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

Supplementary Material

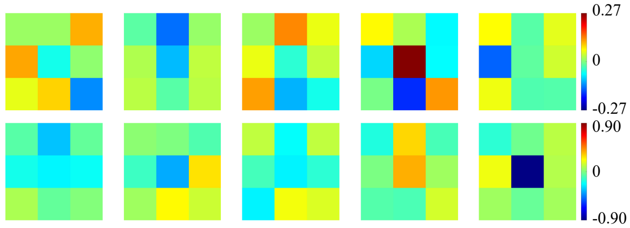


Figure 1. Visualized filters of EDSR-baseline implemented in a recursive manner. We visualize the 1st convolutional layer (Top) and the 2nd (Bottom) of residual blocks.

1. Experimental Details

We used He initialization [2] for the initialization of the partial filters. We provide a visualized explanation of network structures in Figs. 3 and 4. To simplify the implementation, biases in CONV-PS layers of recursive/recurrent models are omitted. Also, to ensure a fair comparison of inference speed, PS models from which biases are removed are compared with original models without biases.

CARN, DRRN, EDSR, and RCAN-PS All PS models, including CARN [1], DRRN [6], EDSR [5], and RCAN [7] mentioned in the main paper, are trained using the official implementation of CARN. We train the models with a batch size of 16, starting with an initial learning rate of 2×10^{-4}, which is reduced by a decay rate of 0.85. Each model is trained from scratch for 2,000K iterations, followed by an additional 2,000K iterations of fine-tuning. The results presented in Tables 3 and 4 are based on models trained for 2,000K iterations without a fine-tuning process. Considering the shape of the filter, we set the partial filter shape to $C{\times}C{\times}1{\times}1$. For CARN, PS is exclusively applied to the 3×3 convolution layers within cascading blocks. In DRRN, PS is restricted to the recursive blocks. For EDSR-baseline-Fixed and RCAN_G5B10-Fixed, the two convolution layers within each repeating residual block are configured to share the same two filters throughout the entire model.

SRFBN-PS We train models with a batch size of 16. The initial learning rate is set to $2{\times}10^{-4}$ with a decay rate of 0.5. Since SRFBN [4] uses a wide kernel, such as $32{\times}32{\times}6{\times}6$ for $\times 2$, we set the partial filter shape to $C{\times}1{\times}K_h{\times}K_w$ for SRFBN-S to enable efficient implementation. We apply PS only to the convolution layers for downsampling.

1.1. Experimental Results of Filter Settings

We report full experimental results of EDSR-baseline-PS with different partial filter settings in Tab. 1.

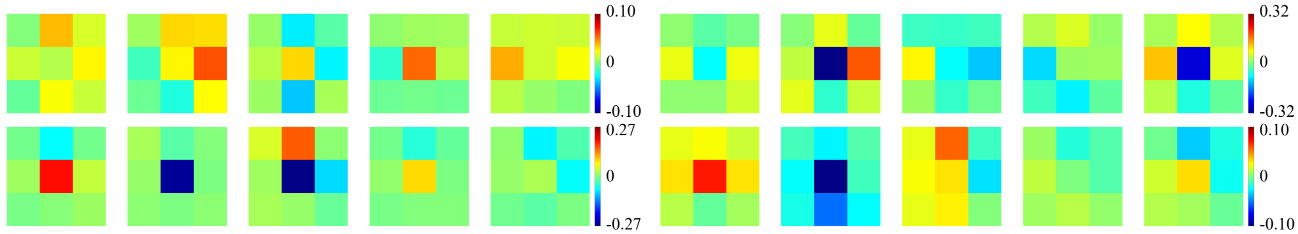
Input v.s. Output Channel We investigate which direction, splitting the input or output channel-wise, yields better results. We compare $64{\times}32{\times}1{\times}1$, $32{\times}64{\times}1{\times}1$, $64{\times}8{\times}3{\times}3$, and $8{\times}64{\times}3{\times}3$ with various ratio values. As shown in Tab. 1, no clear performance difference is observed for each pair. Among these methods, it is observed that smaller partial filters tend to yield superior performance. This superior performance of smaller filters likely arises from a broader range of possible combinations.

2. Supplementary Studies of PS

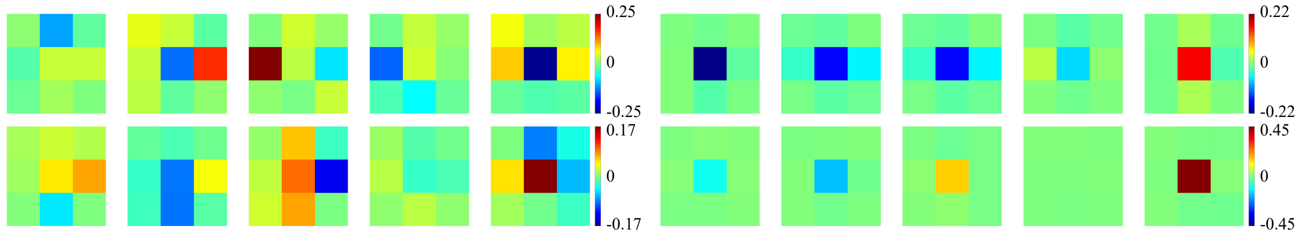
2.1. Filter Diversity

In the main paper, we have demonstrated that Partial filter-Sharing can significantly improve upon previous parameter-sharing SR models. In this section, we visualize the filters of PS networks to show that diverse filters are utilized within the network.

In Figs. 1 and 2, we provide visualizations of the filters for both PS and recursive EDSR-baseline models. For filter visualization, we use the EDSR-baseline-PS with a partial filter shape of $64{\times}64{\times}1{\times}1$ and a N_p/N_{total} ratio of 1/16, which matches the filter size in the EDSR-baseline implemented recursively. We collect the filter values from the first input channel's first five output channels, denoted as W[0:5,0,:,:]. As shown in Figs. 1 and 2, in contrast to recursive models that use the same sets of filters repeatedly, EDSR-baseline-PS employs a diverse range of filters at each depth. This variety of filters enhances the network's representational ability, surpassing



(a) Reconstructed Filter Values from 4th Residual Block (b) Reconstructed Filter Values from 8th Residual Block



(c) Reconstructed Filter Values from 12th Residual Block (d) Reconstructed Filter Values from 16th Residual Block

Figure 2. Visualized reconstructed filters of EDSR-baseline-PS with $N_p/N_{total} = 1/16$. We visualize the 1st convolutional layer (Top) and the 2nd (Bottom) of residual blocks.

Methods	N_p/N_{total}	Params	Set5	Set14	B100	Urban100
Baseline	-	1.37M	38.01/0.9608	33.51/0.9187	32.22/0.9004	32.23/0.9296
Recursive	-	0.26M	37.50/0.9589	32.91/0.9133	31.89/0.8962	30.99/0.9166
D.S.Conv-16 [3]	-	0.34M	37.78/0.9600	33.19/0.9161	32.08/0.8987	31.61/0.9236
D.S.Conv-12 [3]	-	0.60M	37.84/0.9605	33.27/0.9605	32.17/0.8997	31.88/0.9268
D.S.Conv-8 [3]	-	0.86M	37.92/0.9606	33.40/0.9183	32.20/0.9001	32.08/0.9287
D.S.Conv-4 [3]	-	1.11M	37.96/0.9615	33.43/0.9608	32.23/0.9004	32.18/0.9295
64×64×1×1	1/16	0.27M	37.65/0.9597	33.11/0.9156	32.02/0.8980	31.44/0.9219
	2/16	0.35M	37.77/0.9601	33.19/0.9161	32.10/0.8988	31.68/0.9243
	4/16	0.51M	37.89/0.9604	33.29/0.9173	32.16/0.8995	31.93/0.9268
	6/16	0.66M	37.97/0.9606	33.43/0.9183	32.18/0.8998	32.03/0.9277
	8/16	0.82M	38.00/0.9608	33.50/0.9185	32.20/0.9000	32.13/0.9287
	10/16	0.98M	38.01/0.9608	33.46/0.9180	32.20/0.9001	32.17/0.9289
	12/16	1.14M	38.00/0.9608	33.47/0.9180	32.21/0.9001	32.19/0.9291
	14/16	1.30M	38.00/0.9608	33.47/0.9181	32.22/0.9002	32.22/0.9294
64×1×3×3	1/32	0.36M	37.85/0.9603	33.25/0.9164	32.10/0.8988	31.69/0.9244
	1/16	0.53M	37.94/0.9606	33.33/0.9173	32.17/0.8997	31.99/0.9274
	2/16	0.86M	38.01/0.9608	33.43/0.9177	32.21/0.9001	32.15/0.9290
64×32×1×1	1/16	0.29M	37.74/0.9598	33.12/0.9156	32.03/0.8979	31.41/0.9215
	4/16	0.57M	37.92/0.9604	33.33/0.9178	32.16/0.8996	31.94/0.9270
	8/16	0.95M	37.98/0.9608	33.44/0.9175	32.19/0.9000	32.12/0.9286
	12/16	1.32M	38.02/0.9608	33.48/0.9182	32.22/0.9002	32.20/0.9291
32×64×1×1	1/16	0.29M	37.70/0.9598	33.11/0.9157	32.02/0.8980	31.43/0.9217
	4/16	0.57M	37.90/0.9604	33.34/0.9170	32.16/0.8995	31.94/0.9270
	8/16	0.95M	38.01/0.9608	33.47/0.9181	32.20/0.9000	32.16/0.9290
	12/16	1.32M	38.00/0.9607	33.46/0.9182	32.21/0.9002	32.19/0.9293
64×8×3×3	1/16	0.27M	37.71/0.9597	33.02/0.9150	32.00/0.8977	31.33/0.9206
	4/16	0.50M	37.86/0.9604	33.28/0.9165	32.11/0.8991	31.80/0.9255
	8/16	0.81M	37.95/0.9606	33.37/0.9174	32.17/0.8997	32.01/0.9274
	12/16	1.12M	37.99/0.9607	33.46/0.9176	32.20/0.9000	32.14/0.9289
8×64×3×3	1/16	0.27M	37.70/0.9597	33.08/0.9150	32.02/0.8979	31.39/0.9210
	4/16	0.50M	37.84/0.9604	33.28/0.9168	32.13/0.8992	31.83/0.9255
	8/16	0.81M	37.87/0.9604	33.37/0.9175	32.18/0.8998	32.03/0.9278
	12/16	1.12M	37.99/0.9608	33.42/0.9175	32.20/0.9000	32.16/0.9288

Table 1. Experimental results of EDSR-baseline-PS ($\times 2$) with different partial filter shapes and N_p/N_{total} settings. D.S.Conv-n represents EDSR-baseline with n residual block consisting of depth-wise separable convolution.

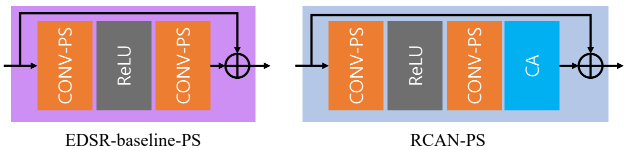


Figure 3. Visualized building blocks of EDSR-baseline-PS, and RCAN-PS.

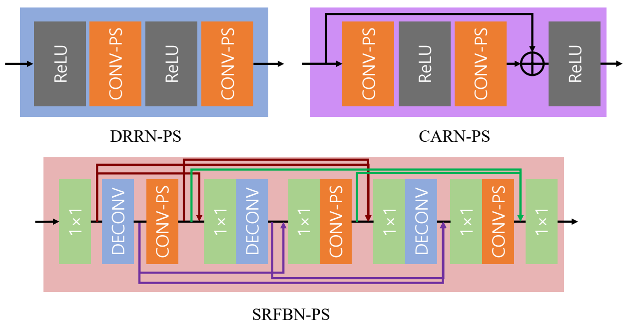


Figure 4. Visualized building blocks of DRRN-PS, CARN-PS, and SRFBN-PS.

Coeff.	Task	DRRN-PS PSNR/ΔPSNR	CARN-PS PSNR/ΔPSNR
	×2	31.73/00.00	32.09/00.00
×2	×3	24.74/-03.11	28.16/-00.03
	×4	23.24/-02.48	26.09/-00.06
	×2	22.71/-09.02	32.08/-00.01
×3	×3	27.85/00.00	28.19/00.00
	×4	23.64/-02.08	26.12/-00.03
	×2	19.28/-12.45	32.01/-00.08
×4	×3	22.80/-05.05	28.17/-00.02
	×4	25.72/00.00	26.15/00.00

Table 2. The experimental results of the up-scaling task with different up-scaling coefficient matrices evaluated on Urban100. Coeff. represents coefficient matrices.

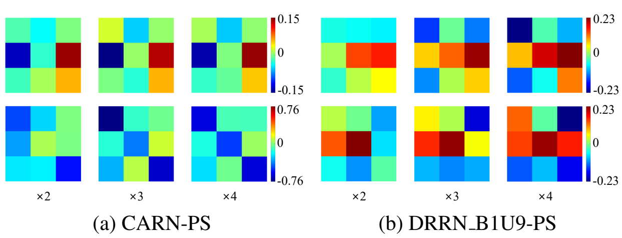


Figure 5. Visualized filters in CARN-PS (Left) with $N_p/N_{total} = 8/9$ and DRRN-PS (Right). The filter values are collected from the two channels of the first convolutional layer in the first block.

the capabilities of traditional parameter-sharing methods.

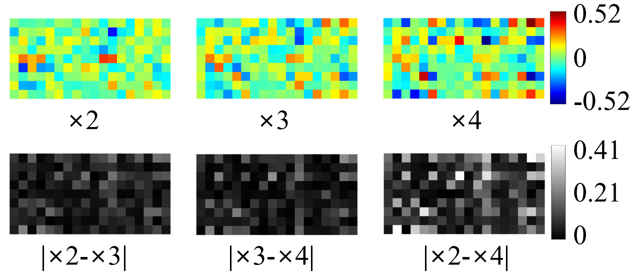


Figure 6. Visualized coefficient matrices (Top) and their difference maps (Bottom) of DRRN_B1U9-PS. Values are collected from the 1st layer of the 1st block.

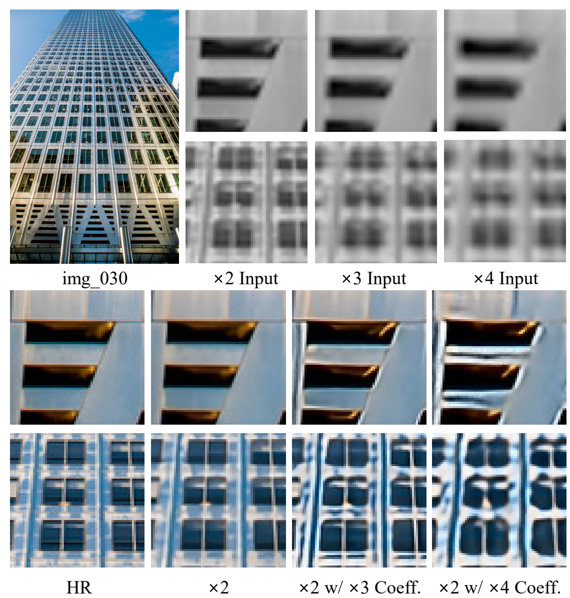


Figure 7. Comparison of characteristics of input images across different scales (Top) and ×2 up-scaling results using coefficient matrices from different up-scaling tasks (Bottom).

2.2. Efficacy in Multi-scale Learning Methods

In this section, we explore the characteristics of the multi-scale learning models employing the proposed method. As shown in Fig. 5, CARN-PS and DRRN_B1U9-PS exhibit similarities across the filters for different up-scaling tasks. We also provide visualizations of the coefficient matrices of the networks and their difference maps in Figs. 6 and 8, which highlight the similarities and differences in the filters across different up-scaling tasks.

Additionally, we investigate their similarities and differences by switching tasks between up-scaling filters. In Tab. 2, we compare the experimental results of the up-scaling task using coefficient matrices from different tasks. We observe that exchanging the tasks of the coefficient matrices results in a minimal performance

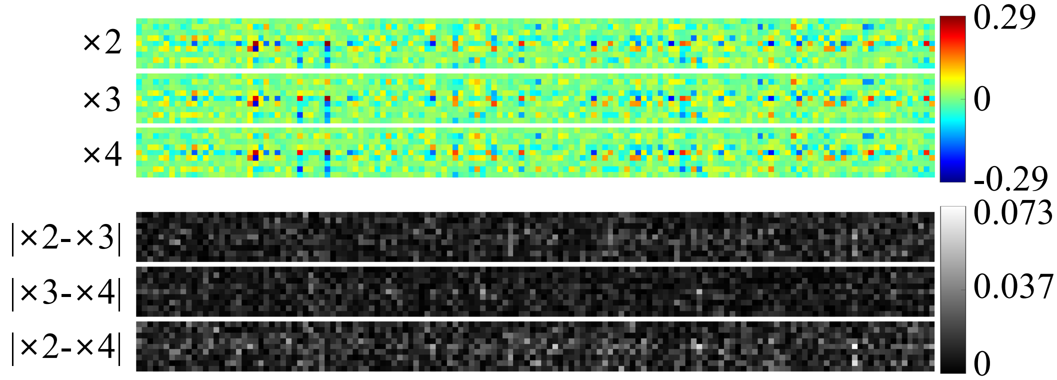


Figure 8. Visualized coefficient matrices (Top) and their difference maps (Bottom) of CARN-PS with $N_p/N_{total} = 8/9$. Values are collected from the 1st layer of the 1st cascading block.

decrease for CARN-PS, ranging from zero to 0.1dB, rather than a significant deterioration. This minimal performance degradation, despite changing the coefficient matrices, suggests that there is a similarity among the filters for different up-scaling tasks. This result explains why multi-scale learning has been successful so far.

In contrast, despite the similarity of filters across different scales, DRRN-PS shows a notable decrease in performance compared to CARN-PS. The differing results between the two networks arise from differences in how they process the inputs. CARN places a pixel-shuffle layer at the end of the network, processing the inputs at the dimension of the input image. On the other hand, DRRN up-scales the input image to the target size using bicubic interpolation before processing. Such pre-processing leads to significant differences in the statistics of the input features at each scale, as shown in Fig. 7 (Top). This variation in input statistics necessitates the network to develop filters adapted to each scale's unique input characteristics. For example, Fig. 7 (Bottom) shows that the $\times 4$ filter, which typically handles severely blurred images, tends to sharpen the input images excessively.

These results demonstrate that the diversification of filters from PS plays a critical role in enhancing DRRN_B1U9-PS's performance. Also, considering the most dramatic performance improvements observed with DRRN_B1U9-PS, PS seems most effective when applied to multi-tasking networks handling diverse input statistics.

2.3. Limitation

To improve parameter-sharing SR methods, we have proposed the Partial filter-Sharing technique. This method allows network layers to utilize a wide range of filters through partial filters and coefficient matrices. In this section, we introduce some of the method's limitations.

In the PS framework, we factorize shared weights into partial filters and coefficient matrices. As noted in Section 4.3, this process results in the loss of channel/spatial correlation information from partial filters, which must be compensated by the coefficient matrices. Therefore, applying PS in networks with smaller capacities could limit the capacity of the coefficient matrix, resulting in less effective filter reconstructions.

To address this, improving the representation of PS filters may be necessary, which would involve refining how partial filters and coefficient matrices are partitioned. One potential solution is to allocate more parameters to the coefficient matrix during weight factorization. Alternatively, adopting adaptive filtering techniques might offer a way forward. This would require an efficient module for calculating the coefficient matrix based on the inputs. Further refinement of PS is left for future works.

References

[1] 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.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.