NTIRE 2023 Efficient SR Challenge Factsheet
Reparameterized Residual Feature Network For Lightweight Image Super-Resolution

Weijian Deng[1], Hongjie Yuan[2]
School of Information and Communication Engineering, Communication University of China
Beijing, China
348957269@qq.com[1], homjayuan@qq.com[2]

Zengtong Lu
Smart Classroom Division, Ruijie Networks Co., Ltd.
Fuzhou Fujian, China
luztxp@gmail.com

1. Team details

- Team name

  CUC_SR

- Team leader name

  Weijian Deng[1]

- Team leader address, phone number, and email

  Team leader address: Communication University of China, Beijing, China

  Phone number: +86 13978942589

  Email: 348957269@qq.com

- Rest of the team members

  Zengtong Lu[2], Hongjie Yuan[3]

- Team website URL (if any)

  N/A

- Affiliation

  [1,3]School of Information and Communication Engineering, Communication University of China.

  [2]Smart Classroom Division, Ruijie Networks Co., Ltd.

- Affiliation of the team and/or team members with NTIRE 2023 sponsors (check the workshop website)

  N/A

- User names and entries on the NTIRE 2023 Codalab competitions (development/validation and testing phases)

  User names: laonafahaodange[1], luztxp[2], Uchiha[3]

  Development/validation phase entries: 7

  Testing phase entries: 3

- Best scoring entries of the team during development/validation phase

PSNR	SSIM	Runtime	Parameters	Extra Data
28.99	0.83	0.03	401504.00	1.00

- Link to the codes/executables of the solution(s)

  https://github.com/laonafahaodange/NTIRE2023-ESR-RepRFN

2. Method details

General method description. We proposed a Reparameterized Residual Feature Network (RepRFN), as shown in the Fig.1. Our work is inspired by RFDN [4], RLFN [3] and ECBSR [5]. We found that the overall structure of RLFN and ECBSR is similar. They both remove the operation of channel concatenation. In fact, in a rough experiment, we found that if the channel concatenation in RFDN were replaced by local residual connections and then retrained with DIV2K training set, although the number of parameters and FLOPs increased slightly (the parameter increases by about 6.7% and the FLOPs increases by about 3.2%), the overall performance was relatively similar (about 0.02dB difference in DIV2K validation set, RFDN scored higher), the inference speed could be improved by about 34% and maximum GPU memory consumed during inference could be

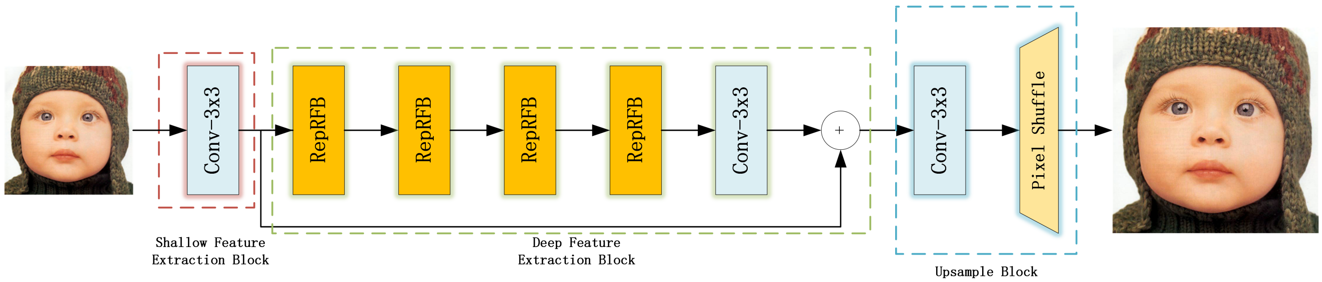

Figure 1. the structure of RepRFN

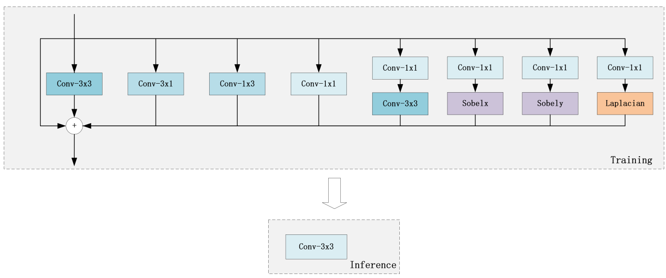

Figure 2. the structure of RepBlock

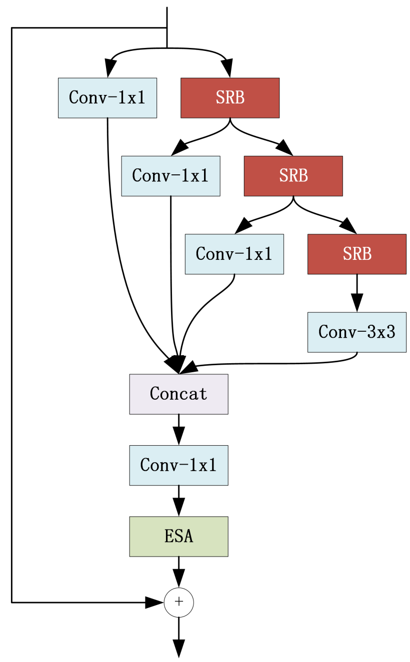

Figure 3. the structure of RFDB

reduced by about 55% . It can be considered that channel concatenation is a factor affecting the speed of inference. Therefore, we rethought RFDN, and mainly improved the following parts.

First, SRB and 1×1 convolution are replaced by 3×3 convolution, and channel concatenation is replaced by local residual connection. However, in consideration of the single receptive field of 3×3 convolution, inspired by the recent reparameterization works [1,2,5], in order to capture features of more patterns as much as possible, as shown in the Fig.2, we design a reparameterized multi-branch module called RepBlock to extract features (including Sobel branch and Laplacian branch in ECBSR) and use residual connections for features fusion. In the training stage, the multi-branch structure is used, and the model is converted into a simple plane structure through the model reparameterization to speed up inference.

In RFDN, as shown in the Fig.3, the feature is distilled by SRB and 1×1 convolution three times in each RFDB, so we replace the first three 3×3 convolution with this multi-branch structure. Considering that some 1×1 convolution in RFDN is to perform channel transformation after channel concatenation, we do not need to perform channel transformation due to residual connection, so these 1×1 convolution can be removed to further compress parameters. Inspired by RLFN, the convolution group in ESA is also reduced to one layer of convolution. The final building block structure of RepRFB is shown in the Fig.4.

In addition, a loss function based on Fourier transform is also used. Specifically, the loss of SR image and HR image after Fourier transform is calculated to guide the model to learn frequency information. The loss can be formulated as:

$$L(x, y) = L_{pix}(fft(x), fft(y)) \tag{1}$$

where L_{pix} defines pixel loss such as L2 loss, L1 loss, Charbonnier loss. $fft(\cdot)$ means Fast Fourier Transform on the image. It should be noted that we only perform Fourier transform on the scale dimension of the image.

Training strategy. In terms of training, DIV2K training set and Flickr2K data set were used, due to time and machine limitations, we did not use the provided LSDIR dataset, which should be better for the model performance after fine-tuning. The HR patches is set to 192×192, random horizontal flip, vertical flip and rotation were introduced into the data augmentation during training. The proposed RepRFN consists of 4 RepRFBs, the number of

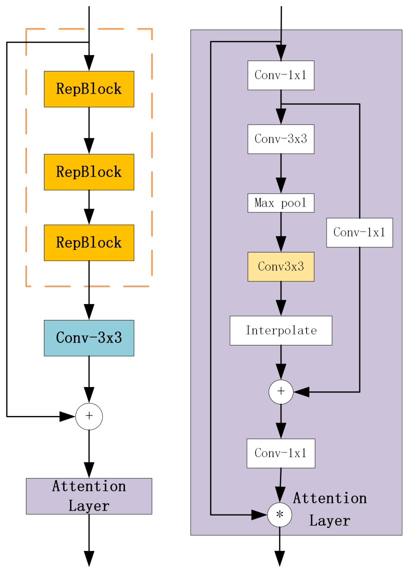

Figure 4. the structure of RepRFB

channels is set to 48. The model is trained from scratch. We used the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$, batchsize was set to 64, and the initial learning rate was set to 5×10^{-4} and halved at every 100 epochs. The total number of epochs is 1001. In the process of training, the loss function used is the combination of pixel loss and loss function based on Fourier transform. In practical application, Charbonnier loss function can avoid the problem that the results generated by L1 loss function and L2 loss function are too smooth [6], in the experiment, we also found that the Charbonnier loss is better than the L1 loss in terms of PSNR, so we choosed Charbonnier loss as L_{pix}. Finally, the loss can be formulated as:

$$L(x, y) = \lambda_1 L_{pix}(x, y) + \lambda_2 L_{pix}(fft(x), fft(y)) \quad (2)$$

where $\lambda_1 = 0.9$ and $\lambda_2 = 0.1$. The hyperparameter ϵ^2 in Charbonnier loss was set to 10^-6.

Table 1. Comparison of RFDN and RepRFN

Model	PSNR (dB)]	Val Time (ms)	Params (M)	FLOPs (G)	Acts (M)	Mem (M)	Conv
RFDN	29.04	41.34	0.433	27.10	112.03	788.13	64
RepRFN	28.99	29.75	0.402	25.23	81.88	344.51	39

As shown in the Tab.1, we compared the differences between the RepRFN and the baseline RFDN on the DIV2K validation set, which was tested at Titan Xp. It can be seen from the table that our model is lighter and faster for inference. During the testing phase, our proposed model scored 27.05.

3. Other details

- Planned submission of a solution(s) description paper at NTIRE 2023 workshop.

  We are planning to submit the solution description paper to NTIRE2023 workshop.

- Other comments

  Can the challenge provide a factsheet filling template (for example, take the baseline as an example)? Because I am the first time to participate in the competition, I don't know where the factsheet needs to be modified. Thank all organizers for their efforts!

References

[1] Xiaohan Ding, Yuchen Guo, Guiguang Ding, and Jungong Han. Acnet: Strengthening the kernel skeletons for powerful cnn via asymmetric convolution blocks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1911–1920, 2019. 2

[2] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Diverse branch block: Building a convolution as an inception-like unit. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10886–10895, 2021. 2

[3] 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. 1

[4] Jie Liu, Wenjie Zhang, Yuting Tang, Jie Tang, and Gangshan Wu. Residual feature aggregation network for image super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2359–2368, 2020. 1

[5] 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. 1, 2

[6] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on computational imaging*, 3(1):47–57, 2016. 3