Reparameterized Residual Feature Network For Lightweight Image Super-Resolution

Weijian Deng[1], Hongjie Yuan[1], Lunhui Deng[1], Zengtong Lu[2]
[1]Communication University of China, China
[2]Ruijie Networks Co., Ltd. China
{348957269, homjayuan}@qq.com, dy3000@cuc.edu.cn, luztxp@gmail.com

Abstract

In order to solve the problem of deploying super-resolution technology on resource-limited devices, this paper explores the differences in performance and efficiency between information distillation mechanism and residual learning mechanism used in lightweight super-resolution, and proposes a lightweight super-resolution network structure based on reparameterization, named RepRFN, which can effectively reduce GPU memory consumption and improve inference speed. A multi-scale feature fusion structure is designed so that the network can learn and integrate features of various scales and high-frequency edges. We rethought the redundancy existing in the overall network framework, and removed some redundant modules without affecting the overall performance as much as possible to further reduce the complexity of the model. In addition, we introduced a loss function based on Fourier transform to transform the spatial domain of the image into the frequency domain, so that the network can supervise and learn the frequency part of the image. The experimental results show that the RepRFN designed in this paper achieves relatively low complexity while ensuring certain performance, which is conducive to the deployment of Edge devices. Code is available at https://github.com/laonafahaodange/RepRFN.

1. Introduction

Super-resolution (SR) is an important branch of image reconstruction in computer vision and a hot research topic in recent years. It is widely used in the fields of medical treatment, security, image and video reconstruction, and even game image enhancement. In recent years, many SR networks based on convolutional neural network (CNN) have been proposed, indicating that CNN plays a role in promoting the development of image SR.

In 2014, Dong et al. applied convolutional neural network to image SR problem for the first time and proposed SRCNN [5]. Compared with traditional methods, SRCNN achieved good results with only three convolution layers, proving the effectiveness of deep learning in this problem. Kim et al. proposed a deeper SR network VDSR [10] by cascading multiple small size convolution to ensure the same receptive field while reducing parameters. The EDSR [17] proposed by Lim et al. further improves the depth by the residual structure and multi-scale technology, achieving better results than SRCNN. This shows that the depth of the network is an important factor affecting the quality of SR image reconstruction, and constructing a deeper network can improve the performance of SR.

However, most SR networks sacrifice the efficiency in order to improve the details of image restoration. In some

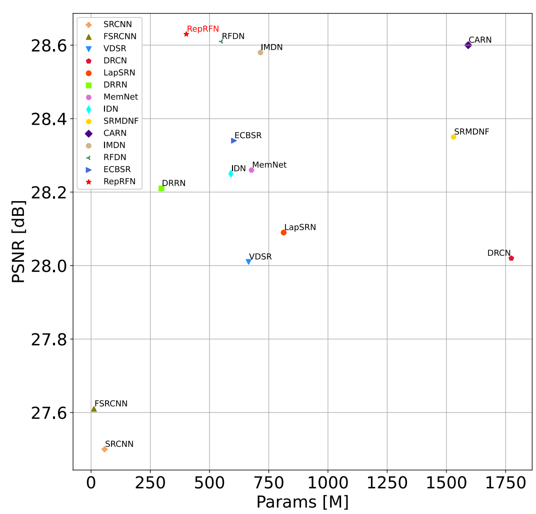


Figure 1. Comparison of PSNR and Parameters of different networks on Set14 [25] (scale=4, Y-channel in Ycbcr color space).

scenarios, real-time performance will also affect user experience. Therefore, how to efficiently extract image edge, texture, structure and other information, and balance the relationship between the performance and complexity of SR network is a key research, which determines whether the network can be deployed on devices with limited resources such as computing and storage units. To this end, we propose a novel Reparameterized Residual Feature Network, referred to as RepRFN. A multi-branch structure is designed to extract features of different receptive fields by using multiple parallel convolution kernels of different sizes, and feature fusion is realized by local residual connection. In order to extract the edge information effectively, the Sobel branch and Laplace branch in the Edge-oriented Convolution Block (ECB) [29] are introduced into the multi-branch structure. In the training stage, we regards the SR task as a multi-task learning problem of spatial domain learning and frequency domain learning. Fourier transform is introduced into the loss function to calculate the loss in the frequency domain to guide the model to recover frequency information. Experiments show that the proposed RepRFN achieves a balance in performance and efficiency.

Our contributions can be summarized as follows:

1. A multi-scale feature fusion structure based on reparameterization is proposed. Features are convolved by multiple parallel convolution of different receptive field and edge-oriented convolution modules to extract features of different modes. Residual connection is used to aggregate features, which improves the expression ability of features.

2. The structure of RFDN [18] model was reconsidered, we analyzed the redundancy of RFDN, and the 1×1 convolution used for channel transformation was removed in our network.

3. Fourier transform is introduced into the loss function, so that the model can learn the frequency information of the image in the process of supervised training, and enhance the recovery ability of frequency details.

2. Related work

The SR network SRCNN [5] has achieved impressive results, but there are some problems such as large computation. Dong et al. achieved a learning upsampling by removing interpolation upsampling, introducing transposed convolution at the end of the network, and using smaller but more convolution kernels for feature extraction. Based on these improvements, they proposed the lightweight SR network FSRCNN [6], which achieved about 17 times of acceleration compared with SRCNN. Kim et al. proposed a deep recursive convolutional network DRCN [11] by recursively invoking the feature extraction layer. DRRN [22] improves DRCN by combining recursion and residual network to achieve better performance with fewer parameters. NamhyukAhn et al. adjusted model efficiency using group convolution, they adopted a mechanism similar to recursive network to share parameters among cascade modules, and proposed a lightweight cascade residual network CARN [1]. Lai et al. proposed LapSRN [13], they removed the preprocessing step of bicubic interpolation of input, transposed convolution is used for upsampling, with low resolution (LR) image as input, feature maps are extracted through the cascade convolutional layer, then a convolution layer is used to learn the residual difference between high-resolution image and up-sampled feature image to complete first-level reconstruction, and multi-scale reconstruction is finally realized through stepwise upsampling. Hui et al. proposed an Information Distillation Network IDN [9]. The key to IDN lies in the information distillation module. Each information distillation module contains an enhancement unit and a compression unit, which can effectively extract local long and short-path features, in addition, IDN uses relatively few convolution kernels and group convolution, so the inference speed is relatively fast. On the basis of IDN, Information Multi-Distillation Network IMDN [8] constructs a cascable Information Multi-Distillation Block IMDB, which consists of distillation and selective fusion, specifically, the distillation module gradually extracted features, and the fusion module determined the importance of candidate features according to the attention mechanism and fused them. IMDN won the first prize in AIM2019 Challenge on Constrained Super-Resolution [27]. Subsequently, Liu et al. reconsidered IMDN and proposed Residual Feature Distillation Network RFDN [18], they think that the key component of both IDN and IMDN is the information distillation mechanism IDM, which explicitly divides extracted features into two parts, one retained and the other further extracted. However, the efficiency of this mechanism is still not enough to integrate the residual connection with IDM, therefore, they designed a Shallow Residual Block SRB as the main module of RFDN, so that the network can achieve lightweight and maximize the use of residual learning. Its E-RFDN won first place in AIM2020 Challenge on Efficient Super-Resolution [26].

3. Method

In view of the excellent performance and efficiency of information distillation mechanism in efficient super-resolution challenge in recent years [26, 27], we focuses on the comparison of information distillation mechanism. The residual network is widely used by researchers because of its simple structure, easy implementation and good optimization effect. Since there is no need to concatenate channels, the inference speed is faster and the operation is more efficient, and there is greater potential for deploying

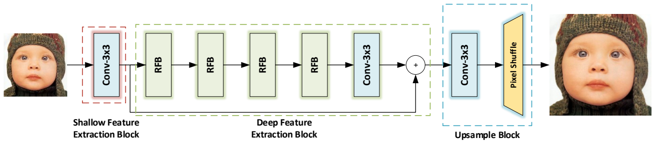
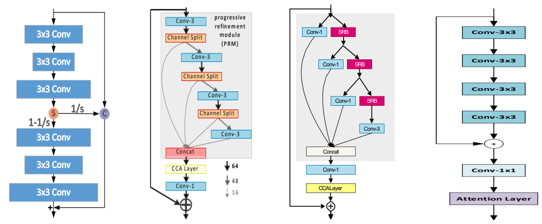


Figure 2. The Structure of Residual Feature Network.



(a) IDN-IDB (b) IMDN-IDB (c) RFDN-IDB (d) RFB

Figure 3. Structure of several information distillation mechanisms and residual learning mechanism.

Table 1. Comparison of model performance and efficiency of different structures (DIV2K [24] validation set, scale=4, RGB color space).

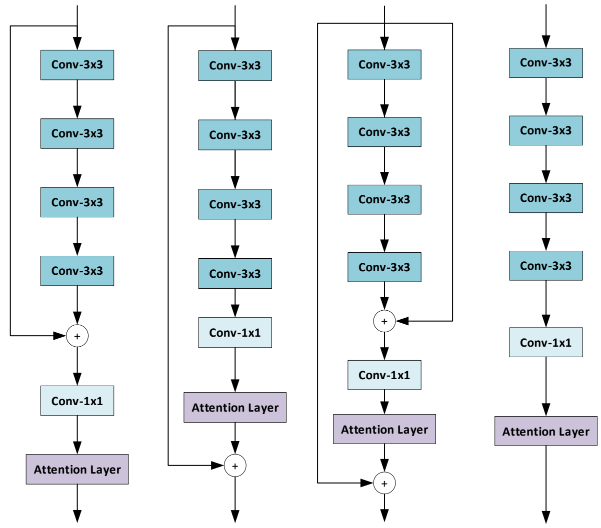
Model	PSNR (dB)	Val Time (ms)	Params (M)	FLOPs (G)	Acts (M)	Mem (M)	Conv
RFDN*	28.93	272.67	0.402	25.06	107.97	758.80	64
RFB1	28.91	179.65	0.429	25.87	94.68	345.04	51
RFB2	28.90	179.76	0.429	25.87	94.68	345.04	51
RFB3	28.91	184.60	0.429	25.87	94.68	345.04	51
Baseline	28.84	177.46	0.429	25.87	94.68	345.04	51

on Edge device with limited resources.

In Section 3.1, we propose a Residual Feature Network (RFN) for lightweight image SR. Compared with the information distillation mechanism, we observe the differences in performance and efficiency between the residual feature learning mechanism and the information distillation mechanism through experiments. In Section 3.2, we review the shortcomings of RFN, make a series of improvements to the model, and propose a multi-scale feature fusion lightweight SR network RepRFN based on Reparameterization [3, 4, 29]. In Section 3.3, we describe a loss function based on Fourier transform, which converts images from spatial domain to frequency domain, so that the model can learn frequency information in the process of training, and improve the performance of lightweight SR model.

3.1. Residual Feature Network (RFN)

The overall network structure is shown in Figure 2. The network consists of three main parts: Shallow Feature-extraction Block (SFB), Deep Feature-extraction Block (DFB) and Upsample Block (UB). The SFB is used to extract the shallow features of the input LR image and DFB carries out further nonlinear mapping on the extracted shallow features to obtain deeper feature expression, then deep feature and shallow feature are fused through residual connection to obtain fusion features. Finally, the UB performs pixel recombination on the fused features to reconstruct a SR image.



(a) RFB1 (b) RFB2 (c) RFB3 (d) baseline

Figure 4. Structure of several residual learning mechanisms.

The SFB consists of a 3×3 convolution layer, which is mainly responsible for extracting shallow feature from the input LR image. The DFB is composed of stacked Residual Feature Block (RFB), which gradually extracts shallow features and uses residual learning to integrate shallow features and deep features to improve the expression ability of features. The UB is the subpixel convolution layer, which is composed of a 3×3 convolution layer and a PixelShuffle layer [21]. The final SR image is obtained by recombining the fused features and pixel mapping. The above process can be expressed as:

$$\begin{aligned} X_{sf} &= F_{sf}(I_{lr}) \\ X_{df}^{i} &= F_{df}^{i-1}, i = 2,3,4,5 \\ X_1 &= X_{sf} \\ X_{fusion} &= conv_{3\times 3}(X_{df}^{5}) + X_{sf} \\ I_{sf} &= Up(X_{fusion}) \end{aligned} \tag{1}$$

where I_{lr} represents the input LR image, and shallow features X_{sf} are obtained after SFB F_{sf}. Then the shallow features obtained are input into the i-th DFB F_{df}^{i-1} of the stacked RFBs, and the i-th deep features X_{df}^{i} are extracted layer by layer. After 3×3 convolution, the extracted deep feature is fused with the shallow feature to obtain the fused features X_{fusion}. Finally, the fused features are input into the UB $Up(\cdot)$ to obtain the reconstructed SR image I_{sr}.

Residual Feature Block The key of the Residual Feature Block lies in the residual feature learning mechanism.

Different from the information distillation mechanism, the information distillation mechanism divides the input feature into two parts along the channel dimension. One part is retained and the other part is input to the next information distillation module for further feature extraction. After several distillation steps, the feature fusion is completed by concatenating along the channel dimension, so as to realize the fusion of distillation information. However, the residual feature learning mechanism does not split the features along the channel dimension, but directly inputs the extracted features into the next part, and only adds and merges the extracted deep features and shallow features in each module, which alleviates the problems of large GPU memory consumption and increased inference time caused by the channel split and concatenation operation. Figure 3 shows several information distillation module. Figure 3a, Figure 3b and Figure 3c respectively represent the structure of Information Distillation Block (IDB) used in IDN [9], IMDN [8] and RFDN [18]. The RFB (Figure 3d) used in this section draws on the structure of RFDN-IDB. Specifically, the difference lies in that the input features are no longer operated by channel split, but directly input to the next convolution layer, and the information fusion mechanism is replaced by residual fusion.

Assume $f^i_{k=n}$ represents the output feature of the i-th $n \times n$ convolution in the RFB, $conv^i_{n\times n}$ represents the operation function of the i-th $n \times n$ convolution layer that has been continuously stacked, f_{fusion} represents the fusion feature generated by residual connection between input feature x_{in} and intermediate feature $f^4_{k=3}$, $Attention$ represents the attention mechanism used by the module, and x_{out} represents the output feature. The calculation process of this module can be expressed as:

$$\begin{aligned} f^4_{k=3} &= conv^4_{3\times3}(x_{in}) \\ f_{fusion} &= f^4_{k=3} + x_{in} \\ f^1_{k=1} &= conv^1_{1\times1}(f_{fusion}) \\ x_{out} &= Attention(f^1_{k=1}) \end{aligned} \tag{2}$$

We explored the performance and efficiency differences between the information distillation mechanism and the residual learning mechanism. RFDN [18] was used as the representative of information distillation mechanism. It is noted that global residual connections are used in RFDBs, and the impact of different residual connections on performance differences was also explored. As shown in Figure 4, RFB1, RFB2 and RFB3 are defined to represent different residual connection modes: local residual connection, global residual connection, and local combined global residual connection. The attention mechanism uses the same Enhanced Spatial Attention (ESA) as RFDN. we used the plane model without any residual connection as the baseline model. The number of output channels of each

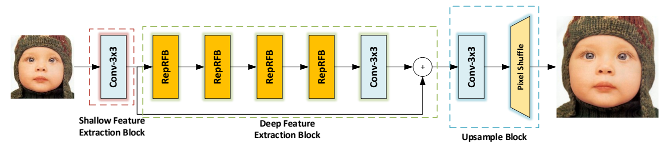


Figure 5. The structure of Reparameterized Residual Feature Network.

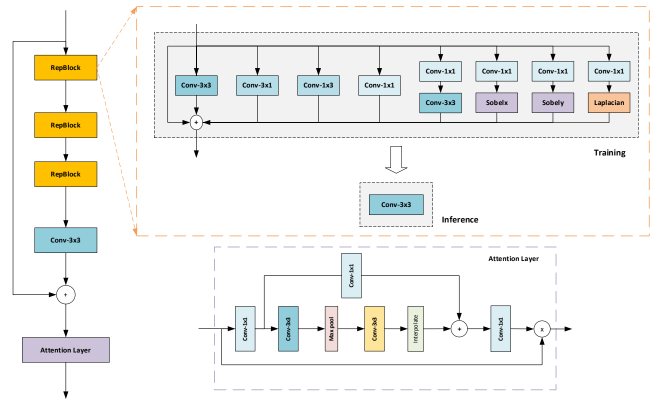


Figure 6. The details of Reparameterized Residual Feature Block.

module was unified as 48, and RFDN was retrained with the above training setting, which is denoted as RFDN*.

As shown in the Table 1, although the network structure based on residual feature learning has a slight increase in the number of model parameters and the amount of computation compared with RFDN, it has a decrease in inference time, number of activations, number of convolution layers, and maximum GPU memory consumed during inference. In particular, Mem decreased by 54.5%. It shows that the residual feature learning mechanism reduces the GPU memory consumed and time cost caused by the channel split and concatenation operation compared with the information distillation mechanism. By comparing different residual connection, it can be seen from the data of RFB1, RFB2 and RFB3 that RFB2 using global residual connection is slightly lower than RFB1 using local residual connection in terms of PSNR when the number of parameters and calculation amount are approximately the same, and RFB1 using only local residual connection has the same performance as RFB3 combined with local residual connection and global residual connection. It can be known that the contribution of global residual connection is lower than that of local residual connection.

3.2. Reparameterized Residual Feature Network (RepRFN)

Despite the low GPU memory usage and fast inference speed of RFN, there are still many problems. From the feature scale of the model, 3×3 convolution layer is mostly used in feature extraction, and the receptive field is relatively simple. Secondly, the structure of the model is still redundant. In addition, the extraction and recovery of high-frequency information in the image feature domain are also deficient. To solve the above problems, this section makes a series of improvements to the model, and proposes a multi-scale feature fusion lightweight SR network RepRFN based on Reparameterization [3, 4, 29]. To solve the problem of simple receptive field of the model, multiple parallel branch structure was designed, and the features of different receptive fields and modes are extracted and fused to make the model benefit from the multi-branch structure as much as possible. At the same time, the reparameterization, decoupling training and inference process are introduced to avoid the problem that the number of parameters and calculation amount increase caused by the introduction of multi-branch structure. To solve the problem of model structure redundancy, we reconsidered and analyzed the structural differences between RFN and RFDN, removes the 1×1 convolution layer used for channel transformation in RFN and makes structural improvements to ESA.

The RepRFN network structure proposed in this paper is shown in Figure 2. RepRFN has the same structure as RFN, the difference is that RepRFN replaces RFBs in RFN with Reparameterized Residual Feature Blocks (RepRFBs). Reparameterized Block (RepBlock) is the main component of RepRFB, and multi-branch structure constitutes RepBlock as shown in Figure 6. Shallow features are gradually extracted from different patterns of features by stacked RepBlocks in RepRFBs and fused through residual connections. Then deep features are obtained through 3×3 convolution layer. Then Shallow features and deep features are fused by local residual connection to improve the expression ability of features. The upsampling module uses subpixel convolution [21] to generate the final SR image. The above process can be expressed similarly as Equation 1.

Reparameterized Residual Feature Block The design of RepRFB proposed in this paper refers to the structure of RFDB in RFDN. In RFDB, the intermediate feature is split three times by the SRB and 1×1 convolution layer in each information distillation module as shown in Figure 3c. Therefore, the first three layers of RepRFB adopt a reparameterized multi-branch structure, which is called RepBlock in this paper, features propagate through multiple paths that perform different operations and eventually fuse them to improve the expressiveness of the model. In RFDB, due to the channel concatenation operation, a channel transformation using 1×1 convolution is required after the concatenation operation to input to the attention layer. However, in RepRFB, due to the existence of local residual connections, the size and number of channels of intermediate features before and after the RepBlock and convolutional layer are always unchanged, so channel transformation operation is not required. Therefore, it can be considered that the 1×1 convolution in RepRFB is redundant, and the 1×1 convolution is removed to further compress the parameters. Kong et al. [12] analyzed the redundancy of ESA in RFDB based on the pruning sensitivity analysis tool of the one-shot structured pruning algorithm [15], and found that the three convolution layers in the convolution group ranked first, third and fourth respectively in terms of redundancy. Therefore, the three-layer convolution of the convolution group in each ESA was reduced to one layer. The final RepRFB structure is shown in Figure 6.

Assume f_{ms}^{i} represents the output feature of the i-th multi-branch structure in the RepRFB, $repblock^{i}(\cdot)$ represents the operation function of the i-th multi branch structure that has been continuously stacked, f_{fusion} represents the fusion feature generated through residual connection between input feature x_in and intermediate feature f_{c3}, $Attention$ represents the attention layer used by the module, and x_{out} represents the output feature, it can be expressed as:

$$\begin{aligned} f_{ms}^{3} &= repblock^{3}(x_{in}) \\ f_{c3} &= conv_{3\times3}(f_{ms}^{3}) \\ f_{fusion} &= f_{c3} + x_in \\ x_{out} &= Attention(f_{fusion}) \end{aligned} \tag{3}$$

3.3. Loss Function Based on Fourier Transform

For the problem of extracting and restoring high-frequency information, in addition to introducing ECB [29] into multi-branch structure, Fourier transform is introduced into loss function to guide the model to learn frequency domain features and restore high-frequency information as much as possible.

The commonly used loss function of SR include L1 loss, L2 loss and Charbonnier loss [13], etc. These loss function can be considered as a kind of pixel loss by measuring the difference between the pixel values of SR images and high resolution (HR) images to guide model learning. How to effectively restore high-frequency information of images in SR has always been a focus of industry attention. During the model training process, the learning of frequency information is achieved by measuring the pixel differences between spectral maps of SR and HR. As expressed in the Equation 4:

$$\mathcal{L}(x, y) = \mathcal{L}_{pix}(fft(x), fft(y)) \tag{4}$$

4. Experiments

4.1. Experimental Setup

In terms of training, DIV2K [24] training set and Flickr2K [24] dataset are used, the HR patches is set to 192×192, random horizontal flip, vertical flip and rotation

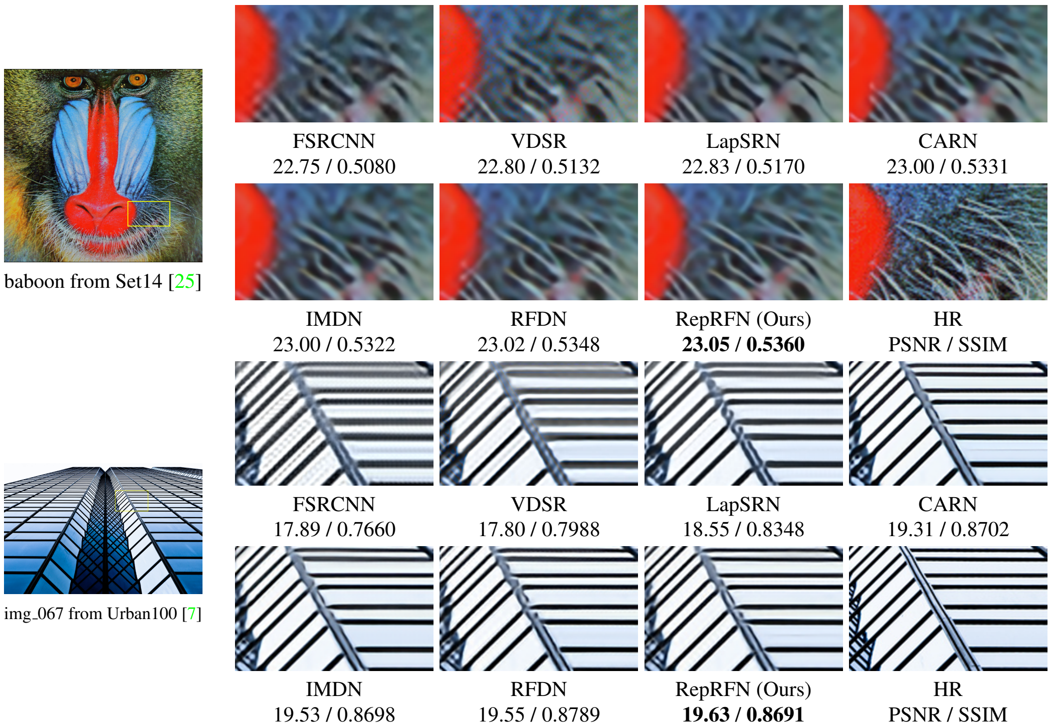


Figure 7. Visualization results.

are introduced into the data augmentation during training. The proposed RepRFN consists of 4 RepRFBs, the number of channels is set to 48. The model is trained from scratch. We use the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$, batchsize is set to 64, and the initial learning rate is 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 [13] can avoid the problem that the results generated by L1 loss and L2 loss are too smooth [30], in the experiment, we also found that the Charbonnier loss is better than the L1 loss in terms of PSNR, so we choosed the Charbonnier loss as L_{pix}. It should be noted that we only perform Fourier transform on the scale dimension of the image. Finally, the loss can be formulated as:

$$\mathcal{L}(x, y) = \lambda_1 L_{pix}(x, y) + \lambda_2 \mathcal{L}_1(fft(x), fft(y)) \quad (5)$$

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

4.2. Quantitative Results

We compared several SR networks based on CNN [1, 5, 6, 8–11, 14, 18, 22, 23, 28, 29]. The PSNR and SSIM of the model were tested on five benchmark datasets [2, 7, 19, 20, 25] to measure model performance. PSNR and SSIM were tested on the Y-channel in the YCbCr color space of the image. In terms of model complexity, assuming that the model output is a 720P image, the Parameters and FLOPs are used to measure the model complexity. In order to maximize the potential performance of the RepRFN proposed in this paper, Geometric Self-ensemble [17] was used in the experiment, and the RepRFN using the Geometric Self-ensemble strategy is denoted as RepRFN+. The experimental results are shown in Table 2. Figure 1 compares the PSNR and Parameters of different networks on the Set14 [25] under the condition of a scale factor of 4. It can be seen that RepRFN achieves performance comparable to other networks with fewer parameters and computational complexity, achieving a better balance in performance and efficiency. The network visualization results are shown in the Figure 7.

In order to further validate the efficiency of model deployment on mobile devices, we also compared the differences in inference time among three popular model deployment schemes on the Android devices of Qualcomm Snapdragon 865 and 820, and Rockchips RK3588 hardware platforms. The three deployment schemes are ONNX, PaddleLite, and Rockchips RKNN. As shown in Table 6, the efficiency of the RepRFN has been verified through experiments, indicating that the proposed RepRFN has certain competitiveness in deployment on hardware platforms.

Table 2. PSNR/SSIM and complexity results of SR with different scale factors for different networks on different benchmark test sets, The best and second-best results are marked in red and blue colors, respectively.

Model	Scale	Set5 [2] PSNR/SSIM	Set14 [25] PSNR/SSIM	BSD100 [19] PSNR/SSIM	Urban100 [7] PSNR/SSIM	Manga109 [20] PSNR/SSIM	Params (K)	FLOPs (G)
Bicubic	2	33.66/0.9299	30.24/0.8688	29.56/0.8431	26.88/0.8403	30.80/0.9339	-	-
SRCNN [5]		36.66/0.9542	32.45/0.9067	31.36/0.8879	29.50/0.8946	35.60/0.9663	57	52.7
FSRCNN [6]		37.00/0.9558	32.63/0.9088	31.53/0.8920	29.88/0.9020	36.67/0.9710	12	6
VDSR [10]		37.53/0.9587	33.03/0.9124	31.90/0.8960	30.76/0.9140	37.22/0.9750	665	612.6
DRCN [11]		37.63/0.9588	33.04/0.9118	31.85/0.8942	30.75/0.9133	37.55/0.9732	1774	17,974.30
LapSRN [14]		37.52/0.9591	32.99/0.9124	31.80/0.8952	30.41/0.9103	37.27/0.9740	813	29.9
DRRN [22]		37.74/0.9591	33.23/0.9136	32.05/0.8973	31.23/0.9188	37.88/0.9749	297	6,796.90
MemNet [23]		37.78/0.9597	33.28/0.9142	32.08/0.8978	31.31/0.9195	37.72/0.9740	677	2662.4
IDN [9]		37.83/0.9600	33.30/0.9148	32.08/0.8985	31.27/0.9196	38.01/0.9749	590	174.1
SRMDNF [28]		37.79/0.9601	33.32/0.9159	32.05/0.8985	31.33/0.9204	38.07/0.9761	1513	247.7
CARN [1]		37.76/0.9590	33.52/0.9166	32.09/0.8978	31.92/0.9256	38.36/0.9765	1592	222.8
IMDN [8]		38.00/0.9605	33.63/0.9177	32.19/0.8996	32.17/0.9283	38.88/0.9774	694	158.8
RFDN [18]		38.05/0.9606	33.68/0.9184	32.16/0.8994	32.12/0.9278	38.88/0.9773	534	123
ECBSR [29]		37.90/0.9615	33.34/0.9178	32.10/0.9018	31.71/0.9250	-	596	137.31
RepRFN (ours)		37.99/0.9609	33.57/0.9179	32.18/0.9004	31.95/0.9261	38.80/0.9774	386	85.12
RepRFN+ (ours)		38.07/0.9612	33.63/0.9184	32.22/0.9009	32.10/0.9274	39.00/0.9779	386	85.12
Bicubic	3	30.39/0.8682	27.55/0.7742	27.21/0.7385	24.46/0.7349	26.95/0.8556	-	-
SRCNN [5]		32.75/0.9090	29.30/0.8215	28.41/0.7863	26.24/0.7989	30.48/0.9117	57	52.7
FSRCNN [6]		33.18/0.9140	29.37/0.8240	28.53/0.7910	26.43/0.8080	31.10/0.9210	12	6
VDSR [10]		33.66/0.9213	29.77/0.8314	28.82/0.7976	27.14/0.8279	32.01/0.9340	665	612.6
DRCN [11]		33.82/0.9226	29.76/0.8311	28.80/0.7963	27.15/0.8276	32.24/0.9343	1774	17,974.00
LapSRN [14]		-	-	-	-	-	-	-
DRRN [22]		34.03/0.9244	29.96/0.8349	28.95/0.8004	27.53/0.8378	32.71/0.9379	297	6,796.90
MemNet [23]		34.09/0.9248	30.00/0.8350	28.96/0.8001	27.56/0.8376	32.51/0.9369	677	2662.4
IDN [9]		34.14/0.9259	30.13/0.8383	28.98/0.8026	27.86/0.8463	33.11/0.9416	590	105.6
SRMDNF [28]		34.12/0.9254	30.04/0.8382	28.97/0.8025	27.57/0.8398	33.00/0.9403	1,530	156.3
CARN [1]		34.29/0.9255	30.29/0.8407	29.06/0.8034	28.06/0.8493	33.50/0.9440	1592	118.8
IMDN [8]		34.36/0.9270	30.32/0.8417	29.09/0.8046	28.17/0.8519	33.61/0.9445	703	71.5
RFDN [18]		34.41/0.9273	30.34/0.8420	29.09/0.8050	28.21/0.8525	33.67/0.9449	541	55.4
ECBSR [29]		-	-	-	-	-	-	-
RepRFN (ours)		34.33/0.9272	30.30/0.8415	29.08/0.8058	27.95/0.8473	33.48/0.9434	392	38.4
RepRFN+ (ours)		34.45/0.9280	30.39/0.8430	29.13/0.8068	28.06/0.8494	33.76/0.9451	392	38.4
Bicubic	4	28.42/0.8104	26.00/0.7027	25.96/0.6675	23.14/0.6577	24.89/0.7866	-	-
SRCNN [5]		30.48/0.8626	27.50/0.7513	26.90/0.7101	24.52/0.7221	27.58/0.8555	57	52.7
FSRCNN [6]		30.72/0.8660	27.61/0.7550	26.98/0.7150	24.62/0.7280	27.90/0.8610	12	4.6
VDSR [10]		31.35/0.8838	28.01/0.7674	27.29/0.7251	25.18/0.7524	28.83/0.8870	665	612.6
DRCN [11]		31.53/0.8854	28.02/0.7670	27.23/0.7233	25.14/0.7510	28.93/0.8854	1774	17,974.00
LapSRN [14]		31.54/0.8852	28.09/0.7700	27.32/0.7275	25.21/0.7562	29.09/0.8900	813	149.4
DRRN [22]		31.68/0.8888	28.21/0.7720	27.38/0.7284	25.44/0.7638	29.45/0.8946	297	6,796.90
MemNet [23]		31.74/0.8893	28.26/0.7723	27.40/0.7281	25.50/0.7630	29.42/0.8942	677	2662.4
IDN [9]		31.82/0.8903	28.25/0.7730	27.41/0.7297	25.41/0.7632	30.04/0.9026	590	81.8
SRMDNF [28]		31.96/0.8925	28.35/0.7787	27.49/0.7337	25.68/0.7731	30.09/0.9024	1530	89.3
CARN [1]		32.13/0.8937	28.60/0.7806	27.58/0.7349	26.07/0.7837	30.47/0.9084	1592	90.9
IMDN [8]		32.21/0.8948	28.58/0.7811	27.56/0.7353	26.04/0.7838	30.45/0.9075	715	40.9G
RFDN [18]		32.24/0.8952	28.61/0.7819	27.57/0.7360	26.11/0.7858	30.58/0.9089	550	31.6
ECBSR [29]		31.92/0.8946	28.34/0.7817	27.48/0.7393	25.81/0.7773	-	603	34.73
RepRFN (ours)		32.15/0.8952	28.63/0.7824	27.60/0.7377	26.09/0.7834	30.52/0.9075	402	22.1
RepRFN+ (ours)		32.28/0.8969	28.68/0.7836	27.65/0.7389	26.18/0.7858	30.79/0.9102	402	22.1

Table 3. Performance differences between RepRFN-P and RepRFN on different benchmark datasets (Y-channel in Ycbcr color space).

Model	Scale	Set5 [2] PSNR/SSIM	Set14 [25] PSNR/SSIM	BSD100 [19] PSNR/SSIM	Urban100 [7] PSNR/SSIM	Manga109 [20] PSNR/SSIM
RepRFN-P	2	37.96/0.9608	33.51/0.9175	**32.18**/0.9003	31.92/0.9260	38.73/0.9774
RepRFN		**37.99/0.9609**	**33.57/0.9179**	**32.18/0.9004**	**31.95/0.9261**	**38.80/0.9774**
RepRFN-P	3	34.29/0.9267	30.27/0.8412	29.05/0.8052	27.89/0.8460	33.38/0.9426
RepRFN		**34.33/0.9272**	**30.30/0.8415**	**29.08/0.8058**	**27.95/0.8473**	**33.48/0.9434**
RepRFN-P	4	**32.17/0.8954**	28.61/0.7820	**27.60/0.7377**	26.05/0.7823	**30.54**/0.9074
RepRFN		32.15/0.8952	**28.63/0.7824**	**27.60/0.7377**	**26.09/0.7834**	30.52/**0.9075**

Table 4. Comparative experiment on model structure redundancy (DIV2K [24] validation set, RGB color space).

Channels	Remove 1×1 Conv	Modify ESA	Params (M)	FLOPs (G)	PSNR (dB)
48	✓	×	0.429	22.68	28.94
48	✓	✓	0.411	22.66	28.97
50	✓	✓	0.443	24.50	28.99
52	✓	✓	0.476	26.41	28.99
50	×	✓	0.432	23.93	28.95

4.3. Ablation Study

Multi-Scale Feature Fusion Module In order to verify the impact of multi-branch structure on model performance,

Table 5. Performance comparison experiment of different loss function on DIV2K [24] validation set (RGB color soace).

L1	Charbonnier	FFT	PSNR (dB)
✓	×	×	28.95
×	✓	×	28.97
×	✓	✓	**28.98**

Table 6. Inference time of RepRFN, IMDN, and RFDN under different deployment schemes.

Model	ONNX(RK3588) Run Time (ms)	PaddleLite(Snapdragon865/820) Run Time (ms)	RKNN(RK3588) Run Time (ms)
IMDN	4605.629395	1636.7737/4245.3735	966.550171
RFDN	3268.103027	905.4292/2847.237	312.565491
RepRFN	**2255.510498**	**645.4514/2049.4033**	**250.944626**

Table 7. NTIRE 2023 ESR Challenge result.

Model	Val PSNR (dB)	Test PSNR (dB)	Val Time (ms)	Test Time (ms)	Ave Time (ms)	Params (M)	FLOPs (G)	Acts (M)	Mem (M)	Conv
RFDN	29.04	27.11	42.41	28.66	35.54	0.433	27.10	112.03	788.13	64
RepRFN	28.99	27.05	**30.58**	**21.37**	**25.97**	**0.402**	**25.23**	**81.88**	**344.51**	**39**

we designed an experiment, which referred to the reparameterized planar model of RepRFN proposed in this paper as RepRFN-P (P: Plain), meaning that RepRFN-P does not contain multi-branch structure, similar to the RFN structure proposed in Section 3.1, In terms of experimental setup, it is the same as RepRFN, and RepRFN-P was retrained to observe the performance differences between it and RepRFN. From the Table 3, it can be seen that RepRFN is relatively superior in performance to RepRFN-P without using a multi branch structure, indicating that the multi-branch structure is beneficial for improving the performance of the model.

Model Structure We explored the redundancy of the model structure. Starting from three aspects: the number of model channels, 1×1 convolution used for channel transformation, and ESA. We investigated the impact of these three factors on model performance and efficiency. The baseline model is based on a model with a channel number of 48, preserving 1×1 convolution and the original ESA module. From the Table 4, it can be seen that the model performance of the modified ESA module has been increased by about 0.03dB, indicating that there is some redundancy in the convolution groups in ESA module. As the number of channels increases to 50, the PSNR also correspondingly improves by 0.02dB, with an 8% increase in both parameter and FLOPs. When the number of channels increases to 52, the PSNR no longer increases and the model complexity continues to increase. This indicates that the impact of channel numbers on model performance is manifested as the larger the number of channels, the more saturated the model performance tends to be, and the higher the complexity. Under the modified ESA module with the same number of channels of 50, removing the 1×1 convolution used for channel transformation can reduce the amount of parameters and FLOPs, but the performance also decreases by about 0.04dB. In order to obtain a model with lower complexity, we sacrificed some performance to design a model with lower complexity, the model RepRFN ultimately adopts the design of channel number 48, modified ESA module, and removes 1×1 convolutions used for channel transformation.

Loss function In order to verify the effectiveness of the proposed Fourier transform based loss function, we explored the impact of L1 loss, Charbonnier loss, and Fourier transform based loss on the performance of the SR model. It can be seen from the Table5 that Charbonnier loss is better than L1 loss in PSNR. After the introduction of Fourier transform based loss function, the performance of the model has been increased, indicating that the Fourier transform based loss function is beneficial to the model performance.

4.4. NTIRE 2023 ESR challenge

We have participated in NTIRE 2023 Efficient Super-Resolution Challenge [16]. This competition aims to devise a network that reduces one or several aspects such as runtime, parameters, FLOPs, activations, and depth of RFDN while at least maintaining PSNR of 29.00dB on validation datasets. Our results are shown in the Table 7.

5. Conclusion

In this paper, we propose a Reparameterized Residual Feature Network for lightweight image super-resolution. A multi branch structure is designed to capture the features of various patterns as much as possible and fuse them, then, reparameterization is introduced to enable complex multi-branch structures to be applied to lightweight networks. In the process of network training, a loss function based on Fourier transform is designed, which converts the image from spatial domain to frequency domain to guide the model to learn frequency information. Experiments have shown that the proposed method achieves better balance in performance and efficiency compared to other networks.

References

[1] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In *Proceedings of the European conference on computer vision (ECCV)*, pages 252–268, 2018. 2, 6, 7

[2] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. 2012. 6, 7

[3] 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. 3, 5

[4] 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. 3, 5

[5] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015. 1, 2, 6, 7

[6] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 391–407. Springer, 2016. 2, 6, 7

[7] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5197–5206, 2015. 6, 7

[8] Zheng Hui, Xinbo Gao, Yunchu Yang, and Xiumei Wang. Lightweight image super-resolution with information multi-distillation network. In *Proceedings of the 27th acm international conference on multimedia*, pages 2024–2032, 2019. 2, 4, 6, 7

[9] Zheng Hui, Xiumei Wang, and Xinbo Gao. Fast and accurate single image super-resolution via information distillation network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 723–731, 2018. 2, 4, 6, 7

[10] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016. 1, 6, 7

[11] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1637–1645, 2016. 2, 6, 7

[12] 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. 5

[13] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 624–632, 2017. 2, 5, 6

[14] 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. 6, 7

[15] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016. 5

[16] 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. 8

[17] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017. 1, 6

[18] 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, 4, 6, 7

[19] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423. IEEE, 2001. 6, 7

[20] 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*, 76:21811–21838, 2017. 6, 7

[21] 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. 3, 5

[22] Ying Tai, Jian Yang, and Xiaoming Liu. Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3147–3155, 2017. 2, 6, 7

[23] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *Proceedings of the IEEE international conference on computer vision*, pages 4539–4547, 2017. 6, 7

[24] 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. 3, 5, 7, 8

[25] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *Curves and Surfaces: 7th International Conference, Avignon, France, June 24-30, 2010, Revised Selected Papers 7*, pages 711–730. Springer, 2012. 1, 6, 7

[26] 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. Springer, 2020. 2

[27] Kai Zhang, Shuhang Gu, Radu Timofte, Zheng Hui, Xiumei Wang, Xinbo Gao, Dongliang Xiong, Shuai Liu, Ruipeng

Gang, Nan Nan, et al. Aim 2019 challenge on constrained super-resolution: Methods and results. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3565–3574. IEEE, 2019. 2

[28] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Learning a single convolutional super-resolution network for multiple degradations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3262–3271, 2018. 6, 7

[29] 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. 2, 3, 5, 6, 7

[30] 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. 6