

# Real-Time 4K Super-Resolution of Compressed AVIF Images.

## AIS 2024 Challenge Survey

Marcos V. Conde<sup>†</sup>      Zhijun Lei<sup>†</sup>      Wen Li<sup>†</sup>      Cosmin Stejerean<sup>†</sup>      Ioannis Katsavounidis<sup>†</sup>  
 Radu Timofte<sup>†</sup>      Kihwan Yoon      Ganzorig Gankhuyag      Jiangtao Lv      Long Sun  
 Jinshan Pan      Jiangxin Dong      Jinhui Tang      Zhiyuan Li      Hao Wei      Chenyang Ge  
 Dongyang Zhang      Tianle Liu      Huaiyan Chen      Yi Jin      Menghan Zhou      Yiqiang Yan  
 Si Gao      Biao Wu      Shaoli Liu      Chengjian Zheng      Diankai Zhang      Ning Wang  
 Xintao Qiu      Yuanbo Zhou      Kongxian Wu      Xinwei Dai      Hui Tang      Wei Deng  
 Qingquan Gao      Tong Tong      Jae-Hyeon Lee      Ui-Jin Choi      Min Yan      Xin Liu  
 Qian Wang      Xiaoqian Ye      Zhan Du      Tiansen Zhang      Long Peng      Jiaming Guo  
 Xin Di      Bohao Liao      Zhibo Du      Peize Xia      Renjing Pei      Yang Wang  
 Yang Cao      Zhengjun Zha      Bingnan Han      Hongyuan Yu      Zhuoyuan Wu  
 Cheng Wan      Yuqing Liu      Haodong Yu      Jizhe Li      Zhijuan Huang      Yuan Huang  
 Yajun Zou      Xianyu Guan      Qi Jia      Heng Zhang      Xuanwu Yin      Kunlong Zuo  
 Hyeon-Cheol Moon      Tae-hyun Jeong      Yoonmo Yang      Jae-Gon Kim      Jinwoo Jeong  
 Sunjei Kim



Figure 1. Sample high-quality 4K images from the testing dataset of the AIS 2024 RTSR Challenge.

## Abstract

This paper introduces a novel benchmark as part of the AIS 2024 Real-Time Image Super-Resolution (RTSR) Challenge, which aims to upscale compressed images from 540p

<sup>†</sup> are the challenge organizers, while the other authors participated in the challenge. Workshop page: <https://ai4streaming-workshop.github.io/>. Benchmark code: <https://github.com/eduardzamfir/NTIRE23-RTSR>

Marcos V. Conde (corresponding author, project lead) and Radu Timofte are with University of Würzburg, CAIDAS & IFI, Computer Vision Lab. Zhijun Lei, Wen Li, Cosmin Stejerean and Ioannis Katsavounidis are with Meta.

to 4K resolution (4x factor) in real-time on commercial GPUs. For this, we use a diverse test set containing a variety of 4K images ranging from digital art to gaming and photography. The images are compressed using the modern AVIF codec, instead of JPEG. All the proposed methods improve PSNR fidelity over Lanczos interpolation, and process images under 10ms. Out of the 160 participants, 25 teams submitted their code and models. The solutions present novel designs tailored for memory-efficiency and runtime on edge devices. This survey describes the best solutions for real-time SR of compressed high-resolution images.

## 1. Introduction

Single image super-resolution (SR) methods generate a high-resolution (HR) image from a single degraded low-resolution (LR) image. This ill-posed problem was initially solved using interpolation methods. However, SR is now commonly approached through the use of deep learning [1, 6, 26, 33, 41]. Image SR assumes that the LR image is obtained through a degradation processes. This can be expressed as:

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

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  $\times s$  (*e.g.*  $\times 2$ ,  $\times 3$ ,  $\times 4$ ,  $\times 8$ ).

Following the foundational efforts by Shi *et al.* [34], optimizing deep neural networks for single image super-resolution has become critical [18, 22, 35, 40, 46, 50]. This focus has inspired the creation of numerous workshops and challenges, for instance [25, 47], which serve as platforms for exchanging ideas and pushing the boundaries of efficient and real-time super-resolution (SR). Moreover, many works consider compressed images as inputs (*e.g.* JPEG), what leads to more practical super-resolution methods [6, 9, 43].

## 2. AIS 2024 Real-Time Image SR Challenge

In conjunction with the 2024 AIS: Vision, Graphics and AI for Streaming workshop, we introduce a new real-time 4K super-resolution challenge.

The challenge aims to upscale a compressed LR image from 540p to 4K resolution using a neural network that complies with the following requirements: (i) improve performance over Lanczos interpolation. (ii) Upscale the image under 33ms. Moreover the images are compressed using different compression factors (QP values) using the modern AVIF codec instead of JPEG. The challenge seeks to identify innovative and advanced solutions for real-time super-resolution of compressed images.

### 2.1. Motivation

AV1 Image File Format (AVIF) is the latest royalty-free image coding format developed based on the Alliance for Open Media’s (AOM) AV1 video coding standard. The compression efficiency and quality of AVIF encoded images is noticeably superior to JPEG and also HEIC, which uses HEVC for image coding. AVIF is also supported in all major web browsers. In the AIS 2024 Real-Time Image SR Challenge, we want to leverage AVIF as the image coding format to evaluate the quality improvement from SR when combining with AVIF.

### 2.2. 4K SR Benchmark Dataset

Following [9], 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 upsampling computer-generated content *e.g.* gaming and rendered content, in addition to photo-realistic imagery, thereby posing a different challenge for existing SR approaches.

The testing set includes diverse content such as rendered gaming content, digital art, as well as high-resolution photo-realistic images of animals, city scenes, and landscapes, totaling 110 test samples.

All the images in the benchmark testing set are at least 4K resolution *i.e.*  $3840 \times 2160$  (some are bigger, even 8K).

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

**Compression and Downsampling** We use `ffmpeg` to produce the LR compressed images. We use 5 different QP values: 31, 39, 47, 55, 63. We use lanczos interpolation to downsample the images. Below we provide an example:

```
ffmpeg -hide_banner -y -loglevel error -i
    ↪ ./1.png -vf 'scale=ceil(iw/4):ceil(
    ↪ ih/4):flags=lanczos+accurate_rnd+
    ↪ full_chroma_int:sws_dither=none:
    ↪ param0=5' -c:v libsvtav1 -qp 31 -
    ↪ preset 5 1_4x_qp31.avif
```

In the context of AVIF and AV1 codecs, larger Quantization Parameter (QP) values imply more compression. Essentially, the QP value dictates the level of quantization applied to the video or image data, where higher quantization reduces the amount of data required to represent the original input, thus leading to higher compression ratios.

The participants can use any publicly available dataset, and produce the corresponding LR images.

### 2.3. Evaluation

The 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 final test phase, the participating teams provided the code, models and results corresponding to the 110 test images. They did not have access to the HR ground-truth. 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.

Team Method	Score	$\Delta$ [dB]	MACs [G]	Runtime [ms]	# Par. [M]	PSNR-RGB [dB] QP31 QP63	PSNR-Y [dB] QP31 QP63	SSIM-RGB QP31 QP63	SSIM-Y QP31 QP63
BasicVison (Sec. 3.10)	27.27	0.350	6.20	0.874	0.012	30.85 26.83	33.30 29.27	0.807 0.719	0.850 0.777
CMVG (Sec. 3.10)	27.55	0.330	5.16	0.833	0.010	30.84 26.82	33.27 29.26	0.807 0.719	0.849 0.777
IVP (Sec. 3.10)	27.08	0.365	6.07	0.905	0.011	30.85 26.83	33.33 29.27	0.808 0.719	0.851 0.777
RVSR (Sec. 3.5)	12.00	0.720	15.6	7.542	0.033	31.52 27.01	33.88 29.43	0.820 0.725	0.859 0.781
VPEG-R (Sec. 3.3)	25.77	0.100	1.56	0.692	0.012	30.59 26.70	32.94 29.13	0.803 0.714	0.845 0.773
VPEG-S (Sec. 3.3)	16.15	0.735	34.2	4.251	0.066	31.57 26.99	33.93 29.41	0.821 0.725	0.859 0.781
ANUNet (Sec. 3.6)	22.77	0.545	9.40	1.642	0.072	31.27 26.86	33.66 29.30	0.814 0.719	0.855 0.778
RESR (Sec. 3.7)	28.68	0.455	7.04	0.914	0.044	31.16 26.87	33.50 29.28	0.813 0.720	0.853 0.777
Megastudy (Sec. 3.11)	13.12	0.210	20.7	3.109	0.040	30.76 26.87	33.01 29.28	0.801 0.721	0.842 0.778
CASR (Sec. 3.4)	33.70	0.205	2.10	0.468	0.026	30.64 26.71	33.11 29.17	0.806 0.715	0.848 0.774
XiaomiMM (Sec. 3.8)	18.60	0.690	13.3	3.010	0.026	31.41 26.96	33.85 29.40	0.819 0.725	0.857 0.781
Team C3 (Sec. 3.8)	20.35	0.500	12.4	1.932	0.024	31.12 26.90	33.52 29.35	0.813 0.723	0.853 0.780
USTC Noah (Sec. 3.9)	25.63	0.485	8.56	1.193	0.045	31.18 26.90	33.52 29.32	0.814 0.722	0.854 0.779
Lanczos++ (Sec. 3.2)	33.24	0.070	1.25	0.399	0.022	30.55 26.71	32.88 29.13	0.800 0.714	0.843 0.772
PixelArtAI (Sec. 3.1)	33.32	0.395	9.68	0.623	0.075	31.06 26.84	33.40 29.26	0.811 0.719	0.852 0.777
RepTCN (Sec. 3.1)	30.91	0.355	5.02	0.685	0.010	30.97 26.83	33.31 29.27	0.809 0.719	0.850 0.777
URPNet (Sec. 3.12)	-	-	1.24	0.621	0.009	30.33 26.65	32.64 29.07	0.798 0.713	0.842 0.773
Baseline Lanczos	-	-	-	-	-	30.36 26.67	32.75 29.12	0.800 0.715	0.843 0.774

Table 1. **Results of the AIS24 Real-Time SR challenge.** All the proposed methods upsample the images under **8ms**. The models can process compressed images with QP factors from 31 to 63. We provide PSNR and SSIM metrics in the RGB domain, and for the Luma (Y) channel. We report the fidelity improvement *w.r.t* the baseline in terms of PSNR  $\Delta$ , considering the average PSNR-Y for QP31 and QP63. We highlight the top-3 (gold, silver, bronze) methods in terms of fidelity, runtime, and computational efficiency (MACs operations).

## 2.4. Architectures and Main Ideas

Here we summarize the core ideas behind the most competitive solutions. Note that most of the ideas follow [9, 46].

- **Re-parameterization** enables training the network using sophisticated blocks [11], while allowing these “Rep-Blocks” to be simplified into a standard  $3 \times 3$  convolutions during inference. This technique has become state-of-the-art in efficient SR [9, 24].
- **Pixel shuffle and unshuffle.** These techniques are also known as depth-to-space, space-to-depth, and sub-pixel convolutions [34]. These are utilized to effectively apply spatial upsampling and downsampling over feature maps.
- **Multi-stage Training:** Given the significant limitations and shallow architecture of the neural networks, this approach enhances learning by varying learning rates and loss functions sequentially.
- **Knowledge distillation** allows to transfer knowledge from complex neural networks into more efficient ones.

## 2.5. Results and Conclusion

In Tab. 1 we provide the challenge benchmark. The models can upsample compressed 540p images and recover the core structural information according to the metrics calculated over Luma (Y). We can also appreciate a notable performance decay at high QP (compression) values. The runtime is the average of 100 runs after GPU warm up, using an NVIDIA 4090 GPU. Following previous work, we also report a “Score” (S) that considers PSNR and runtime, defined as:  $S = (2 \times 2^\Delta) / (T^{0.5} \times C)$  where  $T$  is the runtime,

and  $C = 0.1$  is a constant scaling factor.

Attending to the results, we consider that this score is now obsolete, and we included it only for informative purposes. All the methods achieve runtimes under 10ms. In this scenario, larger fidelity improvements and other memory-related improvements are considered more important for practical applications.

In Sec. 3 we provide the description of the top solutions.

Considering the best methods, we can conclude that there is certain convergence in the model designs. As previously mentioned, re-parameterization is ubiquitous. Edge-oriented filters to extract directly high-frequencies allow to reduce sparsity in the neural network, making effective use of all the kernels (parameters). Upsampling the input image, and enhancing it through a global residual connection is also a common neural network architecture.

**Related Challenges** This challenge is one of the AIS 2024 Workshop associated challenges on: Event-based Eye-Tracking [42], Video Quality Assessment of user-generated content [8], Real-time compressed image super-resolution [7], Mobile Video SR, and Depth Upscaling.

**Acknowledgments** This work was partially supported by the Humboldt Foundation. We thank the AIS 2024 sponsors: Meta Reality Labs, Meta, Netflix, Sony Interactive Entertainment (FTG), and the University of Würzburg (Computer Vision Lab).

Marcos Conde is also supported by Sony Interactive Entertainment (FTG).

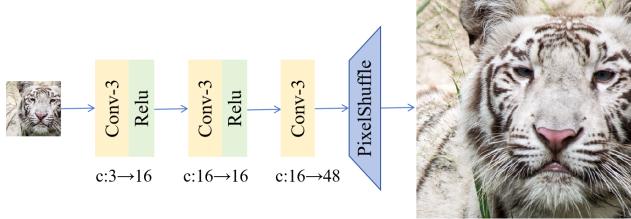


Figure 2. Overview of RepTCN by Team CameraAI.

### 3. Methods and Teams

In the following sections we describe the best challenge solutions. Note that the method descriptions were provided by each team as their contribution to this survey.

#### 3.1. Simple Baselines

##### *Team CameraAI*

*Tianle Liu, Huaian Chen, Yi Jin*

*University of Science and Technology of China*

The team proposes **RepTCN**, a network comprising only three convolutional layers, achieving superior performance over Lanczos interpolation while maintaining exceptional efficiency. To further enhance efficacy, we introduced re-parameterization techniques, replacing the middle convolutional layer with a RepBlock [12] during the training phase. Additionally, we devised a three-stage training strategy to fully exploit the model’s potential.

Figure 2 illustrates our proposed RepTCN. It consists of three convolutional layers, each without bias. A ReLU activation function is applied between every two convolutional layers. During the training phase, we replace the middle convolution with a RepBlock[10]. During inference, we reparameterize the RepBlock into a convolutional layer.

**Implementation details** Our training framework uses Pytorch for training on the RTX3090. We gathered the first 600 images from DIV2K, the first 600 images from Flicker2K, and the first 800 images from GTAV. Subsequently, we cropped these images to  $512 \times 512$  to form our dataset.

During the training phase, the input from the dataset will be randomly cropped into patches, and these patches will undergo random horizontal flips and rotations. The model training can be divided into three stages. In the first stage, we set the batch size to 32 and the patch size to 32. L1 loss are used as target loss functions. We replaced the middle convolutional layer with a RepBlock[10] and trained for 1000k iterations using the Adam optimizer, with a learning rate of  $1 \times 10^{-3}$  decreasing to  $1 \times 10^{-7}$  through the cosine scheduler. In the second stage, we set the batch size

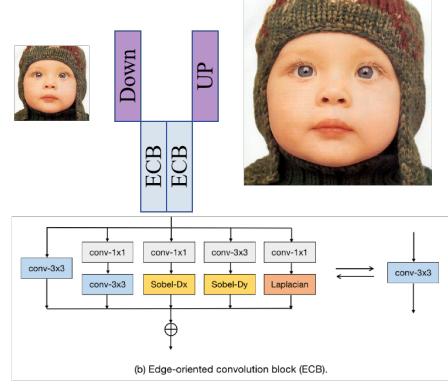


Figure 3. Simple network proposed by PixelArtAI.

to 16 and the patch size to 128. MSE loss are used as target loss functions. We reparameterized the RepBlock into a convolutional layer and trained for 500k iterations using the Adam, with a learning rate of  $5 \times 10^{-4}$  decreasing to  $5 \times 10^{-7}$  through the cosine scheduler. In the third stage, we removed the bias from each convolutional layer and trained for 2000k iterations using the Adam, with a learning rate of  $5 \times 10^{-4}$  decreasing to  $5 \times 10^{-7}$  through the cosine scheduler. The other Settings are the same as in the previous step.

##### *Team PixelArtAI*

*Dongyang Zhang*

*MangoTV*

The team proposes a lightweight and extremely low-time-consuming network is built through re-parameters. Based on the ECB module [48], we designed a lightweight and low-time-consuming network for the competition. The network design points are as follows:

First downsample by a factor of 2 using a convolution with a stride of 2. Downsampling breaks down compression and also improves network inference speed. Then stack two ECB modules and a 8x upsampled pixel shuffle module to return a three-channel image – see Fig. 3.

**Efficiency Metrics** Considering an input 540p and x4 SR, the model has 1.8798 KMACs and a runtime of 1.0367 ms.

**Implementation details** The training data is degraded by FFmpeg with random QP. The input image size is 120x120x3, and the batch size is 96. We use Adam optimizer with the initial learning rate set to 0.001. The training is divided into two stages: First, the learning rate is 0.001 and the loss is L1. This stage is trained for 60k iterations. Second, only the PSNR Loss is calculated, and the initial learning rate set to 0.0002, and is halved by 20k iterations.

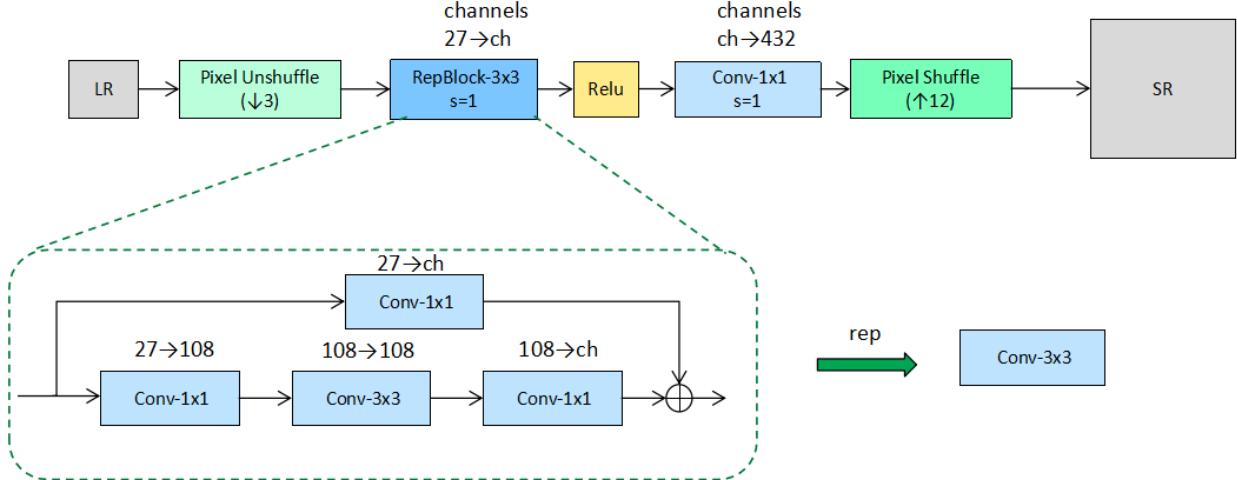


Figure 4. Lanczos++ proposed by Team ZXVIP

Method	PSNR
Lanczos	22.70
Bicubic++	22.92
RCBSR	22.76
Lanczos++	22.77

Table 2. Lanczos++ compared with other fast methods on the 40 images validation set.

### 3.2. Lanczos++:An ultra lightweight image super-resolution network

**Team ZXVIP**

*Si Gao*<sup>1</sup>, *Biao Wu*, *Shaoli Liu*, *Chengjian Zheng*,  
*Diankai Zhang*, *Ning Wang*<sup>2</sup>

<sup>1</sup> Network Intelligence Platform Development Dept. III,  
ZTE

<sup>2</sup> Network Intelligence Platform System Dept., ZTE

The team proposes an ultra lightweight image super-resolution network named Lanczos++.

The highlights of the proposed network (see Fig. 4) are as follows: First, we use PixelShuffle to perform 3x down-sampling on the input LR image while increasing the channel dimension. This design significantly improves the inference efficiency of the network, while basically not losing the representation ability of the model. Second, we design a new type of reparameterization module named RepBlock(see Fig. 4).

In the training phase, we first use 1x1 convolution to increase the input channel to four times, then use 3x3 convo-

lution for feature extraction, and finally use 1x1 convolution to transform the dimension into the output channel dimension, and use it as a residual with the branch that transforms the input channel into the output channel through 1x1 convolution. During the model inference stage, we merge the reparameterization module into a standard 3x3 convolution. Reparameterization can improve the fidelity of the model while maintaining its inference efficiency unchanged.

Third, we remove the bias of the convolutional layer and use 1x1 convolution instead of 3x3 convolution in the reconstruction layer convolution section, which can significantly reduce the running time of the model. Finally, we use Pixelshuffle for 12x up-sampling.

**Implementation details** We use 4450 images from DIV2K, Flickr2K and GTA V datasets for training. To generate the LR data, the images are lanczos downsampled by scale 4 and compressed samples using AVIF with QP 31/39/47/55/63 (5 compression levels).

We implement a three-stage training pipeline: training a basic model, deviation removal of convolutional layers, and final fine-tuning of switching loss functions. In the first stage, we conduct a NAS architecture search to find the optimal network parameter configuration. For the first two stages, we use the L1 loss function for training, and for the final stage, we use the L2 loss function.

We use Adam optimizer by setting  $\beta_1 = 0.9$  and  $\beta_2 = 0.999$ . In the first two stages of training, we start with a learning rate of 5e-4. For the final stage, they start from 2e-4. We use a decaying learning rate scheduler for all stages, where the first 500 epochs are preheated and then the learning rate decays linearly until 1e-8. The total duration of the training process is around 50h using a V100 (32Gb) GPU.

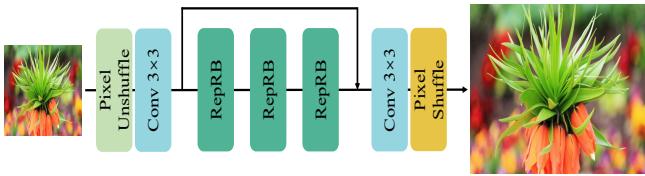


Figure 5. An overview of the proposed VPEG-R model.

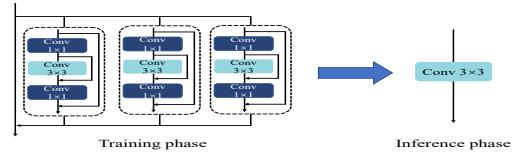


Figure 6. Proposed RepRB for the VPEG-R model.

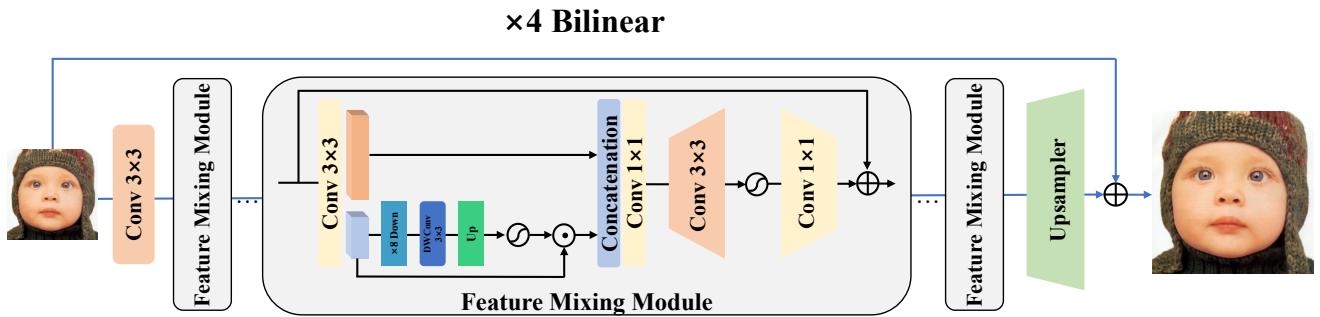


Figure 7. An overview of the proposed SAFMN++ model.

### 3.3. An efficient and fast network for super-resolution using convolution

#### Teams VPEG

Jiangtao Lv, Long Sun, Jinshan Pan, Jiangxin Dong,  
Jinhui Tang

Nanjing University of Science and technology

**SAFMN++: Improved Feature Modulation Network for Real-Time Compressed Image Super-Resolution** We introduce SAFMN++, an enhanced version of SAFMN [36] for solving real-time compressed image SR. This solution is mainly concentrates on improving the effectiveness of the spatially-adaptive feature modulation (SAFM) [36] layer. Different from the original SAFM, as shown in Fig 7, the improved SAFM (SAFM++) is able to extract both local and non-local features. In SAFM++, a  $3 \times 3$  convolution is first utilized to extract local features and a single scale feature modulation is then applied to a portion of the extracted features for non-local feature interaction.

After this process, these two sets of features are aggregated by channel concatenation and fed into a  $1 \times 1$  convolution for feature fusion.

The proposed SAFMN++ is trained by minimizing a combination of the uncertainty-based MSE loss [14, 16] and FFT-based L1 loss [5] with Adam optimizer for a total of 500,000 iterations. We train the proposed SAFMN++ on the DIV2K [1] dataset. The cropped LR image size is  $640 \times 640$  and the mini-batch size is set to 64. We set the initial learn-

Method	Params [M]	FLOPs [G]	Runtime [ms]	Val. PSNR
VPEG-S	0.0662	34.1587	11.5839	23.29
VPEG-R	0.0122	1.556	2.2531	22.77

Table 3. SAFMN++: efficiency results. “FLOPs” and “Runtime” are tested on an LR image of size  $540 \times 960$  with an NVIDIA RTX3060.

ing rate to  $3 \times 10^{-3}$  and the minimum one to  $1 \times 10^{-7}$ , which is updated by the Cosine Annealing scheme [28].

Table 3 presents the efficiency study of SAFMN++.

**A Simple Residual ConvNet with Structural Reparameterization for Real-Time Super-Resolution** The solution VPEG-R is shown in Fig. 5. The proposed method reduces the spatial resolution by a Pixel Unshuffle operation and uses a convolutional layer to transform the input LR image into the feature space, then performs feature extraction using 3 reparameterizable residual blocks (RepRBs), and finally reconstructs the final output by a PixelShuffle [33] convolution.

We use DIV2K [1] as the training data. In order to accelerate the IO speed during training, we crop the 2K resolution HR images to  $640 \times 640$  sub-images, and the mini-batch size is set to 64.

**Implementation details** We use PyTorch and a NVIDIA GeForce RTX 3090 GPU. The training process takes about 44 hours for SAFMN++, and two days for VPEG-R.

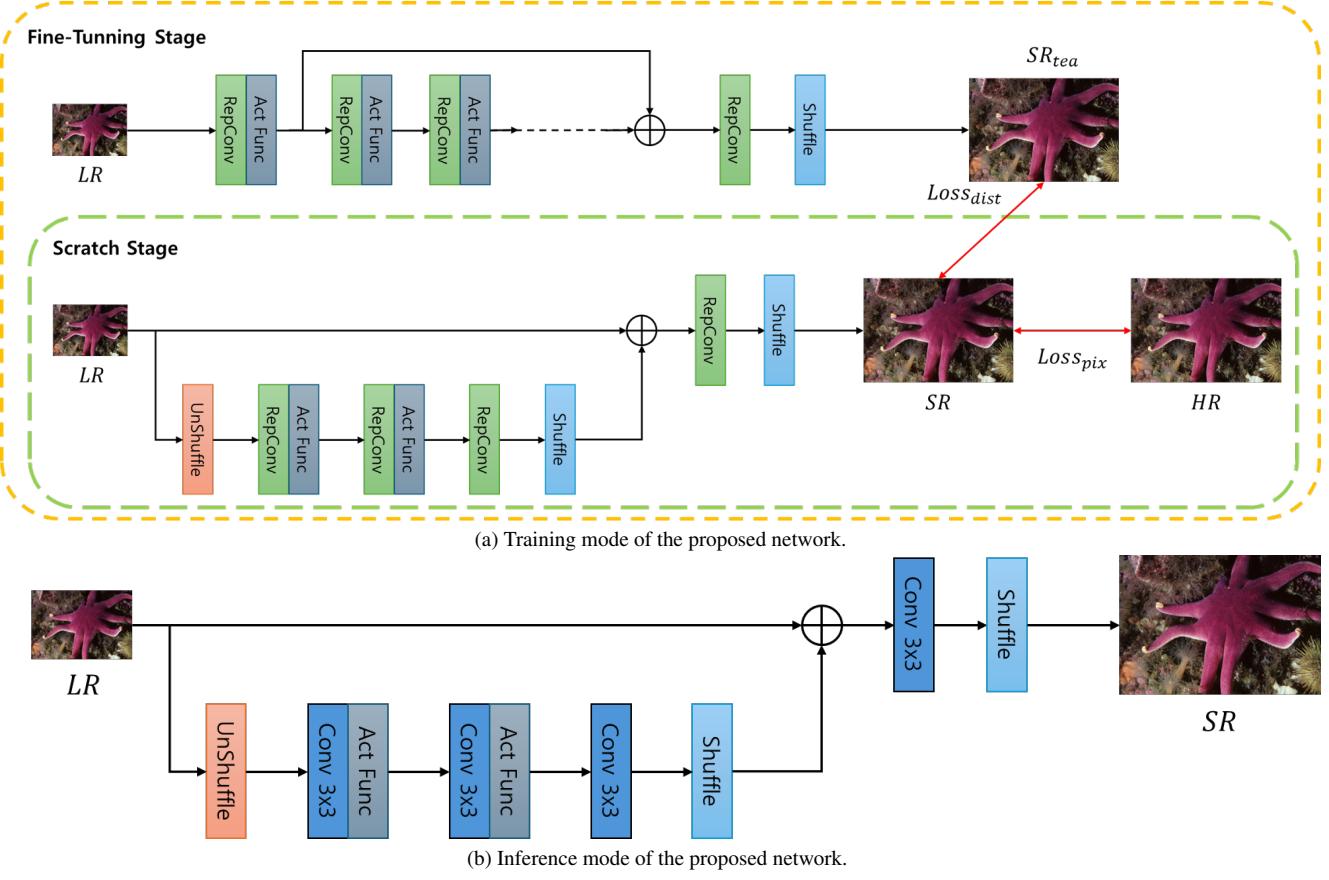


Figure 8. CASR network structure proposed by team Z6.

### 3.4. CASR: Efficient Cascade Network Structure with Channel Aligned method for 4K Real-Time Single Image Super-Resolution

**Team Z6**

Kihwan Yoon<sup>1</sup>, Ganzorig Gankhuyag<sup>2</sup>

<sup>1</sup> The University of Seoul

<sup>2</sup> Korea Electronics Technology Institute (KETI)

We initially reviewed the key factors essential for developing a network structure. Subsequently, we suggest a Cascade Upsampling network structure with Channel Alignment approach for image enhancement, which enhances performance and notably decreases processing time. Lastly, we designed an effective network and integrated reparameterization blocks and knowledge distillation methods to enhance performance without increasing the model’s size [44].

We compared our proposed method with LRSRN [15] which proposed work on the NTIRE 2023 real-time super-resolution challenge [9] in Tab. 4. The score value is calculated from the script of [9]. Our proposed method over-

Method [15]	Div2K Val	AIS2024 Val	Inference Time (ms)
LRSRN [15]	27.26	23.1	4.4890
Proposed method	26.69	22.80	0.5678

Table 4. Ablation study of team Z6.

whelms the previous method and we achieve 0.5678 ms inference time at RTX3090.

**Implementation details** We used two different types of dataset: DIV2K and combined datasets.

- DIV2K: Well-known open dataset. DIV2K training data set used in the scratch training step.
- Combined: The DIV2K training dataset is utilized during the initial training phase from scratch. In contrast, a composite dataset is used for the subsequent second stage. This combined dataset comprises the full DIV2K training set (800 images), the initial 1000 images from the Flickr training set, 121 samples from the GTA training sequences 00 to 19, the first 1000 images from the LSDIR dataset. To generate low resolution, we degrade the random cropped images with avif compression with various

compression factors. For both training stages, we used random cropping, rotation 90, horizontal flip, and vertical flip augmentation.

We trained our model in three steps:

(1) Scratch train step: In the first step, our model was trained from scratch. The LR patches were cropped from LR images with eight mini-batch 98 x 98 sizes. Adam optimizer was used with a learning rate of 0.0005 during scratch training. The total number of epochs was set to 800. We use the  $l_1$  loss.

(ii) Second step: In the second step, the model was initialized with the weights trained in the first step. The distillation method used at this stage. The teacher model was trained with the combined dataset. The detailed illustrated example is shown in Fig. 8b. Fine tuning with loss  $l_2$  improves the PSNR by  $0.01 \sim 0.02$  dB. Also, we turn off the bias term of the reparametrization block at this stage. In this step, the initial learning rate was set to 0.00005 and the Adam optimizer was used along with a cosine warm-up. The total epoch was set at 800 epochs.

(iii) Third step: In the third stage, the model was initialized using the weights trained in the previous step. In addition, the distillation technique was applied in this phase as well. The training hyper-parameters were kept identical to those in the second step. At this point, the bias term of the reparametrization block was deactivated, leading to a decrease in inference time by 0.2 ms. Although there was a slight reduction of 0.02 dB in the precision of the PSNR value, the overall score improved.

We refer the reader to the CASR [44] paper for more details.

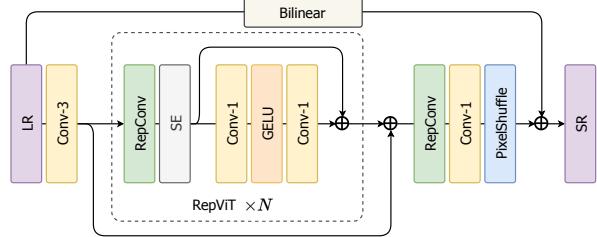
### 3.5. RVSR: Real-Time Super-Resolution with Re-parameterization and ViT architecture

**Team XJTU-AIR**

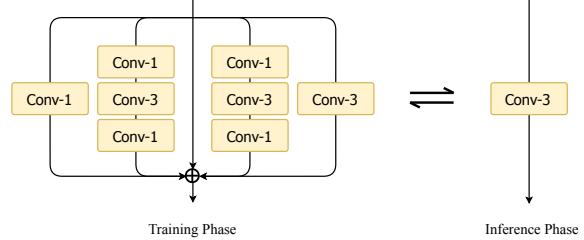
Zhiyuan Li, Hao Wei, Chenyang Ge

*Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University*

We propose a real-time image super-resolution method called RVSR, which is inspired by previous work [13, 39]. Our method leverages the efficient architectural designs of lightweight ViTs and the re-parameterization technique to achieve superior performance in real-time super-resolution tasks. RVSR first applies a  $3 \times 3$  convolution to convert the channel of feature map to the target size (16). Then, RVSR employs 8 stacked RepViT [39] blocks to perform deep feature extraction. As shown in Fig. 9 (a), the RepViT blocks integrate the efficient architectural designs of lightweight ViTs. Inspired by [13], RVSR employs the RepConv module to improve the SR performance while maintaining low complexity, as shown in Fig. 9 (b).



(a) Detailed architecture of RVSR by Team XJTU-AIR.



(b) The RepConv module

Figure 9. Overview of the proposed RVSR by Team XJTU-AIR.

We conducted an end-to-end training of the RVSR model for 5000 epochs, employing a batch size of 32 and optimizing by minimizing the MSE loss with the Adam optimizer. For inference, we re-parameterized the model using standard  $3 \times 3$  convolutions, as illustrated in Fig. 9 (b).

**Implementation details** The method is implemented in PyTorch. For optimization, we utilize the Adam optimizer with  $\beta_1 = 0.99$  and  $\beta_2 = 0.999$ . The learning rate is set to  $5 \times 10^{-4}$  for the first 1000 epochs, after which it linearly decays until reaching  $1 \times 10^{-6}$ .

We trained RVSR on DIV2K dataset (800 images), Flickr2K dataset (2650 images) and LSDIR dataset (first 1000 images). For generating low-resolution images, we employed Lanczos downsampling and AVIF compression, with compression factors ranging from QP 31 to 63. During training, we used random cropping, rotations, and flips augmentations. Besides, the images are normalized to the range [-1, 1]. The experiments were conducted on a Nvidia GeForce RTX 3090 GPU, with the input size set to  $960 \times 540$ . MACs: 15.62 (G), 1883 MACs per pixel, runtime: 12.54 ms (FP32) and 7.36 ms (FP16).

### 3.6. Anchor-based Nested UnshuffleNet for Real-time Super-Resolution (ANUNet)

**Team LeRTSR**

Menghan Zhou, Yiqiang Yan

*Lenovo Research*

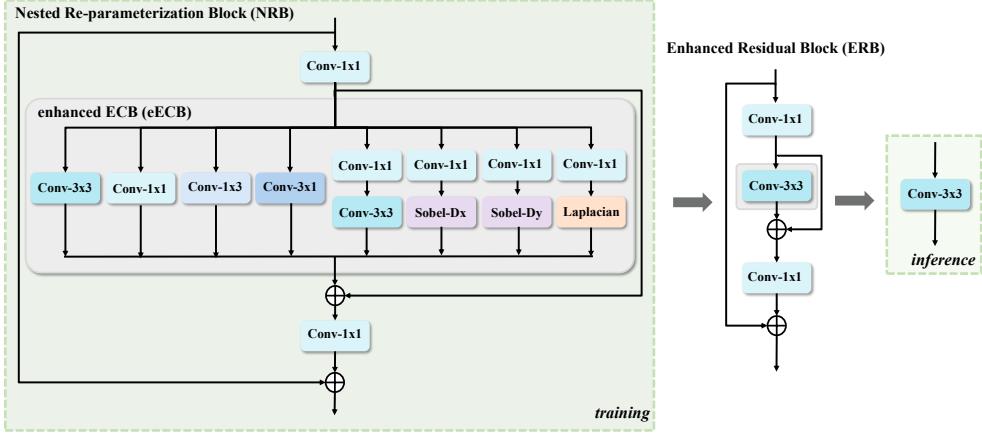


Figure 10. Detail network of the proposed Nested Re-parameterization Block (NRB), Team LeRTSR.

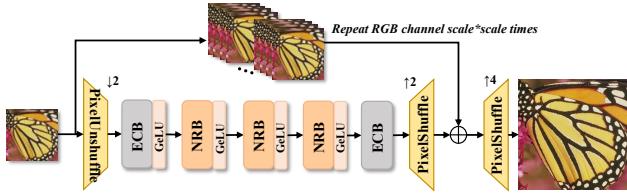


Figure 11. The framework of the proposed Anchor-based Nested UnshuffleNet for Real-time Super-Resolution (ANUNet).

We propose Anchor-based Nested UnshuffleNet for Real-time Super-Resolution (ANUNet). As shown in Fig. 11, the pixel-unshuffle technique [19] is used to reduce the resolution of the image and increase the channel dimension. This design allows for a reduction in the computational overhead of the network while preserving the constant volume of information. After an ECB [49] + GeLU module, the main module is composed of a sequence of Nested Re-parameterization Block (NRB) + GeLU activation, which serves to extract and refine features in a progressive manner. Then, an ECB layer is adopted to transfer features, followed by an upsampling layer for recovering the resolution to LR. While an anchor-based residual learning is applied to directly repeat the RGB channels 16 times in LR space to generate anchors. Finally, a pixel shuffle layer is used to reconstruct the final HR output.

Different from [13] and [49], we design a nested structure, named Nested Re-parameterization Block (NRB). Fig. 10 illustrates the proposed NRB. In the training stage, the NRB employs a nested structure, the outer structure is the ERB RepBlock in the Enhanced Residual Block (ERB) first proposed by [13], the inner structure is an enhanced Edge-oriented Convolution Block (eECB), which includes multiple branches, and can be merged into one normal convolution layer in the inference stage. Performance remains unaffected after re-parameterization in this design.

**Implementation details** We use DIV2K [1] and Flickr2K for training. To generate the compressed LR images, we use AVIF to process the above datasets with the random QP ranges between 31 and 63. Besides, standard augmentations that include all variations of flipping and rotations are also used to improve performance. Additionally, the number of feature channels is set to 28, and the scale of pixel unshuffle and pixel shuffle in the sub-branch is set to 2. After the training, we re-parameterize the model into a network structure (ECB and NRB modules) with regular 3x3 convolutions.

The model is conducted using the PyTorch framework with one NVIDIA A100 40G GPU. Specifically, the training is divided into three stages:

- Initially, the model is trained from scratch with 480x480 patches randomly cropped from high resolution (HR) images with a mini-batch size of 64. We apply a combination of Charbonnier loss [23] and FFT-based frequency loss [5] function for reconstruction. The network is trained for 1000k iterations using the Adam optimizer, with a learning rate  $1 \times 10^{-3}$  decreasing to  $1 \times 10^{-6}$  through the cosine scheduler.

- In the second stage, the model is initialized with the pre-trained weights from the first stage on the same training data as stage 1. Inspired by [6], the auxiliary loss and high-frequency loss are added to our training. Instead of the downsampling bicubic operator used in [6], Lanczos is applied to maintain consistency with the downsampling method in AVIF. The network parameters are optimized for 1000k iterations with the MultiStepLR scheduler, where the initial learning rate is set to  $5 \times 10^{-4}$  and halved at 200k, 400k, 800k-iteration.

- The model is fine-tuned using the L2 loss and FFT loss. The other settings are the same as in stage 2. The network is trained for 1000k iterations in this stage.

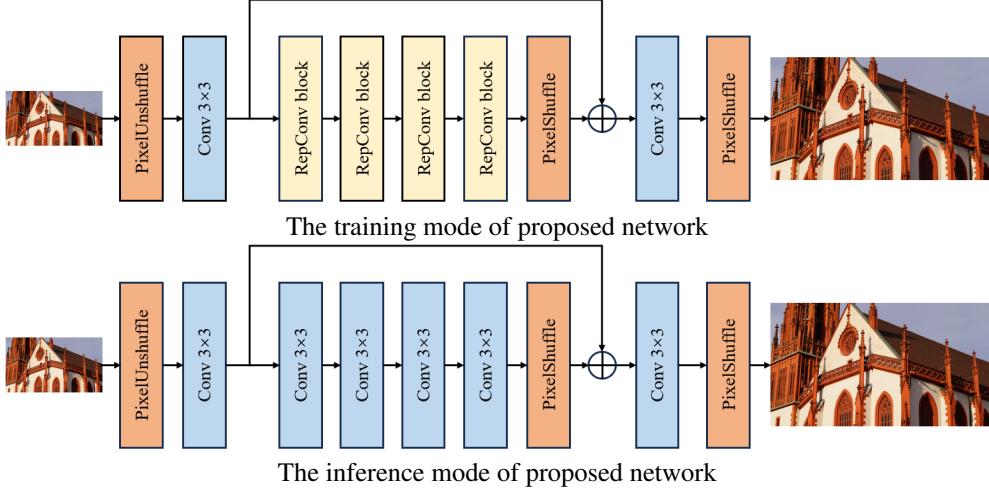


Figure 12. Illustration of proposed RESR (Team FZUQXT).

### 3.7. RESR: Reparameterized and Edge-oriented Network for Real- Time Image Super- Resolution

**Team FZUQXT**

Xintao Qiu<sup>1</sup>, Yuanbo Zhou<sup>1</sup>, Kongxian Wu<sup>1</sup>, Xinwei Dai<sup>1</sup>, Hui Tang<sup>1</sup>, Wei Deng<sup>2</sup>, Qingquan Gao<sup>1</sup>, Tong Tong<sup>1</sup>

<sup>1</sup> Fuzhou University

<sup>2</sup> Imperial Vision Technology

We propose a real-time image super-resolution based on re-parameterization and edge extraction. We use pixel unshuffle to reduce the image resolution and increase the channel dimension. This design reduces the computational cost of the network while keeping the amount of information constant. Meanwhile, we propose a reparameterized image edge extraction block that extracts features in parallel through multiple paths in the training phase, including 3x3 and 1x1 convolution for channel expansion and compression, as well as sobel and laplacian filters for acquiring information about image edges and textures.

In the inference stage, multiple operations can be combined into a 3x3 convolution. The performance of 3x3 convolution is improved without introducing any extra cost.

**Efficiency metrics** Considering the challenge input image, the model has 7.0171 GMACs, 14.0341 GFLOPs and the runtime is 1.64ms (using FP16).

**Implementation details** The datasets we used include the DIV2K training set (800 images) and the Flickr2K training set (2650 images). To increase the speed of IO, we

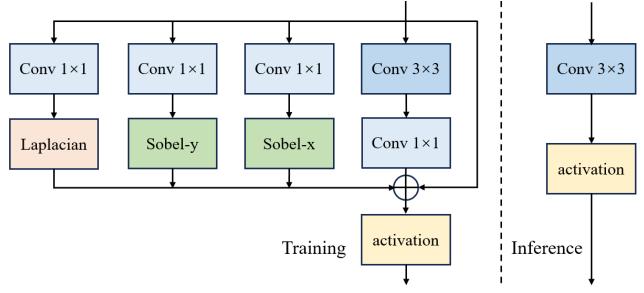


Figure 13. The architecture of RepConv block (Team FZUQXT).

split the original HR (high resolution) and LR (low resolution) images into multiple corresponding 600x600 and 150x150 patches. We randomly flipped these patches by flipping them horizontally, vertically and rotating them by 90-degrees to augment the data.

We use PyTorch and a RTX 3090 GPU (24GB). The models are optimized using Adam with Cosine Warmup. The total duration of the training process is  $\approx 48$ hrs.

In the first training stage, we train our model from scratch. The LR patches cropped from LR images with 128x128 image size and 64 mini-batch. The Adam optimizer uses a 0.00005 learning rate. The cosine warm-up scheduler sets a 0.1 percentage warmup ratio. The total number of epochs in this stage is set to 800.

In the second stage, we initialize the model with the weights trained in the previous stage. In this step, the initial learning rate is set as 0.0001. The cosine warm-up scheduler is set with a 0.1 percentage warm-up ratio. The total number of epochs is set to 200 epochs.

### 3.8. Real Time Swift Parameter-free Attention Network for 4x Image Super-Resolution

*Teams XiaomiMM C3*

Bingnan Han, Hongyuan Yu, Zhuoyuan Wu, Cheng Wan, Yaqing Liu, Haodong Yu, Jizhe Li, Zhijuan Huang, Yuan Huang, Yajun Zou, Xianyu Guan, Qi Jia, Heng Zhang, Xuanwu Yin, Kunlong Zuo

<sup>1</sup> Multimedia Department, Xiaomi Inc.

<sup>2</sup> Georgia Institute of Technology

<sup>3</sup> Dalian university of technology

**Real Time Swift Parameter-free Attention Network for 4x Image Super-Resolution** We propose a convolutional neural network combining swift parameter-free attention block (SPAB) for image SR, the suggested model has very few parameters and fast processing speed for 4x image super resolution.

As shown in Fig. 15, SPAN consists of 2 consecutive SPABs and each SPAB block extracts progressively higher-level features sequentially through three convolutional layers with  $C'$ -channeled  $H' \times W'$ -sized kernels (In our model, we choose  $H' = W' = 3$ ). The extracted features  $H_i$  are then added with a residual connection from the input of SPAB, forming the pre-attention feature map  $U_i$  for that block. The features extracted by the convolutional layers are passed through an activation function  $\sigma_a(\cdot)$  that is symmetric about the origin to obtain the attention map  $V_i$ . The feature map and attention map are element-wise multiplied to produce the final output  $O_i = U_i \odot V_i$  of the SPAB block, where  $\odot$  denotes element-wise multiplication. We use  $W_i^{(j)} \in R^{C' \times H' \times W'}$  to represent the kernel of the  $j$ -th convolutional layer of the  $i$ -th SPAB block and  $\sigma$  to represent the activation function following the convolutional layer. Then the SPAB block can be expressed as:

$$\begin{aligned} O_i &= F_{W_i}^{(i)}(O_{i-1}) = U_i \odot V_i, \\ U_i &= O_{i-1} \oplus H_i, \quad V_i = \sigma_a(H_i), \\ H_i &= F_{c, W_i}^{(i)}(O_{i-1}), \\ &= W_i^{(3)} \otimes \sigma(W_i^{(2)} \otimes \sigma(W_i^{(1)} \otimes O_{i-1})), \end{aligned} \quad (2)$$

where  $\oplus$  and  $\otimes$  represent the element-wise sum between extracted features and residual connections, and the convolution operation, respectively.  $F_{W_i}^{(i)}$  and  $F_{c, W_i}^{(i)}$  are the function representing the  $i$ -th SPAB and the function representing the 3 convolution layers of  $i$ -th SPAB with parameters  $W_i = (W_i^{(1)}, W_i^{(2)}, W_i^{(3)})$ , respectively.  $O_0 = \sigma(W_0 \otimes I_{LR})$  is a  $C'$ -channeled  $H \times W$  feature map from the  $C$ -channeled  $H \times W$ -sized low-resolution input image  $I_{LR}$  undergone a convolutional layer with  $3 \times 3$  sized kernel

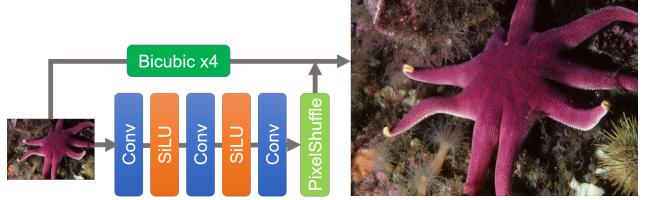


Figure 14. Network architecture of C3.

$W_0$ . This convolutional layer ensures that each SPAB has the same number of channels as input. The whole SPAN neural network can be described as

$$\begin{aligned} I_{HR} &= F(I_{LR}) = \text{PixelShuffle}[W_{f2} \otimes O)], \\ O &= \text{Concat}(O_0, O_1, O_5, W_{f1} \otimes O_6), \end{aligned} \quad (3)$$

where  $O$  is a  $4C'$ -channeled  $H \times W$ -sized feature map with multiple hierarchical features obtaining by concatenating  $O_0$  with the outputs of the first, fifth, and the convolved output of the sixth SPAB blocks by  $C'$ -channeled  $3 \times 3$ -sized kernel  $W_{f1}$ .  $O$  is processed through a  $3 \times 3$  convolutional layer to create an  $r^2C$  channel feature map of size  $H \times W$ . Then, this feature map goes through a pixel shuffle module to generate a high-resolution image of  $C$  channels and dimensions  $rH \times rW$ , where  $r$  represents the super-resolution factor. The idea of computing attention maps directly without parameters from feature extracted by convolutional layers, led to two design considerations for our neural network: the choice of activation function for computing the attention map and the use of residual connections, more details about activation function and SPAB module are in [38].

**C3 network for 4x image super-resolution** A three-layer convolutional neural network for image SR, the suggested model has very few parameters and fast processing speed for 4x image super resolution. This model has 12.39 GFLOPs and 0.024 M parameters. The model is shown in Fig. 14.

**Implementation details** Both models use HAT-L[4] 4x pre-trained network for knowledge distillation.

- **Framework:** Pytorch
- **Optimizer and Learning Rate:** We implement the network with PyTorch (BasicSR framework). The optimizer is Adam with learning rate as  $10^{-4}$ .
- **GPU:** RTX A100
- **Datasets:** We randomly collect the videos from the Internet, and randomly compress them with different QP.
- **Training Time:** We initially utilize L1 loss along with Grad loss for the first step training with 500000 iterations, then for the second step training, we use MSE loss combined with Grad loss with 250000 iterations.

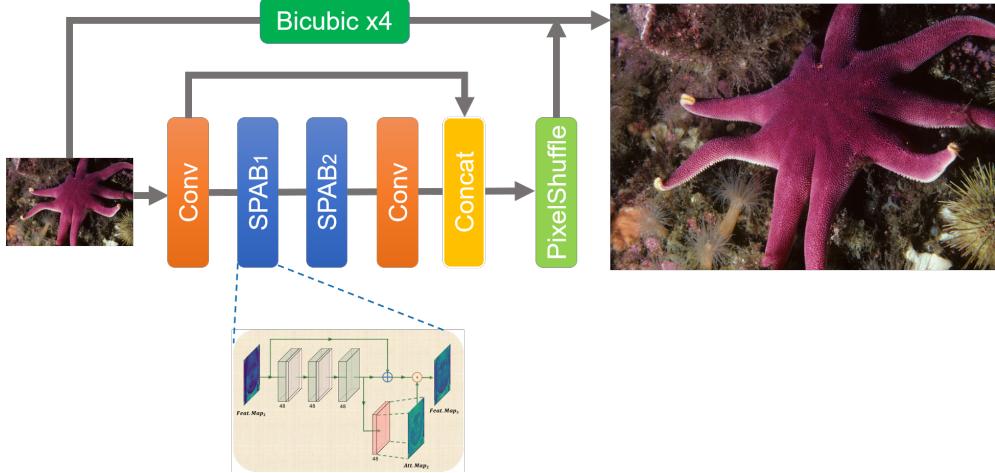


Figure 15. Network architecture of SPANR proposed by Team XiaomiMM.

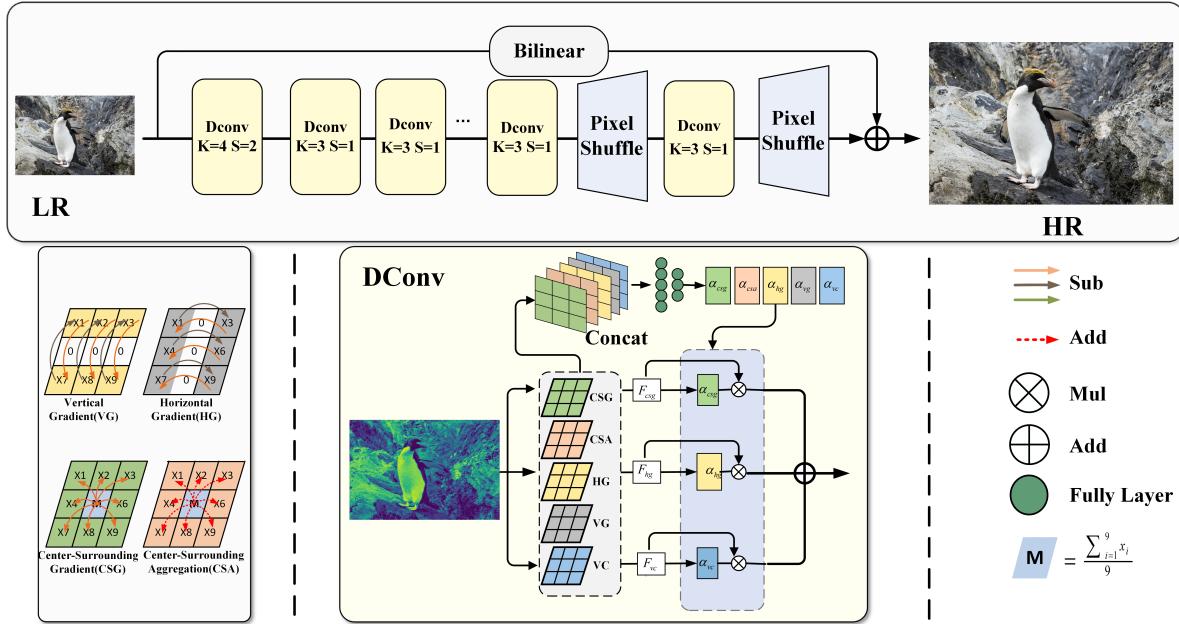


Figure 16. Diagram of the framework proposed by Team USTC Noah. The method utilizes a single DecoupleConv (DConv) with a kernel size of 4 and a stride of 2 to form the feature mapping layer. Concurrently, we construct the feature learning layer using three DConvs, each with a kernel size of 3 and a stride of 1. The resolution of the features is altered through the application of Pixel shuffle.

### 3.9. Efficient Real-Time Image Super-Resolution Via Decouple Convolution

*Team USTC Noah Terminal Vision*

Long Peng<sup>1</sup>, Jiaming Guo<sup>2</sup>, Xin Di<sup>1</sup>, Bohao Liao<sup>1</sup>,  
Zhibo Du<sup>1</sup>, Peize Xia<sup>1</sup>, Renjing Pei<sup>2</sup>, Yang Wang<sup>1</sup>, Yang  
Cao<sup>1</sup>, Zhengjun Zha<sup>1</sup>

<sup>1</sup> University of Science and Technology of China

<sup>2</sup> Huawei Noah's Ark Lab

To enhance the network's perception of gradients and contrast, we have refined the existing vanilla convolution unit by performing feature decoupling within local regions. We innovatively introduce gradient (sub) operators and aggregation (add) operators to convolution to capture detail and contrast relevant properties. Specifically, we have introduced differential operations into the convolutional process to preemptively capture horizontal, vertical, and central-surrounding directions. Furthermore, we have incorporated an aggregation (add) operation into the convolution to boost the network's sensitivity to statistical features. The method

is shown in Fig. 16.

We initially applied the DecoupleConv (with kernel=4 and stride=2) to reduce the spatial resolution while simultaneously increasing the number of channels. Subsequently, we employed four decoupled convolutions with reparameterization, which we designed for feature learning. We then utilized pixel shuffle on the features to upscale the image resolution to its original low resolution (LR) size. Following this, a single decoupled convolution with reparameterization was used for feature map-ping. Finally, another pixel shuffle operation was applied to achieve a 4x super-resolution result.

**Implementation details** We utilized solely the DIV2K dataset and applied the official compression methods to compress the images at various levels, specifically at 31, 39, 47, 55, and 63 compression levels, amounting to a total of five different degrees of compression.

**Training:** We utilized the Adam optimizer with an initial learning rate of 5e-4, performing a total of 1e7 iterations. We employed the stepDecayLR learning rate strategy, which involves a decay every 2e6 iterations with a decay factor of 2. On each card, we set the batch size to 32, resulting in a cumulative batch size of 32\*8 across all cards. The training was conducted over approximately 7 days, distributed across 8 V100 GPUs.

**Inference:** Prior to inference, it is necessary to perform an equivalent transformation of the parameters.

### 3.10. A lightweight Super-resolution Algorithm Based on Re-Parameterization

*Teams BasicVision, CMVG, IVP*

Min Yan<sup>1</sup>, Xin Liu<sup>1</sup>, Qian Wang<sup>1</sup>, Xiaoqian Ye<sup>1</sup>, Zhan Du<sup>1</sup>, Tiansen Zhang<sup>2</sup>

<sup>1</sup> China Mobile Research Institute

<sup>2</sup> Min Zu University of China

**A lightweight Super-resolution Algorithm Based on Re-Parameterization** We propose an efficient super-resolution network, which contains four convolutions and an unshuffle block. First, the network uses a convolutional operation for feature extraction. Then, it utilizes two re-parameterization modules to extract edge and detailed information. The re-parameterization module increases the number of parameters during training, but it is replaced by a single convolution to reduce computational complexity and memory usage during testing. The re-parameterization module we use can extract more edge and detailed information. Subsequently, another convolution operation is used to increase the number of channels to 48, which facilitates the subsequent four-fold super-resolution.

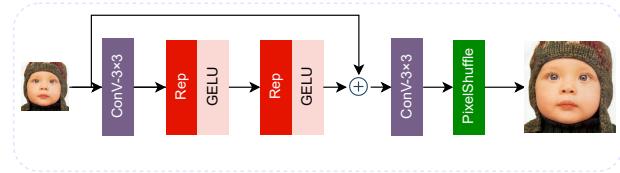


Figure 17. Overall framework of BasicVision.

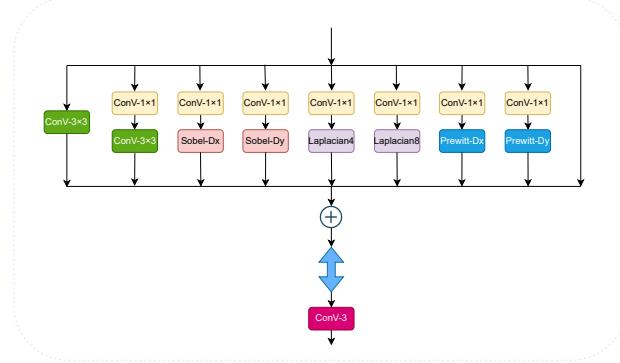


Figure 18. The re-parameterization module of BasicVision.

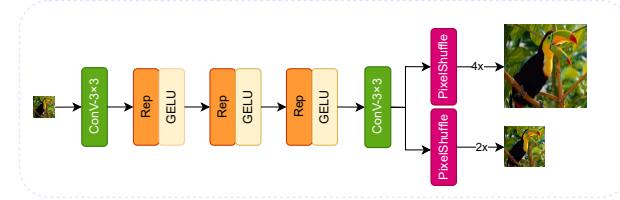


Figure 19. The framework of RDEN (IVP).

Finally, we use an unshuffle block to make the channel-to-space transition. The whole network is shown in Figure 17, and the convolutional layers in the middle(red) are two re-parameterization modules. The re-parameterization module we used is shown in Figure 18.

The local frequency loss (FFL) based on Fast Fourier Transform (FFT) [20] enables the model to dynamically prioritize challenging frequency components while diminishing the influence of easily synthesizable ones. This optimization objective supplements current spatial losses and effectively guards against the degradation of crucial frequency details caused by inherent biases in neural networks. We use the following FFT loss in our training:

$$L_{FFT} = \|FFT(X_S^{SR}) - FFT(X_S^{HR})\| \quad (4)$$

Drawing inspiration from SPSR [29], we propose a gradient loss that aids the model in accurately evaluating the local sharpness intensity of images. We use gradient loss in our training, which is represented as follows:

$$L_{GM} = \|GM(X_S^{SR}) - GM(X_S^{HR})\| \quad (5)$$

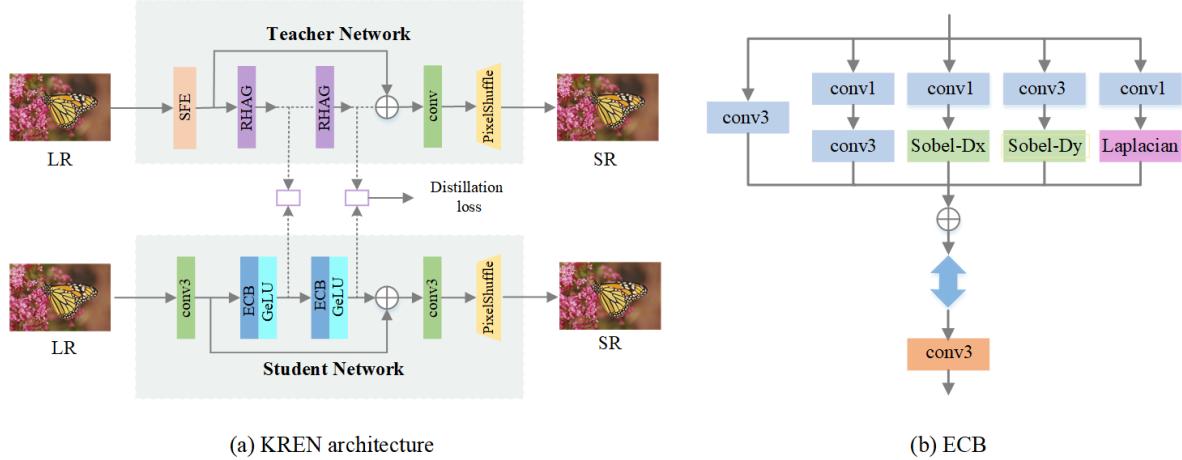


Figure 20. Overall architecture of KREN proposed by CMVG.

The overall loss for training the network is defined as:

$$L_S = \alpha \|X_S^{SR4} - X_S^{HR4}\| + \gamma L_{GM} + \delta L_{FFT} \quad (6)$$

**Real-Time Super-Resolution with auxiliary loss** The network we used is shown in Figure 19, which contains three reparameterized modules, and an auxiliary head with upscale factor 2. We use the ECB [48] model as the re-parameterized module which can achieve competitive performance without computation overhead. Alongside the 4x super-resolution task, we introduce a 2x upsampling head for the 2x SR task. This additional task offers multiple benefits: it functions as a form of simulated annealing, allowing for potential escape from local minima; it serves as a prior, enhancing the delineation of our primary task. The loss associated with the 2x supervision is represented as follows:

$$L_{X2} = \|X_S^{SR2} - X_S^{HR2}\| \quad (7)$$

where the  $X_S^{SRx2}$  denotes the output from 2x upsampling head, and  $X_S^{HRx2}$  are corresponding 2x HR image. Note that we cut the 2x super resolution model off in the testing phase, and the network consists of only five convolutions and one 4x upsampling head.

**Jointly supervision knowledge distillation network for efficient super-resolution** We propose a efficient super-resolution network named KREN based on knowledge distillation and re-parameterization, as shown in Figure 20.

The KREN model is composed of a teacher network and student network, we use the superior SR model HAT [4] as teacher network. The distillation training provides additional effective supervision information for student training, and enhances the performance and generalization ability of student network. The student network is composed

Methods	Time [ms]	Params [M]	FLOPs [G]	Acts [M]	GPU Mem [M]
IMDN [18]	23.508	0.894	58.430	154.141	707.767
RFDN [27]	18.569	0.433	27.046	112.034	791.928
RLFN [22]	12.019	0.317	19.674	80.045	470.753
DIP [45]	10.049	0.243	14.886	72.9672	497.287

Table 1. The lightweight metrics study by Teams BasicVision, CMVG, IVP. The “Time” denotes the average inference time. The “Params” is the total number of parameters. The “FLOPs” and “Acts” are calculated on 256x256 images. The “GPU Mem” represents the GPU memory during the inference.

of two convolution layers and two re-parameterization [49] blocks ECB. The ECB block with complex structure is used in training phase, while it can be merged into a 3\*3 convolution layer for speeding up inference speed during the inference phase. The re-parameterization strategy can effectively improve the feature diversity and boost the feature extraction ability of SR model. In addition, we propose a jointly supervision loss that consists the focal frequency loss(FFL) [21], gradient map loss (GM) [30] , distillation loss and L1 loss. We extract features from the 1st and 3rd blocks of the teacher model, and features from each ECB block to calculate the distillation loss. The constraints on gradients and frequency domain helps super-resolved high quality images. We also propose a multi-stage progressive training strategy to gradually improves the reconstruction quality. The number of feature maps in student network is set to 14.

**Implementation details** We train our model on DIV2K[1], Flickr2K[37] and GTA[32] datasets, and utilize multi-stage training based on Pytorch on NVIDIA V100. The patch size in each training stage is selected from [256,384,512,640]. The mini-batch size is set to 64, and MSE, GM loss[29], and FFT loss[20] are used as

target loss functions. Each stage except for the first stage is fine-tuned based on the result of the previous stage, training for 500 epochs utilizing the Adam algorithm, beginning with a learning rate of  $5 \times 10^{-4}$  and gradually decreasing to  $5 \times 10^{-5}$  following the cosine scheduler.

For the distillation approach (KREN) the training details are described as follows:

**Stage1.** Training teacher network. The teacher network is trained from scratch with teacher loss.

**Stage2.** Training student network. Firstly, we fix the teacher network and pre-train a 2x network to initialize student network. Then we use the jointly supervision loss to train student network. The initial learning rate is set to  $5e-4$  and halved at every 50 epochs and the total number of epochs is 500. The batch size and patch size are set to 64 and 256 separately.

**Stage3.** Fine-tune student network. (1) The student model is initialized from Stage2 and trained with the same settings as Stage2, especially the loss function is only MSE loss. (2) The student model is initialized from the previous step and fine-tuned by MSE loss further, it is worth that the patch size is set to 512. Other parameter settings are not changed.

### 3.11. Enhancing RTSR with ETDS and Edge-oriented Convolutional Blocks.

*Team MegastudyEdu Vision AI*

*Jae-Hyeon Lee, Ui-Jin Choi*

*MegastudyEdu Vision AI*

We introduce a method that leverages the Efficient Transformation and Dual Stream Network (ETDS) [3] conjugated with a Feature-Enhanced Module and an Edge-oriented Convolution Block (ECB) [48].

Our model is based on the Efficient Transformation and Dual Stream Network (ETDS) [3], incorporating a Feature-Enhanced Module inspired by Structure-Preserving Super Resolution with Gradient Guidance (SPSR)[29] and an Edge-oriented Convolution Block (ECB) proposed in ECBSR[48]. This design utilizes the equivalent transformation to convert time-consuming operators into time-friendly operations, alongside a dual stream network structure to reduce redundant parameters.

The architecture of ETDS[3] comprises Dual Stream network to alleviate redundant parameters, as follows:

$$\begin{bmatrix} K_b & K_{r2b} \\ K_{b2r} & K_r \end{bmatrix} \quad (1)$$

Where,  $K_b$  (backbone branch) extracts high-frequency information, while  $K_r$  (residual branch) processes low-frequency information. In our approach, ECB block is applied to  $K_b$  to enhance efficiency, and  $K_{r2b}$  and  $K_r$  consist of 3x3 convolutions. Inspired by SPSR[29], we add

Model	PSNR	# Params. (M)	FLOPs (G)	Runtime (ms)
ETDS [3]	22.844	0.0394	20.342	5.561
Our model	22.912	0.0401	20.677	5.941

Table 6. Ablation study by Team MegastudyEdu.

to restore information from images degraded by compression and downsampling algorithms. We extract gradient information from the input Low-Resolution (LR) images and then enhance the input feature map through a Feature-Enhanced Module. During inference, the Feature-Enhanced Module operates according to

$$z = \begin{bmatrix} W_1 \otimes x + b_1 \\ W_2 \otimes y + b_2 \end{bmatrix} = \begin{bmatrix} W_1 & O \\ O & W_2 \end{bmatrix} \otimes \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad (2)$$

transforming into a concatenate-convolution structure, with  $K_b$  re-parameterized as a 3x3 convolution. Ultimately, all parameters are restructured through equivalent transformation, forming the comprehensive architecture of our model.

To confirm that our solution demonstrates superior performance over previous methods, we conducted a comparison between ETDS and our model. At AIS2024 CVPR, during the Validation phase for Real-Time Compressed Image Super-Resolution, it was observed that ETDS scored 22.844, whereas our proposed model scored 22.912, indicating an improvement in performance.

Our method is trained on DIV2K[1] and Flickr2K datasets, with images processed using AVIF compression with Quality Factor (QF) coefficients ranging from 31 to 63, and scaled by a factor of 4 via Lanczos interpolation. During training we use data augmentation techniques:random cropping to 64x64, random flipping, and random rotation.

ETDS [3] architecture is adapted with ECB [48] to enhance edge detail recovery in high-frequency gradients, while the Feature-Enhanced Module, aids in restoring information lost through compression and downsampling.

#### Implementation details

- **Framework:** PyTorch 2.1.1, PyTorch Lightning
- **Optimizer and Learning Rate:** We employed Adam optimizer with parameters  $\beta_1 = 0.9$  and  $\beta_2 = 0.999$ . The training spanned 100 epochs with an initial learning rate set to 0.0001, halved at the 50th epoch.
- **GPU:** NVIDIA A100 (80GB)
- **Training Time:** The model trained for 24 hours.
- **Training Strategies:** We trained the model using all AVIF images generated within the quality factor range of 31 to 63. This entailed training on a total of 110,400 images comprising 800 from DIV2K and 2,650 from Flickr2K, each at 32 quality factors.

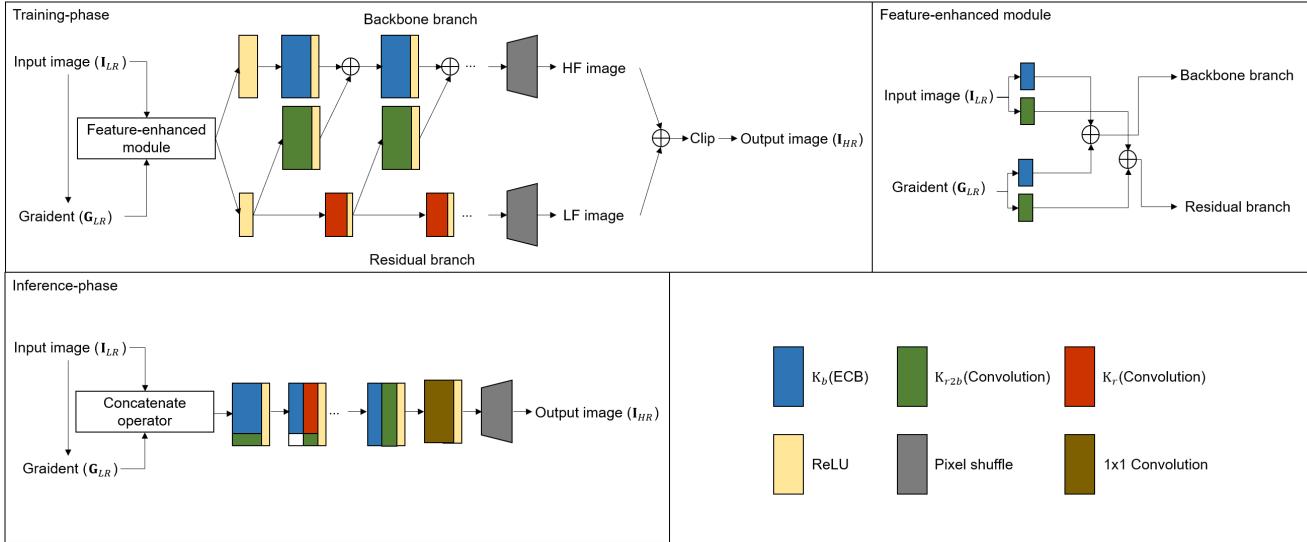


Figure 21. Overview of ETDS Network, conjugated with a Feature-Enhanced Module and an ECB Block – Team MegastudyEdu.

- **Efficiency Optimization Strategies:**

- **Dual Stream Network Architecture:** Utilizing ETDS [3] reduces redundant parameters by separating the processing of high-frequency and low-frequency information. This branch enables more efficient learning and reduces computational overhead.
- **Feature-Enhanced Module with Gradient Guidance:** We incorporated a Feature-Enhanced Module to leverage gradient information from low-resolution inputs. This approach effectively restores high-frequency details lost during compression and downsampling, enhancing model performance without significantly increasing computational demand.

### 3.12. Unshuffle, Re-parameterization, and Pointwise Network (URPNet)

**Team 402Lab**

Hyeon-Cheol Moon<sup>1,2</sup>, Tae-Hyun Jeong<sup>1</sup>, Yoonmo Yang<sup>1</sup>, Jae-Gon Kim<sup>2</sup>, Jinwoo Jeong<sup>1</sup>, Sungjei Kim<sup>1</sup>

<sup>1</sup> Korea Electronics Technology Institute (KETI)

<sup>2</sup> Korea Aerospace University (KAU)

We propose an Unshuffle, Re-parameterization, and Pointwise Network (URPNet) that can achieve higher accuracy at a faster speed compared to previous real-time SR models for 4K images. We applied a pixel unshuffle to the input image to reduce the resolution, and applied the 1x1 pointwise convolution to only the last layer, instead of applying a re-parameterized convolution (RepConv) to all existing convolutions.

We also applied curriculum learning [2] to efficiently learn lightweight models. Since the larger the Quantiza-

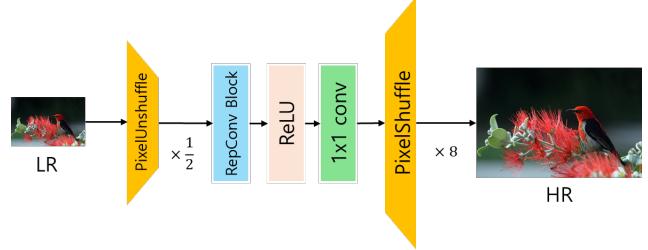


Figure 22. Proposed URPNet structure.

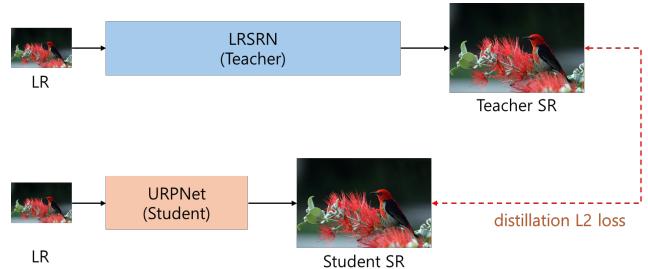


Figure 23. Proposed distillation loss on the fine-tuning stage of URPNet, Team 402Lab.

tion Parameter (QP), the larger the compression artifacts, the worse the performance will be if the lightweight model is trained on high QP data from the beginning. Therefore, we divided the training data into easy (QP 31), medium (QP 39, QP 47), and hard (QP 55, QP 63) sets according to the training difficulty.

Additionally, we applied knowledge distillation (KD) during the fine-tuning stage to achieve higher PSNR than using conventional training. To apply KD, the teacher

model is trained from the scratch on a high-resolution dataset. We train with the L2 loss of the output images of each network between teacher and student [17, 31].

**Efficiency metrics** Considering the challenge input, the model has 0.15K MACs per pixel (4K), a total number of 1.2483 GFlops, and a runtime of 0.62ms in RTX 3090 GPUs

### Implementation details

- **Framework:** PyTorch 1.13 version
- **Optimizer and Learning Rate:** Adam optimizer with a cosine warm-up.  
Initial learning rate: 5e-4 (scratch), 1e-4 (fine-tuning)
- **GPU:** single RTX3090/24GB, 3.2GB (training memory)
- **Datasets:**
  1. DIV2K : We use the DIV2K training dataset (800 images) for scratch training step.
  2. FTCombined : We use a combined dataset for fine-tuning stage, which includes the DIV2K train set (full 800), Flickr train set (2650 full), DIV8K (first 200 samples), and LSDIR (first 1000). Before the training phase, the training data is pre-processed by center cropping it to a resolution of 2040 x 1080. To generate low-resolution images, we degrade the center cropped images with Lanczos downsampling and AVIF compression. For both training stages, we used random cropping, rotation 90, horizontal flip and vertical flip augmentation.
- **Training Time:** 24 hours with single RTX 3090GPUs
- **Training Strategies:**

1. Scratch train step: In the first step, our model was trained from scratch. The LR patches were cropped from LR images with 8 mini-batch 96x96 sizes. The Adam optimizer was used with a 0.0005 learning rate during scratch training. The cosine warm-up scheduler was used. The total number of epochs was set to 500. We use  $l_1$  loss.
2. Fine-tuning step: In the second step, the model was initialized with the weights trained in the first step. To improve the accuracy, we used  $l_2$  and the distillation loss. Fine-tuning with  $l_2$  and distillation loss improves the peak signal-to-noise ratio (PSNR) value by  $0.02 \sim 0.03$  dB. In this step, the initial learning rate was set as 0.0001, and the Adam optimizer was used along with a cosine warm-up. The total epoch was set to 50 epochs.

## 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, 9, 14, 15
- [2] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009. 16
- [3] Jiahao Chao, Zhou Zhou, Hongfan Gao, Jiali Gong, Zhengfeng Yang, Zhenbing Zeng, and Lydia Dehbi. Equivalent transformation and dual stream network construction for mobile image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14102–14111, June 2023. 15, 16
- [4] Xiangyu Chen, Xintao Wang, Jiantao Zhou, Yu Qiao, and Chao Dong. Activating more pixels in image super-resolution transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22367–22377, 2023. 11, 14
- [5] Sung-Jin Cho, Seo-Won Ji, Jun-Pyo Hong, Seung-Won Jung, and Sung-Jea Ko. Rethinking coarse-to-fine approach in single image deblurring. In *ICCV*, 2021. 6, 9
- [6] Marcos V Conde, Ui-Jin Choi, Maxime Burchi, and Radu Timofte. Swinv2r: Swinv2 transformer for compressed image super-resolution and restoration. In *European Conference on Computer Vision*, pages 669–687. Springer, 2022. 2, 9
- [7] Marcos V. Conde, Zhijun Lei, Wen Li, Ioannis Katsavounidis, Radu Timofte, et al. Real-time 4k super-resolution of compressed AVIF images. AIS 2024 challenge survey. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024. 3
- [8] Marcos V. Conde, Saman Zadtootaghaj, Nabajeet Barman, Radu Timofte, et al. AIS 2024 challenge on video quality assessment of user-generated content: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024. 3
- [9] Marcos V Conde, Eduard Zamfir, Radu Timofte, Daniel Motilla, Cen Liu, Zexin Zhang, Yunbo Peng, Yue Lin, Jiaming Guo, Xueyi Zou, 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*, pages 1495–1521, 2023. 2, 3, 7
- [10] Weijian Deng, Hongjie Yuan, Lunhui Deng, and Zengtong Lu. Reparameterized residual feature network for lightweight image super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1712–1721, 2023. 4
- [11] 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
- [12] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *CVPR*, 2021. 4
- [13] Zongcai Du, Ding Liu, Jie Liu, Jie Tang, Gangshan Wu, and Lean Fu. Fast and memory-efficient network towards efficient image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 853–862, 2022. 8, 9

- [14] Stephanie Fu, Mark Hamilton, Laura E. Brandt, Axel Feldmann, Zhoutong Zhang, and William T. Freeman. Featup: A model-agnostic framework for features at any resolution. In *ICLR*, 2024. 6
- [15] 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*, pages 1746–1755, 2023. 7
- [16] Mark Hamilton, Evan Shelhamer, and William T. Freeman. It is likely that your loss should be a likelihood. *CoRR*, abs/2007.06059, 2020. 6
- [17] 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. 17
- [18] 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, 14
- [19] 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. 9
- [20] Liming Jiang, Bo Dai, Wayne Wu, and Chen Change Loy. Focal frequency loss for image reconstruction and synthesis. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 13919–13929, 2021. 13, 14
- [21] Liming Jiang, Bo Dai, Wayne Wu, and Chen Change Loy. Focal frequency loss for image reconstruction and synthesis. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 13919–13929, 2021. 14
- [22] 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, 14
- [23] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Fast and accurate image super-resolution with deep laplacian pyramid networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(11):2599–2613, 2018. 9
- [24] 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. 3
- [25] 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
- [26] 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
- [27] 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. 14
- [28] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *ICLR*, 2017. 6
- [29] Cheng Ma, Yongming Rao, Yean Cheng, Ce Chen, Jiwen Lu, and Jie Zhou. Structure-preserving super resolution with gradient guidance. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7769–7778, 2020. 13, 14, 15
- [30] Cheng Ma, Yongming Rao, Yean Cheng, Ce Chen, Jiwen Lu, and Jie Zhou. Structure-preserving super resolution with gradient guidance. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7769–7778, 2020. 14
- [31] Hyeon-Cheol Moon, Jae-Gon Kim, Jinwoo Jeong, and Sung-jei Kim. Feature-domain adaptive contrastive distillation for efficient single image super-resolution. *IEEE Access*, 11:131885–131896, 2023. 17
- [32] Stephan R. Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22–29, 2017*, pages 2232–2241, 2017. 14
- [33] Wenzhe Shi, Jose Caballero, Ferenc Huszar, 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 *CVPR*, 2016. 2, 6
- [34] 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
- [35] 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
- [36] Long Sun, Jiangxin Dong, Jinhui Tang, and Jinshan Pan. Spatially-adaptive feature modulation for efficient image super-resolution. In *ICCV*, 2023. 6
- [37] 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. 14
- [38] Cheng Wan, Hongyuan Yu, Zhiqi Li, Yihang Chen, Yanjun Zou, Yuqing Liu, Xuanwu Yin, and Kunlong Zuo. Swift parameter-free attention network for efficient super-resolution. *arXiv preprint arXiv:2311.12770*, 2023. 11
- [39] Ao Wang, Hui Chen, Zijia Lin, Hengjun Pu, and Guiguang Ding. Repvit: Revisiting mobile cnn from vit perspective. *arXiv preprint arXiv:2307.09283*, 2023. 8
- [40] 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
- [41] 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
- [42] Zuowen Wang, Chang Gao, Zongwei Wu, Marcos V. Conde, Radu Timofte, Shih-Chii Liu, Qinyu Chen, et al. Event-Based Eye Tracking. AIS 2024 Challenge Survey. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024. 3
- [43] Ren Yang, Radu Timofte, Xin Li, Qi Zhang, Lin Zhang, Fangleong 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 *European Conference on Computer Vision*, pages 174–202. Springer, 2022. 2
- [44] Kihwan Yoon, Ganzorig Gankhuyag, Jinman Park, Haengseon Son, and Kyoungwon Min. Casr: Efficient cascade network structure with channel aligned method for 4k real-time single image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024. 7, 8
- [45] Lei Yu, Xinpeng Li, Youwei Li, Ting Jiang, Qi Wu, Haoqiang Fan, and Shuaicheng Liu. Dipnet: Efficiency distillation and iterative pruning for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1692–1701, 2023. 14
- [46] 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
- [47] 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
- [48] 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. 4, 14, 15
- [49] Xindong Zhang, Hui Zeng, and Lei Zhang. Edge-oriented convolution block for real-time super resolution on mobile devices. In Heng Tao Shen, Yuetong 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. 9, 14
- [50] 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