Supplementary Material of "Iterative Soft Shrinkage Learning for Efficient Image Super-Resolution"

Jiamian Wang[1], Huan Wang[2], Yulun Zhang[3]*, Yun Fu[2], and Zhiqiang Tao[1]*
[1]Rochester Institute of Technology, [2]Northeastern University, [3]ETH Zürich

1. Overview

In this supplementary material, we present additional results and analyses about the proposed method as follows.

- A more detailed and comprehensive illustration to the performance comparison (Section 2).
- Ablation studies under high pruning ratios (Section 3).
- More visualization comparisons (Section 4).
- More discussions about the dynamics of the sparse structure and trainability preserving (Section 5).

2. Compared Methods

ASSL. ASSL [9] prunes the filters of the convolutional layers across different residual blocks, which cannot be easily extended to the novel structures, *e.g.*, multi-layer perceptron, multi-head self-attention, etc. To improve the adaptability of ASSL, we implement this method in a weight pruning fashion by following their designing principles.

Firstly, ASSL selects the last convolutional layers of all residual blocks as *constrained* layers, whose pruned filter indexes should be kept the same across the whole network for the residual connection concern. All the other convolutional layers can be pruned without structural constrains, which are termed as *free* layers. Similarly, we retain this structural constraint by categorizing the learnable layers before the any residual connection as the *constrained* ones, whose pruned wight indexes are consistent to one another, while others are *free* layers without structural constrains.

To select the pruned filter indexes in the *constrained* layers, ASSL firstly employs the weight normalization [8] as an indicator of the filter importance, which in our case, can be directly inferred from the weight magnitudes. Thereby, no additional weight normalization is needed in our setting.

To align the selected filter indexes across different *constrained* layers, ASSL collects the index vectors of each *constrained* layers and encourages their consistency by maximizing the inner-product of the index vectors, yielding a sparsity structure alignment (SSA) penalty. Alternatively, we follow the same practice by regularizing the weight indexes rather than filter indexes.

Except for the above adaptations, we strictly follow the implementation of ASSL. To sum up, by retaining the key design principles (*i.e.*, filter alignment, SSA regularization, etc.) of ASSL, we deliver an enhanced version of ASSL, which is readily compatible with the off-the-shelf SR network designs in the scenario of weight pruning.

SRP. Another representative work of SRP [10] is also dedicated to the pruned filter index alignment considering the residual connection issue. This method selects the indexes of pruned filters across different *constrained* layers in a random manner prior to the training. Then the selected filters are shrunk by an L_2 regularization with a growing schedule. Typically, the regularization encompasses L_2-norm of all unimportant filters in the network.

Here, we directly change the pruning units from the filters to the weights throughout the network and follow the same regularization arrangement as SRP. Notably, both of ASSL and SRP are raised upon a pre-trained SR network. In our setting, the pruning is directly conducted upon the network with random initialization.

N:M Sparsity. Besides compared methods, another emerging trend of weight pruning is to employ the N:M sparsity [2, 6]. This technology serves as an *hardware-driven standard/regulation* proposed to take advantage of the real-world accelerations upon novel computational platforms (*e.g.*, NVIDIA Ampere GPUs). The proposed method fundamentally different from them by delivering a highly-adaptable solution for the diverse SR network architectures *in a algorithm-motivated perspective*. N:M sparsity is orthogonal to our treatment (Section 2 in the manuscript).

However, to better evaluate the effectiveness of the proposed method, We also compare ISS-P with the the most recent method of SLS [7], following N:M sparsity. Note that, this method handles the convolutional networks by customizing the convolutional layers with structural constrains, but not explores and evaluates other types of the neural structures like the Transformer. Therefore, we perform the comparison on the representative EDSR-L [5] backbone.

*Corresponding authors: Yulun Zhang (yulun100@gmail.com) and Zhiqiang Tao (zxtics@rit.edu)

Backbones	Methods	Set5		Set14		B100		Urban100		Manga109	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
EDSR-L	scratch	29.60	0.8522	26.35	0.7312	25.92	0.7003	23.69	0.7303	27.28	0.8570
	L_1 norm [3]	29.61	0.8526	26.36	0.7318	25.93	0.7011	23.70	0.7313	27.35	0.8582
	ASSL [9]	29.85	0.8568	26.54	0.7368	26.07	0.7064	24.09	0.7461	27.93	0.8690
	SRP [10]	29.78	0.8558	26.47	0.7349	26.02	0.7036	23.89	0.7392	27.72	0.8656
	SLS [7]	29.76	0.8558	26.41	0.7339	25.95	0.7029	23.76	0.7369	27.39	0.8576
	ISS-P (ours)	30.23	0.8628	26.74	0.7428	26.21	0.7109	24.43	0.7596	28.51	0.8783

Table 1. A more comprehensive performance comparison of different methods upon the representative CNN backbone, EDSR-L [5] at the scale of the $\times 4$. The method [7] following the N:M sparsity standard is also incorporated. The pruning ratio is 0.95.

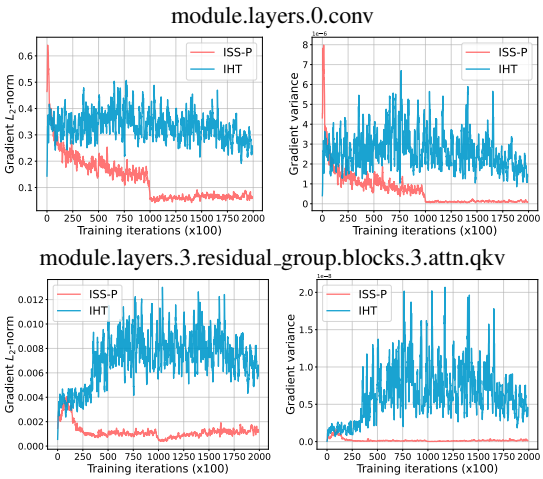


Figure 1. Trainability comparison of ISS-P and IHT. The layer-wise gradient L_2-norm and variance in the pruning stage (1×10^5 iterations) and the first 1×10^5 iterations of fine-tuning stage are plotted. We choose two representative layers, *i.e.*, a convolution (*top*) and a linear layer (*bttom*) from the SwinIR-Lightweight.

Specifically, we employ the SLS by pruning the convolutional layers in the residual blocks and the upsampling module. The N:M sparsity is chosen as 2:32 for a pruning ratio of 0.94. We leverage the same training datasets and data prepossessing operations as the manuscript (Section 4 in the manuscript). By comparison, the proposed method outperforms compared methods including SLS. Considering the orthogonal property, one can easily integrate both trends for a potential hardware acceleration on diverse network designs with weight pruning strategy.

3. Ablation Study

The behaviour of the pruning methods can be different under different pruning ratios. Therefore, we systematically compare the performances of IHT, ISS-R, and ISS-P under very high pruning ratios (*i.e.*, 0.95 and 0.99). As shown by Tables 2 and 3, ISS-P outperforms the ablated methods at different scales, which is consistent with the results under the ratio of 0.9 (Table 2 in the manuscript). Note that the performance gaps between ISS-P and ablated methods grow as the scale or pruning ratio increases. This is because ISS-P not only better selects the sparse structure, but also retains trainability of the sparse network (see Section 5).

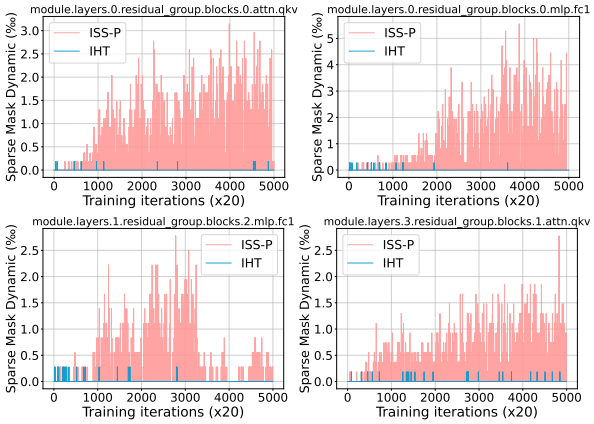


Figure 2. Dynamics of the Sparse structure induced by ISS-P and IHT in the pruning stage. The proposed method, ISS-P, allows a more active sparse pattern exploitation adapting to the optimization. We choose four representative layers (*i.e.*, 1st, 3rd, 36-th, and 80-th) from the SwinIR-Lightweight backbone [4].

4. Visualization Comparison

In Figs. 3∼12, we provide more visualization of different methods under the a very high pruning ratio of 0.99 and scale of $\times 4$, which is the most challenging scenario. The proposed performs better by introducing less distortions, especially on high-frequency textured patterns, indicating the strong modeling capacity of the selected sparse structure.

5. Model Discussion

Both the proposed ISS-P and IHT select the unimportant weights per layer in each iteration. Therefore, measuring the number of weights that are categorized into different importance groups between the consecutive iterations helps infer the effect of the pruning methods. In Fig. 2, we provide more visualizations of representative layers in SwinIR-Lightweight. The weight significance flips in ISS-P happens more frequent than that of IHT, which means the proposed method potentially pursues more sparse possibilities in an active manner. We also provide more visualizations of the gradient statistics on different layers in Fig. 1. The gradient L_2-norm of ISS-P presents a preferred convergence tendency over the IHT regardless of the layer structures, which indicates the promising adaptability of the proposed method across different neural architectures.

Methods	Scale	Set5		Set14		B100		Urban100		Manga109	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
IHT	×2	35.17	0.9448	31.49	0.8978	30.57	0.8781	27.95	0.8740	32.91	0.9519
ISS-R	×2	35.37	0.9462	31.60	0.8983	30.66	0.8790	28.10	0.8730	33.31	0.9543
ISS-P	×2	35.86	0.9496	31.89	0.9015	30.87	0.8819	28.40	0.8777	34.09	0.9584
IHT	×3	32.96	0.9124	29.41	0.8247	28.44	0.7884	26.23	0.8030	30.61	0.9158
ISS-R	×3	32.92	0.9122	29.38	0.8239	28.43	0.7880	26.22	0.8027	30.57	0.9121
ISS-P	×3	33.28	0.9161	29.59	0.8282	28.58	0.7919	26.55	0.8123	31.28	0.9233
IHT	×4	30.61	0.8661	27.57	0.7567	26.93	0.7143	24.49	0.7258	27.49	0.8562
ISS-R	×4	30.69	0.8677	27.61	0.7576	26.96	0.7151	24.52	0.7273	27.61	0.8583
ISS-P	×4	30.97	0.8733	27.77	0.7619	27.05	0.7187	24.73	0.7360	28.00	0.8669

Table 2. Ablation study of different methods under the pruning ratio of 0.95 at different scales.

Methods	Scale	Set5		Set14		B100		Urban100		Manga109	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
IHT	×2	35.17	0.9448	31.49	0.8978	30.57	0.8781	27.95	0.8740	32.91	0.9519
ISS-R	×2	35.37	0.9462	31.60	0.8983	30.66	0.8790	28.10	0.8730	33.31	0.9543
ISS-P	×2	35.86	0.9496	31.89	0.9015	30.87	0.8819	28.39	0.8777	34.09	0.9584
IHT	×3	31.35	0.8853	28.35	0.8035	27.75	0.7697	25.07	0.7630	28.03	0.8742
ISS-R	×3	30.94	0.8806	28.09	0.7990	27.64	0.7660	24.92	0.7578	27.56	0.8676
ISS-P	×3	31.87	0.8960	28.71	0.8102	27.98	0.7753	25.42	0.7754	28.78	0.8889
IHT	×4	29.13	0.8249	26.57	0.7261	26.34	0.6916	23.57	0.6801	25.55	0.7893
ISS-R	×4	29.09	0.8243	26.32	0.7260	26.32	0.6912	23.55	0.6788	25.49	0.7977
ISS-P	×4	29.39	0.8331	26.73	0.7313	26.44	0.6952	23.70	0.6873	25.81	0.8085

Table 3. Ablation study of different methods under the pruning ratio of 0.99 at different scales.

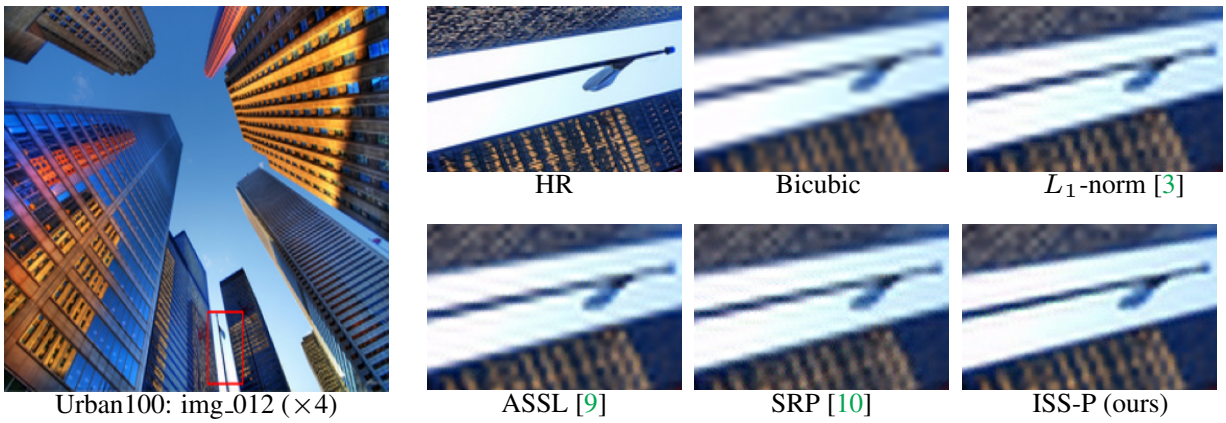


Figure 3. Visualization comparison of different pruning methods on Urban100 [1] dataset. The pruning ratio is 0.99.

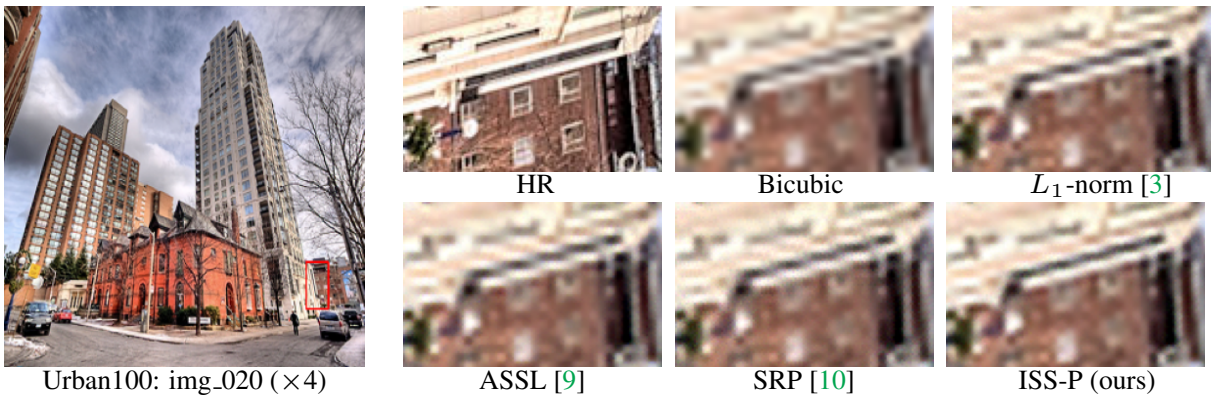
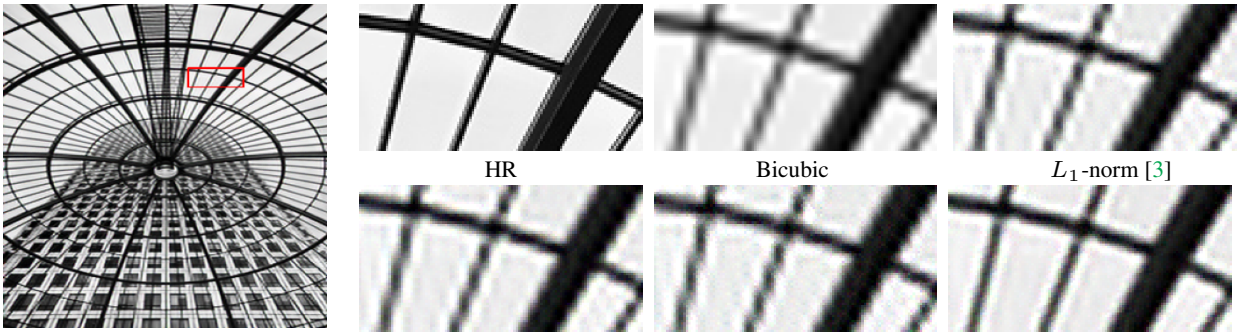
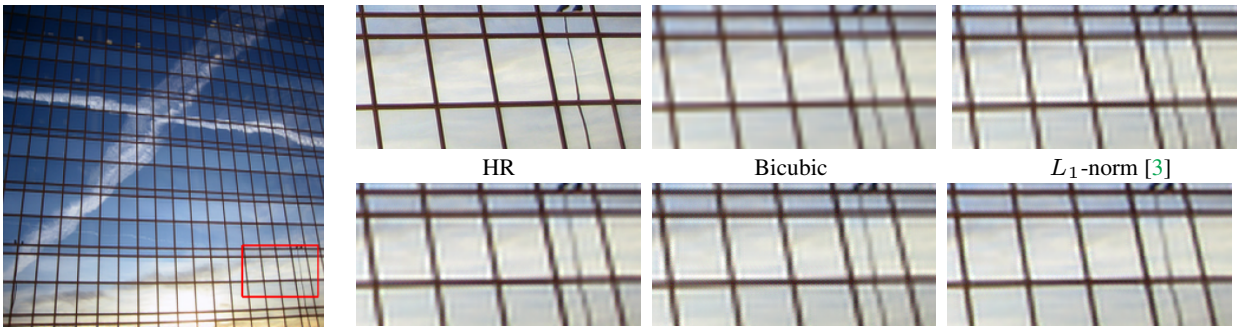


Figure 4. Visualization comparison of different pruning methods on Urban100 [1] dataset. The pruning ratio is 0.99.



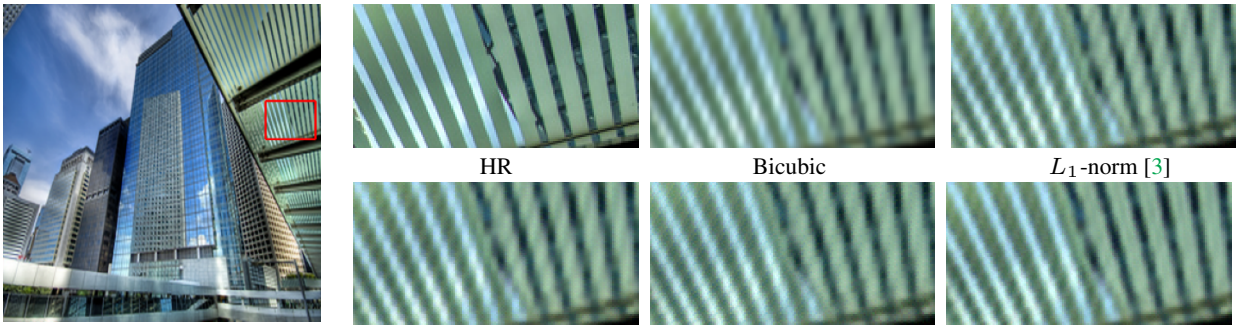
Urban100: img_072 ($\times 4$)

Figure 5. Visualization comparison of different pruning methods on Urban100 [1] dataset. The pruning ratio is 0.99.



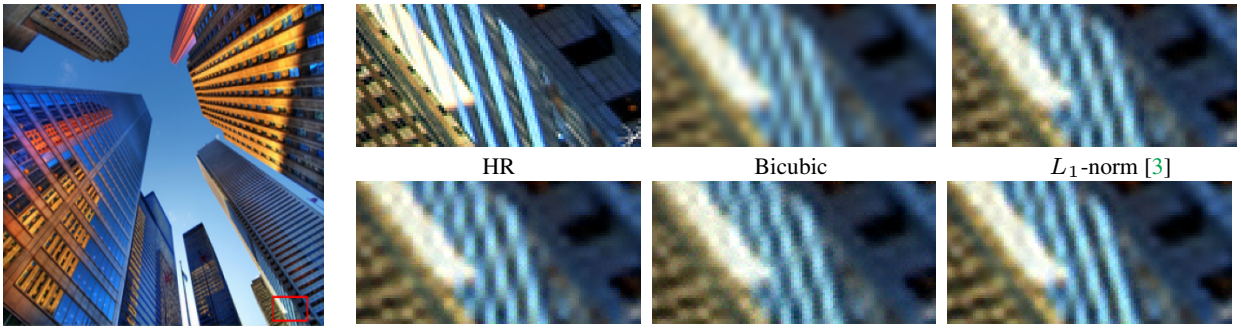
Urban100: img_055 ($\times 4$)

Figure 6. Visualization comparison of different pruning methods on Urban100 [1] dataset. The pruning ratio is 0.99.



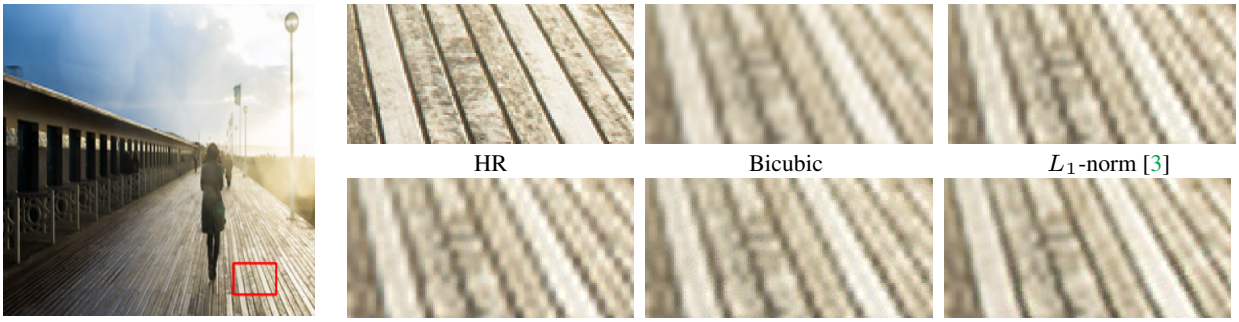
Urban100: img_061 ($\times 4$)

Figure 7. Visualization comparison of different pruning methods on Urban100 [1] dataset. The pruning ratio is 0.99.



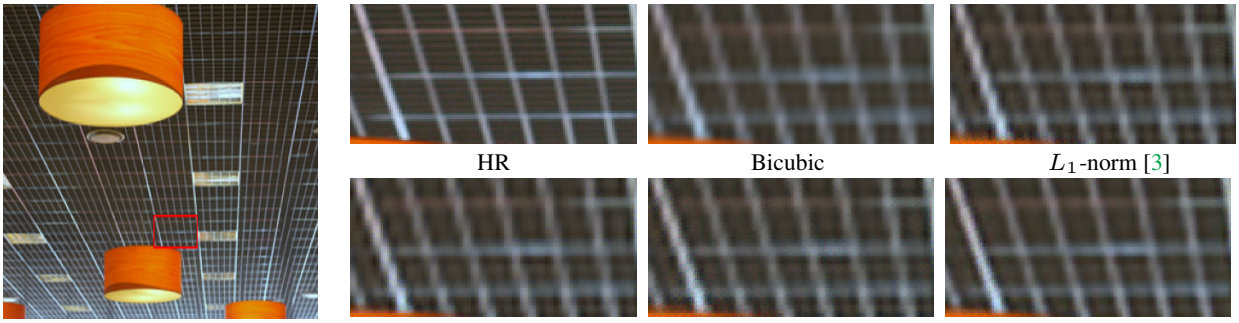
Urban100: img_012 ($\times 4$)

Figure 8. Visualization comparison of different pruning methods on Urban100 [1] dataset. The pruning ratio is 0.99.



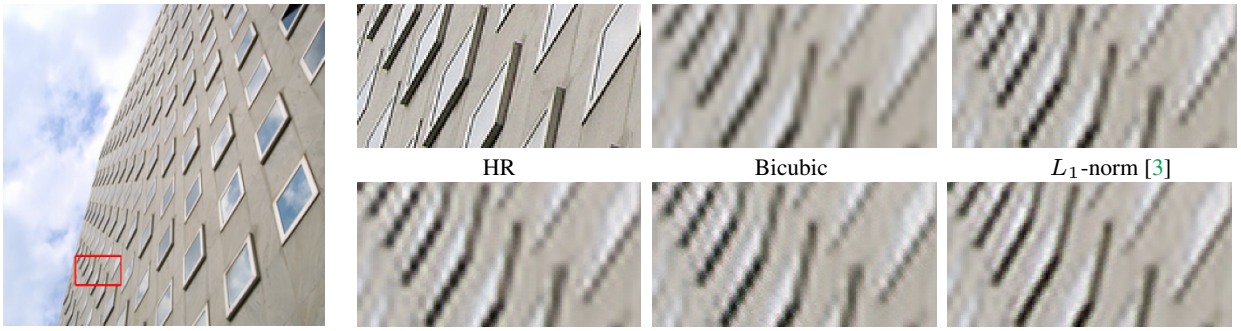
Urban100: img_032 (×4)

Figure 9. Visualization comparison of different pruning methods on Urban100 [1] dataset. The pruning ratio is 0.99.



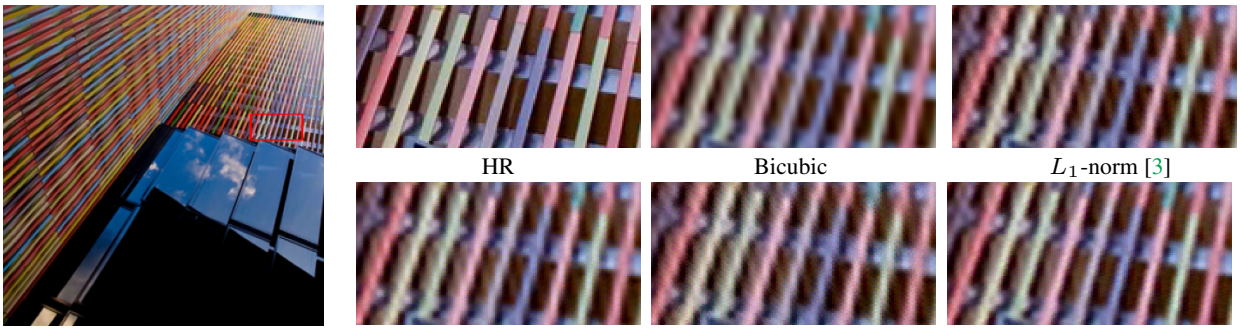
Urban100: img_044 (×4)

Figure 10. Visualization comparison of different pruning methods on Urban100 [1] dataset. The pruning ratio is 0.99.



Urban100: img_010 (×4)

Figure 11. Visualization comparison of different pruning methods on Urban100 [1] dataset. The pruning ratio is 0.99.



Urban100: img_023 (×4)

Figure 12. Visualization comparison of different pruning methods on Urban100 [1] dataset. The pruning ratio is 0.99.

References

[1] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *CVPR*, 2015. 3, 4, 5

[2] Itay Hubara, Brian Chmiel, Moshe Island, Ron Banner, Joseph Naor, and Daniel Soudry. Accelerated sparse neural training: A provable and efficient method to find n: m transposable masks. In *NeurIPS*, 2021. 1

[3] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *ICLR*, 2017. 2, 3, 4, 5

[4] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *ICCV*, 2021. 2

[5] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *CVPR workshop*, 2017. 1, 2

[6] Asit Mishra, Jorge Albericio Latorre, Jeff Pool, Darko Stosic, Dusan Stosic, Ganesh Venkatesh, Chong Yu, and Paulius Micikevicius. Accelerating sparse deep neural networks. *arXiv preprint arXiv:2104.08378*, 2021. 1

[7] Junghun Oh, Heewon Kim, Seungjun Nah, Cheeun Hong, Jonghyun Choi, and Kyoung Mu Lee. Attentive fine-grained structured sparsity for image restoration. In *CVPR*, 2022. 1, 2

[8] Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *Advances in neural information processing systems*, 29, 2016. 1

[9] Yulun Zhang, Huan Wang, Can Qin, and Yun Fu. Aligned structured sparsity learning for efficient image super-resolution. In *NeurIPS*, 2021. 1, 2, 3, 4, 5

[10] Yulun Zhang, Huan Wang, Can Qin, and Yun Fu. Learning efficient image super-resolution networks via structure-regularized pruning. In *ICLR*, 2022. 1, 2, 3, 4, 5