Efficient Deep Models for Real-Time 4K Image Super-Resolution. NTIRE 2023 Benchmark and Report

Marcos V. Conde[†] Eduard Zamfir[†] Radu Timofte[†] Daniel Motilla[‡] Cen Liu
Zexin Zhang Yunbo Peng Yue Lin Jiaming Guo Xueyi Zou Yuyi Chen
Yi Liu Jia Hao Youliang Yan Yuanfan Zhang Gen Li Lei Sun
Lingshun Kong Haoran Bai Jinshan Pan Jiangxin Dong Jinhui Tang
Mustafa Ayazoglu Bahri Batuhan Bilecen Mingxi Li Yuhang Zhang Xianjun Fan
Yankai Sheng Long Sun Zibin Liu Weiran Gou Shaoqing Li Ziyao Yi
Yan Xiang Dehui Kong Ke Xu Ganzorig Gankhuyag Kihwan Yoon Jin Zhang
Gaocheng Yu Feng Zhang Hongbin Wang Zhou Zhou Jiahao Chao
Hongfan Gao Jiali Gong Zhengfeng Yang Zhenbing Zeng Chengpeng Chen
Zichao Guo Anjin Park Yuqing Liu Qi Jia Hongyuan Yu Xuanwu Yin
Kunlong Zuo Dongyang Zhang Ting Fu Zhengxue Cheng Shiai Zhu
Dajiang Zhou Hongyuan Yu Weichen Yu Lin Ge Jiahua Dong Yajun Zou
Zhuoyuan Wu Binnan Han Xiaolin Zhang Heng Zhang Xuanwu Yin Ben Shao
Shaolong Zheng Daheng Yin Baijun Chen Mengyang Liu Marian-Sergiu Nistor
Yi-Chung Chen Zhi-Kai Huang Yuan-Chun Chiang Wei-Ting Chen
Hao-Hsiang Yang Hua-En Chang I-Hsiang Chen Chia-Hsuan Hsieh Sy-Yen Kuo
Tu Vo Qingsen Yan Yun Zhu Jinqiu Su Yanning Zhang Cheng Zhang
Jiaying Luo Youngsun Cho Nakyung Lee Kunlong Zuo

Figure 1. **NTIRE 2023 Real-Time 4K SR.** We introduce a new benchmark and a diverse test set for 4K Super-Resolution.

Abstract

This paper introduces a novel benchmark for efficient upscaling as part of the NTIRE 2023 Real-Time Image Super-Resolution (RTSR) Challenge, which aimed to upscale images from 720*p and* 1080*p resolution to native* 4*K (*$\times$*2 and* $\times$*3 factors) in real-time on commercial GPUs. For this, we use a new test set containing diverse 4K images ranging from digital art to gaming and photography. We assessed the methods devised for 4K SR by measuring their runtime, parameters, and FLOPs, while ensuring a minimum PSNR fidelity over Bicubic interpolation. Out of the 170 participants, 25 teams contributed to this report, making it the most comprehensive benchmark to date and showcasing the latest advancements in real-time SR.*

[†] Organizers and corresponding authors. Computer Vision Lab, University of Würzburg, Germany. [‡] Co-organizer. SIE FTG, USA.
{marcos.conde, radu.timofte}@uni-wuerzburg.de
NTIRE 2023 webpage: https://cvlai.net/ntire/2023/
Code: https://github.com/eduardzamfir/NTIRE23-RTSR

1. Introduction

Single image super-resolution (SR) refers to the process of generating a high-resolution (HR) image from a single degraded low-resolution (LR) image. This ill-posed problem was initially solved using interpolation methods [28, 77–79]. However, with the emergence of deep learning, SR is now commonly approached through the use of deep neural networks [17,24,49,56,57,84,88,99]. Image SR assumes that the LR image is obtained through two major degradation processes: blurring and down-sampling. This can be expressed as:

$$\mathbf{y} = (\mathbf{x} * \mathbf{k}) \downarrow_s, \tag{1}$$

where $*$ represents the convolution operation between the LR image and the blur kernel, and $\downarrow_s$ is the down-sampling operation with respective down-sampling factor $\times s$. Most SR methods are built around the Bicubic model [77,78] with various down-scaling factors (*e.g.* $\times 2$, $\times 3$, $\times 4$, $\times 8$).

The advancements in hardware technologies have led to the training of larger and deeper neural networks for image super-resolution, resulting in significant performance improvements. However, these breakthroughs often come at the cost of introducing more complex approaches [3, 20, 56, 84, 99]. Since the seminal work by Shi *et al*. [70], the design of efficient deep neural networks for single image super-resolution [40, 47, 72, 81, 101] has become pivotal. Various workshops and challenges, such as [42, 53, 94], have emerged as popular forums for sharing ideas and advancing the state-of-the-art in efficient and real-time SR. Publicly available large-scale datasets have been instrumental in driving recent advances in image and video SR [1,32,52,66, 76]. However, with the exception of DIV8K [32] and [95], most existing datasets have images of limited resolution *e.g.* 2K. In addition, the practical challenge of performing real-time SR of images and videos to 4K resolution has received relatively little attention so far.

As the amount of digital content continues to surge, there is a mounting demand for effective SR techniques for rendered content [86, 90]. However, rendering presents unique challenges as it often exhibits significant aliasing, resulting in jagged lines and other sampling artifacts. Consequently, up-scaling rendered content requires a novel approach that involves both anti-aliasing and interpolation, which is distinct from the well-established research on denoising and deblurring in existing SR research [86].

In conjunction with the 2023 New Trends in Image Restoration and Enhancement (NTIRE) workshop, we introduce the real-time 4K super-resolution challenge. The challenge entails super-resolving a LR image from either 720p or 1080p to 4K resolution using a network that reduces one or several aspects, such as runtime, parameters, FLOPs, and memory consumption. The goal is to at least outperform bicubic interpolation on a new and diverse benchmark, while maintaining efficiency. The challenge seeks to identify innovative and advanced solutions for real-time super-resolution, benchmark their efficiency, and identify general trends for designing efficient SR networks.

2. NTIRE 2023 Real-Time Super-Resolution Challenge

The aim of this challenge is to create real-time super-resolution (SR) methods, with a specific focus on up-scaling to 4K resolution. We believe that this area remains largely unexplored within the computer vision community. The challenge has three main objectives: Firstly, to advance research on real-time SR methods. Secondly, to introduce a novel and competitive benchmark for 4K SR, utilizing various image types such as digital art and natural imagery. Thirdly, to facilitate interactions between academic and industry participants and encourage potential collaborations.

2.1. 4K SR Benchmark Dataset

The *4K RTSR benchmark* provides a unique test set comprising ultra-high resolution images from various sources, setting it apart from traditional super-resolution benchmarks. Specifically, the benchmark addresses the increasing demand for upscaling computer-generated content *e.g.* gaming and rendered content, in addition to photorealistic imagery, thereby posing a different challenge for existing SR approaches. The test set includes diverse content such as rendered gaming images, digital art, as well as high-resolution photorealistic images of animals, city scenes, and landscapes, totaling 110 test samples. We created this benchmark with the intention of advancing the development of SR methods, as well as replacing outdated test sets such as Set5 [7], Set14 [93], and Urban100 [39].

All the images in the benchmark testset are at least 4K resolution *i.e.* 3840×2160 (some are bigger, even 8K). The images were filtered manually to ensure there are not unpleasant effects such as noise or strong defocus.

The **distribution** of the 4K RTSR benchmark testset is: 14 real-world captures using a 60MP DSLR camera, 21 rendered images using Unreal Engine [38], 75 diverse images *e.g.* animals, paintings, digital art, nature, buildings, etc.

2.2. Baseline Model

Previous lightweight SR methods [51] such as IMDN [40] or RFDN [60] are not fast enough for this task. For this reason, we use *RT4KSR* [92] as the baseline model for this challenge. The primary objective is to enhance its efficiency in terms of runtime, parameter count and FLOPs. Drawing inspiration from the research presented in [42,53], the baseline design utilizes a shallow convolutional architecture to achieve rapid and precise reconstruction performance. The proposed baseline stacks five simple

3×3 convolutions with a GeLU activation layer and adds a global residual connection with LayerNorm [6] before the standard depth2scale up-sampling operation. Besides, the authors in [92] develop a sophisticated approach that improves model efficiency by downscaling feature maps. To avoid losing important high frequency details that are already scarce, the authors propose extracting HF details from the LR input prior to its downscaling. Additionally, the authors provide a detailed roadmap of their method's development, resulting in a competitive shallow CNN design that can be scaled up and achieves performance comparable to previous state-of-the-art efficient SR models.

2.3. Tracks and Competition

The objective of this challenge is to develop a high-performance SR technique that can upscale a broad range of images to 4K resolution in real-time, while ensuring a PSNR above a traditional Bicubic interpolation.

Track 1: 1080p to 4K. The first challenge track addresses X2 up-scaling from 1080p to 4K resolution.

Track 2: 720p to 4K. The second leg of this NTIRE challenge addresses X3 up-scaling from 720p to 4K resolution.

Challenge Phases. *Development and Validation Phase.* The participants were provided with access to a validation set comprising of 100 images from the DIV2K validation split, along with an additional collection of 50 images that included a variety of content, from videogames to realistic high-resolution photography. The baseline model, scoring function, and evaluation scripts were made available to the participants through GitHub (https://github.com/eduardzamfir/NTIRE23-RTSR). This allowed the participants to benchmark the performance of their models on their systems. During the development phase, the objective was aimed at up-scaling 2K imagery since DIV2K did not include any 4K imagery. *Testing Phase.* During the final test phase, the participating teams received a 4K benchmark comprising 110 diverse images. However, they did not have access to the HR ground-truth. Once the participants generated their super-resolved results, they submitted their code, factsheets and resulting images to the organizers via email. The organizers then validated and executed the submitted code to obtain the final results, which were later conveyed to the participants upon completion of the challenge.

Evaluation Protocol. The quantitative evaluation metrics for this challenge comprise of testing PSNR, runtime, number of parameters, number of FLOPs and maximum GPU memory consumed during inference. The PSNR is calculated on 110 RGB images sourced from our 4K benchmark test set. The corresponding degraded images are obtained through bicubic down-scaling to their respective resolutions (1080p for X2 and 720p for X3 up-scaling). The average runtime is determined by using mixed-precision and repeatedly evaluating randomly initialized tensors of corresponding sizes to overcome any bottlenecks that may arise due to data loading. The FLOPs are evaluated on an input image of size 1920×1080 and 1280×720, respectively.

$$S = \frac{2^{2\times(\text{PSNR}_M - \text{PSNR}_B)}}{C \times T_M^{0.5}} \tag{2}$$

Similar to [42], we determine the final score S of each participant in the challenge by utilizing Eq. (2), in which PSNR_M and T_M represent the PSNR result and runtime of the individual submission. Additionally, the scoring function is designed to prioritize faster runtime over restoration accuracy. However, in cases where two methods have similar runtimes, the PSNR value will be the deciding factor.

Related NTIRE 2023 Challenges. The NTIRE 2023 Real-Time Image Super-Resolution (RTSR) Challenge is part of the NTIRE 2023 Workshop series of challenges on: night photography rendering [71], HR depth from images of specular and transparent surfaces [91], image denoising [55], video colorization [44], shadow removal [80], quality assessment of video enhancement [62], stereo super-resolution [82], light field image super-resolution [85], image super-resolution ($\times 4$) [100], 360° omnidirectional image and video super-resolution [9], lens-to-lens bokeh effect transformation [18], real-time 4K super-resolution [19], HR nonhomogenous dehazing [4], efficient super-resolution [54].

2.4. Architectures and Main Ideas

Here we summarize the core ideas behind the most competitive solutions. Each proposed solution will be covered in the following Sec. 3 and Tab. 2.

1. **Re-parameterization** allows to train the network using complex blocks [22], while during inference the so-called RepBlocks can be reduced to a simple 3×3 convolution.

2. **Pixel shuffle** and unshuffle (also known as depth-to-space and space-to-depth respectively) [70] to efficiently transform the features maps and perform both spatial upsampling and downsampling.

3. **Multi-stage Training.** Since the neural networks are extremely constrained and shallow, this technique allows to maximize learning by alternating different learning rates and loss functions.

https://cvlai.net/ntire/2023/

Table 1. **Results of the NTIRE23 Real-Time SR challenge.** The runtimes are computed using a Nvidia RTX3090 GPU. The teams are ordered by their ranking according to their score. For better comparison we color-code the runtime using < 24 FPS, $30 > x > 24$ FPS, $60 > x > 30$ FPS, $120 > x > 60$ FPS and > 120 FPS, respectively.

Team	Score	# Params (M)	FLOPs (G)	PSNR (dB, ↑)		SSIM (↑)	Runtime (ms, ↓)
				RGB	*Y*	*RGB*	
Track 1: Upscaling from 1080p to 4K resolution.							
Bicubic	-	-	-	33.92	36.66	0.8829	0.46
Noah TerminalVision	**24.13**	2.3523	9.062	35.02	37.74	0.8957	**3.190**
ALONG	**23.81**	0.0668	15.3281	34.63	37.38	0.8906	**1.910**
RTVSR	**23.13**	0.0266	13.7687	34.71	37.50	0.8910	**2.240**
Team OV	19.06	0.0042	8.734	34.62	37.45	0.8899	2.910
DFCDN Team	15.17	0.0064	6.0881	34.63	37.46	0.8916	4.670
DoYouChargeQQCoin	15.07	0.0008	1.6921	34.14	36.97	0.8855	2.380
NJUST-RTSR	14.96	0.0114	23.5893	34.74	37.64	0.8901	5.560
Multimedia	14.09	0.0100	20.4125	34.85	37.61	0.8926	7.300
PixelBE	13.12	0.0137	14.7226	34.70	37.52	0.8908	6.840
z6	12.87	0.0414	85.7309	35.02	37.76	0.8948	11.19
AGSR	12.77	0.0068	14.0673	34.31	37.00	0.8888	4.220
Antins cv	11.25	0.0111	22.9174	34.71	37.56	0.8921	9.470
ECNU SR	10.37	0.1623	83.2094	35.30	37.95	0.8971	25.23
R.I.P. ShopeeVideo	9.68	0.3987	272.7942	35.32	38.01	0.8971	29.73
dh isp	7.63	0.0113	23.4234	33.99	36.89	0.8809	7.600
P.AI.R	6.27	0.0212	38.486	34.65	37.47	0.8905	28.31
NTU BL6	6.07	0.2223	409.8416	35.26	38.04	0.8977	69.37
diSRupt	5.54	0.0500	207.0	34.07	36.86	0.8830	16.00
Touch Fish	5.03	0.0641	132.5777	34.28	37.14	0.8862	26.31
SEU CNII	4.84	0.0299	58.5454	34.24	37.10	0.8858	26.89
KCML2	3.99	0.0392	57.2567	34.24	37.09	0.8851	39.17
NPU SR	3.45	0.2001	0.165 (*)	34.49	37.42	0.8895	74.00
YNOT	2.25	0.4734	422.6991	34.03	36.99	0.8844	92.79
Our Baseline [92]	9.27	0.0445	171.99	34.22	37.01	0.8854	7.090
Track 2: Upscaling from 720p to 4K resolution.							
Bicubic	-	-	-	31.30	33.82	0.8245	0.46
Aselsan Research	**31.26**	0.0504	11.6343	32.06	34.56	0.8344	**1.170**
Team OV	**29.63**	0.0058	5.3748	32.17	34.72	0.8376	**1.510**
ALONG	**28.57**	0.2404	13.8019	32.18	34.66	0.8367	**1.660**
RTVSR	26.89	0.0532	12.2315	32.22	34.77	0.8372	1.960
Noah TerminalVision	26.68	17.797	16.1252	32.65	35.10	0.8455	3.640
NJUST-RTSR	23.51	0.0135	12.4748	32.25	34.90	0.8384	2.680
Antins cv	23.44	0.0127	11.6785	32.63	35.21	0.8457	4.600
DFCDN Team	22.64	0.0075	3.7011	32.07	34.63	0.8371	2.250
Multimedia	21.55	0.0125	11.4361	32.33	34.83	0.8398	3.560
z6	20.90	0.0457	41.9365	32.59	35.05	0.8446	5.470
R.I.P. ShopeeVideo	15.67	0.4073	129.2038	32.84	35.30	0.8469	13.79
ECNU SR	15.39	0.1662	37.8667	32.64	35.17	0.8458	10.75
Touch Fish	11.55	0.1465	134.7748	32.67	35.31	0.8468	19.86
P.AI.R	8.66	0.1280	104.362	32.55	35.04	0.8441	30.03
SEU CNII	6.68	0.0629	55.0807	31.85	34.52	0.8326	19.05
diSRupt	6.34	0.0649	120.0	31.64	34.25	0.8292	16.00
Our Baseline [92]	14.01	0.0575	219.77	31.74	34.37	0.8299	3.740

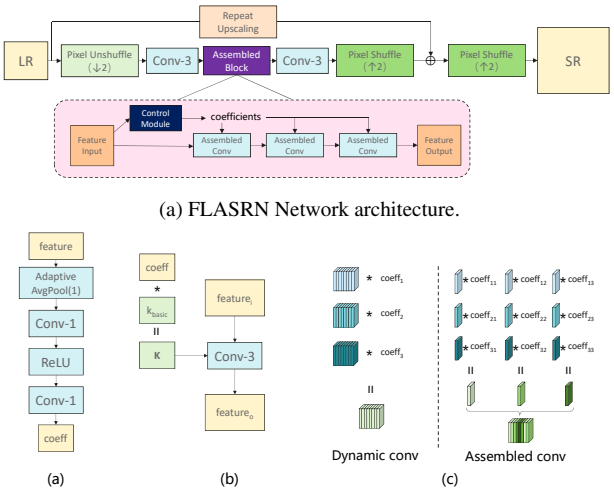
3. Methods and Teams

3.1. AsConvSR

The winning team in Track 1, **Noah_TerminalVision**, proposes a fast and lightweight super-resolution network (AsConvSR) with assembled convolutions [34]. The key points and contributions of the proposed network (see Fig. 2a) are as follows: (i) Pixel unshuffled [41] is used to reduce the resolution of the image and increase the channel dimension. Such design can reduce the computational cost of the network while keep the information volume unchanged. (ii) They remove all residual connections and keep a global skip connection, which repeats each pixel value $4\times$ (or $9\times$ for $\times 3$ SR) [26]. (iii) The authors propose an *assembled convolution* structure Fig. 2b. Different from the dynamic convolution [14] which generates the whole convolution kernel in a linear combination of the basis, assembled convolution generate the optimal kernel coefficient for each output channel, which is more flexible and outperform the dynamic convolution in this task.

Network architecture. Given an input LR image, the resolution would be converted to channel dimension by pixel unshuffle layer. By using a 3x3 convolution, the channel of feature map would be converted to the target size (32 for $\times 2$, 64 for $\times 3$) and then feed into the assembled block. The assembled block contains a control module and three assembled convolutions. As shown in the Fig. 2b, the control module is mainly responsible for generating coefficients for the assembled convolutions. Base on these coefficients, a 3x3 convolution is generated to perform a classical convolution on the feature maps. Therefore, the major computational cost of the assembled convolution is still the 3x3 convolution itself, and runtime of a assembled convolution is only a little higher than the classical convolution. After the assembled block, a 3x3 convolution layer is used to convert the channels size to 48 (108 for $\times 3$ SR) so that the feature map can be restored to target resolution after the pixel shuffle layer. It should be noted that a low resolution images repeated in the channel dimension can also be restored in to the high resolution with a pixel shuffle layer, we divide the final pixel shuffle into two steps in order to import the global skip connection to the network.

Assembled block. As shown in the Fig. 2b, given the input features $F \in R^{B,C,H,W}$, the control module converts the features F into coefficients $\texttt{coeff} \in R^{B,C_o,E}$, B is the batchsize, C_o is the number of output channels, and E is the number of candidate convolution basis. Matrix multiplication is performed on the coefficient $\texttt{coeff}$ and all candidate convolution kernels $k_{basis} \in R^{E,C_i,ks,ks}$ — where C_i is the number of input channels, and ks is the kernel size — to generate a final convolution kernel $K \in R^{B,C_o,C_i,ks,ks}$.



(a) FLASRN Network architecture.

(b) a) Control module. b) Assembled convolution. c) Comparison between dynamic and Assembled convolution.

Figure 2. *Team Noah_TerminalVision* solution.

Because different batches of data require different convolution kernels, the batch dimension of the feature map is reshaped to the channel dimension and the group convolution is used to calculate the output feature maps. As shown in Fig. 2b, dynamic convolution generates the whole convolution kernel (all channels) in a linear combination of the basis. Assembled convolution generate an optimal convolution kernel coefficient for each channel, which is more flexible and outperform the dynamic convolution in this task.

Implementation Details. In the training phase, the training sets include DF2k [2, 75], DIV8K [33], GTAV‘ [68], and LIU4K-V2 [59]. The network is trained by minimizing the charbonnier loss with Adam optimizer. The initial learning rate is 5e-4 and halved at every 2e5 iteration. The total number of training iteration is 3e6 on a Tesla V100 platform.

3.2. Bicubic++

The winning team in Track 2, **Aselsan Research**, proposes a lightweight, single image super-resolution method, named Bicubic++ [8]. Unlike many others lightweight methods where the input image dimensions are fixed throughout the network, Bicubic++ downscales the image first (by half with strided convolutions) to reduce the number of operations greatly on the following network convolutional layers to meet the real-time requirements. Finally they apply $\times 6$ upscaling. The overall structure is given in Fig. 3.

In addition, they follow a three stage training approach, where they train a slightly larger model first, and perform global structured convolutional layers and bias pruning without using heuristic metrics like weight norms on the following two stages. This approach ultimately yields a much faster, real-time model with none to marginal de-

crease in the visual quality. They have not employed quantization or the reparametrization of the convolutional kernels.

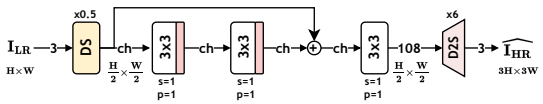


Figure 3. Bicubic++ structure proposed by *Aselsan Research*. The s and p denote stride and padding, respectively. In the final proposed model, ch is 32, all bias terms are removed, and a strided convolution with s=2, p=1 for the downscaling (DS) layer is utilized. Red blocks after 3x3 convolutions are leaky ReLU activations. D2S denotes depth-to-space layer [70].

Implementation Details. The models are trained in PyTorch Lightning. The training is done with mixed precision (FP16) by setting a precision flag in the *Trainer*, and Adam optimizer with $\beta_{1,2}$ parameters 0.99 and 0.999, respectively.

For the first two stages of the training, they start with the learning rate of 5e-4. For the last stage, they start with 1e-4. They utilize a decaying learning rate scheduler for all stages, where after 500 epochs the learning rate decays linarly until we reach to 1e-8.

For all three stages of the training, they train for 1000 epochs using batch size 8. Each epoch consumes 800 randomly cropped and rotated patches of dimension (108,108,3) -for LR- from Q=90 degraded DIV2K [1] dataset. For the validation, they use 48 images with same dimensions (680,452,3) -for LR- from Q=90 degraded DIV2K validation dataset.

3.3. RUNet

Team **ALONG** proposes RUNet: Re-parameterization and Unshuffle Network for Real-time Super- Resolution.

The team mainly considers designing the network following two aspects: (i) Receptive field: the model's ability may be limited if its receptive field is too small. (ii) Computational efficiency: The relationship between runtime and computation is not necessarily positive. A higher level of computational efficiency can result in a shorter runtime.

As shown in 4a, inspiring by [83], initially, they apply the pixel-unshuffle technique, which serves as the inverse process of pixelshuffle [70], to reduce the spatial dimensions and amplify the channel dimensions of the data before feeding them into the main model architecture. Thus, the majority of calculations are performed within a smaller resolution space, leading to a reduction in computational resource consumption and an effective enhancement of the inference speed. Furthermore, this approach can improve the receptive field. Next, a convolutional layer followed by an activation function is applied. This process effectively extracts low-level features from the input image. The body module is composed of a sequence of Re-Parameter blocks (RepBlock) that serves to extract and refine features in a progressive manner. Following the new suggestions in low-level vision task introduced by [53, 58], the Gaussian Error Linear Unit (GeLU) activation function is utilized in the $\times 2$ model, while the Sigmoid Linear Unit (SiLU) activation function is used in the $\times 3$ model, respectively. Finally, the upsampling layer and a skip connection are utilized to increase the image resolution to the desired level. This is achieved by applying a convolutional layer, followed by a pixel-shuffle layer.

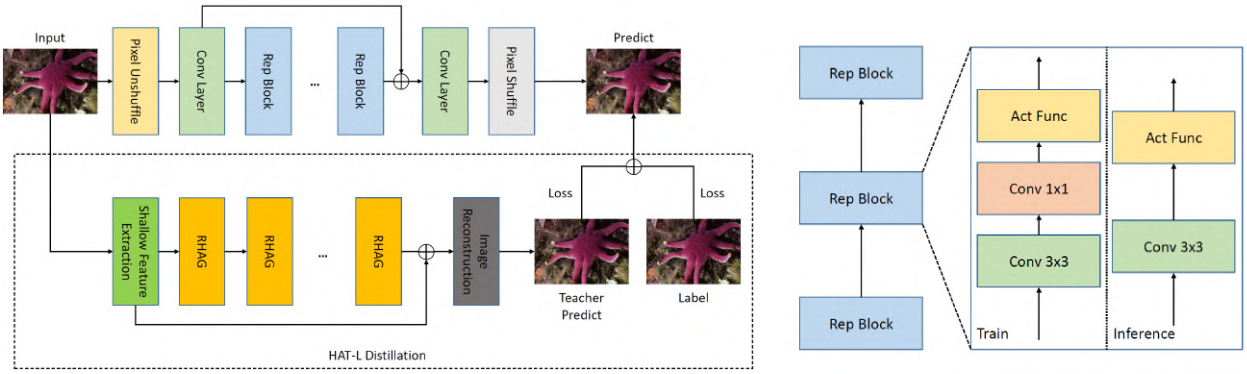
Besides re-parameterization [22], they also use Knowledge Distillation [36] in training. During the training stage, teacher output images and ground-truth images are used to guide the student network via teacher surpervision (TS) and data surpervision (DS), respectively. They use HAT-L model [13] as the teacher model, which is currently considered the SOTA model in the field of super-resolution.

Implementation Details. The method is implemented using Pytorch 1.13. The loss function is $\mathcal{L}1$ for reconstruction and $\mathcal{L}2$ is employed during the fine-tuning and knowledge distillation phases. For the X2 model, the channel employed in the CNN model (student) is 32, and the number of Rep Blocks is 3. Additionally, the scale of pixel unshuffle and pixel shuffle layers is 3. For the X3 model, the channel employed in the CNN model (student) is 64, and the number of Rep Blocks is 5. Additionally, the scale of pixel unshuffle and pixel shuffle layers is 4.

3.4. Team OV

Team OV presents a simple and efficient Convolutional Neural Network architecture that incorporates 3×3 convolutions, GELU activation function, and depth-to-space operations. The network utilizes 12 (for $\times2$) and 16 (for $\times3$) channels and produces the final image output through the depth-to-space operation. These architectural elements are depicted in Figure 5. The team also uses re-parameterization as shown in Fig. 5 (b).

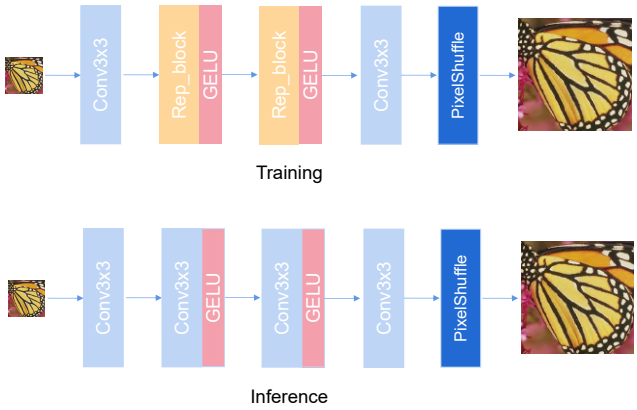
Implementation Details. The network was trained using DF2K (DIV2K+Flickr2K) dataset [2, 75], divided into three stages. Initially, low-resolution (LR) patches having a dimension of 128×128 are randomly cropped from high-resolution (HR) images with a mini-batch size of 64. L1 and FFT losses are used as target loss functions. Following this, network parameters were optimized for $300K$ iterations employing the Adam algorithm, with a learning rate of 1×10^{-3} decreasing to 1×10^{-7} through the cosine scheduler. In the second stage, the model obtained from the first stage was trained similarly for another $300K$ iterations. In the final stage, the model was fine-tuned using L2 loss and



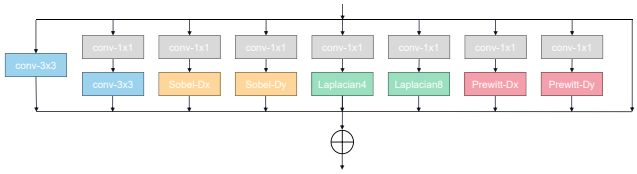
(a) RUNet Architecture.

(b) Reparameter Block.

Figure 4. *Team ALONG*. Overview of the proposed RUNet.



(a) The model architecture proposed by Team OV.



(b) The Rep_block proposed by Team OV.

Figure 5. *Team OV*. Overview of the proposed solution.

FFT loss. Network parameters are optimized for 300k iterations through the Adam algorithm, with a learning rate of 5e–4 reduced to 1e–7 using the cosine scheduler.

3.5. Repnet

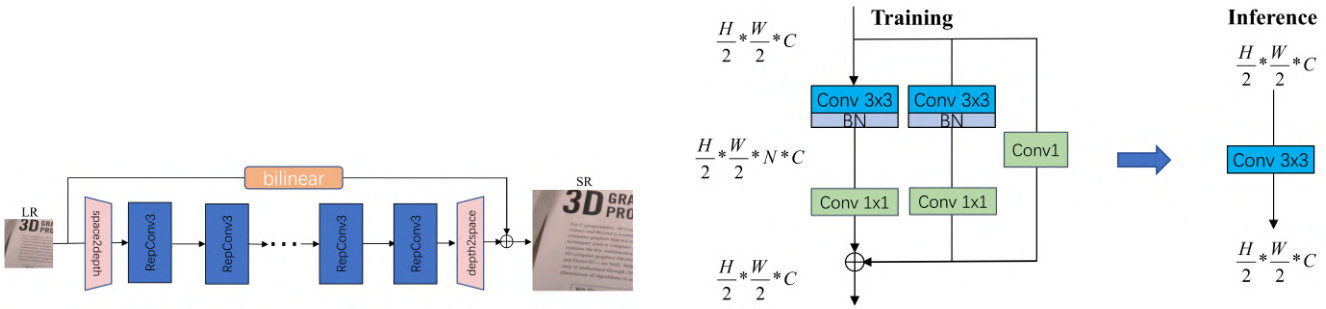
The team **RTVSR** proposes Repnet for Real-Time Super-Resolution. To reduce spatial dimension of the CNN, they first use paired space2depth and depth2space for single image super resolution. Furthermore, they also re-parameterize (conv3-bn-conv1) into a normal 3x3 convolution during inference, effectively improving the performance of the model without increasing the computational complexity of the model. For the $\times 2$ and $\times 3$ tracks, based on time consuming considerations, the model body uses three repconvs and four repconvs, respectively. The network is illustrated in Fig. 6a.

Implementation Details. Their training framework uses Pytorch for training on the A100 GPU. DIV2K, Flicker2K, DIV8K, GTAV datasets are used for training. The model training can be divided into two stages. In the first stage, the reconfigurable parameterized network structure shown in Fig. 6b is used for training. It is trained for 150 epoch using batchsize 32, the patch size is 256x256, and the learning rate is 2e-4. Adam optimizer is used. In the second stage, they use $\mathcal{L}_2$ loss to finetune the model obtained in the previous stage. The batchsize is 16, the patch size is 256x256, he learning rate is 1e-5, and the training time is 50 epochs. After the training (and during inference) they re parameterize the model into a network structure with conventional 3x3 convolution, as shown in Fig. 6b.

3.6. DFCDN

Team DFCDN proposes a novel network for efficient image super-resolution with deep feature complement and distillation network (DFCDN). They use online convolutional re-parameterization to reduce the large extra training cost introduced by re-parameterization.

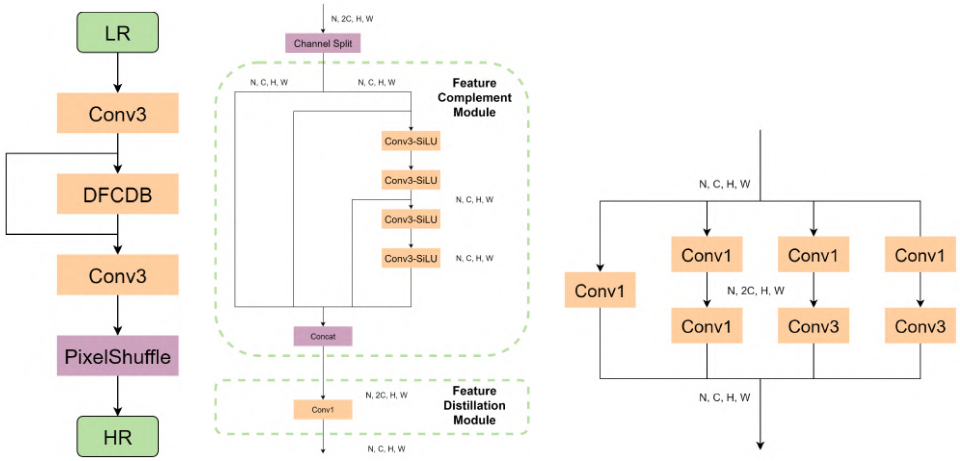
Network Architecture. The overall architecture of Team DFCDN is shown in Fig. 7. The proposed network consists only one deep feature complement and distillation block (DFCDB). Inspired by [35, 67], the input feature map is split equally along the channel dimension in the block. Then several convolutional layers process one of the split feature maps to generate complement features. The input features and complementary features are concatenated to avoid loss of input information and distilled by a conv-1 layer. Besides, the output feature map of DFCDB is further



(a) The architecture of Repconv-based Plain Net for RTSR.

(b) The architecture of reparameterized convolution module.

Figure 6. *Team RTVSR*. Overview of the proposed Repnet.



(a) DFCDN (b) DFCDB (c) RepConv

Figure 7. *Team DFCDN:* The overall architecture of the proposed DFCDN network.

enhanced by efficient spatial attention layer [63].

Online Convolutional Re-parameterization Re-parameterization [96] has improved the performance of image restoration models without introducing any inference cost. However, the training cost is large because of complicated training-time blocks. To reduce the extra training cost, they apply online convolutional re-parameterization [37] by converting the complex convblocks into one single convolutional layer. The architecture of RepConv is shown in Fig. 7 (c). It can be converted to a 3×3 convolution during training, which saves great training cost.

Implementation Details. The number of features is set to 8 and the number of attention channels is set to 16. The DIV2K [1] dataset is used for training and the inputs are in the range of 0-255. First, for training the $\times 2$ (Track 1) models, the setup is as follows: The model is first trained from scratch with 256×256 patches randomly cropped from HR images from DIV2K. The mini-batch size is set to 64. The L1 loss is minimized with Adam optimizer. The initial learning rate is set to 5e-4 with a cosine annealing schedule. The total number of epochs is 1000. At the second stage, the model is initialized with the pre-trained weights of Stage 1. The HR patch size is set to 640. The model is trained with the same settings as in the previous step. At the third stage, the model is initialized with the pre-trained weights of Stage 2. The MSE loss is used for fine-tuning with 640×640 HR patches and a learning rate of 1e-5 for 100 epochs.

The training details of $\times 3$ (Track 2) are as follows: At the first stage, the model is initialized with the pre-trained weights of the model with scale 2. The HR patch size is set to 660. The model is trained with the same settings as X2. At the second stage, the model is initialized with the pre-trained weights of Stage 1. The MSE loss is used for fine-tuning with 660×660 HR patches and a learning rate of 1e-5 for 100 epochs.

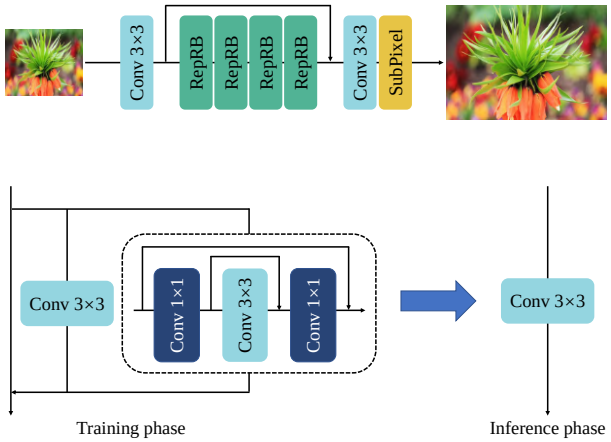
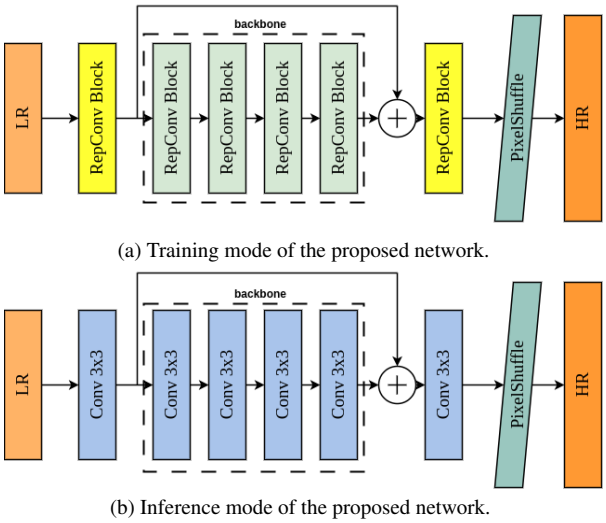


Figure 8. *Team NJUST-RTSR*: The overall architecture of the proposed network. (Bottom) Detail network of the proposed RepRB.



(a) Training mode of the proposed network.

(b) Inference mode of the proposed network.

Figure 9. *Team z6* proposed LRSRN network structure.

3.7. NJUST-RTSR

The team proposes a method that first transforms the input LR image into the feature space using a convolutional layer, then performs feature extraction using four reparameterizable residual blocks (RepRBs), and finally reconstructs the final output by a sub-pixel [70] convolution. The proposed architecture is illustrated in Fig. 8.

To enhance the capability of the model, they use the reparametrization technique [23]. Fig. 8 (Bottom) shows the detail description of the used RepRB module. It contains three branches in the training phase to learn features from different receptive fields, while in the inference phase it can be merged into a 3×3 convolution.

Implementation Details. The team uses DIV2K [2] and Flickr2K [75] as the training data. In order to accelerate the IO speed during training, they crop the 2K resolution images to sub-images — the HR image is cropped into 640×640 and 960×960 sub-images for $\times 2$ and $\times 3$ SR, respectively.

During the training, the data argumentation is performed on the input patches with random horizontal flips and rotations. The HR image patch size is initialized as 128×128 and increases to 256×256, and batch size is set as 64. They use the Adam [46] optimizer with the Cosine Annealing scheme [64]. The initial learning rate to 1×10^{-3} and the minimum one to 1×10^{-6}. The number of total iterations is set to 300k. They use a combination of mean absolute error (MAE) loss and an FFT-based frequency loss function to constrain the model training, which is the same as [73]. All experiments are conducted with the PyTorch framework on an NVIDIA GeForce RTX 3090 GPU.

3.8. LRSRN

Team z6 proposes a Lightweight Real-Time Image Super-Resolution Network (LRSRN) [30] that can deliver higher accuracy at a faster speed compared to previous real-time SR models for 4K images. They apply a reparameterized convolution (RepConv) for all convolution layers to improve the image quality while maintaining the model size and inference speed. The proposed network is an extended version of [29] (previous work of the team), which was designed for Mobile devices. The proposed network is illustrated in Fig. 9.

Implementation Details. The team used Pytorch 1.13. The models were trained in two steps: (i) First, models were trained from scratch. The LR patches were cropped from HR images with mini-batch size 8, and resolution 192 x 192 (Track 1) and 128

×

128 (Track 2). The Adam optimizer was used with a 0.0005 learning rate, and cosine warm-up scheduler. The total number of epochs was set to 800. They use $\mathcal{L}_1$ loss. (ii) In the second step, the model was initialized from previous step. Fine-tuning with $\mathcal{L}_2$ loss improves the PSNR value by 0.01 $\sim$ 0.02 dB. In this step, the initial learning rate was set as 0.0001. The total epoch was set to 200. In particular, the DIV2K [1] was used for scratch training. The combined dataset, which includes DIV2K train set (800 images), Flickr2K (2650 images), GTA (train seq 00 $\sim$ 19), LSDIR [52] (first 1000 images) used for the fine-tuning stage. The training data is preprocessed by center cropping it to a resolution of 2040 x 1080. To generate low-resolution, they degrade the center cropped images

with bicubic downsampling and JPEG compression. During training, they used random cropping, rotations, and flips augmentations.

3.9. SCSYENet

Team Multimedia proposes SCSYENet: A Compact Skip-Concatenated Simple Yet Effective Real-Time Image Super-Resolution based on element-wise multiplication fusion operation and Re-parameter convolution.

They built an end-to-end RTSR network based on element-wise multiplication fusion operation and re-parameter convolution, following previous work [5, 43, 97]. SCSYENet, has only 10/12.5K parameters (in Track 1 (X2) and Track 2 repectively). The network consists of two asymmetrical branches with simple building blocks. To effectively connect the results by asymmetrical branches, a element-wise multiplication fusion operation is proposed. The architecture of SCSYENet is illustrated in Fig. 10a.

Network Structure Inspired by ECBSR [97], SCSYENet employs the re-parameterization technique to boost the SR performance while maintaining high efficiency. The model consists of six ECBs (see Fig. 10b), one PReLu, two fusion blocks and one skip connection (concatenation of input image after preprocessing and intermediate feature map). The number of channels in the network is set to 16. The pixelshuffle is used to produce the final image output. Typically, in the previous multi-branch networks, the fusion of outputs by different branches could be done by concatenation [5, 74] or element-wise addition followed by activation function [21, 31]. In this study, in order to effectively improve the representational power, a element-wise multiplication fusion operation [43], as in Fig. 10a, is employed for the fusion of the results by two branches, where $\otimes$ is the element-wise multiplication, and $\oplus$ is the element-wise addition. During inference, the ECB block can be reparameterized into one single 3×3 convolution.

Implementation Details. The team uses Pytorch 1.21.1, and the training device is the A100 GPU. During training, DIV2K [1] and Flickr2K [75] datasets are used for the whole process. The team follows a 3-stage training: First, the model is trained from scratch. HR patches of size 128 × 128 are randomly cropped from HR images, and the mini-batch size is set to 32. The SCSYENet model is trained by minimizing $\mathcal{L}_1$ loss function with Adam optimizer. The initial learning rate is set to 1×10^{-4} and decayed with cosine annealing scheduler at every 200 epochs. The total number of epochs is 1000. Second, the model is initialized with the pretrained weights, and trained with the same settings as in the previous step. This process repeats once. Third, training settings are the same as Stage 1, except that $\mathcal{L}_2$ loss is used for fine-tuning with 2040 $\times$ 1080 HR patches and an initial learning rate is 1×10^{-5}, the mini-batch size is set to 4.

3.10. ERLFN

Team **Team Antins CV** proposes a method built on Residual Local Feature Network (RLFN) [48]. Based on this network, we prune the architecture and introduce the Enhanced Residual Block (ERB) RepBlock proposed by [51] the runner up solution, and we propose our Enhanced Residual Local Feature Network (ERLFN).

Network Structure. The RLFN proposed by [48] is an efficient network for lightweight super resolution task. While for this real-time super-resolution task, they further prune the network for an ideal speed.

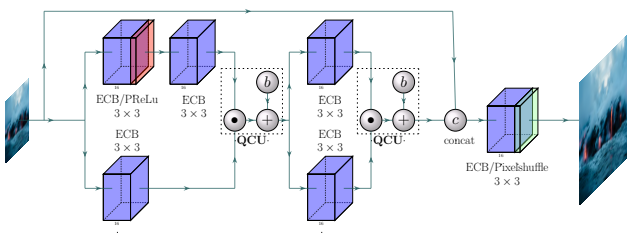
For Track 1 (upscaling from FHD 1080p to 4K) the network requires heavy computation. To balance for speed, we cut the four RLFB blocks in RLFN to two blocks, and shrink the feature channels to 12. The ESA blocks nested in RLFB are removed to reduce computation cost and save time. For Track 2, to upscale from HD 720p to 4K resolution, we cut the four RLFB blocks in RLFN to two blocks, and shrink the feature channels to 27. The ESA blocks are kept and channels are remained as 16.

The team also uses the **ERB RepBlock** in the Enhanced Residual Block (ERB) first proposed by [51] the runner up solution. They replace the 3×3 convolutions in RLFB with the ERB RepBlock. The network and ERB block are shown in Fig. 11. For inference, the ERB RepBlock is reparameterized to a 3×3 convolution. THe team does not experience any performance drop after reparameterization.

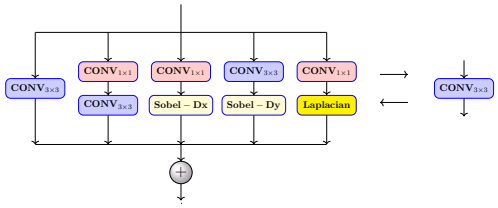
Implementation Details. The ERLFN model is trained for two stages both for Track 1 and Track 2. In the first stage, they train the model from scratch on DIV2K [1], cropped DIV8K, Flickr2K, OST, WED, first 2000 images of FFHQ, and first 1000 images of SCUT-CTW1500 datasets — following [56]. The HR images are randomly cropped to patches of size 256 $\times$ 256 for Track 1, and 192 $\times$ 192 for Track 2. They use Adam optimizer with $\mathcal{L}_1$ loss for this stage. We set the initial learning rate to $5e-4$, with a mini batch size of 64, and train the model for 1000 epochs, and decay the learning rate by 0.5 every 200 epochs. In the second stage, the model is initialized with the pretrained weights from the first stage on the same training data as stage 1. Then the model is finetuned using a cosine learning rate schedule with an initial learning rate of $1e-4$ for 500 epochs, using $\mathcal{L}_2$ loss is applied.

3.11. PCRTSR

Team ECNUSR proposes PCRTSR: Partial convolution based Network for Real-Time Super Resolution. The overall architecture is shown in Fig. 12. The network first



(a) Detailed architecture of SCSYENet.



(b) ECB: In the training stage, the Block employs multiple branches, which can be merged into one normal convolution layer in the inference stage

Figure 10. *Team Multimedia.* Overview of the proposed SCSYENet.

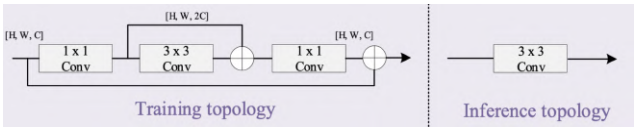


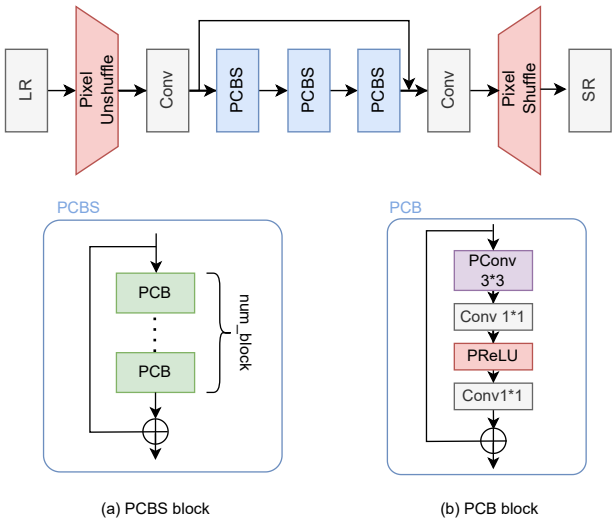
Figure 11. *Team Antins CV* proposed ERLFN network.

involves the pixel unshuffle for faster speed and a larger reception field. Then, the several stacked PCBS Block (Fig. 12 (a)) build up for feature extraction where each PCBS block is composed of several PCB blocks (Fig. 12 (b)) and a residual connection. Finally, the reconstruction module consisting of a 3×3 vanilla convolution and a pixel shuffle operation produces the SR image.

Network Structure. The team designs the models using partial convolution for accelerating the running speed, they do not use pruning or re-parameterization.

PCB Block The high latency of most efficient networks is due to the frequent memory access of the operators, to address this, a PCB block is proposed which consists of partial convolution. Our partial Convolution applies filters on 1/4 of the channels, resulting in lower FLOPs than the vanilla convolution and higher FLOPS than the group convolution. Each PCB block comprises a partial convolution followed by two pointwise convolution layers, with a PReLU activation layer after the middle layer. During feature extraction, there are 3 PCBS blocks which consist of 2, 4 and 2 PCB blocks respectively. The kernel size of the partial convolution and vanilla convolution is 3 × 3. The architecture is symmetrically designed and highly optimized, resulting in lower inference latency.

Implementation Details. The team first trained the models on the DF2K (combined DIV2K and Flickr2K) dataset [75], and then finetuned on the combined datasets consisting of DIV8K, FFHQ, LSDIR [52], and GTA V for the variety of the data. The patches are cropped with the size 256 × 256 and augmented by random flipping and rotation. The model is trained by Adam [46] optimizer with



(a) PCBS block (b) PCB block

Figure 12. *Team ECNUSR* architecture for Partial Convolution based Network for Real-Time Super Resolution (PCRTSR). (a) PCBS block and (b) PCB block.

$\beta_1 = 0.9, \beta_2 = 0.999$. The initial learning rate is set to 5×10^{-4} and decreases by half at 8×10^6 and 1.4×10^7 iterations. $\mathcal{L}_1$ loss is used for training. The model is implemented by PyTorch 1.12 using one 2080Ti GPU.

3.12. R2CNet

Team R.I.P. ShopeeVideo proposes R2CNet using efficient Bottle-in-Bottle blocks for RTSR. As shown in c13, they propose a hardware-efficient *R2C block* with well-designed channel numbers. In the R2C block, they stack efficient 3×3 convolutions [22] inside with small channel numbers, while keeping the channel numbers large outside to improve performance. In R2CNet, a novel downsample-upsample mechanism is also utilized to process images of large size (4K). Neither pruning nor re-parametrization is not used in R2CNet.

Network Structure. The proposed R2C block is illustrated in Fig. 13 (a); an input 1×1 convolution reduces the

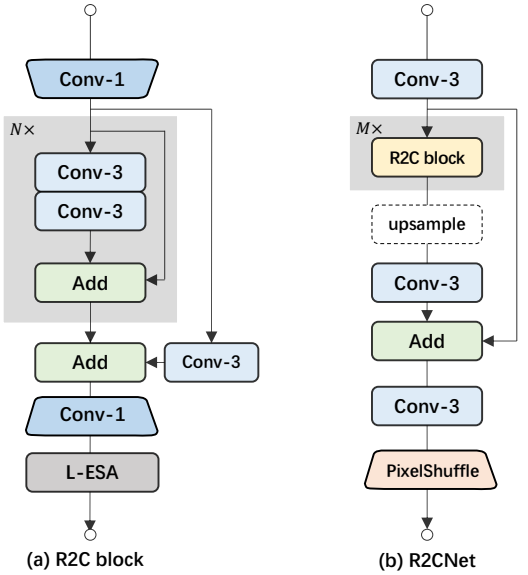


Figure 13. *Team R.I.P. ShopeeVideo* proposed R2C block and R2CNet. (a) **R2C block** uses *L-ESA:*, an improved ESA [60], also Batch Normalization (BN) is applied for each convolution layer to accelerate convergence. (b) **R2CNet:** the macro structure is based on RLFN [47] and M R2C blocks are used.

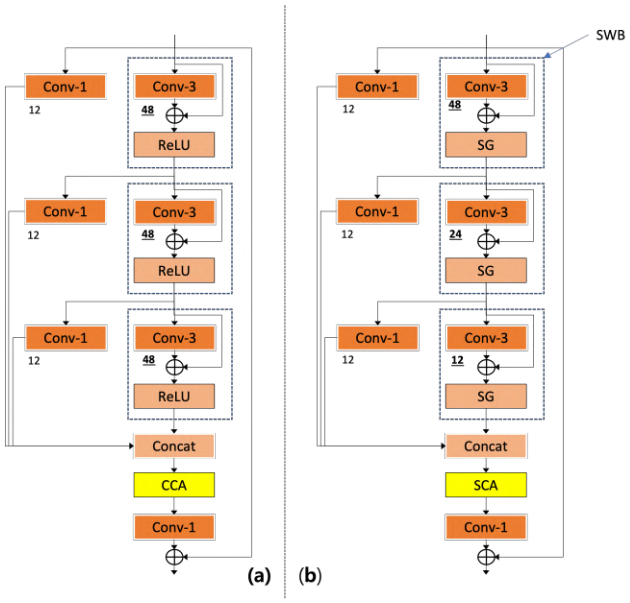


Figure 14. *Team P.AI.R* proposed FADN. Comparison of (a) residual feature distillation block and (b) no attention distillation block.

channel numbers, and the output one to increase. Thus, the channel numbers inside the block is small, making it efficient to stack efficient 3×3 convolutions inside [10, 22], $i.e.$, N basic blocks, and a skip-path 3×3 convolution. The team also proposes L-ESA for efficient and effective spatial attention, in which they simply reset the kernel size and stride of the pooling layer in ESA [60] from 7 and 3 to 11 and 7. Large kernel captures more spatial information and large stride reduces computation and runtime [16]. With R2C block, we build our R2CNet following the macro structure of RLFN [47], as shown in Fig. 13 (b).

To process images of large size (4K) efficiently, they also introduce a new downsample-upsample mechanism into the R2CNet: simply set the stride of the first R2C block as 2 for downsampling and utilize a pixel shuffle layer with factor 2 for upsampling. Specially, in both R2CNet×2 and R2CNet×3, we set N = 4, M = 2, the channel number of the main body as 64 and that inside R2C block is 32.

Implementation Details. The team uses PyTorch for training and inference. They train the models for there stages. Each stage has 100k iterations. The learning rate is set as 5e-4 for the first two stages with first 5k iterations as warm-up, while 2e-4 for the last one without warm-up, and we uses cosine annealing. PSNR loss [12] is utilized. Adam is the optimizer and weight decay is not applied. The global batch size is set to 96 on 3 GPUs. The sizes of HR images during training for R2CNet×3 and R2CNet×2 are 576 and 512, respectively. Before inference, the BN layers in R2C blocks are fused into their corresponding convolution layers for fast inference. The team uses DIV2K [2], Flickr2K, and half LSDIR [52] datasets for training.

3.13. FADN

Team P.AI.R proposes FADN: Few Activation Distillation Networks for Real-time Super-resolution. The solution is mainly based on RFDN [60]. The architecture of the proposed method differs from the RFDN in two ways: 1) the simple gate (SG) introduced in NAFNet [11], which is an element-wise product of feature maps divided into two parts in the channel dimension, was used instead of ReLU in a shallow residual block (SWB). 2) Simplified channel attention (SCA), also introduced in [11], was used instead of contrast-aware channel attention (CCA). The team adopted the SG and SCA to simplify the network, as the SG halves the number of channels, and the SCA is the simplified version of channel attention. In addition, layer normalization was also adopted in the network to ensure a more stable training process. The FADN (see Fig. 14) consists of four no-activation distillation blocks (NADB).

Technical details. The team train the models with ADAM optimizer by setting beta1=0.9, beta2=0.999, and eta=10^{-8}. The learning rate is initialized as 2e-4 and halved at every 100 epochs. The team used LSDIR [52] datasets to train the models, and generated the training LR images by downsampling HR images with bicubic interpolation and JPEG

compression. The model is implemented using the PyTorch framework with an RTX3090 GPU. The number of feature channels was 16 for $\times 2$ SR and 40 for $\times 3$ SR. So then, the number of parameters is 0.0121 M and 0.1280 M respectively.

3.14. Team PixelBE

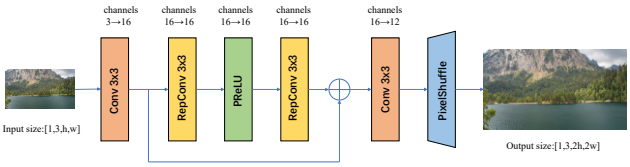
The team proposes: Two-Stage Super-resolution Algorithm Based on Re-Parameterization. They use as reference [98], a re-parameterizable building block, namely Edge-oriented Convolution Block (ECB), for an efficient convolutional module design. This module uses multiple parallel convolution operators in the training phase to improve the SR capability of the model, and fuses the parallel operators into a convolution module in the testing phase to improve inference efficiency. Based on this ECB module [98], they designed a two-stage SR algorithm as follows: (i) First, they downsample by a factor of 2 using a convolution with a stride of 2. Downsampling breaks down jpeg compression and also improves network inference speed. (ii) Then stack two ECB modules and a $\times 2$ upsampling pixel shuffle module to return a three-channel image. (iii) Finally, two ECB modules and a $\times 2$ upsampling pixel shuffle module are used to return a HR image.

Implementation details. The team uses the LSDIR Dataset [52] for training, and the training data is degraded online (*i.e*. downsampling, JPEG compression). The input image size is 128x128x3, the optimizer is Adam. The training is divided into two stages: First, the learning rate is 1e-3 and the jpeg loss and super-resolution loss are calculated at the same time. This stage is trained for 100k iterations. Second, only the super resolution loss ($\mathcal{L}_1$) is calculated, and the learning rate is halved — this is for 150k iterations.

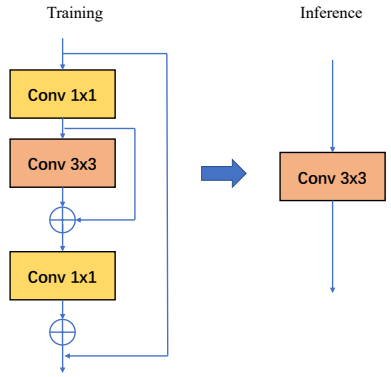
3.15. OELSR

Team AGSR proposes an optimized extreme lightweight super-resolution network (OELSR). The Extreme Low-Power Super Resolution Network in [87] is their baseline. The network (see Fig. 15a) stacks multiple highly optimized convolution+activation layers to achieve a good trade-off between the enhanced quality and model complexity. The team uses the re-parameterizable blocks [25] and replace them to a single convolution to reduce the inference time. Besides, they use a multi-stage training where in each stage, the weights from previous stages are utilized as warm-start to improve the model performance progressively.

Finally, the team obtains a simple yet effective network structure with single frame input (as shown in Fig. 15a) which only have 6 layers, of which only 5 have learnable parameters, including 4 Conv layers and a PReLU activation layer. Besides, they use re-parameterizable blocks to



(a) The overall architecture and the structure of the RepConv block.



(b) The Repblock module.

Figure 15. *Team AGSR*. Overview of the proposed OELSR.

improve the performance of the middle convolution. PixelShuffle operation is used at last to upscale the size of output without introducing more calculation.

Technical details. The team uses DIV2K [1] and Flickr2K as training dataset. In each training batch, 64 cropped LR RGB patches augmented by random flipping and rotation are input to the network. The input data range of the network is *0-255*. The model is trained using PyTorch, Adam [46] optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and they utilize Charbonnier loss (first stage) and $\mathcal{L}_2$ loss (second stage) function separately since they employ a multi-stage training approach.

3.16. Team DoYouChargeQQCoin

The team proposes a ultra fast network for image super-resolution. The network is illustrated in Fig. 16, it consists on 2-layer CNN with a ReLU activation for image SR. This represents the most compact and simple solutions in this challenge; it improves Bicubic upsampling by +0.2dB while running at $\approx$ 2ms.

They implement the network with PyTorch. The optimizer is Adam with learning rate as 10e-4, which is halved for every 200 epochs. The training dataset is DIV2K, using random flips and rotations. The input of the network is in the range *0-255*.

3.17. Team Touch Fish

The team proposes a new attention mechanism. The rationale behind utilizing an attention map with a consider-

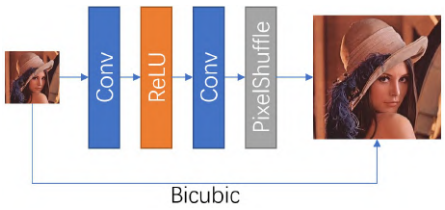


Figure 16. *Team DoYouChargeQQCoin* proposed network.

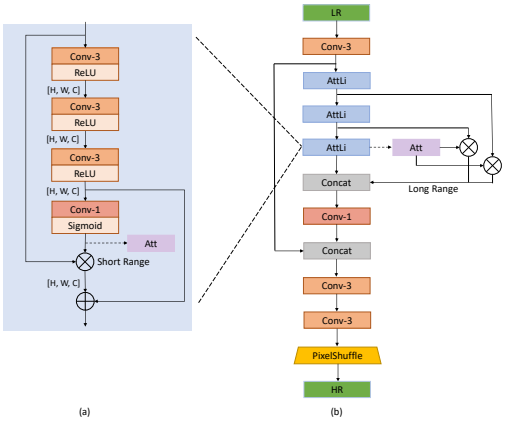


Figure 17. *Team Touch Fish* solution: (a) **Attli block**, pink denotes the generated attention map $\mathcal{M}$. (b) Pipeline $\times 2$ SR.

able perception field is that it can be advantageous for the preceding layers to concentrate their attention on regions of interest. They generate an attention map $\mathcal{M}(i,j)$ as:

$$\mathcal{M}(i,j) = \phi(\text{Conv}_{1\times1}(F_l(i,j))), \tag{3}$$

where $\phi(\cdot)$ denotes the sigmoid function. $F_l(i,j)$ and $F_f(i,j)$ denote the value of the feature map in the position (i,j) from the latter layer and former layers, respectively. Then we use the generated attention map to reweight the features in the former layers as $\mathcal{M}(i,j) \bigodot F_f(i,j)$, where $\bigodot$ denotes the Hadamard product.

As depicted in Fig. 17 (b), an attention map is generated for each block, which is subsequently utilized to reweight the feature maps originating from distinct levels.

They also use re-parameterization (rep) [22] to enhance the efficiency of the inference phase. This technique has been incorporated into each convolutional block depicted in Fig. 17. In contrast to prior techniques that employ stride convolutions, pooling, and upsampling, the team merely uses the generated mask. This modification has resulted in a significant acceleration of both inference and training times, as well as a reduction in the memory footprint.

Technical details. The number of channels is set to 24 (x2) and 32 (x3). The learning rate is 5×10^{-4} and undergoes a halving process every 2×10^5 iterations. The network

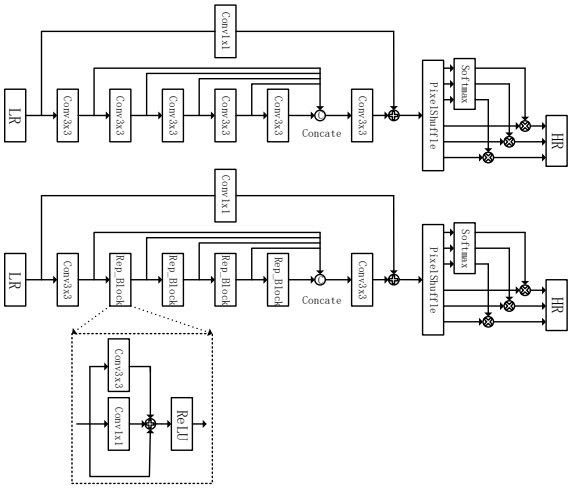


Figure 18. *Team DH ISP* proposed solution.

is trained for a total of 10^6 iterations, with the $\mathcal{L}_1$ loss, batch sizes of 64, and Adam optimizer [45]. Subsequently, fine-tuning is executed using the $\mathcal{L}_1$ and $\mathcal{L}_2$ loss functions, with an initial learning rate of 1×10^{-5} for 5×10^5 iterations, and HR patch size of 512. The dataset utilized for training comprises of DIV2K [1] and LSDIR [52].

3.18. Team DH ISP

The team designed a simple lightweight network for image super resolution. The model consists of two 3x3 convolution layers, one 1x1 convolution layer and four Re-Parameterizable blocks (RepBlock), the final output is obtained using the pixel shuffle. Re-parameterizable blocks can learn features at different scales during the training phase, then, in during inference, they can be converted into a 3x3 convolutions to accelerate the inference speed. The network structure is shown in the Figure 18.

Two branches are used for feature extraction. (i) four re-parameterizable blocks and a 3x3 convolution, which is used to extract the deep features of the image. (ii) a 1x1 convolution is used to extract the shallow features of the input image. Finally, the features extracted from the two branches are added together for fusion, and the upsampled features are obtained through the pixelshuffle layer and the final output is obtained through the structure of self-attention.

Technical details. The training data set includes Flickr2K and DIV2K [1]. The training of the model is divided into two stages: (i) the network is trained from scratch. The input image size is 256 × 256, the batch size is 16, the loss function is $\mathcal{L}_1$, Adam optimizer with the initial learning rate set to 0.001, the learning rate is halved every 200 epoch,

and a total of 800 batches of training. (ii) on the basis of the training in the first stage, the $\mathcal{L}_2$ loss was used to continue training for 200 epochs, with an initial learning rate of 0.0001, halved every 50 batches. Finally, the heavy parameter module in the network is re-parameterized by 3x3 convolution, and the trained model parameters are transformed to achieve faster inference.

3.19. PRFDN

Team SEU CNII proposes PRFDN: High Parallelism Distillation Network For Image Super-Resolution.

The proposed Parallel RFDN (PRFDN) is based on the pre-trained RFDN [60] as shown in Fig. 19a. The method disentangles the sequentially computed trunks in RFDN into branches (Fig. 19b) and performs re-parametrization to make these branches inference in parallel on single devices. After that, they further perform pruning on the model (Fig. 19d) and fine-tune it to achieve higher performance.

Network Structure. *Branching*. To accelerate the inference, authors first consider reducing the data dependency in the model to achieve higher parallelism. Thus, the method disentangles the sequentially computed trunks into branches. As shown in Fig. 19b, after the branching, the major part of the model will consist of four independent branches that can calculate in parallel. To improve the performance, authors also design small SR blocks (SRFDB) based on [60], and add them before the input of each branch.

Re-parameterization. Without much data dependency, branches in the model can be computed in parallel. As shown in Fig. 19c, the major part of these four branches (RFDBs and SRFDBs) have exactly the same structure but different parameters, so we can merge and re-parameterized the RFDBs and SRFDBs into a single branch.

Pruning. To further accelerate the inference, they apply channel pruning on the re-parameterized model, as shown in Fig. 19d, using Torch-Pruning [27], and fine-tune the model between each pruning step.

Technical details. The authors use Pytorch and Torch-Pruning [27]. The models are trained using Adam [46] with learning rate 1e-5 before re-parameterization, 1e-6 after re-parameterization. The training datasets are LSDIR [52] and DIV2K [1]. Since they only change the data flow, but not the structure of RFDB, the pre-trained RFDN parameters can still be loaded into the major part of our branch model (only except for those SRFDBs). To benefit from the pre-training, they load the pre-trained RFDN parameters into our branch model before training our branch model.

3.20. LFDN

Team NTU-BL6F adopts LFDN [47] model as the backbone. The authors reduce the number of channels and utilize pre-trained weights by selecting the necessary channels to match the compressed channel quantities. The authors also find that using channel quantities whose power is 2 can result in faster processing speed compared to other channel quantities. The model is illustrated in Fig. 20.

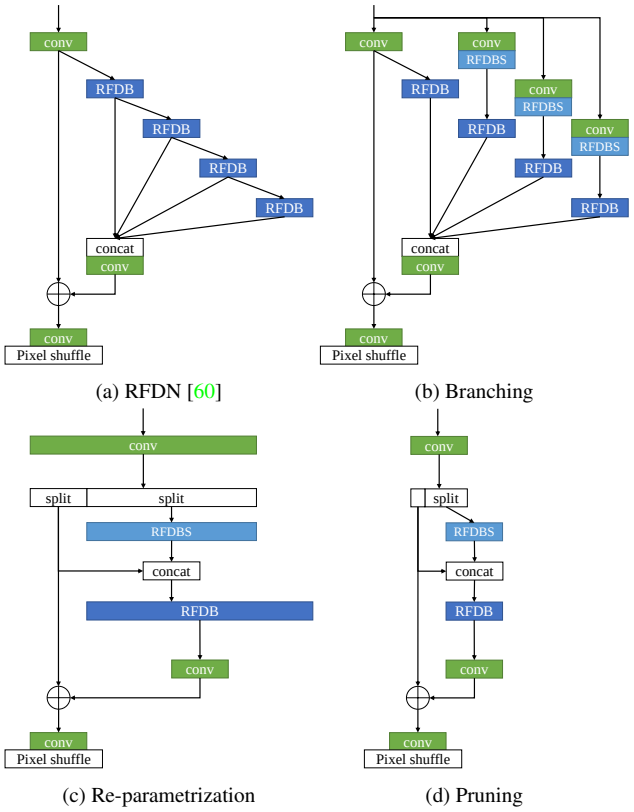


Figure 19. *Team SEU CNII* proposed PRFDN including: branching, re-parameterization, and pruning.

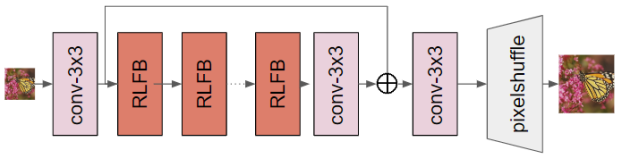


Figure 20. *Team NTU-BL6F* solution based on LFDN [47]. They adjust the channel number of RLFB and use mixed precision training to improve the model.

Technical details. The team uses LFDN [47] model pre-trained on the DIV2K dataset [1]. The network input range is from 0 to 255, and mixed precision was used for fine-tuning. The team uses the DIV2K [1], Flickr [76], OTS [50] and GTA [69] datasets to train the model. The authors adopt $\mathcal{L}_1$ loss to optimize the network. The optimizer is

Adam [46] with learning rate 5e-4. In the test phase, they feed the whole-size image to the model and the inference speed is approximately 18ms per image.

3.21. DRCNN

Team diSRupt proposes Depthwise-Residual Convolutional Neural Network (DRCNN).

DRCNN (see Fig. 21) extends the SCSRN architecture, which was introduced in [43]. On top of the existing architecture, DRCNN performs nearest-neighbors upsampling to provide the SCSRN stage with an upsampled baseline image. In order to maintain efficiency through GPU parallelism, a space-to-depth transformation is applied to the upscaled LR image, forcing the following convolutional layers to operate on feature maps having the same dimensions as the LR image. The same depthwise-upsampled LR image is added to the feature map generated through the SCSRN, forcing the network to learn the residual between the naive interpolation and the HR image, thus enhancing the convergence speed and the overall performance.

Implementation details. The authors use Tensorflow 2. The network was trained for 70 epochs on the entire Div2K training set [1], using the Adam [46] optimizer with a 3e-4 learning rate, a batch size of 16, a patch size of 128, classical augmentations, and optimizing for MSE. The model accepts RGB images of any resolution. No re-parameterization, pruning or quantization was applied.

3.22. ELIS

Team KCML2 proposes Enhanced Lightweight Image Super-resolution (ELIS), which is inspired by XLSR [5] with the addition of the advanced attention mechanism. The main idea is to use channel splitting to separate the feature maps and process them in parallel with attention. Besides this, the authors use a multi-stage warm-start training strategy. In each stage, the pre-trained weights from previous stages are utilized to improve the model performance. The network is illustrated in Fig. 22.

The authors add a spatial operation to the original block from XLSR [5] to enhance the performance as each pixel is considered differently at each pixel location. They design the ECSB block, which contains a channel splitting mechanism, convolution operation, and an enhanced spatial attention block (ESA) as shown on Fig. 22 (*bottom*).

Implementation details. The authors use DIV2K and Flickr2K [1] for training set, and randomly crop the images to the size of 512×512. All images are normalized to range $0-1$. During training, they randomly crop LR patches of size 256×256 and use horizontal flipping, vertical flipping along with random intensity scaling for augmentation. As the loss function, we employ the Charbornier loss with $\eta = 0.1$. The number of ECSB is set to 5 and the number of channels inside ECSB to 32. The model is trained using a multi-stages training strategy with cyclic learning rate scheduler, Adam optimizer [46] and batch size of 64 . The authors did not use any pruning or re-parameterization technique, only using channel splitting and attention.

3.23. Team NPU SuperResolution

The team proposes a model based on ECBSR [96] with some improvements. The authors found that the edge operator can not make a relatively large contribution to the performance improvement of the whole model, so they propose to replace the edge operator with wavelet transform. The experiment proves that the wavelet transform has a certain effect on the improvement of the model.

The authors also use ideas from MWCNN [61] and other models that use wavelet transform to achieve super-resolution. In their model, LL, HL, LH, and HH after wavelet transformation will be concatenated in the channel dimension, which can ensure that messages will not be lost, thereby further improving the performance of the model. They chose a very simple model with only one branch, so that the speed of the model can be guaranteed. In a block, they remove the branches that do not significantly improve the model effect, and only keep the branch that contributes the most. In addition, they also use re-parameterization as ECBSR [96], so that each block can be re-parameterized into one or two 3x3 convolutions, so that during the inference process of the model, have faster speed.

Technical details. The team uses Pytorch to implement the model. The optimizer used is Adam [46], the learning rate is 5e-4, and the GPU is A100. The training dataset combines DIV2K [1], Flicker2k, manga [65], and some pictures obtained on the internet – the authors find that the data set can significantly improve the performance of the model. The obtained model is re-parameterized.

3.24. Team YNOT

The team utilized an image processing method based on Fast Fourier Convolution (FFC) [15], which has different advantages from conventional convolution-based image processing (*i.e.* it can utilize both global and local information), and Wavelet Analysis [89] image processing techniques. By utilizing information at the frequency level, they aimed for better performance while lightening the baseline architecture of IMDN [40].

The authors found that FFC [15] can be used to replace traditional CNNs, but it may not be suitable for real-time super-resolution. However, by utilizing the information available in the spectral domain (*e.g.* Fourier Transform, Wavelet Transform), they were able to lighten the architecture of the IMDN [40] model used to satisfy some of the

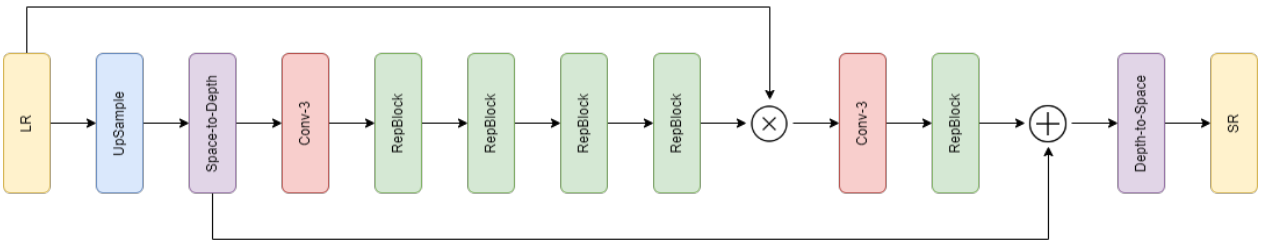


Figure 21. *Team diSRupt* proposed DRCNN.

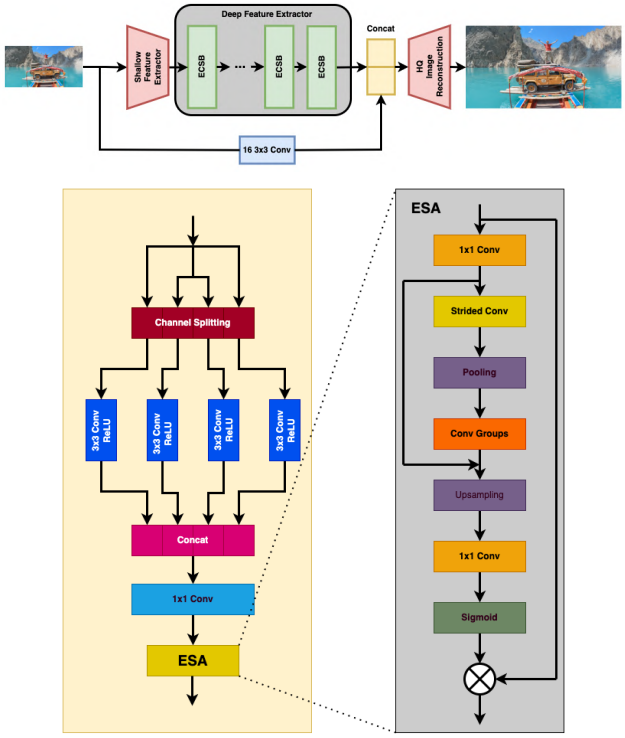


Figure 22. *Team KCML2* proposed enhanced lightweight image super-resolution network. (Bottom) Architecture of ECSB with ESA (Enhanced Spatial Attention) [51].

computational and performance tradeoffs.

Technical details. The authors use Pytorch 1.7.1 to develop the models. The models are trained for 500 epochs using $\mathcal{L}_1$ loss, Adam optimizer [46], a learning rate of 2e-4, and MultiStepLR with a gamma of 0.5. The team only uses DIV2K [1] for training the models.

4. Qualitative Results Comparison

We provide qualitative comparisons in Fig. 24, Fig. 25 and Fig. 26 between the top-3 proposed methods. All high-resolution images and the results from each top team, are available in our project website and github. All the top

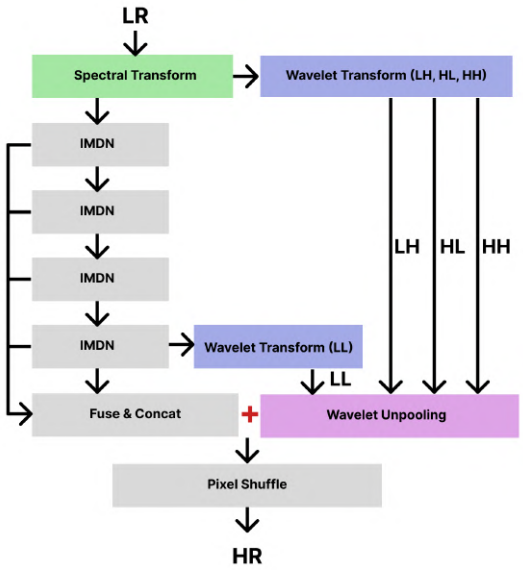


Figure 23. *Team YNOT* proposed solution.

methods can recover details from the LR 1080p and 720p, and produce high-quality 4K images.

5. Conclusion

This paper introduces a novel benchmark for efficient upscaling as part of the NTIRE 2023 Real-Time Image Super-Resolution (RTSR) Challenge, which aimed to upscale images from 720p and 1080p resolution to native 4K ($\times 2$ and $\times 3$ factors) in real-time on commercial GPUs. For this, we use a new test set containing diverse 4K images ranging from digital art to gaming and photography. We assessed the methods devised for 4K SR by measuring their runtime, parameters, and FLOPs, while ensuring a minimum PSNR fidelity over Bicubic interpolation. These methods allow processing at 60 FPS and even beyond. Out of the 170 participants, 25 teams contributed to this report, making it the most comprehensive benchmark to date and showcasing the latest advancements in real-time SR.

Figure 24. *Qualitative results.* Comparison of the best methods using the test sample 11. The image corresponds to a real capture using a 60MP camera. Complete HQ uncompressed results -for the top teams- can be consulted in our project website.

LR Input

AsConvSR [34]

Bicubic++ [8]

HR Ground-truth

RUNet

RT4KSR [92]

Figure 25. *Qualitative results.* Comparison of the best methods using the test sample 59, a real world capture using a SONY ILCE-7M3. Image credit: "Asakusa" by @mosdesign.

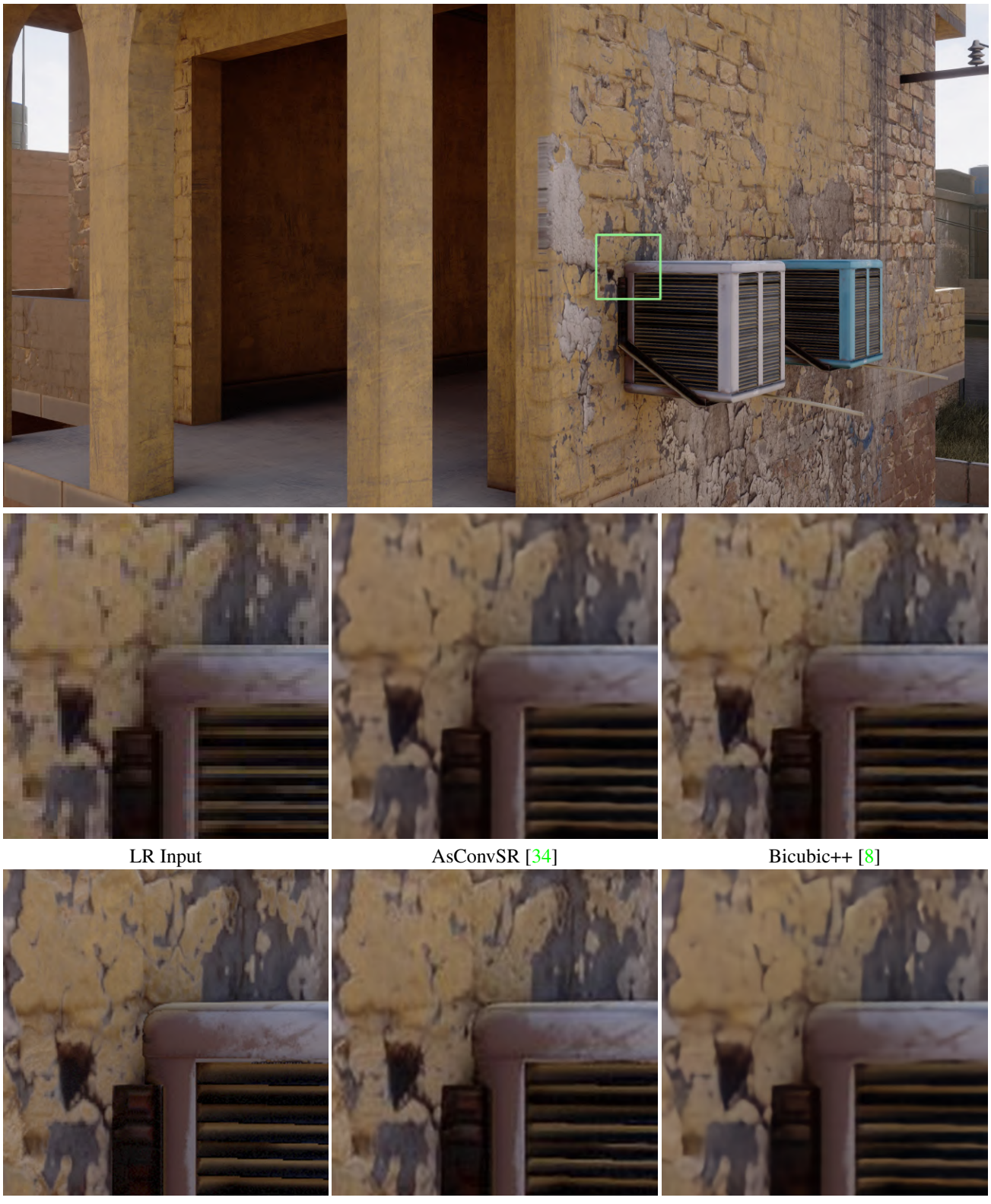


Figure 26. *Qualitative results.* Comparison of the best methods using the test sample 114, rendered content using Unreal Engine [38].

Table 2. We provide **Additional Training Details** to facilitate reproducibility of the solutions. The teams indicate the resolution of the input RGB image during training, the training time in hours, and the GPU device.

Method	Input	Training Time (h)	Attention	Quantization	# Params. (M)	GPU
AsConvSR $\times 2$	120×120	30	No	No	2.3	V100
AsConvSR $\times 3$	80×80	30	No	No	17	V100
RUNet $\times 2$	192×192	24	No	No	0.0668	RTX3090
RUNet $\times 3$	192×192	20	No	No	0.24	RTX3090
Team OV	128×128	21	No	No	0.005	RTX3090
Repnet $\times 2$	256×256	8	No	No	0.0266	A100
Repnet $\times 3$	256×256	12	No	No	0.0532	A100
Bicubic++ $\times 3$	108×108	3	No	No	0.0504	V100
DFCDN $\times 2$	320×320	44	Yes	No	0.0064	RTX3090
DFCDN $\times 3$	220×220	44	Yes	No	0.0075	RTX3090
NJUST-RTSR $\times 2$	256×256	16	No	No	0.014	RTX3090
LRSRN $\times 2$	192×192	48	No	No	0.0046	A6000
LRSRN $\times 3$	128×128	16	No	No	0.0046	A6000
SCSYENet $\times 2$	512×512	27	No	No	0.01	A100
SCSYENet $\times 3$	540×540	18	No	No	0.0125	A100
ERLFN $\times 2$	256×256	71	ESA	No	0.0111	V100x4
ERLFN $\times 3$	192×192	47	ESA	No	0.0666	V100x4
PCRTSR $\times 2$	256×256	30	No	No	0.162288	2080Ti
R2CNet $\times 2$	512×512	180	L-ESA	No	0.3987	V100
R2CNet $\times 3$	576×576	180	L-ESA	No	0.4073	V100
FADN $\times 2$	256×256	130	Yes	No	0.0212	RTX3090
PixelBE $\times 2$	128×128	96	No	No	0.137	V100
OELSR $\times 2$	512×512	8	No	No	0.0068	2080Ti
QQCoin $\times 2$	256×256	48	No	No	0.00082	RTX3090
Touch Fish $\times 2$	256×256	60	Yes	No	0.064	A100x8
Touch Fish $\times 3$	256×256	60	Yes	No	0.183	A100x8
dh ISP	256×256	5	Yes	No	0.01	2080Ti
PRFDN $\times 2$	678×1020	16	No	No	0.0299	RTX3070
PRFDN $\times 3$	512×680	16	No	No	0.0629	RTX3070
NTU-BL6F (LFDN) $\times 2$	256×256	12	Yes	Yes	0.22	RTX3090
DRCNN $\times 2$	128×128	5	No	No	0.0499	NVIDIA T4
DRCNN $\times 3$	128×128	3	No	No	0.0649	NVIDIA T4
ELIS	256×256	10	ESA	No	0.039	TITAN RTX
NPU-SR (ECBSR) $\times 2$	1080×1920	10	No	Yes	0.2	A100
YNOT $\times 2$	256×256	4	Yes	No	0.4648	A100

Acknowledgments This work was partly supported by the Humboldt Foundation and Sony Interactive Entertainment. We thank the NTIRE 2023 sponsors: Sony Interactive Entertainment, Meta Reality Labs, ModelScope, ETH Zürich (Computer Vision Lab) and University of Würzburg (Computer Vision Lab).

6. Appendix

6.1. NTIRE 2023 Team

Title: NTIRE 2023 Real-Time Super-Resolution Challenge Organization

Members: Marcos V. Conde [1], Eduard Zamfir [1], Radu Timofte [1], Daniel Motilla [2]

Affiliations: [1] Computer Vision Lab, CAIDAS, IFI, Uni-

versity of Würzburg, Germany
[2] Sony Interactive Entertainment, CA.

6.2. Noah_TerminalVision

Title: AsConvSR: Fast and Lightweight Super-Resolution Network with Assembled Convolutions
Members: Jiaming Guo, Xueyi Zou, Yuyi Chen, Yi Liu, Jia Hao, Youliang Yan
Affiliations: Huawei Technologies Co., Ltd.

6.3. Aselsan Research

Title: Bicubic++: Slim, Slimmer, Slimmest - Designing an Industry-Grade Super-Resolution Network
Members: Mustafa Ayazoglu, Bahri Batuhan Bilecen
Affiliations: Aselsan Research, Türkiye. https://www.aselsan.com/tr

6.4. ALONG

Title: RUNet: Re-parameterization and Unshuffle Network for Real-time Super-Resolution
Members: Cen Liu, Zexin Zhang, Yunbo Peng, Yue Lin
Affiliations: NetEase Games AI Lab

6.5. Team OV

Title: An Efficient ConvNet for Real-time Image Super-resolution
Members: Lingshun Kong, Haoran Bai, Jinshan Pan, Jiangxin Dong, Jinhui Tang
Affiliations: Nanjing University of Science and Technology

6.6. RTVSR

Title: Repnet for Real-Time Super-Resolution
Members: Yuanfan Zhang, Gen Li, Lei Sun
Affiliations: Tencent

6.7. DFCDN Team

Title: DFCDN: Deep Feature Complement and Distillation Network
Members: Mingxi Li, Yuhang Zhang, Xianjun Fan, Yankai Sheng
Affiliations Attrsense

6.8. z6

Title: Lightweight Efficient Real-Time Image Super-Resolution Network (LER- SRN)
Members: Ganzorig Gankhuyag, Kihwan Yoon
Affiliations: Korea Electronics Technology Institute (KETI)

6.9. NJUST-RTSR

Title: A Simple Residual ConvNet with Progressive Learning for Real- Time Super-Resolution
Members: Long Sun, Jinshan Pan, Jiangxin Dong, Jinhui Tang
Affiliations: Nanjing University of Science and Technology

6.10. Multimedia

Title: SCSYENet: A Compact Skip-Concatenated Simple Yet Effective Real- Time Image Super-Resolution based on element-wise multiplication fusion operation and Re-parameter convolution
Members: Zibin Liu, Weiran Gou, Shaoqing Li, Ziyao Yi, Yan Xiang, Dehui Kong, Ke Xu
Affiliations: Sanechips Co Ltd

6.11. Antins_CV

Title: Enhanced Residual Local Feature Network (ERLFN)
Members: Jin Zhang, Gaocheng Yu, Feng Zhang, Hongbin Wang
Affiliations: Ant Group

6.12. ECNU_SR

Title: Partial convolution based Network for Real-Time Super Resolution (PCRTSR)
Members: Zhou Zhou, Jiahao Chao, Hongfan Gao, Jiali Gong, Zhengfeng Yang, Zhenbing Zeng
Affiliations: East China Normal University

6.13. R.I.P ShopeeVideo

Title: Efficient Bottle-in-Bottle Block for Real-Time Super-Resolution
Members: Chengpeng Chen, Zichao Guo
Affiliations: Shopee https://shopee.com/

6.14. DoYouChargeQQCoin

Title: Ultra fast network for image super-resolution.
Members: Yuqing Liu, Qi Jia, Hongyuan Yu, Xuanwu Yin, Kunlong Zuo
Affiliations: Dalian University of Technology; Xiaomi Inc.

6.15. PixelBE

Title: Two-Stage Super-resolution Algorithm Based on Re-Parameterization
Members: Dongyang Zhang
Affiliations: Mango TV (MGTV)

6.16. AGSR

Title: Optimized Extreme Lightweight Super Resolution
Members: Ting Fu, Zhengxue Cheng, Shiai Zhu, Dajiang Zhou
Affiliations: Ant Group antgroup.com

6.17. dh_isp

Title: Lightweight network for image super-resolution.
Members: Ben Shao, Shaolong Zheng
Affiliations: Zhejiang Dahua Technology Co., Ltd.

6.18. Touch_Fish

Title: Attention Block for Real-time Super-Resolution
Members: Hongyuan Yu, Weichen Yu, Lin Ge, Jiahua Dong, Yajun Zou, Zhuoyuan Wu, Binnan Han, Xiaolin Zhang, Heng Zhang, Xuanwu Yin, Kunlong Zuo
Affiliations: Multimedia Department, Xiaomi Inc.

6.19. P.A.I.R

Title: Few Activation Distillation Networks for Real-time Super-resolution
Members: Anjin Park
Affiliations: Korea Photonic Technology Institute

6.20. SEU_CNII

Title: PRFDN: High Parallelism Distillation Network For Image Super-resolution
Members: Daheng Yin, Baijun Chen, Mengyang Liu
Affiliations: School of Computer Science and Engineering, Southeast University

6.21. diSRupt

Title: Depthwise-Residual Convolutional Neural Network (DRCNN)
Members: Marian-Sergiu Nistor
Affiliations: University "Al. I. Cuza" Iasi

6.22. NTU-BL6

Title: Finetuning and pruning for Real-Time Super-Resolution
Members: Yi-Chung Chen[3], Zhi-Kai Huang[2], Yuan-Chun Chiang[2], Wei-Ting Chen[1], Hao-Hsiang Yang[2], Hua-En Chang[2], I-Hsiang Chen[2], Chia-Hsuan Hsieh[4], Sy-Yen Kuo[2]
Affiliations: [1]Graduate Institute of Electronics Engineering, National Taiwan University, Taiwan
[2]Department of Electrical Engineering, National Taiwan University, Taiwan
[3]Graduate Institute of Communication Engineering, National Taiwan University, Taiwan
[4] ServiceNow, USA

6.23. NPU_Superresolution

Title: ECBSR,
Members: Qingsen Yan, Yun Zhu, Jinqiu Su, Yanning Zhang, Cheng Zhang, Jiaying Luo
Affiliations: Northwestern Polytechnical University

6.24. KCML2

Title: Enhanced Lightweight Image Super-resolution (ELIS)
Members: Tu Vo
Affiliations: KC Machine Learning Lab

6.25. YNOT

Title: Super Resolution with Spectral Transform and Wavelet Transform
Members: Youngsun Cho, Nakyung Lee
Affiliations: CJ OliveNetworks AI Research

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017. 2, 6, 8, 9, 10, 13, 14, 15, 16, 17

[2] 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. 5, 6, 9, 12

[3] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In *European Conference on Computer Vision*, pages 252–268, 2018. 2

[4] Codruta O Ancuti, Cosmin Ancuti, Florin-Alexandru Vasluianu, Radu Timofte, et al. NTIRE 2023 challenge on nonhomogeneous dehazing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[5] Mustafa Ayazoglu. Extremely lightweight quantization robust real-time single-image super resolution for mobile devices. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2472–2479, 2021. 10, 16

[6] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 3

[7] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie line Alberi Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In *British Machine Vision Conference*, pages 135.1–135.10, 2012. 2

[8] Bahri Batuhan Bilecen and Mustafa Ayazoglu. Bicubic++: Slim, slimmer, slimmest - designing an industry-grade super-resolution network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 5, 18, 19, 20

[9] Mingdeng Cao, Chong Mou, Fanghua Yu, Xintao Wang, Yinqiang Zheng, Jian Zhang, Chao Dong, Ying Shan, Gen Li, Radu Timofte, et al. NTIRE 2023 challenge on 360° omnidirectional image and video super-resolution: Datasets, methods and results. In *Proceedings of the*

IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2023. 3

[10] Chengpeng Chen, Zichao Guo, Haien Zeng, Pengfei Xiong, and Jian Dong. Repghost: A hardware-efficient ghost module via re-parameterization. *arXiv preprint arXiv:2211.06088*, 2022. 12

[11] Liangyu Chen, Xiaojie Chu, Xiangyu Zhang, and Jian Sun. Simple baselines for image restoration. *arXiv preprint arXiv:2204.04676*, 2022. 12

[12] Liangyu Chen, Xin Lu, Jie Zhang, Xiaojie Chu, and Chengpeng Chen. Hinet: Half instance normalization network for image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 182–192, June 2021. 12

[13] Xiangyu Chen, Xintao Wang, Jiantao Zhou, and Chao Dong. Activating more pixels in image super-resolution transformer. *arXiv preprint arXiv:2205.04437*, 2022. 6

[14] 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. 5

[15] Lu Chi, Borui Jiang, and Yadong Mu. Fast fourier convolution. *Advances in Neural Information Processing Systems*, 33:4479–4488, 2020. 16

[16] Xiaojie Chu, Liangyu Chen, Chengpeng Chen, and Xin Lu. Improving image restoration by revisiting global information aggregation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VII*, pages 53–71. Springer, 2022. 12

[17] Marcos V Conde, Ui-Jin Choi, Maxime Burchi, and Radu Timofte. Swin2SR: Swinv2 transformer for compressed image super-resolution and restoration. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2022. 2

[18] Marcos V Conde, Manuel Kolmet, Tim Seizinger, Thomas E. Bishop, Radu Timofte, et al. Lens-to-lens bokeh effect transformation. NTIRE 2023 challenge report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[19] Marcos V Conde, Eduard Zamfir, Radu Timofte, et al. Efficient deep models for real-time 4k image super-resolution. NTIRE 2023 benchmark and report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

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

[21] Xiaohan Ding, Yuchen Guo, Guiguang Ding, and J. Han. Acnet: Strengthening the kernel skeletons for powerful cnn via asymmetric convolution blocks. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1911–1920, 2019. 10

[22] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13733–13742, 2021. 3, 6, 11, 12, 14

[23] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *CVPR*, 2021. 9

[24] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *European Conference on Computer Vision*, pages 184–199, Cham, 2014. Springer International Publishing. 2

[25] Zongcai Du, Ding Liu, Jie Liu, Jie Tang, Gangshan Wu, and Lean Fu. Fast and memory-efficient network towards efficient image super-resolution, 2022. 13

[26] Zongcai Du, Jie Liu, Jie Tang, and Gangshan Wu. Anchor-based plain net for mobile image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2494–2502, 2021. 5

[27] Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. Depgraph: Towards any structural pruning. *The Thirty-Fourth IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 15

[28] William T Freeman, Thouis R Jones, and Egon C Pasztor. Example-based super-resolution. *IEEE Computer graphics and Applications*, 22(2):56–65, 2002. 2

[29] Ganzorig Gankhuyag, Jingang Huh, Myeongkyun Kim, Kihwan Yoon, HyeonCheol Moon, Seungho Lee, Jinwoo Jeong, Sungjei Kim, and Yoonsik Choe. Skip-concatenated image super-resolution network for mobile devices. *IEEE Access*, 2022. 9

[30] Ganzorig Gankhuyag, Kihwan Yoon, Jinman Park, Haeng Seon Son, and Kyoungwon Min. Lightweight real-time image super-resolution network for 4k images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 9

[31] Michaël Gharbi, Jiawen Chen, Jonathan T. Barron, Samuel W. Hasinoff, and Frédo Durand. Deep bilateral learning for real-time image enhancement. *ACM Transactions on Graphics (TOG)*, 36:1 – 12, 2017. 10

[32] Shuhang Gu, Andreas Lugmayr, Martin Danelljan, Manuel Fritsche, Julien Lamour, and Radu Timofte. Div8k: Diverse 8k resolution image dataset. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3512–3516, 2019. 2

[33] Shuhang Gu, Andreas Lugmayr, Martin Danelljan, Manuel Fritsche, Julien Lamour, and Radu Timofte. Div8k: Diverse 8k resolution image dataset. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3512–3516. IEEE, 2019. 5

[34] Jiaming Guo, Xueyi Zou, Yuyi Chen, Yi Liu, Jia Hao, and Youliang Yan. Asconvsr: Fast and lightweight super-resolution network with assembled convolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 5, 18, 19, 20

[35] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. Ghostnet: More features from cheap operations. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 1577–1586. Computer Vision Foundation / IEEE, 2020. 7

[36] Zibin He, Tao Dai, Jian Lu, Yong Jiang, and Shu-Tao Xia. Fakd: Feature-affinity based knowledge distillation for efficient image super-resolution. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 518–522. IEEE, 2020. 6

[37] Mu Hu, Junyi Feng, Jiashen Hua, Baisheng Lai, Jianqiang Huang, Xiaojin Gong, and Xian-Sheng Hua. Online convolutional re-parameterization. *CoRR*, abs/2204.00826, 2022. 8

[38] Yaoyu Hu, Wenshan Wang, Huai Yu, Weikun Zhen, and Sebastian Scherer. Orstereo: Occlusion-aware recurrent stereo matching for 4k-resolution images, 2021. 2, 20

[39] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2015. 2

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

[41] Andrey Ignatov, Radu Timofte, Maurizio Denna, and Abdel Younes. Real-time quantized image super-resolution on mobile npus, mobile ai 2021 challenge: Report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2525–2534, 2021. 5

[42] Andrey Ignatov, Radu Timofte, Maurizio Denna, Abdel Younes, Ganzorig Gankhuyag, Jingang Huh, Myeong Kyun Kim, Kihwan Yoon, Hyeon-Cheol Moon, Seungho Lee, et al. Efficient and accurate quantized image super-resolution on mobile npus, mobile ai & aim 2022 challenge: report. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*, pages 92–129. Springer, 2023. 2, 3

[43] Andrey Ignatov, Radu Timofte, Shuai Liu, Chaoyu Feng, Furui Bai, Xiaotao Wang, Lei Lei, Ziyao Yi, Yan Xiang, Zibin Liu, Shaoqing Li, Keming Shi, Dehui Kong, Ke Xu, Minsu Kwon, Yaqi Wu, Jiesi Zheng, Zhihao Fan, Xun Wu, Feng Zhang, Albert No, Minhyeok Cho, Zewen Chen, Xiaze Zhang, Ran Li, Juan Wang, Zhiming Wang, Marcos V. Conde, Ui-Jin Choi, Georgy Perevozchikov, Egor Ershov, Zheng Hui, Mengchuan Dong, Xin Lou, Wei Zhou, Cong Pang, Haina Qin, and Mingxuan Cai. Learned Smartphone ISP on Mobile GPUs with Deep Learning, Mobile AI & AIM 2022 Challenge: Report. *arXiv e-prints*, page arXiv:2211.03885, Nov. 2022. 10, 16

[44] Xiaoyang Kang, Xianhui Lin, Kai Zhang, Zheng Hui, Wangmeng Xiang, Jun-Yan He, Xiaoming Li, Peiran Ren, Xuansong Xie, Radu Timofte, et al. NTIRE 2023 video colorization challenge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[45] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 14

[46] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 9, 11, 13, 15, 16, 17

[47] Fangyuan Kong, Mingxi Li, Songwei Liu, Ding Liu, Jingwen He, Yang Bai, Fangmin Chen, and Lean Fu. Residual local feature network for efficient super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 766–776, 2022. 2, 12, 15

[48] Fangyuan Kong, Mingxi Li, Songwei Liu, Ding Liu, Jingwen He, Yang Bai, Fangmin Chen, and Lean Fu. Residual local feature network for efficient super-resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2022, New Orleans, LA, USA, June 19-20, 2022*, pages 765–775. IEEE, 2022. 10

[49] 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 *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4681–4690, 2017. 2

[50] Boyi Li, Wenqi Ren, Dengpan Fu, Dacheng Tao, Dan Feng, Wenjun Zeng, and Zhangyang Wang. Benchmarking single-image dehazing and beyond. *IEEE Transactions on Image Processing*, 28(1):492–505, 2019. 15

[51] Yawei Li, Kai Zhang, Luc Van Gool, Radu Timofte, et al. NTIRE 2022 challenge on efficient super-resolution: Methods and results. In *CVPR Workshops*, 2022. 2, 10, 17

[52] Yawei Li, Kai Zhang, Jingyun Liang, Jiezhang Cao, Ce Liu, Rui Gong, Yulun Zhang, Hao Tang, Yun Liu, Denis Demandolx, Rakesh Ranjan, Radu Timofte, and Luc Van Gool. LSDIR: a large scale dataset for image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 2, 9, 11, 12, 13, 14, 15

[53] Yawei Li, Kai Zhang, Radu Timofte, Luc Van Gool, Fangyuan Kong, Mingxi Li, Songwei Liu, Zongcai Du, Ding Liu, Chenhui Zhou, et al. Ntire 2022 challenge on efficient super-resolution: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1062–1102, 2022. 2, 6

[54] Yawei Li, Yulun Zhang, Luc Van Gool, Radu Timofte, et al. NTIRE 2023 challenge on efficient super-resolution: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[55] Yawei Li, Yulun Zhang, Luc Van Gool, Radu Timofte, et al. NTIRE 2023 challenge on image denoising: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[56] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration

using swin transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1833–1844, 2021. 2, 10

[57] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 136–144, 2017. 2

[58] Zudi Lin, Prateek Garg, Atmadeep Banerjee, Salma Abdel Magid, Deqing Sun, Yulun Zhang, Luc Van Gool, Donglai Wei, and Hanspeter Pfister. Revisiting rcan: Improved training for image super-resolution. *arXiv preprint arXiv:2201.11279*, 2022. 6

[59] J. Liu, D. Liu, W. Yang, S. Xia, X. Zhang, and Y. Dai. A comprehensive benchmark for single image compression artifacts reduction. In *arXiv*, 2019. 5

[60] 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, 12, 15

[61] Pengju Liu, Hongzhi Zhang, Kai Zhang, Liang Lin, and Wangmeng Zuo. Multi-level wavelet-cnn for image restoration. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 773–782, 2018. 16

[62] Xiaohong Liu, Xiongkuo Min, Wei Sun, Yulun Zhang, Kai Zhang, Radu Timofte, Guangtao Zhai, Yixuan Gao, Yuqin Cao, Tengchuan Kou, Yunlong Dong, Ziheng Jia, et al. NTIRE 2023 quality assessment of video enhancement challenge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[63] Zhuoqun Liu, Meiguang Jin, Ying Chen, Huaida Liu, Canqian Yang, and Hongkai Xiong. Mfdnet: Towards real-time image denoising on mobile devices. *CoRR*, abs/2211.04687, 2022. 8

[64] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017. 9

[65] Yusuke Matsui, Kota Ito, Yuji Aramaki, Azuma Fujimoto, Toru Ogawa, Toshihiko Yamasaki, and Kiyoharu Aizawa. Sketch-based manga retrieval using manga109 dataset. *Multimedia Tools and Applications*, 76(20):21811–21838, 2017. 16

[66] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Heewon Lee, Radu Timofte, and Kyoung Mu Lee. Ntire 2019 challenge on video restoration and enhancement: Methods and results. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2019. 2

[67] Ying Nie, Kai Han, Zhenhua Liu, An Xiao, Yiping Deng, Chunjing Xu, and Yunhe Wang. Ghostsr: Learning ghost features for efficient image super-resolution. *CoRR*, abs/2101.08525, 2021. 7

[68] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 102–118. Springer, 2016. 5

[69] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *European Conference on Computer Vision (ECCV)*, volume 9906 of *LNCS*, pages 102–118. Springer International Publishing, 2016. 15

[70] 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. 2, 3, 6, 9

[71] Alina Shutova, Egor Ershov, Georgy Perevozchikov, Ivan A Ermakov, Nikola Banic, Radu Timofte, Richard Collins, Maria Efimova, Arseniy Terekhin, et al. NTIRE 2023 challenge on night photography rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[72] Dehua Song, Chang Xu, Xu Jia, Yiyi Chen, Chunjing Xu, and Yunhe Wang. Efficient residual dense block search for image super-resolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12007–12014, 2020. 2

[73] Long Sun, Jinshan Pan, and Jinhui Tang. ShuffleMixer: An efficient convnet for image super-resolution. In *NeurIPS*, 2022. 9

[74] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, D. Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2014. 10

[75] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, and Lei Zhang. Ntire 2017 challenge on single image super-resolution: Methods and results. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 114–125, 2017. 5, 6, 9, 10, 11

[76] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, Lei Zhang, Bee Lim, et al. Ntire 2017 challenge on single image super-resolution: Methods and results. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017. 2, 15

[77] Radu Timofte, Vincent De Smet, and Luc Van Gool. Anchored neighborhood regression for fast example-based super-resolution. In *IEEE Conference on International Conference on Computer Vision*, pages 1920–1927, 2013. 2

[78] Radu Timofte, Vincent De Smet, and Luc Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *ACCV*, 2014. 2

[79] Radu Timofte, Rasmus Rothe, and Luc Van Gool. Seven ways to improve example-based single image super resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1865–1873, 2016. 2

[80] Florin-Alexandru Vasluianu, Tim Seizinger, Radu Timofte, et al. NTIRE 2023 image shadow removal challenge report. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[81] Longguang Wang, Xiaoyu Dong, Yingqian Wang, Xinyi Ying, Zaiping Lin, Wei An, and Yulan Guo. Exploring sparsity in image super-resolution for efficient inference. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4917–4926, 2021. 2

[82] Longguang Wang, Yulan Guo, Yingqian Wang, Juncheng Li, Shuhang Gu, Radu Timofte, et al. NTIRE 2023 challenge on stereo image super-resolution: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[83] Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1905–1914, 2021. 6

[84] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *European Conference on Computer Vision Workshops*, pages 701–710, 2018. 2

[85] Yingqian Wang, Longguang Wang, Zhengyu Liang, Jungang Yang, Radu Timofte, Yulan Guo, et al. NTIRE 2023 challenge on light field image super-resolution: Dataset, methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[86] Lei Xiao, Salah Nouri, Matt Chapman, Alexander Fix, Douglas Lanman, and Anton Kaplanyan. Neural supersampling for real-time rendering. *ACM Transactions on Graphics (TOG)*, 39(4):142–1, 2020. 2

[87] Tianyu Xu, Zhuang Jia, Yijian Zhang, Long Bao, and Heng Sun. Elsr: Extreme low-power super resolution network for mobile devices, 2022. 13

[88] Ren Yang, Radu Timofte, Xin Li, Qi Zhang, Lin Zhang, Fanglong Liu, Dongliang He, Fu Li, He Zheng, Weihang Yuan, et al. Aim 2022 challenge on super-resolution of compressed image and video: Dataset, methods and results. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*, pages 174–202. Springer, 2023. 2

[89] Jaejun Yoo, Youngjung Uh, Sanghyuk Chun, Byeongkyu Kang, and Jung-Woo Ha. Photorealistic style transfer via wavelet transforms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9036–9045, 2019. 16

[90] Hongliang Yuan, Boyu Zhang, Mingyan Zhu, Ligang Liu, and Jue Wang. High-quality supersampling via mask-reinforced deep learning for real-time rendering. *arXiv preprint arXiv:2301.01036*, 2023. 2

[91] Pierluigi Zama Ramirez, Fabio Tosi, Luigi Di Stefano, Radu Timofte, et al. NTIRE 2023 challenge on hr depth from images of specular and transparent surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[92] Eduard Zamfir, Marcos V Conde, and Radu Timofte. Towards real-time 4k image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 2, 3, 4, 18, 19, 20

[93] 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, 2010. 2

[94] Kai Zhang, Martin Danelljan, Yawei Li, Radu Timofte, Jie Liu, Jie Tang, Gangshan Wu, Yu Zhu, Xiangyu He, Wenjie Xu, et al. Aim 2020 challenge on efficient super-resolution: Methods and results. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 5–40, 2020. 2

[95] Kaihao Zhang, Dongxu Li, Wenhan Luo, Wenqi Ren, Björn Stenger, Wei Liu, Hongdong Li, and Ming-Hsuan Yang. Benchmarking ultra-high-definition image super-resolution. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14769–14778, 2021. 2

[96] Xindong Zhang, Hui Zeng, and Lei Zhang. Edge-oriented convolution block for real-time super resolution on mobile devices. In Heng Tao Shen, Yueting Zhuang, John R. Smith, Yang Yang, Pablo César, Florian Metze, and Balakrishnan Prabhakaran, editors, *MM '21: ACM Multimedia Conference, Virtual Event, China, October 20 - 24, 2021*, pages 4034–4043. ACM, 2021. 8, 16

[97] Xindong Zhang, Huiyu Zeng, and Lei Zhang. Edge-oriented convolution block for real-time super resolution on mobile devices. *Proceedings of the 29th ACM International Conference on Multimedia*, 2021. 10

[98] 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. 13

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

[100] Yulun Zhang, Kai Zhang, Zheng Chen, Yawei Li, Radu Timofte, et al. NTIRE 2023 challenge on image super-resolution (x4): Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2023. 3

[101] Hengyuan Zhao, Xiangtao Kong, Jingwen He, Yu Qiao, and Chao Dong. Efficient image super-resolution using pixel attention. In *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 56–72. Springer, 2020. 2