PlainUSR: Chasing Faster ConvNet for Efficient Super-Resolution

Yan Wang, Yusen Li*, Gang Wang, and Xiaoguang Liu

School of Computer Science, Nankai University, Tianjin, China
{wangy, liyusen, wgzwp, liuxg}@nbjl.nankai.edu.cn
https://github.com/icandle/PlainUSR

Abstract. Reducing latency is a roaring trend in recent super-resolution (SR) research. While recent progress exploits various *convolutional blocks*, *attention modules*, and *backbones* to unlock the full potentials of the convolutional neural network (ConvNet), achieving real-time performance remains a challenge. To this end, we present PlainUSR, a novel framework incorporating three pertinent modifications to expedite ConvNet for efficient SR. For the convolutional block, we squeeze the lighter but slower MobileNetv3 block into a heavier but faster vanilla convolution by reparameterization tricks to balance memory access and calculations. For the attention module, by modulating input with a regional importance map and gate, we introduce local importance-based attention to realize high-order information interaction within a 1-order attention latency. As to the backbone, we propose a plain U-Net that executes channel-wise discriminate splitting and concatenation. In the experimental phase, PlainUSR exhibits impressively low latency, great scalability, and competitive performance compared to both state-of-the-art latency-oriented and quality-oriented methods. In particular, compared to recent NGswin, the PlainUSR-L is 16.4× faster with competitive performance.

Keywords: Super-resolution · Reparamterization · Attention

1 Introduction

Image super-resolution (SR) has been a long-consisting low-level vision task aiming at restoring a proper high-resolution (HR) image from its numerous low-resolution degradation. Since the burgeon of deep learning, convolutional SR networks [10, 23, 30, 56] have become mainstream and achieved remarkable success. Despite the rising PSNR, the correspondingly increasing latency limits practical usage on mobile devices (*e.g.*, SR for cloud gaming and live streaming), which forces researchers to refine a faster and more efficient model. Generally, prevailing works developed more efficient ConvNets from three perspectives: 1) *convolution block*, 2) *attention module*, and 3) *backbone*. We revisit these components and ease the "bottleneck" restricting their latency reduction.

* Corresponding author

For *convolution block*, a primary attempt is to adopt residual learning to improve the representation capability and ameliorate optimization. For example, EDSR [30] utilized residual block [16] and RFDN [32] leveraged shallow residual block. While residual learning contributes to heightened restoration quality, it maintains comparatively intensive computations and increased memory usage. To address this limitation, the following works [28, 47, 59] focused on convolution decomposition to stack numerous simplified convolutions as the replacement for standard convolution. A notable example of this decomposition is the MobileNet [18] block, which decouples convolution into depthwise and pointwise convolutions, showcasing exceptional efficiency across diverse tasks [44]. Despite remarkably reducing parameters and calculations, the latency is instead growing due to higher memory access, especially for tasks with large input image. For instance, the MBConv is slower than vanilla convolution in the SR task. Inspired by reparameterization technology [46, 55] that transfers the complex training-time block into a simple one for inference, we equivalently transform the reparameterizable MBConv (RepMBConv) into a vanilla convolution to reach a more reasonable trade-off. Compared to the original MBConv, RepMBConv has more MACs but lower memory access, attaining lower latency and better extraction.

For *attention module*, RCAN [56] first absorbed channel attention into the residual block and achieved prominent performance. Building upon RCAN's success, subsequent works [38,57] delved into further explorations of attention mechanisms in super-resolution tasks. Specifically, RFANet [33] proposed effective spatial attention (ESA), which works its magic on both performance-oriented and lightweight frameworks [32]. However, these attention mechanisms facilitate only 1-order information interaction with a limited receptive region. Recent studies [29, 31] apply 2-order interaction, *e.g.*, non-local attention [37] and self-attention [54], to realize more comprehensive long-range modeling. However, these approaches introduce additional runtime due to the quadratic complexity. To strike a balance, we propose local importance-based attention (LIA) that realizes 2-order interaction with extremely simple operators. Specifically, LIA measures the local importance at the downscaled feature map and calibrates the attention map with one channel gate, thereby ensuring a relatively low latency.

For *backbone*, three typical designs are commonly employed: VGG [9]-style, ResNet [16]-style, and IMDN [22]-style, with the increasing performance but descending throughput. The VGG-style networks, exemplified by ESPCN [42] and QuickSRNet [2], demonstrate optimal deployment while encountering quality drops when enlarging model size. For prevailing efficient SR models [24, 32, 46], they tend to employ the EDSR/IMDN-like framework for guaranteed performance but ignore latency. In our endeavor to balance performance and latency, we introduce a U-Net design into the plain network by proposing a PlainU-Net, which executes a channel-dimension "U" shape processing.

Incorporating the aforementioned refinements: RepConv, LIA, and PlainU-Net, we introduce a new-generation ConvNet termed PlainUSR, aimed at achieving faster runtimes for efficient super-resolution tasks. Our contributions can be summarized as follows:

1) We revisit the efficient SR designs from three fundamental components (convolutional block, attention module, and backbone), and explore and alleviate bottlenecks of latency by following refinements:
   - We propose a reparameterized MBConv (RepMBConv) that seamlessly integrates MBConv into vanilla convolution without any performance degradation but 2.9× acceleration.
   - We introduce local importance-based attention (LIA), surpassing existing attention mechanisms in terms of both quality and efficiency.
   - We introduce a novel backbone, PlainU-Net, which remains straightforward during inference yet attains enhanced representation and optimization properties compared to existing backbones.
2) Based on these modifications, we present a simple yet effective framework, namely PlainUSR, which records significantly lower latency, greater scalability, and relatively fewer calculations than existing efficient SR models.

2 Related Work

2.1 Efficient Super-Resolution

To alleviate the complexity of upscaling LR images on devices with constrained resources, numerous strategies [10,22,46] have been introduced for efficient ConvNet design. The representative work, IMDN [22] employed information multi-distillation and contrast-aware channel attention (CCA), reaching remarkable performance within 1M parameters. Based on IMDN, RFDN [32] replaced channel splitting operation and CCA with feature distillation connection and more powerful ESA, to improve the flexibility and representation capability. Then, several researches [11, 24, 52] removed the skip connections and utilized structural parameterization to enable lossless improvement on a faster backbone. FMEN [11] proposed parameterizable residual in residual block (RRRB) and backbone with only one global residual connection, which spent extremely low memory and latency. Subsequent studies [4, 12] rehashed the plain VGG-like backbone for compact feature processing. However, an unavoidable challenge is the poor optimization of the plain structure. Recent work attempted to resolve the issue by reparameterization. For instance, QuickSRNet [2] adopted reparameterizable parameter initialization. ETDS [4] squeezed dual stream network into a plain one. These plain networks obtained better trade-offs between performance and latency.

2.2 Reparameter and Attention in Efficient Super-Resolution

Re-parameterization. Since pioneer RepVGG [9] equivalently transformed a three-branch block into a single layer after training for "free" improvement, structure parameterization has become an essential strategy for efficient SR. ECBSR [55] proposed an edge-oriented convolution block (ECB) that leveraged the Sobel filter for better edge generating. EFDN [46] further developed ECB

with Laplacian branches and edge loss to achieve better learning capability. SESR [11] collapsed the linear blocks into single convolution layers to add the training-time network depth for better optimization.

Attention. The attention module is an adaptive discriminating selection to retain useful features and automatically ignore noisy responses according to the input. As to efficient SR, most existing works adopt the existing attention modules. RLFN [24] employed ESA [33] and PAN [57] leveraged pixel-attention [34]. Recently, 2-order attention mechanisms, *e.g.*, self-attention [6, 29] and non-local attention [37, 51], bring higher performance. However, they are rarely used for low-latency SR models due to their computationally expensive costs.

3 Methodology

3.1 Convolution: Reparameterized MBConv

The MobileNetV3 [18] Block (MBConv) has achieved remarkable success in various computer vision tasks [44, 45]. Despite its parameter and calculation efficiency, its higher memory access of depth-wise and point-wise convolution results in non-negligible delay, thereby reducing the running speed. For I/O-bound devices and I/O-demand tasks, this drawback is exacerbated, for example, the efficient super-resolution task that conducts full-size image super-resolution on restricted devices. As indicated in Tab. 1, despite MBConv attaining fewer parameters and calculations, the model based on MBConv spends more time than vanilla convolution in $\times 4$ SR task on DIV2K-valid [1]. Drawing inspiration from reparameterization technology [8, 9, 55], we aim to address these limitations by reparameterizing the MBConv to a vanilla convolution. In essence, we reparameterize the MBConv to convolution (RepMBConv) through three steps, as illustrated in Fig. 1.

Replacing nonlinear: As the nonlinear operators (GELU [17] activation and SE [19] module) are un-reparameterizable, we opt to replace them with the linear operator or move them to the tail. For the activation function, we replace the first GELU with a scaling function and move the second one to the tail. For the SE module, based on *stripe observation* [58] that channel attention's values of varied inputs in the dataset tend towards a constant vector, we replace the input-relative SE with the learnable tensor-relative SE module.

Complex module: As RepMBConv deployed as vanilla convolution during inference, adding the complexity of training topology is harmless to improve representation capability. Thus, we replace the depth-wise convolution with standard convolution and identity to chase a better performance as previous works [8, 55].

Deployment: Following [8, 9], we train the complicated structure and squeeze it to convolution during inference for deployment. Given the input feature $\mathbf{X} \in \mathbb{R}^{C\times H\times W}$, the training process of RepMBConv $\mathcal{C}(\cdot)$ can be formulated by:

$$\mathcal{C}(\mathbf{X}) = \{[(\mathbf{X} * \mathbf{K}_1 + \mathbf{b}_1) \odot \mathbf{s} * \mathbf{K}_2 + \mathbf{b}_2] \odot \mathcal{F}_{SE}(\mathbf{v}) * \mathbf{K}_3 + \mathbf{b}_3\} + \mathbf{X}, \quad (1)$$

where $\odot$ and $*$ are scale-dot and convolution, respectively. ($\mathbf{K}_1 \in \mathbb{R}^{C_m\times C\times 1\times 1}$, $\mathbf{b}_1 \in \mathbb{R}^{C_m}$) are the kernel and bias weight for expanding point-wise convolution,

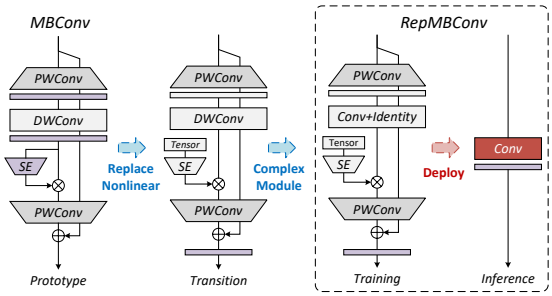

Fig. 1: Illustration of MBConv and RepMBConv.

Metrics	MBConv	Conv
Channel	64	64
Params	26.0k	36.9k
MACs	1.01G	2.12G
Memory locate	176.9M	42.5M
Memory access	40.6M	7.38M
Activations	342.8M	74.5M
Latency	110.9ms	39.3ms

Table 1: Comparisons for MBConv and vanilla convolution with 256×256 inputs. Latency is validated on RTX 4060.

$[\mathbf{K}_2 \in \mathbb{R}^{C_m \times C_m \times k \times k}, \mathbf{b}_2 \in \mathbb{R}^{C_m}]$ are the reparameterized kernel and bias for immediate 3×3 convolution and identity; $\{\mathbf{K}_3 \in \mathbb{R}^{C \times C_m \times 1 \times 1}, \mathbf{b}_3 \in \mathbb{R}^{C}\}$ are weights of squeezing point-wise convolution. $\mathbf{s}, \mathbf{v} \in \mathbb{R}^{C_m}$ are learnable scalar and vector to replace nonlinear operators. $\mathcal{F}_{SE}(\cdot)$ is SE module calculating channel attention. By unfolding Eq. (1), we can easily obtain the reparameterized kernel and bias for deployment:

$$\begin{aligned} \mathbf{K}_{Rep} &= \mathbf{K}_1 \odot \mathbf{s} * \mathbf{K}_2 \odot \mathcal{F}_{SE}(\mathbf{v}) * \mathbf{K}_3 + \mathbf{I}, \\ \mathbf{b}_{Rep} &= (\mathbf{b}_1 \odot \mathbf{s} * \mathbf{K}_2 + \mathbf{b}_2) \odot \mathcal{F}_{SE}(\mathbf{v}) * \mathbf{K}_3 + \mathbf{b}_3, \\ \mathcal{C}(\mathbf{X}) &= \mathbf{X} * \mathbf{K}_{Rep} + \mathbf{b}_{Rep}, \end{aligned} \tag{2}$$

where $\mathbf{K}_{Rep} \in \mathbb{R}^{C \times C \times k \times k}$ and $\mathbf{b}_{Rep} \in \mathbb{R}^{C}$ are reparameterized weights. $\mathbf{I} \in \mathbb{R}^{C \times C \times k \times k}$ is the reparameterized identity operator that satisfies $\mathbf{I}[i, i, \lfloor, \frac{k}{2}\rfloor, \frac{k}{2}\rfloor] = 1, i \in [0, C)$ but other values are zeros.

3.2 Attention: Local Importance-based Attention

Another impediment to a faster SR network is the spatial attention calculation. Existing spatial attention mechanisms can be primitively classified according to their orders of interaction [40], *e.g.*, ESA [33] (1-order: one element-wise multiplication) and Self-Attention [29] (2-order: two matrix multiplications). However, they suffer different defects that limit their usage in lightweight SR models: the weak performance for 1-order attention and quadric complexity for 2-order attention. Thus, we revisit the existing work and tailor a simplified 2-order spatial attention to trade off computation and performance.

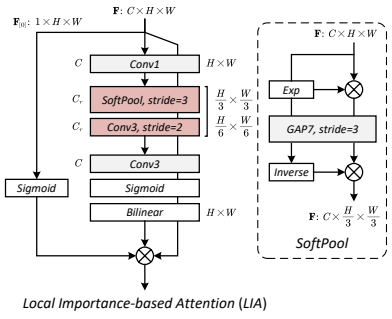

Fig. 2: Illustration of the proposed local importance-based attention (LIA).

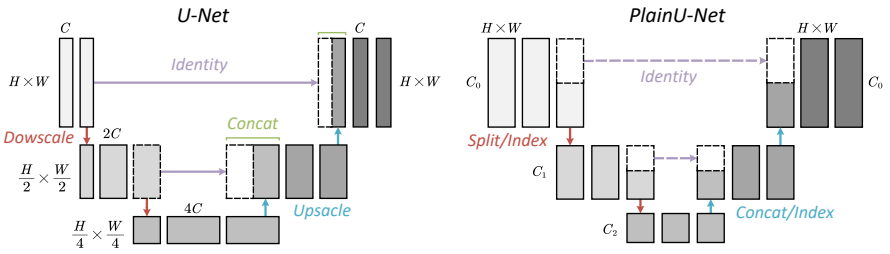

Fig. 3: Visual comparison between U-Net and PlainU-Net.

The core of the attention mechanism is to adaptively enhance helpful information but weaken useless one according to their input-relative importance. In prevailing studies, the importance map is measured by subnetwork or matrix multiplication. Inspired by [14, 43] that attains the local importance by regional softmax, we calculate the importance value of the pixel $\mathbf{x}$ within the surround $\mathbf{R}$ as follows:

$$\mathcal{I}(\mathbf{X})|_{\mathbf{x}} = \sum_{k \in \mathbf{R}} \sum_{i \in \mathbf{R}_k} \frac{e^{\mathbf{x}_i}}{\sum\limits_{j \in \mathbf{R}_k} e^{\mathbf{x}_j}} \cdot \mathbf{w}_k, \tag{3}$$

where $\mathcal{I}(\mathbf{X})|_{\mathbf{x}}$ is the local importance for $\mathbf{x}$. $\mathbf{R}$ refers to neighborhood centered at $\mathbf{x}$. $\mathbf{w}$ represents the learnable weight to refine the measured importance. As shown in Fig. 2, we instantiate Eq. (3) by stacking a softpool [43] and 3×3 convolution for efficiency and applicability. Similar to 1-order attention like ESA, we leverage stride and squeeze convolution to reduce calculation and enlarge the receptive field while sigmoid and bilinear for activation and rescaling.

To re-calibrate the local importance avoiding artifacts brought by stride convolution and bilinear interpolation, we use the gate mechanism [7, 47] for feature refining of local importance $\mathcal{I}(\mathbf{X})$. Differing from existing gate units applying additional networks, we choose the first channel $\mathbf{X}_{[0]}$ maps from the input to perform as a gate for simplicity. Overall, the LIA $\mathcal{A}(\cdot)$ can be summarised as:

$$\mathcal{A}(\mathbf{X}) = \sigma(\mathbf{X}_{[0]}) \odot \psi(\sigma(\mathcal{I}(\mathbf{X}))) \odot \mathbf{X}, \tag{4}$$

where $\sigma(\cdot)$ and $\psi(\cdot)$ are sigmoid activation and bilinear interpolation.

3.3 Backbone: PlainU-Net

Despite the above accelerations, the backbone determines the overall latency. Recent studies [2, 12] indicate that a plain backbone (VGG-style) surpasses other backbones, *e.g.*, ResNet [30] or IMDN [22] in terms of runtime and memory. Nevertheless, the plain VGG-style backbone uniformly processes the intermediate feature during the forward pass, wasting massive computations. Inspired by U-Net [41], which effectively encodes and decodes features at the spatial dimension, we introduce the PlainU-Net. This architecture employs a "U" shape

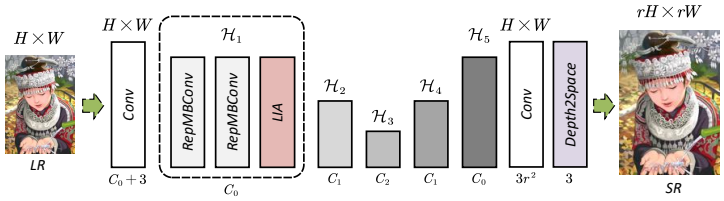

Fig. 4: Overview of the proposed PlainUSR framework.

processing at the channel dimension. As exhibited in Fig. 3, the plainU-Net splits and concatenates channel features to conduct hierarchical encoding and decoding throughout the training stage. For inference, it is plain by using a channel index. Given the input feature $\mathbf{F}_0 = \mathbf{F} \in \mathbb{R}^{C_0 \times H \times W}$ and the channel number $C_2 \leq C_1 \leq C_0$ of each stage, these procedures can be given by:

$$\begin{aligned}
\mathbf{F}_1, \mathbf{F}_{0_} &= \mathcal{S}(\mathcal{H}_1(\mathbf{F}_0), [C_1, C_0 - C_1]) &&\Longleftrightarrow \mathbf{F}[:C_0] = \mathcal{H}_1(\mathbf{F}[:C_0]),\\
\mathbf{F}_2, \mathbf{F}_{1_} &= \mathcal{S}(\mathcal{H}_2(\mathbf{F}_1), [C_2, C_1 - C_2]) &&\Longleftrightarrow \mathbf{F}[:C_1] = \mathcal{H}_2(\mathbf{F}[:C_1]),\\
\mathbf{F}_3 &= \mathcal{H}_3(\mathbf{F}_2) &&\Longleftrightarrow \mathbf{F}[:C_2] = \mathcal{H}_3(\mathbf{F}[:C_2]),\\
\mathbf{F}_4 &= \mathcal{H}_4(\mathcal{L}(\mathbf{F}_3, \mathbf{F}_{1_})) &&\Longleftrightarrow \mathbf{F}[:C_1] = \mathcal{H}_4(\mathbf{F}[:C_1]),\\
\mathbf{F}_5 &= \mathcal{H}_5(\mathcal{L}(\mathbf{F}_4, \mathbf{F}_{0_})) &&\Longleftrightarrow \mathbf{F}[:C_0] = \mathcal{H}_5(\mathbf{F}[:C_0]),
\end{aligned} \tag{5}$$

where $\mathcal{H}_i(\cdot)$, $\mathcal{S}(\cdot)$, and $\mathcal{L}(\cdot)$ are convolutional block, split function, and concatenation function, respectively. The left functions formulate the training process while the right ones indicate inference deployment, which employs non-residual forward and shares the memory space of $\mathbf{F}$.

3.4 Overall: PlainUSR

Building upon RepMBConv, LIA, and PlainU-Net, we build a framework named PlainUSR. As depicted in Fig. 4, we stack two RepMBConv and an LIA to instantiate $\mathcal{H}_i$ in Eq. (5). In addition to the PlainU-Net backbone, we incorporate a shallow extraction head and reconstruction tail, consistent with prior work [24].

4 Experiment

Datasets and metrics. Following prevailing works [26,32,46], we utilize DF2K (800 images from DIV2K [1] and 2650 images from Flickr2K [30]) as the training datasets. During the testing phase, we assess our models on five classic benchmarks, including Set5 [3], Set14 [53], BSD100 [36], Urban100 [20], and DIV2K-valid [1]. Additionally, following [5,25], we utilize DIV8K [15] to generate Test2K and Test4K as more comprehensive benchmarks to prevent model overfitting on traditional test sets. For quality evaluation, we use PSNR and SSIM [49] metrics.

Table 2: Quantitative comparison (average PSNR/SSIM on RGB, Parameters, MACs, Latency, Memory Footprint, and Activations) with state-of-the-art approaches for efficient image SR (×4). Overall best results are in red. MACs are measured under the setting of the input image to 256×256. Latency and Memory footprint are reported on the DIV2K-valid [1] dataset with Mobile AMD Ryzen 7940hs (CPU), RTX 4060-Laptop (GPU1) and Tesla V100 (GPU2).

Method	CPU (ms)	GPU1 (ms)	GPU2 (ms)	Para (K)	MACs (G)	Mem (M)	Acts (M)	DIV2KV [1] PSNR/SSIM	Test2K [15] PSNR/SSIM	Test4K [15] PSNR/SSIM
IMDN [22]	692	78.3	32.6	893	58.53	473.7	154.14	29.13/0.8221	26.24/0.7614	27.71/0.8078
RFDN [32]	680	56.5	25.8	433	27.10	832.0	112.03	29.04/0.8201	26.23/0.7601	27.70/0.8067
PlainUSR-L	554	49.8	23.0	734	44.22	327.7	80.62	29.12/0.8221	26.29/0.7622	27.77/0.8086
EFDN [46]	420	43.4	20.4	276	16.73	710.0	111.12	29.00/0.8187	26.17/0.7580	27.63/0.8049
ECBSR [55]	413	39.7	19.7	622	40.66	231.6	77.59	28.86/0.8190	26.13/0.7564	27.57/0.8033
FMEN [11]	629	31.5	18.2	341	22.28	205.9	72.09	29.00/0.8190	26.17/0.7586	27.62/0.8051
RLFN [24]	346	33.9	17.1	317	19.70	521.5	80.05	29.00/0.8190	26.20/0.7593	27.67/0.8060
DIPNet [52]	343	29.1	16.0	243	14.90	550.4	72.97	29.04/0.8184	26.11/0.7550	27.52/0.8018
PlainUSR-B	289	26.8	14.1	333	18.69	327.7	46.93	28.96/0.8181	26.18/0.7579	27.63/0.8048

For efficiency evaluation, referring to NTIRE challenge [26], we exploit parameters, FLOPs, latency, activations, and max GPU memory footprint to examine the proposed models with state-of-the-art methods.

Training details. We implement PlainUSR-**U**tra/**T**iny/**S**mall/**M**edium/**B**ase/**L**arge with varied settings to explore comprehensive trade-offs between performance and complexity. Specifically, for **U**ltra/**T**iny models, we leverage 2-stage PlainU-Net backbones with channel numbers {16, 8} and {32, 16}. For **S**mall/**M**edium/**B**ase/**L**arge models, we employ 3-stage PlainU-Net and set channel numbers {32, 16, 8}, {48, 32, 16}, {64, 48, 32}, and {80, 64, 48}, respectively. Regarding the training procedure, we randomly sample 64×64 LR images with a batch size of 64 for each iteration. The optimization objective is the ℓ_1 loss, and we employ AdamW [35](β_1=0.9, β_2=0.99) to optimize PlainUSR. The learning rate is initialized as 5×10^{-4} and halved at {250k, 400k, 450k, 475k} within 500k iterations. All experiments are conducted using the PyTorch [39] framework on 4 Tesla V100 GPUs.

4.1 Comparison with SOTA SR models

Quantitative comparison under similar performance. Following NTIRE series Efficient SR competitions [26, 27], we compare the models under similar performance (around 29.00dB on DIV2K-valid [1], including IMDN [22], RFDN [32], EFDN [46], ECBSR [55], FMEN [11], RLFN [24], DIPNet [52], and the PlainUSR-B/L). To avoid overfitting on the specific dataset, we also add Test2K and Test4K for fair performance validation. As shown in Tab. 2, our PlainUSR-B attains the lowest latency and fewest activations while PlainUSR-L reaches the highest PSNR/SSIM. Specifically, compared to DIPNet [52], PlainUSR-B obtains 0.11dB improvement on Test4K and 10%-15%faster. Compared to more complex IMDN [22], PlainUSR-L advances by 0.06dB but is 30% faster.

Table 3: Quantitative comparison (average PSNR/SSIM on Y of YCbCr, Parameters, MACs, Latency, and Activation) with state-of-the-art approaches for varied trade-offs in image SR (×4). The models are divided into six groups according to their throughput on the Tesla V100, *i.e.*, from top to bottom: *240 fps* (<4.16ms), *180 fps* (<5.55ms), *120 fps* (<8.33ms), *90 fps* (<11.11ms), *60 fps* (<16.66ms), and others. The best results of each group are in red. For this and the following tables Tabs. 4 and 5, MACs are measured under the setting of the input image to 256×256. Latency is reported on the DIV2K-valid [1] dataset with (i) RTX 4060-Laptop (ms) and (ii) Tesla V100 (ms).

Method (×4)	Latency		Para	MACs	Acts	Set5 [3]	Set14 [53]	BSD100 [36]	Urban [20]
	(i)	(ii)	(K)	(G)	(M)	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM
ESPCN [42]	10.9	2.7	37.2	2.43	9.44	30.66/0.8688	27.66/0.7581	26.94/0.7152	24.56/0.7263
PlainUSR-U	9.5	3.2	19.8	1.29	10.16	30.77/0.8698	27.72/0.7602	27.03/0.7173	24.71/0.7327
ECBSR-M4C16 [55]	10.4	5.5	16.8	1.09	8.39	30.89/0.8735	27.81/0.7623	27.06/0.7181	24.76/0.7366
QuickSRNet-S [2]	10.1	4.6	33.3	2.17	9.44	30.91/0.8746	27.85/0.7627	27.06/0.7183	24.76/0.7373
ETDS-S [4]	11.6	4.7	54.1	3.53	16.78	31.19/0.8806	28.01/0.7678	27.18/0.7226	25.03/0.7479
PlainUSR-T	10.8	4.5	60.1	3.93	16.97	31.23/0.8809	28.03/0.7678	27.20/0.7230	25.04/0.7480
ECBSR-M4C32 [55]	12.5	7.8	51.9	3.39	13.63	31.26/0.8809	28.06/0.7683	27.22/0.7234	25.08/0.7496
QuickSRNet-M [2]	12.1	6.2	61.0	3.98	15.73	31.35/0.8827	28.10/0.7690	27.22/0.7241	25.10/0.7508
ETDS-M [4]	12.1	5.6	72.6	4.74	20.97	31.41/0.8843	28.13/0.7705	27.27/0.7251	25.20/0.7544
PlainUSR-S	12.4	7.1	68.6	4.19	20.60	31.56/0.8855	28.17/0.7714	27.30/0.7268	25.32/0.7597
ECBSR-M6C40 [55]	15.8	8.7	105.4	6.88	21.50	31.42/0.8825	28.21/0.7714	27.33/0.7265	25.38/0.7602
PCEVA-M [60]	16.4	10.3	67.7	4.32	19.16	31.58/0.8867	28.25/0.7718	27.36/0.7279	25.48/0.7618
ETDS-L [4]	16.8	8.5	170.0	11.11	31.46	31.69/0.8889	28.31/0.7751	27.37/0.7302	25.47/0.7643
PlainUSR-M	18.1	10.4	165.5	9.82	32.56	31.79/0.8889	28.38/0.7760	27.43/0.7313	25.68/0.7716
DIPNet [52]	29.1	16.0	243.3	14.90	72.97	31.81/0.8898	28.44/0.7773	27.45/0.7318	25.75/0.7751
ECBSR-M16C64 [55]	29.9	15.3	621.6	40.66	77.59	31.92/0.8946	28.34/0.7798	27.48/0.7339	25.81/0.7773
QuickSRNet-L [2]	29.3	15.3	435.9	28.51	56.62	31.88/0.8913	28.46/0.7788	27.48/0.7332	25.76/0.7754
PlainUSR-B	26.8	14.1	333.1	18.69	46.93	32.02/0.8922	28.54/0.7800	27.54/0.7351	25.97/0.7814
IMDN [22]	78.3	32.6	893.9	58.53	154.1	32.21/0.8948	28.58/0.7811	27.56/0.7353	26.04/0.7838
RLFN [24]	62.7	24.7	543.7	33.99	126.5	32.24/0.8952	28.62/0.7813	27.60/0.7364	26.17/0.7877
DDistill-SR [48]	130.2	50.7	675.8	37.30	226.6	32.28/0.8961	28.69/0.7833	27.64/0.7383	26.25/0.7892
PlainUSR-L	49.8	23.0	734.0	44.22	80.62	32.28/0.8959	28.70/0.7838	27.65/0.7385	26.35/0.7943

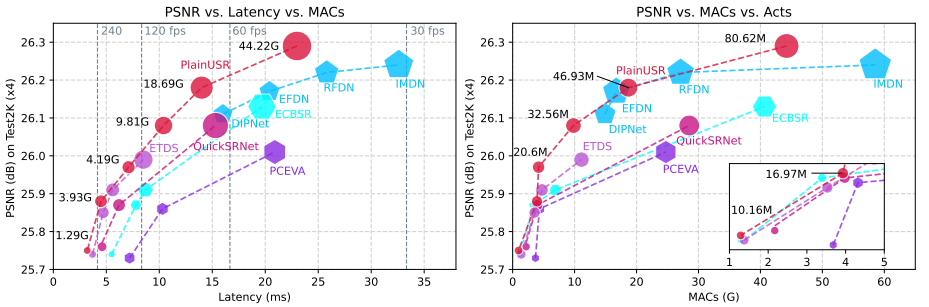

Fig. 5: Trade-off curves between restoration performance and runtime complexity.

Quantitative comparison under similar latency. We also execute comparisons under different performance-complexity trade-offs to demonstrate the scalability and generality of the proposed framework. Tab. 3 exhibits the ×4 SR results, where our PlainUSR family achieves incredible improvements over pre-

Table 4: Quantitative comparison (average PSNR/SSIM on Y of YCbCr, Parameters, MACs, Latency, and Activation) with state-of-the-art approaches for varied trade-offs in image SR (×2). The best results of each group are in red.

Method (×2)	Latency (i)	Latency (ii)	Para (K)	MACs (G)	Acts (M)	Set5 [3] PSNR/SSIM	Set14 [53] PSNR/SSIM	BSD100 [36] PSNR/SSIM	Urban [20] PSNR/SSIM
ESPCN [42]	19.2	4.9	26.8	1.75	7.08	36.87/0.9559	32.62/0.9086	31.40/0.8898	29.61/0.8973
PlainUSR-U	17.4	4.4	13.6	0.89	7.80	37.10/0.9569	32.75/0.9101	31.56/0.8914	29.98/0.9037
ECBSR-M4C16 [55]	21.2	5.9	11.5	0.75	6.03	37.18/0.9577	32.79/0.9105	31.61/0.8928	30.18/0.9064
QuickSRNet-S [2]	20.6	5.3	22.9	1.49	7.08	37.18/0.9573	32.82/0.9104	31.61/0.8921	30.17/0.9064
ETDS-S [4]	23.8	6.8	41.5	2.71	12.06	37.38/0.9587	32.96/0.9124	31.77/0.8943	30.62/0.9121
PlainUSR-T	21.4	6.5	48.7	3.18	14.61	37.47/0.9584	32.97/0.9124	31.77/0.8943	30.53/0.9110
ECBSR-M4C32 [55]	24.4	7.2	41.5	2.71	11.27	37.39/0.9587	33.03/0.9126	31.79/0.8953	30.69/0.9128
QuickSRNet-M [2]	25.5	7.5	50.6	3.30	13.37	37.42/0.9584	33.05/0.9127	31.81/0.8951	30.77/0.9140
ETDS-M [4]	29.8	8.9	60.0	3.92	16.25	37.54/0.9593	33.09/0.9133	31.86/0.8963	30.87/0.9149
PlainUSR-S	37.3	10.1	57.2	3.45	18.24	37.64/0.9593	33.16/0.9146	31.93/0.8968	31.06/0.9178
ECBSR-M6C40 [55]	49.5	14.2	92.4	6.03	19.14	37.61/0.9596	33.18/0.9139	31.94/0.8972	31.09/0.9174
ETDS-L [4]	53.8	17.8	152.2	9.95	26.74	37.64/0.9597	33.24/0.9145	31.98/0.8977	31.22/0.9188
PlainUSR-M	53.2	16.7	148.9	8.74	30.20	37.79/0.9597	33.37/0.9166	32.06/0.8984	31.53/0.9228
ECBSR-M16C64 [55]	166.9	50.0	600.7	39.29	72.88	37.90/0.9615	33.34/0.9178	32.10/0.9018	31.71/0.9250
QuickSRNet-L [2]	121.6	34.2	415.0	27.14	51.90	37.87/0.9600	33.45/0.9164	32.10/0.8988	31.76/0.9246
PlainUSR-B	117.2	35.2	311.4	17.27	44.57	37.95/0.9603	33.60/0.9184	32.16/0.8996	32.00/0.9273
IMDN [22]	338.0	96.2	873.2	57.17	151.8	38.00/0.9605	33.63/0.9177	32.19/0.8996	32.17/0.9283
RLFN [24]	232.7	75.7	526.9	32.89	124.1	38.07/0.9607	33.72/0.9187	32.22/0.9000	32.33/0.9299
PlainUSR-L	219.1	67.4	707.1	42.46	78.29	38.07/0.9607	33.82/0.9202	32.27/0.9009	32.53/0.9320

vailing models, additionally including ESPCN [42], QuickSRNet [2], PCEVA [60], ETDS [4], and DDistill-SR [48]. In general, the PlainUSR can maintain a comparative throughput (fps) while advancing 0.1dB on Urban100 for almost all groups. Particularly, the PlainUSR-B/M is as fast as QuickSRNet-L/PCEVA-M but attains 0.21dB improvement on Urban100. Compared to ESPCN, PlainUSR-U accelerates 1.4ms and leads by 0.15dB. To enable a more comprehensive comparison, we validate these models on Test2K [15] and visualize them in Fig. 5. Specifically, the latency-PSNR curve shows the PlainUSR is more competitive than other plain architecture, *e.g.*, ETDS [4] and QuickSRNet [2]. Moreover, compared to these latency-oriented models, our PlainUSR can reach better trade-offs between computations and performance. We also provide ×2 results in Tab. 4, where PlainUSR consistently improves performance in terms of both PSNR/SSIM under similar latency. These upshots demonstrate the effectiveness of the PlainUSR framework.

More comparison with quality-oriented methods. In the above comparisons, we compare the proposed PlainUSR with models trading off quality and latency. Here, we provide a detailed comparison with more quality-oriented methods (*e.g.*, transformers and dedicated ConvNet) in Tab. 5. In a word, PlainUSR has the lowest latency and highest quality. Specifically, compared to NGswin [6], the PlainUSR-L is 16.4× faster and attains higher PSNR/SSIM on BSD100. Compared to SwinIR-light, the PlainUSR-L+ accelerates by 12.0× and maintains similar restoration quality.

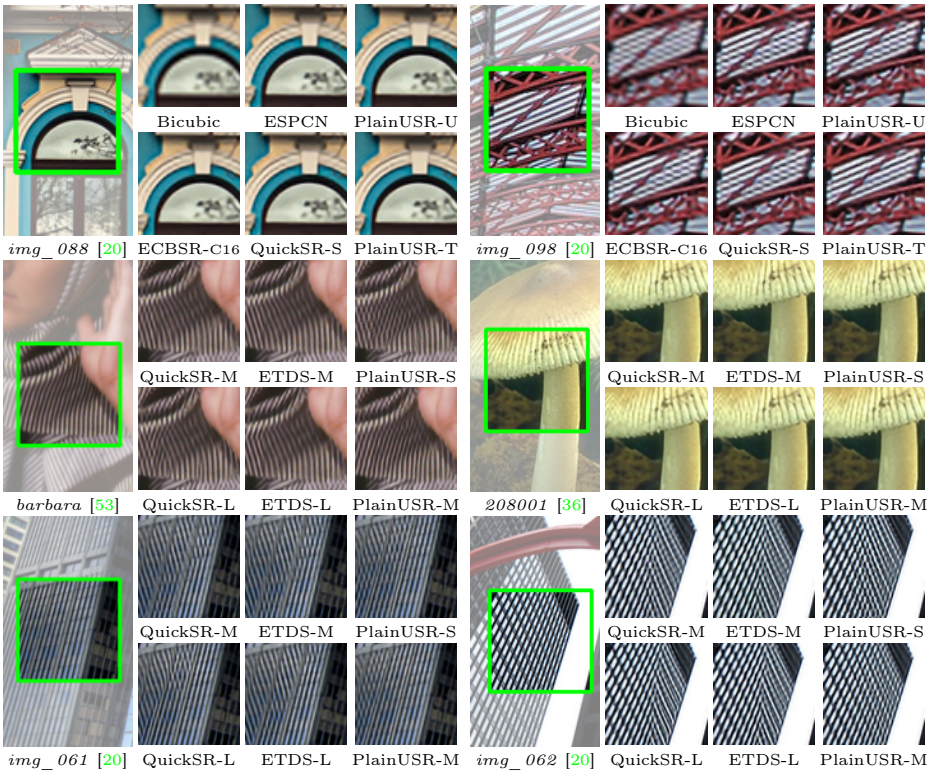

Fig. 6: Visual comparison for efficient SR models with an upscaling factor ×2. (Best viewed by zooming)

Method	Latency (ms)	Para (K)	MACs (G)	Acts (M)	Set5 [3] PSNR/SSIM	Set14 [53] PSNR/SSIM	BSD100 [36] PSNR/SSIM	Urban100 [20] PSNR/SSIM
FDIWN [13]	821.6	629.2	77.58	1008.1	38.07/0.9608	33.75/0.9201	32.23/0.9003	32.40/0.9305
NGswin [6]	1104.9	998.4	111.09	-	38.05/0.9610	33.79/0.9199	32.27/0.9008	32.53/0.9324
CoMoNet-S [21]	831.4	722.1	58.70	273.68	38.06/0.9606	33.84/0.9206	32.23/0.9003	32.45/0.9315
PlainUSR-L	67.4	707.1	169.84	78.29	38.07/0.9607	33.82/0.9202	32.27/0.9009	32.53/0.9320
SwinIR-light [29]	3096.9	910.2	217.51	-	38.14/0.9611	33.86/0.9206	32.31/0.9012	32.76/0.9340
PlainUSR-L+	257.6	707.1	169.84	78.29	38.13/0.9610	33.88/0.9211	32.32/0.9014	32.73/0.9335

Table 5: Quantitative comparison with quality-oriented models for efficient SR (×2).

Qualitative comparison. In Figs. 6 and 7, we display the visual comparison between the PlainUSR family with other SOTA approaches. Generally, our methods recover substantially clearer and more accurate structural content than others. Concretely, for *barbara* from Fig. 6, PlainUSR-S/M are the only two models restoring the correct cloth texture. For *img_020* in Fig. 7, PlainUSR-B/L recovers more legible patterns of windows than models with similar latency.

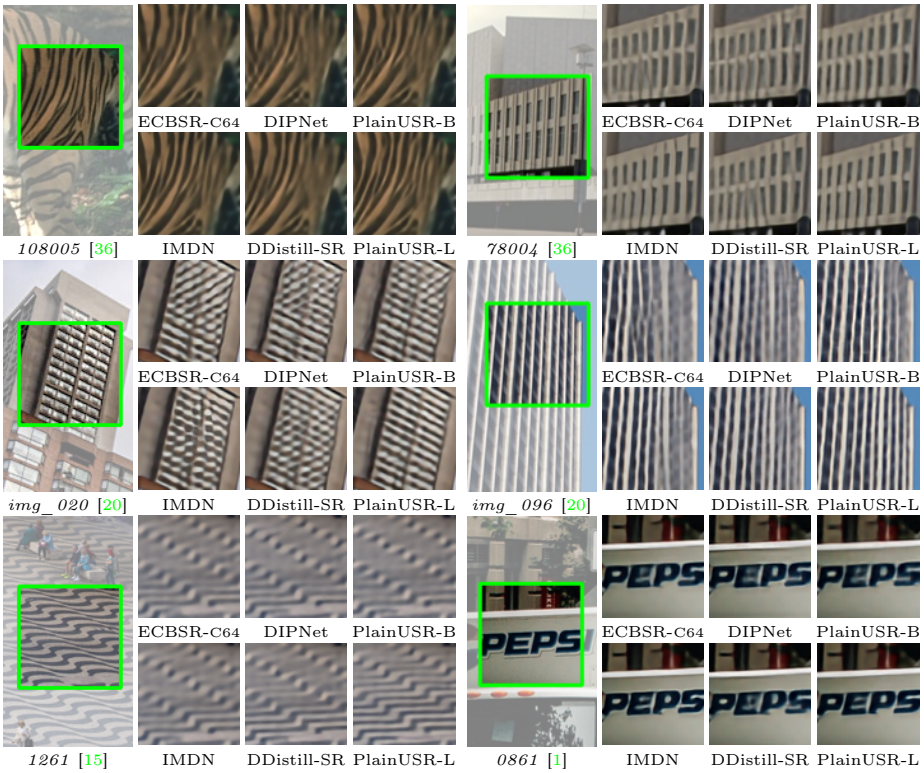

Fig. 7: Visual comparison for efficient SR models with an upscaling factor ×4.

4.2 Ablation studies

Effectiveness of RepMBConv. In Tab. 6, we offer the roadmap of transferring RepMBConv into convolution and more comparisons with other rep-strategies like RepVGG [9], ECB [55], and RRRB [11] on the proposed PlainUSR. First, we compare the performance between vanilla convolution and MBConv [18]. While MBConv leads by over 0.05dB and is lighter than standard convolution, the latter is in turn 3× faster due to the lower memory access and activation. To endow MBConv with faster inference, *i.e.*, convolution, we replace the nonlinear layers and use the complex module to propose RepMBConv. Replacing nonlinear makes the module reparameterizable but induces a huge performance drop (0.13dB on DIV2KV). Employing complex modules alleviates and even improves the performance decrease. Overall, the proposed RepMBConv can attain similar performance as the original MBConv but maintain the inference as a single convolution. Moreover, RepMBConv surpasses others by over 0.06dB/0.002 PSNR/SSIM improvement on DIV2KV, demonstrating its effectiveness.

Effectiveness of LIA. To investigate the effectiveness of LIA, we compare it with several existing attention modules, including SE [19], CBAM [50], ESA [33],

Table 6: Quantitative comparison between varied parameterization, *i.e.* RepConv *vs.* RepMBConv on PlainUSR for ×4 SR tasks. For "†" items, the metrics {Para, MACs, Mem, Acts} are reported at the training time.

Method	Latency (ms)	Para (K)	MACs (G)	Mem (M)	Acts (M)	DIV2KV [1] PSNR/SSIM	Test2K [15] PSNR/SSIM	Test4K [15] PSNR/SSIM
Convolution	26.8	333	18.69	327.7	46.93	28.81/0.8148	26.08/0.7550	27.51/0.8021
RepVGG† [9]	26.8	583	35.00	328.7	80.48	28.80/0.8147	26.09/0.7554	27.53/0.8025
ECB† [55]	26.8	989	61.27	479.2	348.9	28.83/0.8153	26.10/0.7557	27.54/0.8028
RRRB† [11]	26.8	1192	74.86	649.1	181.1	28.83/0.8151	26.10/0.7558	27.54/0.8030
MBConv [18]	78.3	261	10.24	399.1	181.1	28.88/0.8166	26.12/0.7568	27.57/0.8039
Replacing Nonlinear†	26.8	263	10.24	399.1	181.1	28.75/0.8133	26.06/0.7539	27.49/0.8014
Complex Module†	26.8	1249	74.86	658.4	181.1	28.89/0.8171	26.13/0.7573	27.58/0.8043
RepMBConv	26.8	333	18.69	327.7	46.93	28.89/0.8171	26.13/0.7573	27.58/0.8043

Table 7: Quantitative comparison between varied attention modules on ×4 SR tasks.

Method	Attention	Latency (ms)	Para (K)	MACs (G)	Mem (M)	Acts (M)	DIV2KV [1] PSNR/SSIM	Test2K [15] PSNR/SSIM	Test4K [15] PSNR/SSIM
PlainUSR	Baseline	20.2	280	18.34	262.5	41.09	28.61/0.8092	26.00/0.7508	27.38/0.7978
	SE [19]	22.5	287	18.34	270.7	41.09	28.62/0.8095	26.00/0.7509	27.38/0.7978
	CBAM [50]	55.7	289	18.37	263.6	41.42	28.64/0.8097	25.99/0.7506	27.36/0.7972
	ESA [50]	30.9	337	19.20	395.5	70.05	28.86/0.8157	26.11/0.7562	27.56/0.8032
	NLSA [37]	631.5	326	21.28	6748.9	62.06	28.91/0.8173	26.13/0.7575	27.59/0.8043
	LIA	26.8	333	18.69	327.7	46.93	28.89/0.8171	26.13/0.7573	27.58/0.8043
RFDN [32]	ESA [33]	56.5	433	27.10	832.0	112.03	28.96/0.8182	26.17/0.7585	27.64/0.8054
	LIA	50.3	449	26.92	755.3	96.28	28.97/0.8185	26.19/0.7588	27.65/0.8057

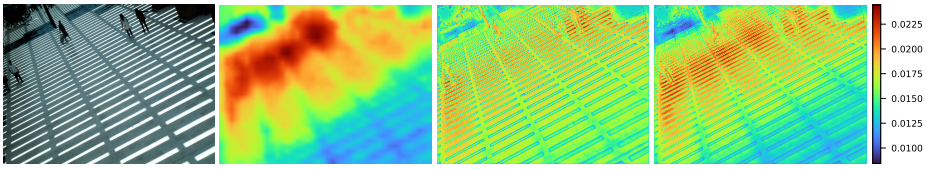

Fig. 8: Visual activation maps of each term in Eq. (4).

and NLSA [37] on the proposed PlainUSR and RFDN [32] in Tab. 7. In general, the LIA can catch up with dedicated 2-order attention and surpasses 1-oder attention by large margins. Particularly, the PSNR of LIA decreases a maximum of 0.02dB than NLSA while accelerating 23×. Compared to ESA, the proposed LIA spends fewer calculations and time and obtains notable gains over three challenging benchmarks on both PlainUSR and RFDN. These numerical results imply the efficiency and effectiveness of the LIA module.

To evaluate the influence of individual components of LIA, we ablate them in Tab. 8 with six variants. The variant I is the baseline without attention mechanism. For variant II and III, we degrade LIA with only 1-order information interaction. Removing any of them (local importance or gate mechanism) will result in huge performance drops. Moreover, local importance plays a more im-

Table 8: Quantitative comparison between varied LIA variants on ×4 SR tasks.

Variant	$\mathcal{A}(\mathbf{X})$	Latency (ms)	Para (K)	MACs (G)	Mem (M)	Acts (M)	Urban100 [20] PSNR/SSIM	Test2K [15] PSNR/SSIM
I	$\mathbf{I} \odot \mathbf{X}$	20.2	280	18.34	262.5	41.09	25.43/0.7634	26.00/0.7508
II	$\sigma(\mathbf{X}_{[0]}) \odot \mathbf{X}$	21.9	280	18.34	262.5	41.09	25.54/0.7671	26.03/0.7522
III	$\psi(\sigma(\mathcal{I}(\mathbf{X}))) \odot \mathbf{X}$	24.6	333	18.69	262.7	46.93	25.68/0.7727	26.07/0.7543
IV	$\sigma(\mathbf{X}_{[0]}) \odot \psi(\sigma(\mathcal{I}'(\mathbf{X}))) \odot \mathbf{X}$	25.9	333	18.69	327.7	46.93	25.78/0.7760	26.09/0.7552
V	$\sigma(\mathbf{X}_{[0]}) \odot \sigma(\psi(\mathcal{I}(\mathbf{X}))) \odot \mathbf{X}$	28.5	333	18.69	327.7	46.93	25.88/0.7800	26.13/0.7573
VI	$\sigma(\mathbf{X}_{[0]}) \odot \psi(\sigma(\mathcal{I}(\mathbf{X}))) \odot \mathbf{X}$	26.8	333	18.69	327.7	46.93	25.88/0.7800	26.13/0.7573

Variant	Latency (ms)	Mem (M)	Acts (M)	Urban100 [20] PSNR/SSIM	Test2K [15] PSNR/SSIM
VGG [9]-style	17.7	167.2	29.39	25.51/0.7672	26.00/0.7519
ResNet [30]-style	19.8	228.5	29.19	25.52/0.7674	26.02/0.7523
PlainU-Net	18.1	247.2	32.56	25.60/0.7702	26.04/0.7531

Table 9: Quantitative comparison between varied backbones on ×4 SR tasks.

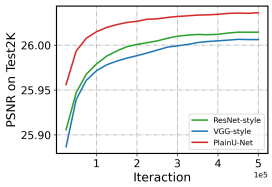

Fig. 9: Training curve.

portant role in restoration. For variant IV, we replace the softmax-based importance with the max-based one, inducing a 0.1dB/0.004 decline on Urban100. For variant V, we swap the sigmoid activation and bilinear interpolation. This operation has no impact on performance but increases the 1.7ms latency. In Fig. 8, we exhibit the activation maps for Eq. (4), where the local importance captures the approximate responses and then gate rectifies it for more precise attention.

Effectiveness of PlainU-Net backbone. In Tab. 9 and Fig. 9, we present the comparison between the proposed PlainU-Net with the other two style backbones: VGG [9]-style and ResNet [16]-style. We set the channel number 32 for these two backbone and {48,32,16} for PlainU-Net to maintain similar complexity. After fair training under the same settings, we observe that PlainU-Net provides a better-optimizing curve than the other two backbones. The PSNR improvements of PlainU-Net over VGG/ResNet are 0.09dB/0.08dB on Urban100.

5 Conclusion

This paper presents PlainUSR, a novel framework chasing faster ConvNets for image super-resolution tasks. Our methodology focuses on reducing inference latency through three key modifications: RepMBConv, LIA module, and PlainU-Net, targeting convolution, attention, and backbone components. Through comprehensive experiments, we demonstrate that each of these modifications surpasses existing schemes in both inference latency and reconstruction quality. By synergistically integrating them, the PlainUSR family achieves remarkable trade-offs between latency, complexity, and quality.

Acknowledgements This research is supported by NSF of China (grant numbers 62293510/62293513, 62272252, 62272253), NSF of Tianjin under grant 21JCYBJC00070.

References

1. Agustsson, E., Timofte, R.: NTIRE 2017 challenge on single image super-resolution: Dataset and study. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 1122–1131 (2017)
2. Berger, G., Dhingra, M., Mercier, A., Savani, Y., Panchal, S., Porikli, F.: Quicksrnet: Plain single-image super-resolution architecture for faster inference on mobile platforms. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 2186–2195 (2023)
3. Bevilacqua, M., Roumy, A., Guillemot, C., Alberi-Morel, M.: Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In: British Machine Vision Conference (BMVC). pp. 1–10 (2012)
4. Chao, J., Zhou, Z., Gao, H., Gong, J., Yang, Z., Zeng, Z., Dehbi, L.: Equivalent transformation and dual stream network construction for mobile image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 14102–14111 (2023)
5. Chen, B., Lin, M., Sheng, K., Zhang, M., Chen, P., Li, K., Cao, L., Ji, R.: Arm: Any-time super-resolution method. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 254–270. Springer (2022)
6. Choi, H., Lee, J., Yang, J.: N-gram in swin transformers for efficient lightweight image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2071–2081 (June 2023)
7. Dauphin, Y.N., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks. In: International Conference on Machine Learning (ICML). Proceedings of Machine Learning Research, vol. 70, pp. 933–941. PMLR, Sydney, Australia (2017)
8. Ding, X., Zhang, X., Han, J., Ding, G.: Diverse branch block: Building a convolution as an inception-like unit. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10886–10895. Computer Vision Foundation / IEEE, virtual (2021)
9. Ding, X., Zhang, X., Ma, N., Han, J., Ding, G., Sun, J.: Repvgg: Making vgg-style convnets great again. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 13733–13742. Computer Vision Foundation / IEEE, virtual (2021)
10. Dong, C., Loy, C.C., Tang, X.: Accelerating the super-resolution convolutional neural network. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Proceedings of the European Conference on Computer Vision (ECCV). vol. 9906, pp. 391–407 (2016)
11. Du, Z., Liu, D., Liu, J., Tang, J., Wu, G., Fu, L.: Fast and memory-efficient network towards efficient image super-resolution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 853–862 (2022)
12. Du, Z., Liu, J., Tang, J., Wu, G.: Anchor-based plain net for mobile image super-resolution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 2494–2502 (2021)
13. Gao, G., Li, W., Li, J., Wu, F., Lu, H., Yu, Y.: Feature distillation interaction weighting network for lightweight image super-resolution. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). vol. 36, pp. 661–669 (2022)
14. Gao, Z., Wang, L., Wu, G.: Lip: Local importance-based pooling. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 3355–3364 (2019)

15. Gu, S., Lugmayr, A., Danelljan, M., Fritsche, M., Lamour, J., Timofte, R.: Div8k: Diverse 8k resolution image dataset. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW). pp. 3512–3516 (2019)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778. IEEE Computer Society, Las Vegas, USA (2016)
17. Hendrycks, D., Gimpel, K.: Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415 (2016)
18. Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al.: Searching for mobilenetv3. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 1314–1324 (2019)
19. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 7132–7141 (2018)
20. Huang, J., Singh, A., Ahuja, N.: Single image super-resolution from transformed self-exemplars. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5197–5206 (2015)
21. Huang, Y., Li, J., Hu, Y., Huang, H., Gao, X.: Deep convolution modulation for image super-resolution. IEEE Transactions on Circuits and Systems for Video Technology (2023)
22. Hui, Z., Gao, X., Yang, Y., Wang, X.: Lightweight image super-resolution with information multi-distillation network. In: Proceedings of the ACM International Conference on Multimedia (MM). pp. 2024–2032 (2019)
23. Kim, J., Lee, J.K., Lee, K.M.: Accurate image super-resolution using very deep convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1646–1654 (2016)
24. Kong, F., Li, M., Liu, S., Liu, D., He, J., Bai, Y., Chen, F., Fu, L.: Residual local feature network for efficient super-resolution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 766–776 (2022)
25. Kong, X., Zhao, H., Qiao, Y., Dong, C.: Classsr: A general framework to accelerate super-resolution networks by data characteristic. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12016–12025 (2021)
26. Li, Y., Zhang, K., Gool, L.V., Timofte, R., et al.: Ntire 2022 challenge on efficient super-resolution: Methods and results. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (2022)
27. Li, Y., Zhang, Y., Timofte, R., Van Gool, L., Yu, L., Li, Y., Li, X., Jiang, T., Wu, Q., Han, M., et al.: Ntire 2023 challenge on efficient super-resolution: Methods and results. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 1921–1959 (2023)
28. Li, Z., Liu, Y., Chen, X., Cai, H., Gu, J., Qiao, Y., Dong, C.: Blueprint separable residual network for efficient image super-resolution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 833–843 (2022)
29. Liang, J., Cao, J., Sun, G., Zhang, K., Gool, L.V., Timofte, R.: Swinir: Image restoration using swin transformer. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW). pp. 1833–1844 (2021)

30. Lim, B., Son, S., Kim, H., Nah, S., Lee, K.M.: Enhanced deep residual networks for single image super-resolution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 1132–1140 (2017)
31. Liu, D., Wen, B., Fan, Y., Loy, C.C., Huang, T.S.: Non-local recurrent network for image restoration. In: Advances in Neural Information Processing Systems (NIPS). pp. 1680–1689. Montréal, Canada (2018)
32. Liu, J., Tang, J., Wu, G.: Residual feature distillation network for lightweight image super-resolution. In: Proceedings of the European Conference on Computer Vision Workshops (ECCVW). vol. 12537, pp. 41–55 (2020)
33. Liu, J., Zhang, W., Tang, Y., Tang, J., Wu, G.: Residual feature aggregation network for image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2356–2365 (2020)
34. Liu, N., Han, J., Yang, M.H.: Picanet: Learning pixel-wise contextual attention for saliency detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3089–3098 (2018)
35. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (ICLR) (2018)
36. Martin, D.R., Fowlkes, C.C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). pp. 416–425 (2001)
37. Mei, Y., Fan, Y., Zhou, Y.: Image super-resolution with non-local sparse attention. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3517–3526 (2021)
38. Niu, B., Wen, W., Ren, W., Zhang, X., Yang, L., Wang, S., Zhang, K., Cao, X., Shen, H.: Single image super-resolution via a holistic attention network. In: Proceedings of the European Conference on Computer Vision (ECCV). vol. 12357, pp. 191–207. Springer, Glasgow, UK (2020)
39. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems (NIPS). pp. 8024–8035. Vancouver, Canada (2019)
40. Rao, Y., Zhao, W., Tang, Y., Zhou, J., Lim, S.N., Lu, J.: Hornet: Efficient high-order spatial interactions with recursive gated convolutions. Advances in Neural Information Processing Systems (NIPS) **35**, 10353–10366 (2022)
41. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI). pp. 234–241. Springer (2015)
42. Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1874–1883 (2016)
43. Stergiou, A., Poppe, R., Kalliatakis, G.: Refining activation downsampling with softpool. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 10357–10366 (2021)
44. Wang, H., Bhaskara, V., Levinshtein, A., Tsogkas, S., Jepson, A.: Efficient super-resolution using mobilenetv3. In: Proceedings of the European Conference on Computer Vision Workshops (ECCVW). pp. 87–102. Springer (2020)
45. Wang, H., Chen, X., Ni, B., Liu, Y., Liu, J.: Omni aggregation networks for lightweight image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 22378–22387 (2023)

46. Wang, Y.: Edge-enhanced feature distillation network for efficient super-resolution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 777–785 (June 2022)
47. Wang, Y., Li, Y., Wang, G., Liu, X.: Multi-scale attention network for single image super-resolution. arXiv preprint arXiv:2209.14145 (2022)
48. Wang, Y., Su, T., Li, Y., Cao, J., Wang, G., Liu, X.: DDistill-SR: Reparameterized dynamic distillation network for lightweight image super-resolution. IEEE Transactions on Multimedia (2022)
49. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing **13**(4), 600–612 (2004)
50. Woo, S., Park, J., Lee, J.Y., Kweon, I.S.: Cbam: Convolutional block attention module. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 3–19 (2018)
51. Xia, B., Hang, Y., Tian, Y., Yang, W., Liao, Q., Zhou, J.: Efficient non-local contrastive attention for image super-resolution. arXiv preprint arXiv:2201.03794 (2022)
52. Yu, L., Li, X., Li, Y., Jiang, T., Wu, Q., Fan, H., Liu, S.: Dipnet: Efficiency distillation and iterative pruning for image super-resolution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 1692–1701 (2023)
53. Zeyde, R., Elad, M., Protter, M.: On single image scale-up using sparse-representations. In: Curves and Surfaces - 7th International Conference. vol. 6920, pp. 711–730 (2010)
54. Zhang, X., Zeng, H., Guo, S., Zhang, L.: Efficient long-range attention network for image super-resolution. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 649–667 (2022)
55. Zhang, X., Zeng, H., Zhang, L.: Edge-oriented convolution block for real-time super resolution on mobile devices. In: Proceedings of the ACM International Conference on Multimedia (MM). pp. 4034–4043. ACM, Virtual Event, China (2021)
56. Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: Proceedings of the European Conference on Computer Vision (ECCV). vol. 11211, pp. 294–310 (2018)
57. Zhao, H., Kong, X., He, J., Qiao, Y., Dong, C.: Efficient image super-resolution using pixel attention. In: Bartoli, A., Fusiello, A. (eds.) Proceedings of the European Conference on Computer Vision Workshops (ECCVW). vol. 12537, pp. 56–72. Springer, Glasgow, UK (2020)
58. Zhong, S., Huang, Z., Wen, W., Qin, J., Lin, L.: Asr: Attention-alike structural re-parameterization. arXiv preprint arXiv:2304.06345 (2023)
59. Zhou, L., Cai, H., Gu, J., Li, Z., Liu, Y., Chen, X., Qiao, Y., Dong, C.: Efficient image super-resolution using vast-receptive-field attention. In: Proceedings of the European Conference on Computer Vision Workshops (ECCVW). pp. 256–272 (2022)
60. Zhou, Z., Chao, J., Gong, J., Gao, H., Zeng, Z., Yang, Z.: Enhancing real-time super resolution with partial convolution and efficient variance attention. In: Proceedings of the ACM International Conference on Multimedia (MM). pp. 5348–5357 (2023)