swin transformer. In: ICCVW. pp. 1833–1844 (2021)
19. Liu, Z., Shen, Z., Savvides, M., Cheng, K.T.: ReActNet: Towards precise binary neural network with generalized activation functions. In: ECCV. pp. 143–159 (2020)
20. Liu, Z., Wu, B., Luo, W., Yang, X., Liu, W., Cheng, K.T.: Bi-Real Net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In: ECCV. pp. 722–737 (2018)

21. Mittal, A., Soundararajan, R., Bovik, A.C.: Making a “completely blind” image quality analyzer. *IEEE Signal Processing Letters* **20**(3), 209–212 (2012)
22. Neseem, M., McCullough, C., Hsin, R., Lechner, C., Li, S., Chong, I.S., Howard, A., Lew, L., Reda, S., Rautio, V.M., et al.: PikeLPN: Mitigating overlooked inefficiencies of low-precision neural networks. In: *CVPR*. pp. 15996–16005 (2024)
23. Park, S., Lee, S., Jin, K.H., Jung, S.W.: Im-lut: Interpolation mixing look-up tables for image super-resolution. In: *ICCV*. pp. 14317–14325 (2025)
24. Petersen, F., Borgelt, C., Kuehne, H., Deussen, O.: Deep differentiable logic gate networks. In: *NeurIPS*. pp. 2006–2018 (2022)
25. Petersen, F., Kuehne, H., Borgelt, C., Welzel, J., Ermon, S.: Convolutional differentiable logic gate networks. In: *NeurIPS*. pp. 121185–121203 (2024)
26. Ren, D., Zuo, W., Hu, Q., Zhu, P., Meng, D.: Progressive image deraining networks: A better and simpler baseline. In: *CVPR*. pp. 3937–3946 (2019)
27. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: *Proceedings of the Medical Image Computing and Computer Assisted Intervention*. pp. 234–241 (2015)
28. Schmidt, U., Roth, S.: Shrinkage fields for effective image restoration. In: *CVPR*. pp. 2774–2781 (2014)
29. Sheikh, H.: LIVE image quality assessment database release 2. <http://live.ece.utexas.edu/research/quality> (2005)
30. Shi, X., Ning, X., Guo, L., Zhao, T., Liu, E., Cai, Y., Dong, Y., Yang, H., Wang, Y.: Memory-oriented structural pruning for efficient image restoration. In: *AAAI*. pp. 2245–2253 (2023)
31. Wang, J., Chan, K.C., Loy, C.C.: Exploring clip for assessing the look and feel of images. In: *AAAI*. pp. 2555–2563 (2023)
32. Xia, B., Zhang, Y., Wang, Y., Tian, Y., Yang, W., Timofte, R., Gool, L.V.: Basic binary convolution unit for binarized image restoration network. In: *ICLR* (2023)
33. Xin, J., Wang, N., Jiang, X., Li, J., Huang, H., Gao, X.: Binarized neural network for single image super resolution. In: *ECCV*. pp. 91–107 (2020)
34. Yue, C., Jha, N.K.: Learning interpretable differentiable logic networks. *IEEE Transactions on Circuits and Systems for Artificial Intelligence* **1**(1), 69–82 (2024)
35. Zamir, S.W., Arora, A., Khan, S., Hayat, M., Khan, F.S., Yang, M.H.: Restormer: Efficient transformer for high-resolution image restoration. In: *CVPR*. pp. 5728–5739 (2022)
36. Zhang, H., Sindagi, V., Patel, V.M.: Image de-raining using a conditional generative adversarial network. *IEEE TCSVT* **30**(11), 3943–3956 (2019)
37. Zhang, K., Zuo, W., Chen, Y., Meng, D., Zhang, L.: Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE TIP* **26**(7), 3142–3155 (2017)
38. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: *CVPR*. pp. 586–595 (2018)
39. Zhang, X., Zhou, X., Lin, M., Sun, J.: ShuffleNet: An extremely efficient convolutional neural network for mobile devices. In: *CVPR*. pp. 6848–6856 (2018)
40. Zhang, Y., Qin, H., Zhao, Z., Liu, X., Danelljan, M., Yu, F.: Flexible residual binarization for image super-resolution. In: *ICML*. vol. 235, pp. 59731–59740 (2024)
41. Zhang, Y., Tian, Y., Kong, Y., Zhong, B., Fu, Y.: Residual dense network for image super-resolution. In: *CVPR*. pp. 2472–2481 (2018)
42. Zhang, Y., Wang, H., Qin, C., Fu, Y.: Learning efficient image super-resolution networks via structure-regularized pruning. In: *ICLR* (2022)