Activating More Pixels in Image Super-Resolution Transformer Supplementary Material

Xiangyu Chen[1,2,3] Xintao Wang[4] Jiantao Zhou[1] Yu Qiao[2,3] Chao Dong[2,3†]
[1]State Key Laboratory of Internet of Things for Smart City, University of Macau
[2]Shenzhen Key Lab of Computer Vision and Pattern Recognition,
Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
[3]Shanghai Artificial Intelligence Laboratory [4]ARC Lab, Tencent PCG
{chxy95, xintao.alpha}@gmail.com jtzhou@um.edu.mo {yu.qiao, chao.dong}@siat.ac.cn

1. Training Details

We use DF2K (DIV2K+Flicker2K) with 3450 images as the training dataset when training from scratch. The low-resolution images are generated from the ground truth images by the "bicubic" down-sampling in MATLAB. We set the input patch size to 64×64 and use random rotation and horizontally flipping for data augmentation. The mini-batch size is set to 32 and total training iterations are set to 500K. The learning rate is initialized as 2e-4 and reduced by half at [250K,400K,450K,475K]. For $\times 4$ SR, we initialize the model with pre-trained $\times 2$ SR weights and halve the iterations for each learning rate decay as well as total iterations. We adopt Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.99$ to train the model. For the same-task pre-training, the full ImageNet dataset with 1.28 million images is first exploited to pre-train the model for 800K iterations. The initial learning rate is also set to 2e-4 but reduced by half at [300K,500K,650K,700K,750k]. Then, we adopt DF2K dataset to fine-tune the pre-trained model. For fine-tuning, we set the initial learning rate to 1e-5 and halve it at [125K,200K,230K,240K] for total 250K training iterations.

2. Analysis of Model Complexity

We conduct experiments to analyze the computational complexity of our method from three aspects: window size for calculation of self-attention, overlapping cross-attention block (OCAB) and channel attention block (CAB). We also compare our method with the Transformer-based method SwinIR. The $\times 4$ SR performance on Urban100 are reported and the number of Multiply-Add operations is counted at the input size of 64×64. Note that pre-training techniques (including $\times 2$ pre-training) are **NOT** used for all the models in this section. The experimental setup is completely fair.

First, we use the standard Swin Transformer block as the backbone to explore the influence on different window sizes. As shown in Tab. 1, enlarging window size can bring a large performance gain (+0.36dB) with a little increase in parameters and $\sim$%19 increase in Multi-Adds.

Table 1. Model complexity comparison of window sizes.

window size	#Params.	#Multi-Adds.	PSNR
(8, 8)	11.9M	53.6G	27.45dB
(16, 16)	12.1M	63.8G	27.81dB

Table 2. Model complexity comparison of OCAB and CAB.

Method	#Params.	#Multi-Adds.	PSNR
Baseline	12.1M	63.8G	27.81dB
w/ OCAB	13.7M	74.7G	27.91dB
w/ CAB	19.2M	92.8G	27.91dB
Ours	20.8M	103.7G	27.97dB

Table 3. Model complexity comparison of CAB sizes.

β in CAB	#Params.	#Multi-Adds.	PSNR
1	33.2M	150.1G	27.97dB
2	22.7M	107.1G	27.92dB
3 (default)	19.2M	92.8G	27.91dB
6	15.7M	78.5G	27.88dB
w/o CAB	12.1M	63.8G	27.81dB

Table 4. Model complexity comparison of SwinIR and HAT.

Method	#Params.	#Multi-Adds.	PSNR
SwinIR	11.9M	53.6G	27.45dB
HAT-S (ours)	9.6M	54.9G	27.80dB
SwinIR-L1	24.0M	104.4G	27.53dB
SwinIR-L2	23.1M	102.4G	27.58dB
HAT (ours)	20.8M	103.7G	27.97dB

Then, we use window size 16 as the baseline to investigate the computational complexity of the proposed OCAB and CAB. As illustrated in Tab. 2, our OCAB obtains a performance gain with a limited increase of parameters and Multi-Adds. It demonstrates that the effectiveness and efficiency of the proposed OCAB. Besides, adding CAB to the