Restore Globally, Refine Locally: A Mask-Guided Scheme to Accelerate Super-Resolution Networks

Xiaotao Hu[1,2], Jun Xu[2] ⋆, Shuhang Gu[3], Ming-Ming Cheng[1], and Li Liu[4]

[1] College of Computer Science, Nankai University
[2] School of Statistics and Data Science, Nankai University
[3] The University of Sydney
[4] Inception Institute of Artificial Intelligence

Abstract. Single image super-resolution (SR) has been boosted by deep convolutional neural networks with growing model complexity and computational costs. To deploy existing SR networks onto edge devices, it is necessary to accelerate them for large image (4K) processing. The different areas in an image often require different SR intensities by networks with different complexity. Motivated by this, in this paper, we propose a Mask Guided Acceleration (MGA) scheme to reduce the computational costs of existing SR networks while maintaining their SR capability. In our MGA scheme, we first decompose a given SR network into a Base-Net and a Refine-Net. The Base-Net is to extract a coarse feature and obtain a coarse SR image. To locate the under-SR areas in the coarse SR image, we then propose a Mask Prediction (MP) module to generate an error mask from the coarse feature. According to the error mask, we select K feature patches from the coarse feature and refine them (instead of the whole feature) by Refine-Net to output the final SR image. Experiments on seven benchmarks demonstrate that our MGA scheme reduces the FLOPs of five popular SR networks by 10% $\sim$ 48% with comparable or even better SR performance. The code is available at https://github.com/huxiaotaostasy/MGA-scheme.

Keywords: Single image super-resolution, network acceleration

1 Introduction

Single image super-resolution (SR) aims to recover high-resolution (HR) images from the corresponding low-resolution (LR) ones. During the last decades, this problem is widely studied in academia and industry of the computer vision field [8, 19, 26, 47, 30, 25]. Ever since the pioneer work of SRCNN [7], numerous methods [19, 48, 47, 33, 5] have been developed to improve the SR performance along with the advances in network backbones [13, 16, 15, 41, 39]. However, this advancement is achieved at the expense of designing larger SR networks with

⋆ Jun Xu is the corresponding author (email: nankaimathxujun@gmail.com).

more computational costs [7, 23, 47, 34, 33, 14, 50, 6], hindering the corresponding SR networks from being deployed on edge devices such as cameras and mobiles.

For real-world applications, researchers shift to developing efficient SR networks for high-resolution image processing. The efficiency is mainly judged by three aspects: running time, number of parameters, or floating-point operations (FLOPs). Early works mainly report the running time [7, 9, 19], which is largely determined by the hardware and implementations of basic operations (*e.g.*, 3×3 conv.) in deep learning frameworks [35, 1]. Later works mainly resort to reducing the number of parameters [37, 38, 27] or FLOPs [29, 40] for network efficiency. However, pursuing reduction on parameters (or FLOPs) may increase the running time [37] and FLOPs (or parameters) amount. In all, expertise knowledge on deep neural networks is essential for developing efficient SR networks.

Instead of developing new SR networks, a recent trend is to reduce the parameter amount and/or FLOPs of existing ones while maintaining their SR capability [21]. As far as we know, ClassSR [21] is a pioneer work in this direction. It classifies the patches of an LR image into the "easy" (restoration difficulty), "medium", and "hard" categories, and restores these patches by three SR networks in different widths. However, the extra classification network in ClassSR largely limits its practical efficiency due to the additional computational costs. Besides, since the three SR networks in [21] share the same network architecture but with different widths, ClassSR suffers from a huge growth on parameters.

In this paper, we develop a new scheme to reduce the FLOPs and running time on edge devices of popular SR networks. Our scheme divides an existing SR network along the depth, *e.g.*, RCAN [47], into a base network (Base-Net) and a refine network (Refine-Net). The Base-Net restores globally the LR image to extract a coarse feature and output a coarse SR image. Then we propose a Mask Prediction (MP) module to estimate an error mask from the coarse feature by Base-Net, indicating the gap between the coarse and desired SR images. This mask is used to select the feature patches of under super-resolved (under-SR) areas in the coarse SR image. The selected feature patches will be fed into the Refine-Net for further refinement. Since Refine-Net only needs to refine the selected areas, instead of the whole coarse SR image, the computational costs (*e.g.*, FLOPs) of the original SR network, *i.e.*, RCAN [47], can be largely reduced. The refined image is output as the final SR image.

Our Mask-Guided Acceleration (MGA) scheme is different from ClassSR [21] on at least two aspects. Firstly, ClassSR has an extra classification network and three parallel SR networks, resulting in a huge growth on the number of parameters. But our MGA scheme only brings a marginal increment on parameters by our lightweight MP module. Secondly, ClassSR has to perform classification and SR sequentially during the inference, making the SR process time-consuming. However, in our MGA scheme, after the coarse SR by the Base-Net, the Refine-Net of an existing SR network only needs to handle a few selected complex areas. As shown in Figure 1, our MAG scheme reduces the computational FLOPs of FSRCNN [9], SRResNet [49], and RCAN [47] by 42%, 33%, and 23%, respectively, while arriving at comparable PSNR results on Test2K [12]. Experiments

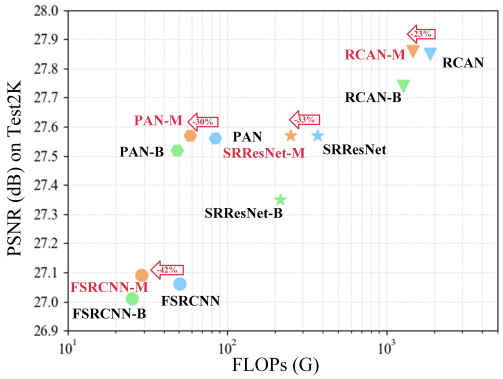

Fig. 1: **Average PSNR (dB) and FLOPs (G) of different SR networks *w/* or *w/o* our Mask-Guided Acceleration scheme** on Test2K [12] at scale factor 4. "B" and "M" denote the baseline and mask-guided models, respectively.

on seven benchmark datasets validate the effectiveness of our MGA Scheme on accelerating five popular SR networks [9, 49, 29, 47].

Our main contributions are summarized as follows:

- **We propose a Mask-Guided Acceleration (MGA) scheme to reduce the FLOPs amount and running time of existing SR networks while preserving their SR capability**. Given an existing SR network, our MGA scheme divides it into a Base-Net to restore the LR image globally and a Refine-Net to further refine the selected under-SR areas locally.
- **We propose a lightweight Mask Prediction module** to adaptively select the feature patches of under-SR areas, which are from the coarse SR image by the Base-Net and further refined by the Refine-Net.
- Experiments on seven benchmark datasets demonstrate that **our Mask-Guided Acceleration scheme can accelerate five popular SR networks in different scales** while achieving comparable SR performance.

2 Related Work

2.1 Single Image Super-Resolution

Single image SR has been advanced by convolutional neural networks (CNNs) ever since SRCNN [7]. The work of VDSR [19] introduces a residual learning scheme to avoid direct SR prediction. The integration of residual and dense connections is later exploited in RDN [48]. Despite the discriminative learning framework, generative learning [10] is also employed in SRGAN [23] to produce visually pleasing SR results. Attention mechanisms have also been utilized in many SR networks. For example, RCAN [47] incorporates channel attention [15]

into residual learning [13]. Self-attention is adopted in [33] to provide global correlation information for SR. The success of the transformer framework in natural language processing inspires the work of IPT [5]. These SR networks achieve performance gain at the expense of increasing parameters and FLOPs, hindering their applications on edge devices [3, 29, 24]. In this work, we aim to accelerate general SR networks while preserving their SR capability.

2.2 Lightweight Super-Resolution

Lightweight super-resolution aims to achieve efficient SR by reducing the running time, the number of parameters and/or floating point operations (FLOPs). To reduce the running time, FSRCNN [9] performs upsampling at the final stage and uses small convolution kernels, while LapSRN [22] gradually upsamples the feature maps. But running time is greatly influenced by the hardware, the actual implementations, and the deep learning frameworks [46, 35, 1]. The parameter amount is also a useful criterion to evaluate model efficiency [37, 49]. To this end, DRRN [37] learns shared residual blocks in a recursive manner, while PAN [49] incorporates the self-calibrated convolution [28] and introduces a pixel attention module. However, the reduction on parameter amount does not necessarily bring clear improvements on the model efficiency [46]. Recent works [29, 40] often tend to reduce the number of FLOPs. RFDN [29] replaces standard residual blocks with shallow variants, arriving at similar SR results with fewer FLOPs. Overall, it is challenging to design better lightweight SR networks. In this paper, we propose a general scheme to accelerate general SR networks.

2.3 Accelerating Super-Resolution Networks

Accelerating super-resolution networks is an alternative way for lightweight SR. Along this direction, SMSR [40] adaptively tackles different areas by efficient convolutions, which are not plug-and-play to general SR networks. AdderSR [36] replaces amounts of multiplications by additions in convolutions to reduce the energy consumption. But it suffers from clear degradation in SR performance. ClassSR [21] is a pioneer framework to accelerate SR networks. It assigns suitable SR networks to process local areas with different restoration difficulties, which are determined by a classification network. Though with less computational costs, ClassSR also brings a huge amount of extra parameters. FADN [44] replaces the original residual block with an efficient one to accelerate SR networks. However, each replacement increases the parameter amounts of the SR networks. In this paper, we introduce a mask-guided scheme to accelerate popular SR networks by performing coarse SR on the LR image and further refinement on under-SR areas only. This provides a flexible trade-off between SR capability and network efficiency with a little parameter growth.

3 Our Mask-Guided Acceleration Scheme

3.1 Motivation and Overview

In image super-resolution (SR), the quality of SR images is mainly degraded by the information loss that occurs in complex areas such as edges and textures [40]. To well recover these areas, most SR methods resort to developing networks larger than those required for the plain areas [11].

Increasing the network width (*i.e.*, number of channels) is a feasible choice to improve its capability of restoring complex areas [21]. But for different areas of an image, an SR network with a single branch can only extract the features with equal channel dimension. Thus, the work of [21] restores the areas with different complexities by multiple SR networks with respective widths. However, this brings a great increase in the amount of network parameters [21].

Increasing the network depth (*i.e.*, number of layers) is an alternative way to well recover complex areas. However, handling different areas simply by multiple SR networks with adaptive depths still suffers from significant parameter growth. To alleviate this problem, we propose to process different areas by the sub-networks with adaptive depths of a single-branch SR network. This is feasible by decomposing an SR network into different parts along the depth dimension: the shallow part restores the whole image while the deep part only recovers the complex areas. Since complex areas are sparse in the image, the computational costs of the decomposed SR network can be largely reduced.

Take the PAN network [49] for example. The original PAN has 16 SCPA blocks. We denote as PAN-B the PAN with 12 SCPA blocks. Given an LR image in Figure 2 (1), we employ PAN-B and PAN to obtain a coarse SR image and a final SR image in Figure 2 (2) and Figure 2 (3), respectively. The absolute difference (error map) between these two SR images is calculated on the luminance channel and shown in Figure 2 (4). We plot its histogram in Figure 2 (5), and observe that most areas are smooth with small errors (<0.1) while large errors (*e.g.*, >0.1) are sparsely dispersed in textures and edges. This demonstrates that most areas in the LR image can be well recovered by the shallower PAN-B, while only a few areas need "stronger" restoration by the deeper PAN. This motivates us to accelerate an SR network by decomposing it into a base network for coarse restoration and a refine network to further enhance the (sparsely) complex areas.

Overview. In this work, we propose a Mask-Guided Acceleration (MGA) scheme to accelerate general SR networks. Specifically, we design a Mask Prediction (MP) module (will be introduced in §3.3) to indicate the areas with large errors between coarse and final SR images. We then incorporate the MP module into a decomposed SR network (D-Net) to contain three parts: the base network (Base-Net), the MP module, and the refine network (Refine-Net). Our MGA scheme is illustrated in Figure 3. The LR image is first fed into the Base-Net to obtain a feature map $\boldsymbol{F}_c$, which is upsampled to produce a coarse SR image $\boldsymbol{I}_c$. Then we use the proposed MP module to generate an error mask $\boldsymbol{M}$ from $\boldsymbol{F}_c$. Guided by the mask $\boldsymbol{M}$, we select a few areas with the largest errors and crop the respective feature patches from $\boldsymbol{F}_c$. The feature patches are fed into Refine-Net for further

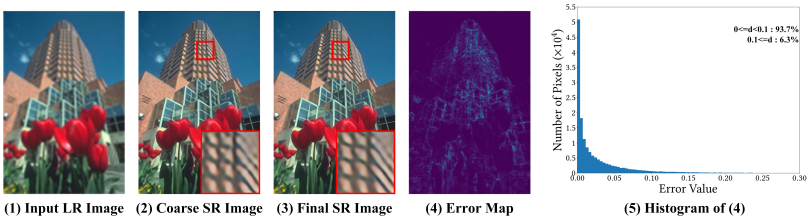

Fig. 2: **Difference between the coarse and final SR images**. The LR image (1) is restored by PAN-B and PAN to produce the coarse SR image (2) and the final SR image (3), respectively. (4): the error map between (2) and (3).

refinement. Finally, we replace the respective patches in the coarse SR image $\boldsymbol{I}_c$ by the refined patches to output the final SR image $\boldsymbol{I}_{SR}$.

3.2 Base Network for Global Super-Resolution

Given an input LR image $\boldsymbol{I}_{LR}$, the goal of Base-Net is to extract its coarse feature map $\boldsymbol{F}_c$, which is upsampled to output a coarse SR image $\boldsymbol{I}_c$. The smooth areas dominating the LR image $\boldsymbol{I}_{LR}$ can be well restored with no need of further processing. Since the Base-Net is shallower than the decomposed SR network, the complex areas like edges and textures are still relatively under-SR when compared to the smooth areas. By varying the depth of Base-Net, it is feasible to trade-off its SR capability and model efficiency.

3.3 Mask Prediction and Feature Patch Selection

To locate the under-SR areas in $\boldsymbol{I}_c$ for further refinement, we feed the coarse feature $\boldsymbol{F}_c$ into our Mask-Prediction module to obtain an error mask $\boldsymbol{M}$, and accordingly select K feature patches $\{\boldsymbol{F}_k\}_{k=1}^{K}$ from $\boldsymbol{F}_c$ with the largest errors.
Mask prediction. Given the coarse feature $\boldsymbol{F}_c \in \mathbb{R}^{H\times W\times C}$, our Mask Prediction (MP) module is to estimate an adaptive error mask $\boldsymbol{M}$ indicating the under-SR areas in the coarse SR image $\boldsymbol{I}_c$. Specifically, as shown in Figure 4, the coarse feature $\boldsymbol{F}_c$ is input into two 3×3 convolutional layers with a Global Average Pooling (GAP) to produce a spatial feature map $\boldsymbol{F}_s \in \mathbb{R}^{\frac{1}{p}H\times\frac{1}{p}W\times 2}$, where $p\geq 1$ is the spatial size of the feature patch $\boldsymbol{F}_k$. Note that each spatial element of $\boldsymbol{F}_s$ corresponds to a feature patch $\boldsymbol{F}_k \in \mathbb{R}^{p\times p\times C}$. Then, a softmax operation is performed on $\boldsymbol{F}_s$ along the channel dimension to output a spatial mask $\boldsymbol{M}_s \in \mathbb{R}^{\frac{1}{p}H\times\frac{1}{p}W\times 2}$. The two channels in $\boldsymbol{M}_s$, denoted as $\boldsymbol{M}_s^1$ and $\boldsymbol{M}_s^2$, indicate the possibilities of being well-SR and under-SR patches, respectively.

The under-SR mask $\boldsymbol{M}_s^2 \in \mathbb{R}^{\frac{1}{p}H\times\frac{1}{p}W}$, obtained by the spatial-wise softmax operation, only reflects the position-wise possibilities of being under-SR patches. To integrate the surrounding information for more comprehensive mask prediction, we utilize a convolution operation over the $\boldsymbol{M}_s^2$ for larger spatial perception.

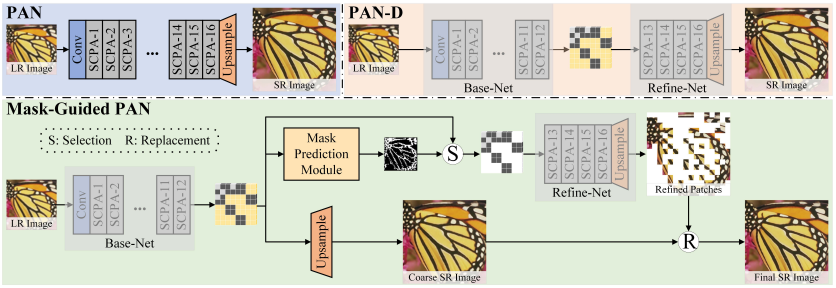

Fig. 3: **Illustration of our Mask-Guided Acceleration (MGA) scheme to accelerate super-resolution (SR) networks, *e.g.*, PAN [49]. PAN** (top-left) mainly contains 16 SCPA blocks. **PAN-D** (top-right) is the decomposed PAN consisting of a Base-Net with 12 SCPA blocks and a Refine-Net with 4 SCPA blocks. **Mask-Guided PAN** (bottom) is built upon the PAN-D, but accelerated by our MGA scheme. Here, the LR image is fed into the Base-Net to output a coarse feature. This coarse feature is upsampled to produce a coarse SR image, and used to estimate an error mask by the proposed Mask Prediction module. K patches from coarse feature with largest errors are selected for further refinement by the Refine-Net. The refined image patches are used to replace the respective patches in the coarse SR image to output the final SR image.

To achieve position-adaptive interaction, here we implement the dynamic convolution (D-Conv) [18] over the $\boldsymbol{M}_s^2$ to generate position-wise filters by integrating the corresponding surrounding features in $\boldsymbol{F}_c$. The mask $\boldsymbol{M}_s^2$ processed by the D-Conv is employed as the final error mask, denoted as $\boldsymbol{M} \in \mathbb{R}^{\frac{1}{p}H \times \frac{1}{p}W}$.

Feature patch selection. Here, each element in the error mask $\boldsymbol{M}$ corresponds to a feature patch in the coarse feature $\boldsymbol{F}_c$. We select K feature patches from $\boldsymbol{F}_c$ by the K largest elements in $\boldsymbol{M}$. These selected feature patches $\{\boldsymbol{F}_k \in \mathbb{R}^{p \times p \times C}\}_{k=1}^{K}$, instead of $\boldsymbol{F}_c$, will be fed into the Refine-Net for further refinement. This greatly reduces the amounts of computational costs for the original SR network.

3.4 Refine Network for Local Enhancement

The K feature patches $\{\boldsymbol{F}_k\}_{k=1}^{K}$ selected by our MP module, mainly from the complex areas under-SR by the Base-Net, are fed into the Refine-Net for further enhancement. The outputs are the reconstructed high-resolution image patches $\{\boldsymbol{P}_k \in \mathbb{R}^{sp \times sp \times 3}\}_{k=1}^{K}$, where s is the SR scale factor (*e.g.*, 4). In this way, the Refine-Net avoids processing the whole coarse feature $\boldsymbol{F}_c \in \mathbb{R}^{H \times W \times C}$. Since one can set $Kp^2 \ll HW$, the computational costs of the original SR network can be largely reduced on the Refine-Net. This is the key reason that our MGA scheme is able to accelerate the original SR network. Finally, the refined patches $\{\boldsymbol{P}_k\}_{k=1}^{K}$ are used to replace the respective patches in the coarse SR image $\boldsymbol{I}_c$, to output the final SR image $\boldsymbol{I}_{SR}$.

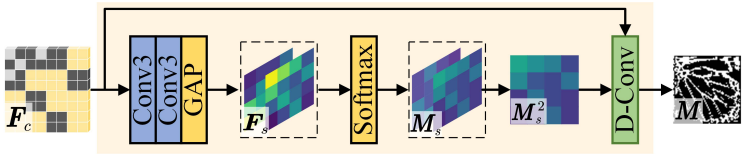

Fig. 4: **Our Mask Prediction module**. The feature $\boldsymbol{F}_c$ is fed into two 3×3 convolutional layers and a Global Average Pooling (GAP) to produce a spatial feature $\boldsymbol{F}_s$. A softmax operation is performed on $\boldsymbol{F}_s$ along the channel dimension to output a spatial mask $\boldsymbol{M}_s = [\boldsymbol{M}_s^1, \boldsymbol{M}_s^2]$. The feature $\boldsymbol{F}_c$ and mask $\boldsymbol{M}_s^2$ are fed into a dynamic convolution (D-Conv) to output final error mask $\boldsymbol{M}$.

3.5 Training Strategy

When directly trained end-to-end from scratch, the mask-guided SR network suffers from clear performance drops, as will be shown in §4.4 (Table 4). The main reason is that, in the early training, the initial features extracted by the Base-Net are of low quality and change greatly, making our MP-module unstably updated and failing to reach a good local optimum. To avoid this problem, we propose to train the Mask-Guided SR network in a three-step strategy.
Step 1. We train the decomposed SR network (D-Net) on all the training data (will be introduced in §4.2) with an ℓ_1 loss function.
Step 2. This step aims to endow the Refine-Net with the capability to process small feature patches. To this end, once the Base-Net extracts the coarse feature $\boldsymbol{F}_c$, we first crop small feature patches from it and then feed these cropped feature patches into the Refine-Net. The D-Net trained with all cropped feature patches is denoted as All-Net, which is trained with the same setting as that in **Step 1**.
Step 3. This step aims to train the mask-guided SR network, in which the Base-Net and Refine-Net are initialized by the All-Net in **Step 2**. Now we fix the learned weights of the Base-Net and Refine-Net, and only train our MP-module. To supervise the mask prediction process for better SR performance, we deploy the absolute difference between the coarse and final SR images obtained by the Base-Net and All-Net in **Step 2**, respectively, as the "ground-truth" for the corresponding mask $\boldsymbol{M}$ predicted by our MP module. Here, we also penalty the mask prediction by an ℓ_1 loss function.

4 Experiments

4.1 Implementation Details

Firstly, we train the decomposed SR network (D-Net). The mini-batch size per GPU is set as 32. We use the Adam optimizer [20] with the default setting. We set $p = 4$. The total number of iterations is 600K and divided into three identical periods. For each period, we use a cosine annealing strategy with warm-up to adjust the learning rate. Each period contains 200K iterations and is subdivided

into two stages. In the first 2K iterations, called warmup, the learning rate is increased from 4×10^{-5} to 4×10^{-4}. In the remaining 198K, the learning rate is decayed from 4×10^{-4} to 1×10^{-7}. Secondly, we use the trained D-Net as the initialization model of the All-Net. Then we train the All-Net for another 600K iterations in the same way as that for D-Net. Finally, we employ the pre-trained All-Net to initialize the Mask-Guided SR network, and only train the MP module while fixing the Base-Net and Refine-Net. We train the MP module for 100K iterations using the cosine annealing learning rate strategy, with 2K iterations for warm-up. The other settings are the same as the training of D-Net. We perform data argumentation with randomly horizontal/vertical flipping and rotation with 90°. For a fair comparison with the Base-Net of the original SR network, we also retrain it from scratch with the same settings as training the D-Net. All models are trained on two GeForce RTX 2080Ti GPUs.

4.2 Datasets and Metrics

Training set. We use the DIV2K [2] and Flickr2K [42] datasets for network training. We crop the HR images in DIV2K and Flickr2K into 480×480 sub-images, and use these sub-images as the HR images. We downsample the HR images by scale factors of 2, 3, or 4 to obtain the corresponding LR images. We randomly crop a 64×64 patch from each LR image and a 128×128 (or 192×192, 256×256) patch from the corresponding HR image as the paired training samples for the $\times 2$ (or $\times 3$, $\times 4$) SR task.
Test set. We evaluate different methods on seven standard datasets: Set5 [4], Set14 [45], B100 [31], Manga109 [32], Urban100 [17], Test2K and Test4K. For Test2K (or Test4K), as suggested in ClassSR [21], we downsample the 200 images (index: 1201–1400) from DIV8K [12] to 2K (or 4K) resolution as HR images. Here, we only provide the results for $\times 3$ and $\times 4$ SR tasks on Urban100, Test2K, and Test4K. Please refer to the *Supplementary File* for more comparison results.
Metric. We calculate PSNR and SSIM [43] on the Y channel of the YCbCr color space to evaluate different comparison methods.

4.3 Comparison Results

Comparison of SR network variants. We implement our MGA scheme into five popular SR networks with diverse parameter amounts, *i.e.*, FSRCNN [9] ($\sim$42K), PAN [49] ($\sim$300K), RFDN [29] ($\sim$700K), SRResNet [23] ($\sim$2M) and RCAN [47] ($\sim$16M). The details of decomposing different SR networks are provided in the *Supplementary File*. For each network, we evaluate its Base network (-B), the Decomposed network (-D) before acceleration, the All-Net trained with all cropped feature patches (-A), and the mask-guided network accelerated by our scheme (-M). The results on $\times 3$ and $\times 4$ SR tasks are listed in Table 1. One can see that the mask-guided networks (appended by "-M") achieve comparable PSNR/SSIM results with the decomposed ones (appended by "-D", achieving close SR performance to their original networks, respectively), but reducing the FLOPs amounts of these five SR networks by 10%$\sim$48% with a little

Table 1: **Comparison results of different SR networks** on the Urban100, Test2K and Test4K datasets. "-B": Base-Net. "-D": D-Net. "-A": ALL-Net. "-M": network accelerated by our MGA scheme. See §3.5 and §4.3.

Scale	Method	# Params	Urban100			Test2K			Test4K		
			PSNR	SSIM	FLOPs(G)	PSNR	SSIM	FLOPs(G)	PSNR	SSIM	FLOPs(G)
×3	FSRCNN-B	23K	25.87	0.7919	11.25	28.47	0.8187	26.27	29.87	0.8573	100.77
	FSRCNN-D	42K	26.07	0.7986	22.08	28.56	0.8211	51.55	29.99	0.8594	197.76
	FSRCNN-A	42K	26.24	0.8036	22.36	28.60	0.8220	52.20	30.05	0.8603	200.26
	FSRCNN-M	43K	26.22	0.8025	13.45	28.59	0.8216	28.61	30.03	0.8598	103.85
	PAN-B	204K	27.80	0.8453	27.69	29.10	0.8387	64.66	30.71	0.8749	248.07
	PAN-D	296K	27.97	0.8481	49.42	29.17	0.8395	115.38	30.80	0.8757	442.68
	PAN-A	296K	28.04	0.8499	67.18	29.20	0.8407	156.84	30.84	0.8768	601.75
	PAN-M	311K	27.96	0.8475	36.34	29.17	0.8396	74.94	30.78	0.8757	266.49
	RFDN-B	424K	27.89	0.8473	34.42	29.15	0.8393	80.37	30.79	0.8759	308.37
	RFDN-D	698K	28.08	0.8507	57.04	29.21	0.8405	133.18	30.85	0.8765	510.98
	RFDN-A	698K	28.32	0.8556	81.35	29.26	0.8420	189.94	30.91	0.8780	728.73
	RFDN-M	721K	28.28	0.8545	45.16	29.23	0.8403	93.67	30.84	0.8761	334.38
	SRResNet-B	1.10M	27.62	0.8420	120.13	28.93	0.8361	280.49	30.36	0.8712	1076.11
	SRResNet-D	1.67M	28.06	0.8501	194.93	29.18	0.8393	455.14	30.76	0.8751	1746.16
	SRResNet-A	1.67M	28.08	0.8503	234.22	29.20	0.8397	546.86	30.78	0.8753	2098.07
	SRResNet-M	1.71M	28.01	0.8485	144.70	29.18	0.8391	309.26	30.76	0.8747	1125.70
	RCAN-B	11.05M	28.57	0.8613	934.84	29.37	0.8456	2182.72	31.03	0.8808	8374.18
	RCAN-D	16.07M	29.10	0.8705	1360.41	29.55	0.8497	3176.36	31.25	0.8845	12186.37
	RCAN-A	16.07M	29.16	0.8717	1851.63	29.56	0.8499	4323.29	31.25	0.8847	16586.64
	RCAN-M	16.11M	29.10	0.8706	1110.19	29.55	0.8493	2362.27	31.23	0.8841	8574.54
×4	FSRCNN-B	23K	24.32	0.7192	10.93	27.01	0.7499	25.43	28.22	0.7984	97.57
	FSRCNN-D	42K	24.43	0.7230	21.63	27.06	0.7512	50.31	28.29	0.7997	193.01
	FSRCNN-A	42K	24.52	0.7271	21.78	27.09	0.7525	50.67	28.33	0.8009	194.41
	FSRCNN-M	43K	24.50	0.7261	14.60	27.09	0.7523	29.19	28.31	0.8004	101.74
	PAN-B	215K	25.87	0.7780	20.84	27.52	0.7716	48.47	28.91	0.8187	185.96
	PAN-D	313K	26.01	0.7828	36.05	27.56	0.7725	83.86	28.96	0.8195	321.74
	PAN-A	313K	26.05	0.7840	45.53	27.57	0.7730	105.91	28.97	0.8200	406.34
	PAN-M	328K	26.02	0.7825	29.74	27.57	0.7728	58.30	28.96	0.8195	200.37
	RFDN-B	433K	26.01	0.7839	19.89	27.53	0.7730	46.27	28.95	0.8202	177.51
	RFDN-D	717K	26.09	0.7862	33.11	27.60	0.7743	77.03	29.02	0.8213	295.53
	RFDN-A	717K	26.18	0.7886	46.83	27.63	0.7751	108.95	29.05	0.8221	418.02
	RFDN-M	740K	26.17	0.7878	29.94	27.61	0.7741	57.75	29.03	0.8210	196.15
	SRResNet-B	1.25M	25.74	0.7773	92.59	27.35	0.7696	215.40	28.49	0.8144	826.38
	SRResNet-D	1.97M	26.08	0.7857	159.59	27.57	0.7731	371.26	28.94	0.8195	1424.36
	SRResNet-A	1.97M	26.08	0.7859	181.77	27.57	0.7736	422.85	28.93	0.8199	1622.31
	SRResNet-M	2.01M	26.07	0.7850	124.04	27.57	0.7735	249.20	28.92	0.8197	871.89
	RCAN-B	11.02M	26.56	0.8002	547.86	27.74	0.7791	1274.50	29.16	0.8251	4889.71
	RCAN-D	16.00M	26.95	0.8115	808.21	27.85	0.7838	1880.14	29.31	0.8296	7213.30
	RCAN-A	16.00M	27.00	0.8128	1085.53	27.86	0.7842	2525.29	29.32	0.8300	9688.46
	RCAN-M	16.04M	26.96	0.8119	728.52	27.86	0.7837	1457.51	29.31	0.8295	5084.44

parameter growth. We also compare the visual results by different variants of RCAN/SRResNet in Figure 5. We observe that the mask-guided variants (-M) obtain comparable or better quality than the decomposed ones (-D), validating the effectiveness of our MGA scheme on preserving their SR capability.

Comparison with ClassSR [21]. We compare our MGA scheme with ClassSR on the Urban100, Test2K, and Test4K datasets. For a fair comparison, we retrain each SR network accelerated by ClassSR [21] on DIV2K and Flickr2K, while the other settings are kept unchanged. In Table 2, we compare the PSNR results on RGB color space, and the average FLOPs of ClassSR processing a whole image as our MGA does. Note that the ClassSR here obtains higher FLOPs than those reported in [21]. The reason is that the original ClassSR only calculates the average FLOPs on the cropped 32×32 image patches, while needing to consider the FLOPs of the overlapping areas between 32×32 image patches when processing a whole image. From Figure 5, we observe that the SR networks

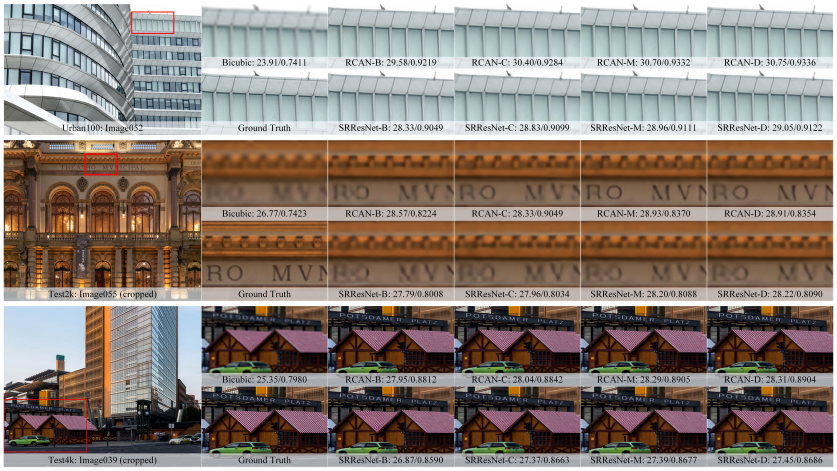

Fig. 5: **SR images by different methods on the $\times 4$ SR task**. "-B": BaseNet. "-C": network accelerated by ClassSR. "-M": network accelerated by our MGA scheme. "-D": D-Net. Please refer to §3.5 and §4.3 for more details.

Table 2: **Results of different SR networks accelerated by ClassSR and our MGA scheme** on the Urban100, Test2K and Test4K datasets. "-M": network accelerated by our MGA scheme. "-C": network accelerated by ClassSR.

Scale	Method	# Params	Urban100		Test2K		Test4K	
			PSNR(RGB)	FLOPs(G)	PSNR(RGB)	FLOPs(G)	PSNR(RGB)	FLOPs(G)
$\times 4$	FSRCNN-C	113K	22.89	20.07	25.61	36.77	26.91	139.72
	FSRCNN-M	43K	23.01	14.60	25.66	29.19	26.94	101.74
	SRResNet-C	3.06M	24.53	149.92	26.20	298.18	27.66	1135.60
	SRResNet-M	2.01M	24.55	124.04	26.20	249.20	27.66	871.89
	RCAN-C	30.11M	25.14	741.76	26.39	1380.80	27.88	5255.70
	RCAN-M	16.04M	25.43	728.52	26.46	1457.51	27.96	5084.44

of RCAN and SRResNet accelerated by our MGA scheme recover the structure and textures more clearly than those accelerated by ClassSR.

Speed on mobile devices. To test the speed of SR networks accelerated by our MGA scheme and ClassSR on mobile devices, we deploy the accelerated RCAN on Kirin 980 using the Pytorch Mobile framework[1]. The average speed is calculated for the $\times 4$ SR task with $256 \times 256 \times 3$ images. The average speeds of RCAN-D, RCAN-C and RCAN-M are 45.61s, 38.91s and 35.51s, respectively.

4.4 Ablation Study

Now we conduct a more detailed examination of our MGA scheme on SR to assess: 1) the design of our Mask Prediction (MP) module; 2) effectiveness of our MP module on mask prediction; 3) how to train mask-guided SR networks

[1] https://pytorch.org/mobile/home/

Fig. 6: **Visualization** of the LR image, the HR image, the "ground-truth" error mask (MASK_GT), and the error mask predicted by our MP module.

Table 3: **Results of PAN-M with the spatial feature map $\boldsymbol{F}_s$ of one channel or two channels in our MP module** on Urban100, Test2K, and Test4K. "1C": the spatial feature $\boldsymbol{F}_s$ is of one channel. The scale factor is 4.

Method	Urban100			Test2K			Test4K		
	PSNR	SSIM	FLOPs(G)	PSNR	SSIM	FLOPs(G)	PSNR	SSIM	FLOPs(G)
PAN-B	25.87	0.7780	20.84	27.52	0.7716	48.47	28.91	0.8187	185.96
PAN-M(1C)	25.96	0.7812	29.74	27.55	0.7725	58.30	28.94	0.8193	200.37
PAN-M	26.02	0.7825	29.74	27.57	0.7728	58.30	28.96	0.8195	200.37
PAN-D	26.01	0.7828	36.05	27.56	0.7725	83.86	28.96	0.8195	321.74

with our MP module; 4) the impact of decomposing manner to SR networks in our MGA scheme; 5) how the size of feature patches affects our MGA scheme; 6) how the order of selecting elements from the error mask influences the SR performance. More ablation studies are provided in the *Supplementary File*.

1) How to design our MP module? A trivial design of our MP module is to generate the spatial feature map $\boldsymbol{F}_s$ with only one channel, which needs to remove the softmax function. Taking PAN as an example, we denote this variant as "PAN-M (1C)". The comparison results bewteen PAN-M and "PAN-M (1C)", on Urban100, Test2K, and Tesk4K at $\times 4$ SR task, are listed in Table 3: PAN-M outperforms clearly "PAN-M (1C)" on PSNR and SSIM, but with close FLOPs. This shows the necessity to design a two-channel spatial feature map $\boldsymbol{F}_s$.

2) Effectiveness of our MP module. To study this problem, we visualize the mask predicted by our MP module in Figure 6. We observe that most of the under-SR pixels are successfully predicted by our MP module. The ℓ_1 error between the predicted MASK and the "ground-truth" mask "MASK_GT" (as described in §3.5) is very small (*i.e.*, 0.0346). This demonstrates that our MP module can indeed accurately predict the error masks indicating under-SR areas.

3) How to train the mask-guided SR networks with our MP module? To answer this question, we propose to train the mask-guided network with two other strategies. The first is to train the mask-guided network with our MP module end-to-end from scratch, denoted as "E2E". The second is to train the mask-guided network by the strategy introduced in §3.5, but without supervision on our MP module in **Step 3**, denoted as "w/o S". By taking FSRCNN for example, the results listed in Table 4 show that FSRCNN-M achieves higher PSNR and SSIM results than those of FSRCNN-M (E2E) and FSRCNN (w/o

Table 4: **Results of FSRCNN-M with different training strategies for our Mask Prediction (MP) module** on the Urban100, Test2K, and Test4K datasets. "E2E": end-to-end train the mask-guided network with our MP module from scratch. "w/o S": train our MP module separately in **Step 3** (see §3.5) without supervision. The scale factor is 2.

Method	Urban100			Test2K			Test4K		
	PSNR	SSIM	FLOPs(G)	PSNR	SSIM	FLOPs(G)	PSNR	SSIM	FLOPs(G)
FSRCNN-B	29.15	0.8913	12.38	31.52	0.9107	28.65	33.32	0.9326	109.92
FSRCNN-M(E2E)	29.10	0.8910	13.63	31.50	0.9110	30.23	33.27	0.9326	113.15
FSRCNN-M(w/o S)	29.50	0.8970	13.63	31.64	0.9124	30.23	33.47	0.9341	113.15
FSRCNN-M	29.56	0.8973	13.63	31.67	0.9128	30.23	33.49	0.9343	113.15
FSRCNN-D	29.38	0.8951	23.80	31.62	0.9122	55.08	33.43	0.9337	211.33

Table 5: **Results of PAN-M with different decomposing manners** on the Urban100, Test2K, and Test4K datasets. "$a + b$" means that the Base-Net has a SCPA blocks and the Refine-Net has b SCPA blocks. The scale factor is 2.

Method	Urban100			Test2K			Test4K		
	PSNR	SSIM	FLOPs(G)	PSNR	SSIM	FLOPs(G)	PSNR	SSIM	FLOPs(G)
PAN-M($10 + 6$)	31.67	0.9236	50.58	32.41	0.9234	110.33	34.39	0.9428	408.74
PAN-M($12 + 4$)	31.81	0.9252	55.14	32.45	0.9240	122.25	34.45	0.9433	457.42
PAN-M($14 + 2$)	31.90	0.9263	59.69	32.50	0.9247	134.17	34.50	0.9439	506.09
PAN-D	31.86	0.9259	73.91	32.46	0.9242	171.09	34.48	0.9436	656.39

S). This validates the advantages of our training strategy in §3.5 over the two variant strategies of "E2E" and "w/o S", for our MGA scheme.

4) How the depths of the Base-Net and Refine-Net influence the SR performance of the decomposed network? To study this problem, by taking PAN for example, we compare the three mask-guided SR networks obtained by decomposing the PAN-D into three variants: "PAN-M(10+6)" with 10 SCPAs in Base-Net and 6 SCPAs in Refine-Net; "PAN-M(12+4)" with 12 SCPAs in Base-Net and 4 SCPAs in Refine-Net; "PAN-M(14+2)" with 14 SCPAs in Base-Net and 2 SCPAs in Refine-Net. The results listed in Table 5 show that, by varying the depth of Refine-Net, our MGA scheme provides a flexible trade-off between the capability (on PSNR/SSIM) and efficiency (FLOPs) of an SR network.

5) How does the size p of feature patches influence the performance of the mask-guided SR networks? To this end, we compare the mask-guided SR networks with different sizes of feature patches in our MGA scheme. By taking FSRCNN for example, we implement the FSRCNN-M with $p = 2, 4, 8$ and denote the resulting variants as FSRCNN-M(2), FSRCNN-M(4), and FSRCNN-M(8). The results are listed in Table 6. One can see that, on Urban100 and Test2K, FSRCNN-M(2) (or FSRCNN-M(4)) slightly outperforms FSRCNN-M(4) (or FSRCNN-M(8)) with a little growth on FLOPs. In summary, larger p reduces the FLOPs amount but usually degrades the SR performance.

Table 6: **Results of FSRCNN-M with different sizes of feature patches (selected between MP module and Refine-Net)** on the Urban100, Test2K, and Test4K datasets. "2": 2×2. "4": 4×4. "8": 8×8. The scale factor is 4.

Method	Urban100			Test2K			Test4K		
	PSNR	SSIM	FLOPs(G)	PSNR	SSIM	FLOPs(G)	PSNR	SSIM	FLOPs(G)
FSRCNN-B	24.32	0.7192	10.93	27.01	0.7499	25.43	28.22	0.7984	97.57
FSRCNN-M(2)	24.50	0.7264	14.73	27.08	0.7527	29.26	28.30	0.8003	101.82
FSRCNN-M(4)	24.50	0.7261	14.60	27.09	0.7523	29.19	28.31	0.8004	101.74
FSRCNN-M(8)	24.48	0.7226	14.52	27.07	0.7517	28.64	28.29	0.7999	101.71
FSRCNN-D	24.43	0.7230	21.63	27.06	0.7512	20.31	28.29	0.7997	193.01

Table 7: **Results of RFDN-M with different orders of selecting K elements from error mask**, on the Urban100, Test2K, and Test4K datasets. "S", "R" or "L" means selecting K smaller, random or larger errors, respectively. "All": selecting all errors for each image. The scale factor is 2.

Method		K=0	K=1000, S	K=1000, R	K=1000, L	K=All/2, S	K=All/2, R	K=All/2, L	K=All
Urban100	PSNR	31.99	31.99	32.01	32.26	32.03	32.14	32.36	32.36
	SSIM	0.9265	0.9265	0.9270	0.9296	0.9273	0.9283	0.9304	0.9306
Test2K	PSNR	32.48	32.48	32.49	32.57	32.49	32.55	32.62	32.62
	SSIM	0.9246	0.9246	0.9248	0.9254	0.9249	0.9253	0.9261	0.9262
Test4K	PSNR	34.51	34.51	34.53	34.59	34.53	34.59	34.66	34.66
	SSIM	0.9440	0.9440	0.9441	0.9445	0.9441	0.9445	0.9451	0.9451

6) How does the order of selecting elements from error mask M influence our MGA on SR? In our MGA scheme, we select K largest elements from the error mask $\boldsymbol{M}$. To validate this point, we change the order of selecting the K elements from "Larger" (L) to "Smaller" (S) or "Random" (R). By taking RFDN for example, we list the results of different variants in Table 7. We observe that RFDN-M with "L" outperforms the other variants obviously. This validates the effectiveness of selecting K largest elements in our MGA scheme.

5 Conclusion

In this paper, we proposed a Mask Guided Acceleration (MGA) scheme to accelerate popular single image super-resolution (SR) networks. Our MGA scheme decomposes an SR network into a Base-Net to extract a coarse feature and a Refine-Net to refine the mostly under-SR areas. To locate these areas, we designed a Mask Prediction module for error mask generation. Some feature patches were selected accordingly from the coarse feature to trade-off model capability and efficiency. Experiments on seven benchmark datasets demonstrated that, our MGA scheme largely reduces the computational costs of five SR networks with different complexities, while preserving well their SR capability.

Acknowledgements. This work was supported by The National Natural Science Foundation of China (No. 62002176 and 62176068).

References

1. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., GhemawFat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D.G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., Zheng, X.: Tensorflow: A system for large-scale machine learning. In: Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation. p. 265–283. USENIX Association, USA (2016)
2. Agustsson, E., Timofte, R.: Ntire 2017 challenge on single image super-resolution: Dataset and study. In: IEEE Conf. Comput. Vis. Pattern Recog. Worksh. pp. 126–135 (2017)
3. Ahn, N., Kang, B., Sohn, K.A.: Fast, accurate, and lightweight super-resolution with cascading residual network. In: Eur. Conf. Comput. Vis. pp. 252–268 (2018)
4. Bevilacqua, M., Roumy, A., Guillemot, C., Alberi-Morel, M.L.: Low-complexity single-image super-resolution based on nonnegative neighbor embedding (2012)
5. Chen, H., Wang, Y., Guo, T., Xu, C., Deng, Y., Liu, Z., Ma, S., Xu, C., Xu, C., Gao, W.: Pre-trained image processing transformer. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 12299–12310 (2021)
6. Dai, T., Cai, J., Zhang, Y., Xia, S.T., Zhang, L.: Second-order attention network for single image super-resolution. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 11065–11074 (2019)
7. Dong, C., Loy, C.C., He, K., Tang, X.: Learning a deep convolutional network for image super-resolution. In: Eur. Conf. Comput. Vis. Lecture Notes in Computer Science, vol. 8692, pp. 184–199. Springer (2014)
8. Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks. IEEE Trans. Pattern Anal. Mach. Intell. **38**(2), 295–307 (2015)
9. Dong, C., Loy, C.C., Tang, X.: Accelerating the super-resolution convolutional neural network. In: Eur. Conf. Comput. Vis. pp. 391–407. Springer (2016)
10. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Adv. Neural Inform. Process. Syst. **27** (2014)
11. Gu, J., Dong, C.: Interpreting super-resolution networks with local attribution maps. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 9199–9208 (2021)
12. Gu, S., Lugmayr, A., Danelljan, M., Fritsche, M., Lamour, J., Timofte, R.: Div8k: Diverse 8k resolution image dataset. In: Int. Conf. Comput. Vis. Worksh. pp. 3512–3516. IEEE (2019)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 770–778 (2016)
14. He, X., Mo, Z., Wang, P., Liu, Y., Yang, M., Cheng, J.: Ode-inspired network design for single image super-resolution. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1732–1741 (2019)
15. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7132–7141 (2018)
16. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4700–4708 (2017)
17. Huang, J.B., Singh, A., Ahuja, N.: Single image super-resolution from transformed self-exemplars. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5197–5206 (2015)
18. Jia, X., De Brabandere, B., Tuytelaars, T., Gool, L.V.: Dynamic filter networks. Adv. Neural Inform. Process. Syst. **29**, 667–675 (2016)

19. Kim, J., Lee, J.K., Lee, K.M.: Accurate image super-resolution using very deep convolutional networks. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1646–1654. IEEE Computer Society (2016)
20. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Int. Conf. Learn. Represent. (2015)
21. Kong, X., Zhao, H., Qiao, Y., Dong, C.: Classsr: A general framework to accelerate super-resolution networks by data characteristic. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 12016–12025 (2021)
22. Lai, W.S., Huang, J.B., Ahuja, N., Yang, M.H.: Deep laplacian pyramid networks for fast and accurate super-resolution. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 624–632 (2017)
23. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4681–4690 (2017)
24. Li, W., Zhou, K., Qi, L., Jiang, N., Lu, J., Jia, J.: Lapar: Linearly-assembled pixel-adaptive regression network for single image super-resolution and beyond. In: Adv. Neural Inform. Process. Syst. (2020)
25. Li, Y., Gu, S., Mayer, C., Gool, L.V., Timofte, R.: Group sparsity: The hinge between filter pruning and decomposition for network compression. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 8018–8027 (2020)
26. Lim, B., Son, S., Kim, H., Nah, S., Mu Lee, K.: Enhanced deep residual networks for single image super-resolution. In: IEEE Conf. Comput. Vis. Pattern Recog. Worksh. pp. 136–144 (2017)
27. Lin, S., Ryabtsev, A., Sengupta, S., Curless, B.L., Seitz, S.M., Kemelmacher-Shlizerman, I.: Real-time high-resolution background matting. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 8762–8771 (2021)
28. Liu, J.J., Hou, Q., Cheng, M.M., Wang, C., Feng, J.: Improving convolutional networks with self-calibrated convolutions. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 10096–10105 (2020)
29. Liu, J., Tang, J., Wu, G.: Residual feature distillation network for lightweight image super-resolution. In: Eur. Conf. Comput. Vis. pp. 41–55. Springer (2020)
30. Liu, J., Zhang, W., Tang, Y., Tang, J., Wu, G.: Residual feature aggregation network for image super-resolution. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 2359–2368 (2020)
31. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Int. Conf. Comput. Vis. vol. 2, pp. 416–423. IEEE (2001)
32. Matsui, Y., Ito, K., Aramaki, Y., Fujimoto, A., Ogawa, T., Yamasaki, T., Aizawa, K.: Sketch-based manga retrieval using manga109 dataset. Multimedia Tools and Applications **76**(20), 21811–21838 (2017)
33. Mei, Y., Fan, Y., Zhou, Y.: Image super-resolution with non-local sparse attention. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3517–3526 (2021)
34. 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: Eur. Conf. Comput. Vis. pp. 191–207. Springer (2020)
35. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Adv. Neural Inform. Process. Syst. (2019)

36. Song, D., Wang, Y., Chen, H., Xu, C., Xu, C., Tao, D.: Addersr: Towards energy efficient image super-resolution. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 15648–15657 (2021)
37. Tai, Y., Yang, J., Liu, X.: Image super-resolution via deep recursive residual network. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 2790–2798. IEEE Computer Society (2017)
38. Tai, Y., Yang, J., Liu, X., Xu, C.: Memnet: A persistent memory network for image restoration. In: Int. Conf. Comput. Vis. pp. 4549–4557. IEEE Computer Society (2017)
39. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Adv. Neural Inform. Process. Syst. pp. 5998–6008 (2017)
40. Wang, L., Dong, X., Wang, Y., Ying, X., Lin, Z., An, W., Guo, Y.: Exploring sparsity in image super-resolution for efficient inference. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4917–4926 (2021)
41. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7794–7803 (2018)
42. Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Qiao, Y., Change Loy, C.: Esrgan: Enhanced super-resolution generative adversarial networks. In: Eur. Conf. Comput. Vis. pp. 0–0 (2018)
43. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
44. Xie, W., Song, D., Xu, C., Xu, C., Zhang, H., Wang, Y.: Learning frequency-aware dynamic network for efficient super-resolution. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4308–4317 (2021)
45. Zeyde, R., Elad, M., Protter, M.: On single image scale-up using sparse-representations. In: International Conference on Curves and Surfaces. pp. 711–730. Springer (2010)
46. Zhang, K., Danelljan, M., Li, Y., Timofte, R., Liu, J., Tang, J., Wu, G., Zhu, Y., He, X., Xu, W., et al.: Aim 2020 challenge on efficient super-resolution: Methods and results. In: Eur. Conf. Comput. Vis. pp. 5–40. Springer (2020)
47. Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: Eur. Conf. Comput. Vis. pp. 286–301 (2018)
48. Zhang, Y., Tian, Y., Kong, Y., Zhong, B., Fu, Y.: Residual dense network for image super-resolution. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 2472–2481 (2018)
49. Zhao, H., Kong, X., He, J., Qiao, Y., Dong, C.: Efficient image super-resolution using pixel attention. In: Eur. Conf. Comput. Vis. pp. 56–72. Springer (2020)
50. Zhou, S., Zhang, J., Zuo, W., Loy, C.C.: Cross-scale internal graph neural network for image super-resolution. Adv. Neural Inform. Process. Syst. **33**, 3499–3509 (2020)