Partial filter-Sharing: Improved Parameter-sharing Method for Single Image Super-Resolution Networks

Karam Park Nam Ik Cho
Department of ECE, INMC, Seoul National University

Abstract

Numerous deep learning techniques have been developed for Single Image Super-Resolution (SISR), leading to significant performance improvements. However, these techniques have also resulted in a substantial increase in parameter size. As a result, there is a growing interest in reducing network complexity for more practical usage while still maintaining high SR quality. One such method is parameter-sharing, which includes recursive, recurrent, and multi-scale learning approaches. However, sharing identical kernels across layers or up-scaling tasks can reduce the network's representational capacity. To address this, we propose Partial filter-Sharing (PS), a new parameter-sharing method that preserves the network's representational power more effectively than previous approaches. Instead of sharing a single filter, PS shares segments of filters, called partial filters, across layers. This approach enables parameter-sharing layers to use diverse filters for each layer or task, striking a balance between parameter efficiency and the network's representational ability without imposing excessive computational or parameter overhead. Furthermore, the PS framework provides precise control over the network's performance and complexity by adjusting the quantity of partial filters. Extensive experiments demonstrate that our PS framework outperforms traditional parameter-sharing super-resolution (SR) methods without incurring excessive additional parameters or computational cost. Our code is available here: https://github.com/saturnian77/Partial_filter-Sharing.

1. Introduction

Single Image Super-Resolution (SISR) is a technique aimed at improving the resolution of low-resolution (LR) images to generate high-resolution (HR) images. SISR has garnered significant attention in both academic and industrial fields due to its wide range of applications, including medical imaging [35], satellite imagery [43], and surveillance [36]. In recent years, deep learning-based SISR methods have made significant advancements, surpassing classical approaches [5,21,44,50]. The first notable method, SRCNN [10], demonstrated the exceptional performance of convolutional neural networks (CNNs) for SISR using a simple network with three convolutional layers. Subsequent research expanded the depth of networks, leading to substantial improvements in SR performance [20, 28, 39]. However, as the network depth increased, the number of parameters grew significantly, making these models less practical for real-world applications. As a result, there has been growing interest in exploring methods to reduce network parameters to more practical levels.

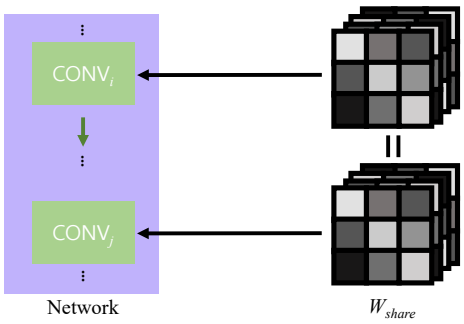
There are various techniques for reducing the number of parameters in neural networks. One approach is parameter-sharing, combined with a recursive or recurrent structure. These structures have a repetitive design, with layers sharing limited sets of filters. The use of recursive structures has been investigated in several studies [19, 41], which have demonstrated significant reductions in network parameters relative to their depth. However, the use of fixed filters in SR networks imposes significant constraints on their capacity, resulting in reduced performance. Another example of parameter-sharing frameworks is multi-scale learning methods [2, 20, 41], which utilize a single network for all scales, eliminating the need to train individual models for different scale factors. For example, CARN [2] reduces the number of parameters by using a single network to perform super-resolution (SR) across scale factors from 2 to 4. However, this approach introduces a challenge: the network must handle various up-scaling tasks with the same filters, which hinders learning appropriate filters for each specific scale.

In this paper, we introduce a new approach called Partial filter-Sharing (PS) to overcome the limitations of previous parameter-sharing SR methods. PS employs a novel filter assignment strategy that involves using fragments of filters, which we refer to as **"partial filters."** To reconstruct the weights of convolutional layers from these partial filters, network layers use **"coefficient matrices,"** which contain the information necessary to merge partial filters into complete filters. By utilizing coefficient matrices,

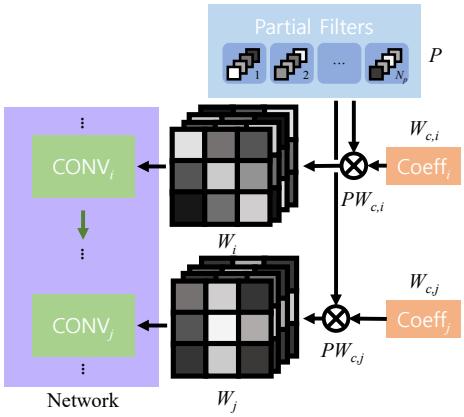
the network can take advantage of parameter-sharing while also employing a variety of filters for each layer or scale. This approach mitigates the risk of compromising representational ability due to reduced parameters, a common issue in traditional parameter-sharing methods. Fig. 1 provides a visual explanation of the differences between PS and traditional parameter-sharing methods. In general, PS improves the performance of parameter-sharing SR by enabling the network to utilize diverse filters in addition to the benefits of parameter-sharing. Furthermore, PS allows for customization of network performance and complexity by adjusting the number of partial filters, demonstrating its ability to adapt network functionality to specific applications or settings.

The contributions of this paper can be summarized as follows:

- We introduce a novel technique for parameter-sharing SR that enables the network to utilize diverse filters, thereby enhancing its representational ability.
- Our method can be applied to existing parameter-sharing SR networks, allowing for a trade-off between network complexity and performance by adjusting the number of partial filters.
- We provide a guideline for determining partial filter shapes within the PS framework, aiming to strike a balance between performance and efficiency.
- We conduct extensive experiments to analyze the behavior of our new network structure with PS. The results demonstrate that our partial filter-sharing technique can significantly improve the performance and efficiency of previous parameter-sharing SR methods.



(a) Traditional Parameter-sharing Method



(b) Proposed Method

Figure 1. Comparison of the traditional parameter-sharing method and the proposed method: In conventional approaches, parameter-sharing layers perform convolution operations using identical sets of filters. In contrast, our approach employs partial filters and coefficient matrices, enabling the reconstruction of diverse filters tailored for each layer. This diversity in filters enriches the network's representations and improves its performance.

2. Related Works

Single Image Super-Resolution The field of image processing has been significantly impacted by the advancement of deep learning technologies, including the SISR field. Dong *et al*. [10] introduced the first CNN for the SR and demonstrated that CNNs have great potential. Shi *et al*. [39] proposed the pixel-shuffle layer to handle the up-sampling process efficiently. Influenced by ResNet [12], Ledig *et al*. proposed SRResNet [24], showing that residual connections are an effective method to improve SR performance. The combination of pixel-shuffle operations and residual connections led to dramatic improvements in the SR network performance through depth-wise expansion [28]. Building on depth-wise expansion, attention mechanisms [45] further enhanced network performance by allowing the network to consider channel and spatial correlations of meaningful features [9, 54, 56].

Meanwhile, as network performance improves with increased complexity, the demand for practical SR networks has also grown. Recursive and recurrent structure [19, 27, 41, 42] are among the options to reduce network complexity, as they share the same filters through an iterative structure. However, these structures repetitively reuse the same filter operations to refine features, which can compromise their representational ability relative to their computational complexity. To address this, researchers have proposed low-complexity structures with high representational ability to enhance network performance efficiently. These networks typically feature shallow depths and fewer channels to minimize complexity. These approaches refine features using efficient and compact computational units, such as information distillation [16], feature distillation [29], lattice blocks [30], and feature mining units [47]. Additionally,

to counteract the limited representation capacity caused by shallow depths, they incorporate complex handcrafted residual connections [13, 34], dense connections [16, 29], or locally adaptive kernels [48]. While these methods have shown promising results, they lack versatility for broader applications, as they are difficult to implement in networks with diverse structural configurations.

In addition to the efficient network designs mentioned above, alternative approaches aim to reduce complexity with minimal performance degradation. For instance, pruning [37, 55] removes redundant parameters to decrease inference time and network complexity while maintaining performance. Quantization [8, 25] replaces parameters represented in floating-point with short-length integers to improve network efficiency. Although pruning and quantization are also common techniques for improving SR network efficiency, these methods are typically applied to pre-trained networks, which differs significantly from the focus of our work.

Parameter-sharing in SR Weight-sharing CNNs, such as recursive and recurrent networks, typically have fewer trainable parameters compared to conventional CNNs [23]. These efficient structures have been widely adopted in various tasks, including scene parsing [40], object recognition [4], natural language processing [18], and image super-resolution [19, 27, 41, 42]. However, they often face limitations in representational capacity, which can hinder their reconstruction performance.

Meanwhile, multi-scale learning is another example of network parameter reduction through parameter-sharing. This approach allows a single network to handle multiple scales, instead of requiring separate models for each scale. While this method is efficient in terms of parameters, its performance per parameter is generally lower compared to single-scale models, as the use of the same sets of filters across different up-scaling tasks limits the network's ability to learn scale-specific features. In contrast to previous parameter-sharing methods, the PS technique incorporates diverse filters within a parameter-sharing framework. By using partial filters and coefficient matrices, separate convolution operations are enabled for each layer, thereby improving the representational capacity of the parameter-sharing network.

Various Convolutions Sainath *et al*. [38] introduced low-rank matrix factorization, a method that represents a weight matrix as the product of two smaller matrices, achieving a reduction in parameters by 30 to 50%. MobileNet [14] presented depth-wise separable convolution, a technique that splits convolution into group and pixel-wise convolutions, significantly enhancing computational efficiency. Input-dependent convolution improves network performance by adaptively modifying filters based on the input. Li *et al*. [26] proposed a method that merges the

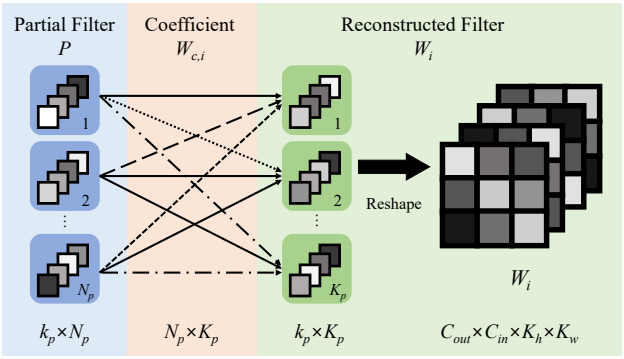


Figure 2. Visualized explanation of the proposed method. The matrix multiplication of partial filter P and coefficient matrix $W_{c,i}$ gives reconstructed filter W_i.

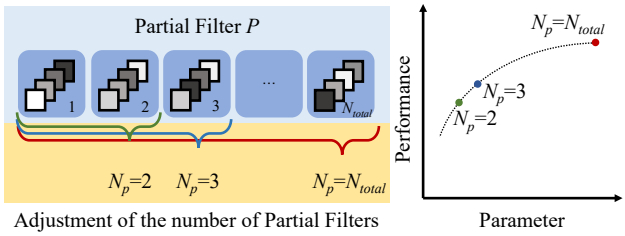


Figure 3. Visualized explanation of the parameter adjustment with PS.

predefined sets of filters to apply distinct filters to each pixel in the RGB space, where the network dynamically calculates coefficients to blend these filters based on the input image. Conditional Convolution [49] and Dynamic Convolution [6] generate dynamic kernels by aggregating multiple kernels, referred to as experts. An attention module computes coefficients to integrate these kernels based on input features. These approaches increase model complexity without expanding the network's depth or width. In contrast, our PS technique improves the performance of parameter-sharing networks with minimal extra parameters, avoiding additional computations and parameter overload by not relying on the inclusion of more experts, thus distinguishing it from dynamic convolution.

3. Proposed Method

3.1. Partial filter-Sharing Method

A convolution layer is represented by $W \in \mathbb{R}^{C_{out} \times C_{in} \times K_h \times K_w}$, where W contains the weights used to process input features. To introduce filter diversity in a parameter-sharing framework, PS shares several segments of filters, termed partial filters, instead of using whole filter weights. Let $p \in \mathbb{R}^{c_{out} \times c_{in} \times k_h \times k_w}$ be a partial filter, where c_{out}, c_{in}, k_h, k_w are the dimensions corresponding to the full filter's C_{out}, C_{in}, K_h, K_w. A

collection of N_p partial filters is represented by the partial filter matrix $P \in \mathbb{R}^{k_p \times N_p}$, where k_p denotes the size of each partial filter, $c_{out} \cdot c_{in} \cdot k_h \cdot k_w$.

On the other hand, the i-th parameter-sharing layer has a coefficient matrix $W_{c,i} \in \mathbb{R}^{N_p \times K_p}$ instead of the full filter, where K_p is determined by the size of the partial filter p:

$$K_p = (C_{out} \cdot C_{in} \cdot K_h \cdot K_w)/k_p. \quad (1)$$

Through matrix multiplication, the coefficient matrix $W_{c,i}$ reconstructs the weight W_i to be used within the network:

$$W_i = Reshape(PW_{c,i}), \quad (2)$$

where $Reshape(\cdot)$ is a function that rearranges the result of the matrix multiplication to match the filter shape $C_{out} \times C_{in} \times K_h \times K_w$. Through this process, parameter-sharing networks utilize separate filters for each layer by sharing partial filters rather than full filters, while the coefficient matrix enables the reconstruction of the full filters. In Fig. 2, we provide a visual explanation of the PS framework and its filter reconstruction process.

3.2. Parameter Adjustment

Besides filter diversification with PS, another key benefit is the precise control over network capacity. Let W_{total} be the total weights of N_{layers} convolutional layers. By dividing the size of W_{total} by the size of a partial filter p, we can determine the total number of partial filters N_{total} needed to fully represent W_{total} as in the original network:

$$N_{total} = Size(W_{total})/k_p. \quad (3)$$

The network's capacity can be adjusted by modifying the number of partial filters N_p. For example, by setting N_p smaller than N_{total}, we can reduce the number of partial filters required to represent the N_{layers} layers, thus decreasing the parameter size. This approach enables precise control over the network's capacity, depending on the size of the partial filters. A visual explanation of the parameter adjustment through PS is provided in Fig. 3.

On the other hand, representing the network with PS introduces computational and parameter overhead due to matrix multiplication between $P \in \mathbb{R}^{k_p \times N_p}$ and $W_c \in \mathbb{R}^{N_p \times K_p}$, which is required to restore the weights. The resulting variations in parameter and computational cost can be calculated as follows:

$$\begin{aligned} Params &= N_{layers} \cdot K_p \cdot N_p - (N_{total} - N_p) \cdot k_p, \\ Multi\text{-}Adds &= N_{layers} \cdot K_p \cdot k_p \cdot (N_p - 1). \end{aligned} \quad (4)$$

By considering Eq. (4), we can determine efficient partial filter shapes for networks that apply PS, tailored to their structural characteristics.

3.3. Filter Shape Decision

The shape of the partial filter plays a critical role in determining both the additional parameters and the overall performance of the network. Each partial filter is smaller than the full filter, but multiple partial filters can be combined to reconstruct the original filter. For instance, if the target filter is 64×64×3×3, the filter can be divided spatiallly into 64×64×1×1. Alternatively, it could be split along the input or output channels, as in 1×64×3×3 or 64×1×3×3. A third option is to partition the filter in both the spatial and the channel directions, such as 32×64×1×1. The smaller the size of the partial filter, the more variations are possible in representing the entire filter, thereby improving the representational capacity of the network. However, if the partial filter is too small, the coefficient matrix can lead to a significant increase in parameters, as shown in Eq. (4). In Sec. 4.3, we explore how the shape of the partial filter impacts network performance and offer guidelines for selecting the appropriate partial filter shape.

4. Experimental Results

4.1. Settings

Datasets and Evaluation All models in the experiments presented in this paper are trained on the DIV2K dataset [1], which consists of 800 images. For training, RGB patches of size 48×48 are used, with data augmentation performed through flipping and rotation. To evaluate the performance, we use four benchmark datasets: Set5 [3], Set14 [51], B100 [31], and Urban100 [15]. Evaluation is based on peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) on the Y channel. For assessing computational complexity, Multi-Adds are calculated with the output shape set to 720p (1280×720).

Training We use the official code provided by the authors of CARN [2] and SRFBN [27] to train and implement their PS counterparts. Since a PyTorch implementation by the authors is unavailable, we adapt the CARN [2] code to implement DRRN [41], which also performs multi-scale learning. The proposed method is applied to the parameter-sharing convolution layers within the building blocks of the SR networks. We use the L1 loss, following conventional SR methods [2, 27, 28]. All experiments are conducted on a single NVIDIA TITAN XP GPU. Models implemented with the proposed method are denoted as "-PS." Additional network structure visualizations and implementation details can be found in the Supplementary Material.

4.2. Application to Parameter-Sharing SR Methods

In this section, we provide qualitative and quantitative comparisons of recursive [41], recurrent [27], and multi-

Scale	Methods	N_p/N_{total}	Params	Set5	Set14	B100	Urban100
2	SRFBN-S$^{\triangle}$ [27]	-	0.28M	37.78/0.9597	33.35/0.9156	32.00/0.8970	31.41/0.9207
	SRFBN-S-PS	**3/4**	**0.28M**	**37.74/0.9596**	**33.34/0.9153**	**32.03/0.8979**	**31.47/0.9213**
	DRRN_B1U9* [41]	-	0.30M	37.66/0.9589	33.19/0.9133	32.01/0.8969	31.02/0.9164
	DRRN_B1U25* [41]	-	0.30M	37.74/0.9591	33.23/0.9136	32.05/0.8973	31.23/0.9188
	DRRN_B1U9*-PS	**1**	**0.31M**	**37.79/0.9601**	**33.41/0.9158**	**32.07/0.8986**	**31.73/0.9240**
	CARN* [2]	-	1.59M	37.76/0.9590	33.52/0.9166	32.09/0.8978	31.92/0.9256
	CARN*-PS	**6/9**	**1.43M**	**37.86/0.9603**	**33.59/0.9180**	**32.13/0.8995**	**31.99/0.9271**
	CARN*-PS	**8/9**	**1.59M**	**37.90/0.9603**	**33.62/0.9180**	**32.16/0.8999**	**32.09/0.9280**
3	DRRN_B1U9* [41]	-	0.30M	33.93/0.9234	29.94/0.8339	28.91/0.7992	27.38/0.8331
	DRRN_B1U25* [41]	-	0.30M	34.03/0.9244	29.96/0.8349	28.95/0.8004	27.53/0.8378
	DRRN_B1U9*-PS	**1**	**0.31M**	**34.12/0.9251**	**30.09/0.8374**	**28.97/0.8018**	**27.85/0.8440**
	SRFBN-S$^{\triangle}$ [27]	-	0.38M	34.20/0.9255	30.10/0.8372	28.96/0.8010	27.66/0.8415
	SRFBN-S-PS	**19/24**	**0.37M**	**34.24/0.9256**	**30.18/0.8386**	**29.01/0.8027**	**27.84/0.8451**
	CARN* [2]	-	1.59M	34.29/0.9255	30.29/0.8407	29.06/0.8034	28.06/0.8493
	CARN*-PS	**6/9**	**1.43M**	**34.31/0.9265**	**30.33/0.8424**	**29.07/0.8053**	**28.11/0.8517**
	CARN*-PS	**8/9**	**1.59M**	**34.37/0.9268**	**30.35/0.8430**	**29.10/0.8059**	**28.19/0.8534**
4	DRRN_B1U9* [41]	-	0.30M	31.58/0.8864	28.18/0.7701	27.35/0.7262	25.35/0.7576
	DRRN_B1U25* [41]	-	0.30M	31.68/0.8888	28.21/0.7720	27.38/0.7284	25.44/0.7638
	DRRN_B1U9*-PS	**1**	**0.31M**	**31.85/0.8897**	**28.36/0.7755**	**27.41/0.7301**	**25.72/0.7718**
	SRFBN-S$^{\triangle}$ [27]	-	0.48M	31.98/0.8923	28.45/0.7779	27.44/0.7313	25.71/0.7719
	SRFBN-S-PS	**5/6**	**0.48M**	**32.04/0.8932**	**28.51/0.7796**	**27.51/0.7338**	**25.88/0.7786**
	CARN* [2]	-	1.59M	32.13/0.8937	28.60/0.7806	27.58/0.7349	26.07/0.7837
	CARN*-PS	**2/3**	**1.43M**	**32.15/0.8939**	**28.62/0.7825**	**27.58/0.7371**	**26.09/0.7864**
	CARN*-PS	**8/9**	**1.59M**	**32.17/0.8943**	**28.61/0.7831**	**27.59/0.7381**	**26.15/0.7890**

Table 1. Quantitative Comparison of parameter-sharing SR models and the proposed method. Multi-scale learning methods are marked with $*$ symbol. Models trained with DF2K dataset are marked with $\triangle$ symbol. Our methods are marked in **bold**.

Model	Multi-Adds	N_p/N_{total}	Time
SRFBN-S	852.9G	-	0.0933s
SRFBN-S-PS	852.9G+0.06G	**5/6**	**0.0938s**
DRRN_B1U9	6.8T	-	35.05s
DRRN_B1U9-PS	6.8T+0.05G	**1**	**35.09s**
CARN	90.9G	-	3.439s
CARN-PS	90.9G+0.09G	**8/9**	**3.506s**

Table 2. Comparison of inference speed on Urban100 ($\times 4$).

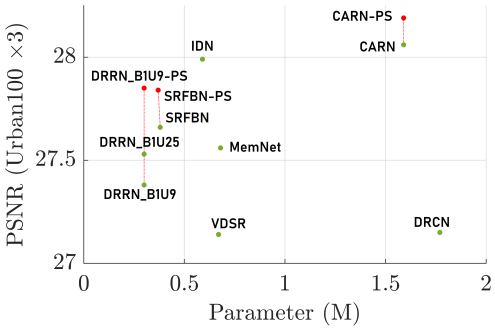


Figure 4. Comparison of performance on Urban100 ($\times 3$) dataset.

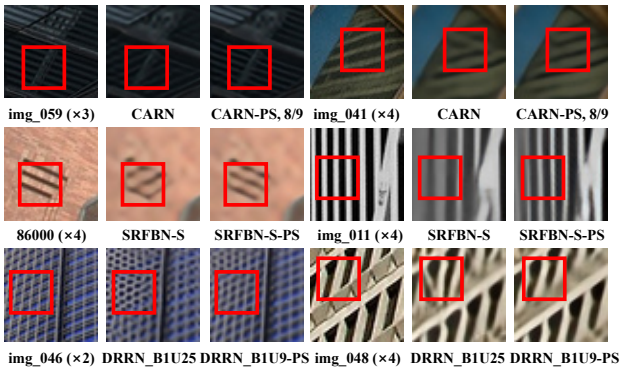


Figure 5. Visualized results of baselines and PS methods.

scale learning [2, 41] methods, as shown in Tab. 1. We employ well-known parameter-sharing SR methods: DRRN_B1U9 [41], CARN [2], and SRFBN-S [27]. While these methods do not achieve the current state-of-the-art (SOTA) performance, they are selected for comparison due to their established reputation and relatively low complexity, making them suitable for practical use. To evaluate the enhancements introduced by the proposed method, we compare the models implemented with the PS framework to their original counterparts. We adjust the N_p/N_{total} ratio to align the parameters with those of the original network, thus minimizing the impact of additional parameters from the coefficient matrix on network performance.

As shown in Tab. 1, models with PS achieve significant performance improvements over their original counterparts. Even with 10% fewer parameters, CARN-PS with a 2/3 ratio outperforms the original model, and adjusting the ratio to 8/9 leads to even better results. In multi-scale

Methods	Multi-Adds	N_p/N_{total}	Params	Set5	Set14	B100	Urban100
EDSR-b-Fixed	0.3T	-	0.26M	37.50/0.9589	32.91/0.9133	31.89/0.8962	30.99/0.9166
EDSR-b-PS	0.3T+0.02G	1/16	0.27M	37.65/0.9597	33.11/0.9156	32.02/0.8980	31.44/0.9219
RCAN_G5B10-Fixed	1.0T	-	0.48M	37.73/0.9598	33.31/0.9156	32.01/0.8983	31.62/0.9234
RCAN_G5B10-PS	1.0T+0.06G	1/50	0.50M	37.79/0.9601	33.43/0.9164	32.08/0.8991	31.85/0.9256

Table 3. Quantitative comparisons of fixed filter and PS models on $\times 2$ scale.

Methods	N_p/N_{total}	Params	Set5	Urban100
Baseline	-	1.37M	38.01/0.9608	32.23/0.9296
Recursive	-	0.26M	37.50/0.9589	30.99/0.9166
64×64×1×1	1/16	0.27M	37.65/0.9597	31.44/0.9219
	2/16	0.35M	37.77/0.9601	31.68/0.9243
	4/16	0.51M	37.89/0.9604	31.93/0.9268
	6/16	0.66M	37.97/0.9606	32.03/0.9277
	8/16	0.82M	38.00/0.9608	32.13/0.9287
	10/16	0.98M	38.01/0.9608	32.17/0.9289
	12/16	1.14M	38.00/0.9608	32.19/0.9291
	14/16	1.30M	38.00/0.9608	32.22/0.9294
64×1×3×3	1/32	0.36M	37.85/0.9603	31.69/0.9244
	1/16	0.53M	37.94/0.9606	31.99/0.9274
	2/16	0.86M	38.01/0.9608	32.15/0.9290
64×32×1×1	1/16	0.29M	37.74/0.9598	31.41/0.9215
	4/16	0.57M	37.92/0.9604	31.94/0.9270
	8/16	0.95M	37.98/0.9608	32.12/0.9286
	12/16	1.32M	38.02/0.9608	32.20/0.9291
64×8×3×3	1/16	0.27M	37.71/0.9597	31.33/0.9206
	4/16	0.50M	37.86/0.9604	31.80/0.9255
	8/16	0.81M	37.95/0.9606	32.01/0.9274
	12/16	1.12M	37.99/0.9607	32.14/0.9289

Table 4. Experimental results of EDSR-baseline-PS ($\times 2$) with different partial filter shapes and N_p/N_{total} settings. Additional experimental results are available in the Supplementary Material.

SR networks, which share filters across all scales, there is a risk of biased learning toward a specific scale, which degrades performance on other scales. In contrast, PS allows the network to use different filters for each scale, thus enhancing its representational ability. Although SRFBN-S-PS still shares a limited sets of filters in most layers, applying PS to the downsample block allows it to achieve comparable performance for a $\times 2$ scale factor and superior performance for other factors, despite discrepancies in the training dataset. It is worth noting that original SRFBN-S models are trained on the DF2K dataset, which contains 3550 images. Finally, DRRN_B1U9-PS outperforms the original model and surpasses the deeper DRRN_B1U25. Since DRRN is both a recursive and multi-scale learning model, the representational capacity of its filters is inherently limited, which negatively impacts its performance. The proposed PS method addresses this limitation, leading to a significant improvement.

Additionally, to evaluate the additional cost and latency introduced by the proposed method, we compare the inference speed of CARN, DRRN, and SRFBN on the Urban100 ($\times 4$). Tab. 2 shows the inference times for networks using PS and their original counterparts. We observe less than a 2% difference in running time between the models with the proposed method and the original ones, indicating that the impact on inference time is negligible. Together with the minimal computational cost outlined in Tabs. 2 and 5, these results highlight the cost-efficiency of PS. A visual comparison between the models with the proposed method and the original counterparts is provided in Fig. 5.

Furthermore, to investigate the versatility of PS across different network structures, we experiment with models that do not employ a parameter-sharing framework. Specifically, we select EDSR [28] and RCAN [54] as representatives of lightweight and large-scale SR models, respectively. These models are commonly used as baselines in the design and application of SR networks [9, 13, 33, 46, 52]. We compare the performance of these models between the case of using fixed filters in a recursive manner and employing PS. In the fixed-filter models, convolution layers are shared across repeated residual blocks, while in the models with PS, a specific number of partial filters are used to match the weight size of the fixed-filter counterparts. For RCAN, to enable training with fixed filters, we reduce the number of residual groups from 10 to 5 and the number of residual blocks from 20 to 10. In the PS networks, all parameter-sharing layers use filters divided spatially as $C_{out} \times C_{in} \times 1 \times 1$. As shown in Tab. 3, the results reveal that our approach consistently outperforms the fixed-filter models, despite sharing the same network architecture. This demonstrates that PS adds a level of flexibility to the network, leading to improved performance.

4.3. Ablation and Analysis

In this subsection, we explore how the shape of the partial filter and the number of partial filters affect network performance. We also examine the influence of coefficient matrices on the network's functionality in the proposed method. The experimental results are summarized in Tab. 4 and Fig. 7. Since the EDSR-baseline consists of 32 convolutional layers in 16 residual blocks, we adjust the ratio in increments of 1/32 and 1/16. Full experimental results can be found in the Supplementary Material.

Filter Shape & N_p/N_{total} Adjustment Firstly, we investigate the impact of partial filter shape and ratio on the performance of PS networks. As shown in

N_p/N_{total}	64×64×1×1 ΔMulti-Adds/ΔParameters	64×1×3×3 ΔMulti-Adds/ΔParameters
1/32	+9.43M/-1.14M	+74.32M/-1.01M
1/16	+20.05M/-1.10M	+149.82M/-0.84M
2/16	+41.29M/-1.02M	+300.81M/-0.51M
4/16	+83.76M/-0.86M	+602.80M/+0.16M
6/16	+126.22M/-0.71M	+904.79M/+0.84M
8/16	+168.69M/-0.55M	+1206.78M/+1.51M
10/16	+211.16M/-0.39M	+1508.77M/+2.18M
12/16	+253.62M/-0.23M	+1810.76M/+2.85M
14/16	+296.09M/-0.07M	+2112.75M/+3.52M

Table 5. Comparison of 64×64×1×1 and 64×1×3×3 on variation in Multi-Adds and parameters of EDSR-baseline-PS.

Methods	N_p/N_{total}	Δ Params	Params	Set5	Urban100
FSRCNN [11]	-	-	0.01M	37.00	29.88
MAFFSRN [32]	-	-	0.40M	37.97	31.96
FALSR [7]	-	-	0.41M	37.66	31.24
EDSR-b-PS	1/16	-0.84M	**0.53M**	**37.94**	**31.99**
IDN [17]	-	-	0.55M	37.83	31.27
ECBSR [53]	-	-	0.60M	37.90	31.71
RCAN-PS	1/100	-14.83M	**0.61M**	**38.03**	**32.51**
DRSAN [34]	-	-	0.65M	38.08	32.16
VDSR [20]	-	-	0.66M	37.53	30.76
MemNet [42]	-	-	0.68M	37.78	31.31
IMDN [16]	-	-	0.69M	38.00	32.17
LatticeNet [30]	-	-	0.76M	38.15	32.43
LapSRN [22]	-	-	0.81M	37.52	30.41

Table 6. Comparison of the SOTA lightweight methods and PS models (×2).

Tab. 4, the performance of the network approaches that of the baseline as the ratio increases. Notably, for a simple dataset like Set5, performance comparable to the original network is achieved even with only 60% of the parameters, as seen with the $64\times64\times1\times1$ configuration. Among the configurations, $64\times1\times3\times3$ achieves the highest performance at the same ratio, followed by $64\times64\times1\times1$. When partial filters are split both spatial- and channel-wise, performance tends to be lower than when the filters are split in just one direction. This is because, unlike conventional filters that learn both channel and spatial correlations simultaneously, any correlation not captured by the partial filter must be learned by the coefficient matrix. However, since the coefficient matrix typically has a limited number of parameters and capacity, it is challenging to capture the complex correlations between channel and spatial directions when the partial filter is split in both directions. Therefore, in this study, we use partial filters that are split only in either the spatial or the channel direction.

Computational & Parameter Overhead On the other hand, the side effects of additional parameters resulting from the partial filter shape must be considered. As the partial filter size decreases, K_p increases, which leads to a higher overall parameter increase, as shown in Eq. (4). As observed in Tab. 5, for $64\times1\times3\times3$, there is a noticeable increase in both computational cost and parameter overhead compared to other methods with the same ratio. While it shows superior performance, the permissible range of N_p/N_{total} for parameter reduction becomes quite narrow. We present the variations in computational cost and model parameters for EDSR-baseline-PS in Tab. 5, with different values of N_p/N_{total}. Considering that most networks operate in the range of several G Multi-Adds, the computational overhead required for filter reconstruction is relatively minor.

Study of Network Capacity Additionally, we evaluate the capacity of PS networks by comparing their performance with other parameter reduction techniques and lightweight networks [7, 11, 14, 16, 17, 20, 22, 30, 32, 34, 42, 53]. We find that all methods using PS with a ratio of 1/16 outperform the recursive model. As shown in Fig. 6, models with this ratio show performance improvements of 0.3 to 1.0 dB over the recursive model. By allowing each layer to use diverse filters, derived from combinations of partial filters, the proposed method enhances representational ability compared to recursive models with fixed shared filters. Furthermore, we compare our method with the well-known parameter reduction technique, depth-wise separable convolution (D.S.Conv) from MobileNet [14]. For the SR task, we omit the normalization layers and apply D.S.Conv only to the convolutional layers in the residual blocks of the baseline model. As shown in Fig. 7, the performance of networks employing our method is comparable to that of networks using depth-wise separable convolution. These results demonstrate that PS effectively preserves representational capacity during parameter reduction. Moreover, to evaluate how well the performance of networks with a small N_p/N_{total} ratio in PS is preserved, we compare models employing PS [28, 54] against lightweight methods [7, 11, 16, 17, 20, 22, 30, 32, 34, 42, 53] with a similar number of parameters, as shown in Tab. 6. To assess performance under extreme parameter reduction, we set the filter shape to $64\times64\times1\times1$ and the ratio to 1/100 for RCAN-PS. As shown in Fig. 8, models implemented with PS demonstrate competitive performance compared to other lightweight methods with a similar parameter count. These results suggest that the proposed approach, despite reducing parameters and the network's overall capacity, successfully retains representational ability comparable to that of lightweight methods.

Filter Shape Decision Strategy In summary, to strike a balance between performance and efficiency, we determine the partial filter shape as follows:

- Split the filters in either the channel direction ($C\times1\times K_h\times K_w$) or the spatial direction

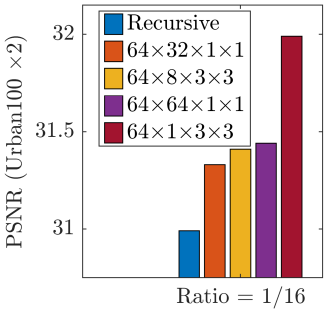


Figure 6. Comparison of various partial filter shapes with the ratio of 1/16.

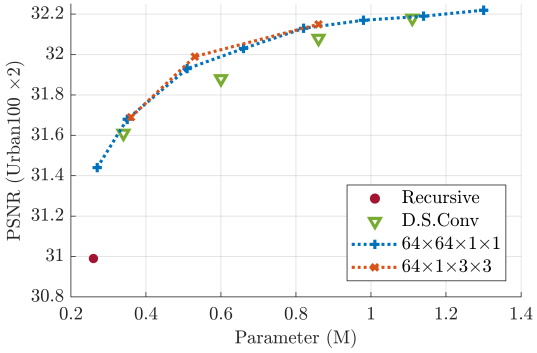


Figure 7. Comparison of parameter reduction methods and PS. D.S.Conv represents EDSR-baselines with 4, 8, 12, and 16 residual blocks consisting of depth-wise separable convolution.

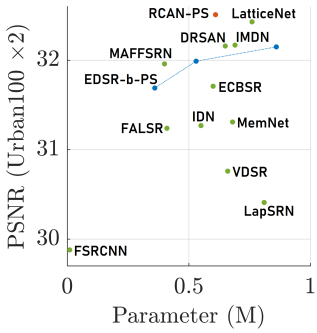


Figure 8. Comparison of PS methods with extremely low N_p/N_{total} setting and lightweight SR methods.

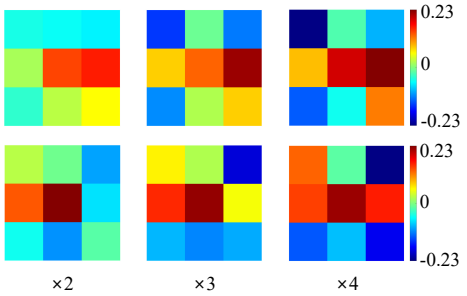


Figure 9. Visualized reconstructed kernels in DRRN-PS. The kernel values are collected from the two channels of the first convolutional layer in the first block.

$(C\times C\times 1\times 1)$.

- If the kernel size exceeds the channel size ($K_h \cdot K_w > C$), split the filters in the channel direction.
- If the channel size exceeds the kernel size ($K_h \cdot K_w < C$), split the filters in the spatial direction.

Coefficient Matrices Coefficient matrices enable parameter-sharing networks to employ diverse kernels across layers. As shown in Fig. 9, the reconstructed kernels for different up-scaling tasks in DRRN-PS illustrate that each task uses distinct kernels, unlike DRRN, which shares the same filter across all scales. We further investigate the advantages of coefficient matrices by swapping them within the DRRN-PS model. Tab. 7 shows a significant degradation in image quality when the matrices are swapped, highlighting that DRRN-PS learns task-specific filters, unlike traditional networks that use shared filters. This demonstrates that our method supports distinct filter operations for each scale. Additional experimental results can be found in the Supplementary Material.

Coeff.	Task	DRRN-PS PSNR/ΔPSNR
	×2	31.73/00.00
×2	×3	24.74/-03.11
	×4	23.24/-02.48
	×2	22.71/-09.02
×3	×3	27.85/00.00
	×4	23.64/-02.08
	×2	19.28/-12.45
×4	×3	22.80/-05.05
	×4	25.72/00.00

Table 7. The experimental results of the up-scaling task with different up-scaling coefficient matrices evaluated on Urban100. Coeff. represents coefficient matrices.

5. Conclusion

We have introduced a novel parameter-sharing technique for SISR networks, called Partial filter-Sharing. This method allows the network to utilize a variety of filters, thereby enhancing its representational capacity. By adjusting the N_p/N_{total} ratio, it is possible to precisely control the network's capacity according to specific requirements. Our experimental results demonstrate that our approach improves the performance of existing parameter-sharing networks for SISR, while maintaining a similar number of parameters.

Acknowledgement

This research was supported in part by Samsung Electronics Co., Ltd., and in part by the BK21 FOUR program of the Education and Research Program for Future ICT Pioneers, Seoul National University in 2024.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 126–135, 2017. 4

[2] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 252–268, 2018. 1, 4, 5

[3] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. 2012. 4

[4] Hieu Minh Bui, Margaret Lech, Eva Cheng, Katrina Neville, and Ian S Burnett. Using grayscale images for object recognition with convolutional-recursive neural network. In *2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)*, pages 321–325. IEEE, 2016. 3

[5] Hong Chang, Dit-Yan Yeung, and Yimin Xiong. Super-resolution through neighbor embedding. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–I. IEEE, 2004. 1

[6] Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11030–11039, 2020. 3

[7] Xiangxiang Chu, Bo Zhang, Hailong Ma, Ruijun Xu, and Qingyuan Li. Fast, accurate and lightweight super-resolution with neural architecture search. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 59–64. IEEE, 2021. 7

[8] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. *Advances in neural information processing systems*, 28, 2015. 3

[9] Tao Dai, Jianrui Cai, Yongbing Zhang, Shu-Tao Xia, and Lei Zhang. Second-order attention network for single image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11065–11074, 2019. 2, 6

[10] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015. 1, 2

[11] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *European conference on computer vision*, pages 391–407. Springer, 2016. 7

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[13] Xiangyu He, Zitao Mo, Peisong Wang, Yang Liu, Mingyuan Yang, and Jian Cheng. Ode-inspired network design for single image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1732–1741, 2019. 3, 6

[14] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 3, 7

[15] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5197–5206, 2015. 4

[16] Zheng Hui, Xinbo Gao, Yunchu Yang, and Xiumei Wang. Lightweight image super-resolution with information multi-distillation network. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 2024–2032, 2019. 2, 3, 7

[17] Zheng Hui, Xiumei Wang, and Xinbo Gao. Fast and accurate single image super-resolution via information distillation network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 723–731, 2018. 7

[18] Ozan Irsoy and Claire Cardie. Deep recursive neural networks for compositionality in language. *Advances in neural information processing systems*, 27, 2014. 3

[19] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1637–1645, 2016. 1, 2, 3

[20] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR Oral)*, June 2016. 1, 7

[21] Kwang In Kim and Younghee Kwon. Single-image super-resolution using sparse regression and natural image prior. *IEEE transactions on pattern analysis and machine intelligence*, 32(6):1127–1133, 2010. 1

[22] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Fast and accurate image super-resolution with deep laplacian pyramid networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(11):2599–2613, 2018. 7

[23] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 3

[24] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017. 2

[25] Huixia Li, Chenqian Yan, Shaohui Lin, Xiawu Zheng, Baochang Zhang, Fan Yang, and Rongrong Ji. Pams:

Quantized super-resolution via parameterized max scale. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*, pages 564–580. Springer, 2020. 3

[26] Wenbo Li, Kun Zhou, Lu Qi, Nianjuan Jiang, Jiangbo Lu, and Jiaya Jia. Lapar: Linearly-assembled pixel-adaptive regression network for single image super-resolution and beyond. *Advances in Neural Information Processing Systems*, 33:20343–20355, 2020. 3

[27] Zhen Li, Jinglei Yang, Zheng Liu, Xiaomin Yang, Gwanggil Jeon, and Wei Wu. Feedback network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3867–3876, 2019. 2, 3, 4, 5

[28] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017. 1, 2, 4, 6, 7

[29] Jie Liu, Jie Tang, and Gangshan Wu. Residual feature distillation network for lightweight image super-resolution. In *Computer vision–ECCV 2020 workshops: Glasgow, UK, August 23–28, 2020, proceedings, part III 16*, pages 41–55. Springer, 2020. 2, 3

[30] Xiaotong Luo, Yuan Xie, Yulun Zhang, Yanyun Qu, Cuihua Li, and Yun Fu. Latticenet: Towards lightweight image super-resolution with lattice block. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 272–289. Springer, 2020. 2, 7

[31] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423. IEEE, 2001. 4

[32] Abdul Muqeet, Jiwon Hwang, Subin Yang, JungHeum Kang, Yongwoo Kim, and Sung-Ho Bae. Multi-attention based ultra lightweight image super-resolution. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 103–118. Springer, 2020. 7

[33] Ben Niu, Weilei Wen, Wenqi Ren, Xiangde Zhang, Lianping Yang, Shuzhen Wang, Kaihao Zhang, Xiaochun Cao, and Haifeng Shen. Single image super-resolution via a holistic attention network. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*, pages 191–207. Springer, 2020. 6

[34] Karam Park, Jae Woong Soh, and Nam Ik Cho. Single image super-resolution with dynamic residual connection. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 1–8. IEEE, 2021. 3, 7

[35] Sharon Peled and Yehezkel Yeshurun. Superresolution in mri: application to human white matter fiber tract visualization by diffusion tensor imaging. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 45(1):29–35, 2001. 1

[36] Pejman Rasti, Tonis Uiboupin, Sergio Escalera, and Gholamreza Anbarjafari. Convolutional neural network super resolution for face recognition in surveillance monitoring. In *International conference on articulated motion and deformable objects*, pages 175–184. Springer, 2016. 1

[37] Russell Reed. Pruning algorithms-a survey. *IEEE transactions on Neural Networks*, 4(5):740–747, 1993. 3

[38] Tara N Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6655–6659. IEEE, 2013. 3

[39] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. 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*, pages 1874–1883, 2016. 1, 2

[40] Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 129–136, 2011. 3

[41] Ying Tai, Jian Yang, and Xiaoming Liu. Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3147–3155, 2017. 1, 2, 3, 4, 5

[42] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *Proceedings of the IEEE international conference on computer vision*, pages 4539–4547, 2017. 2, 3, 7

[43] Matt W Thornton, Peter M Atkinson, and DA Holland. Sub-pixel mapping of rural land cover objects from fine spatial resolution satellite sensor imagery using super-resolution pixel-swapping. *International Journal of Remote Sensing*, 27(3):473–491, 2006. 1

[44] Radu Timofte, Vincent De Smet, and Luc Van Gool. Anchored neighborhood regression for fast example-based super-resolution. In *Proceedings of the IEEE international conference on computer vision*, pages 1920–1927, 2013. 1

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2

[46] Longguang Wang, Yingqian Wang, Zaiping Lin, Jungang Yang, Wei An, and Yulan Guo. Learning a single network for scale-arbitrary super-resolution. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4801–4810, 2021. 6

[47] Jun Xiao, Wenqi Jia, and Kin-Man Lam. Feature redundancy mining: Deep light-weight image super-resolution model. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1620–1624. IEEE, 2021. 2

[48] Jun Xiao, Qian Ye, Rui Zhao, Kin-Man Lam, and Kao Wan. Self-feature learning: An efficient deep lightweight network for image super-resolution. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 4408–4416, 2021. 3

[49] Brandon Yang, Gabriel Bender, Quoc V Le, and Jiquan Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. *Advances in neural information processing systems*, 32, 2019. 3

[50] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE transactions on image processing*, 19(11):2861–2873, 2010. 1

[51] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *International conference on curves and surfaces*, pages 711–730. Springer, 2010. 4

[52] Xiaoming Zhang, Tianrui Li, and Xiaole Zhao. Boosting single image super-resolution via partial channel shifting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13223–13232, 2023. 6

[53] Xindong Zhang, Hui Zeng, and Lei Zhang. Edge-oriented convolution block for real-time super resolution on mobile devices. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 4034–4043, 2021. 7

[54] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 286–301, 2018. 2, 6, 7

[55] Yulun Zhang, Huan Wang, Can Qin, and Yun Fu. Learning efficient image super-resolution networks via structure-regularized pruning. In *International conference on learning representations*, 2021. 3

[56] Hengyuan Zhao, Xiangtao Kong, Jingwen He, Yu Qiao, and Chao Dong. Efficient image super-resolution using pixel attention. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, August 2020. 2