Multi-scale Residual Network for Image Super-Resolution

Juncheng Li[1], Faming Fang[1](✉), Kangfu Mei[2], and Guixu Zhang[1]

[1] Shanghai Key Laboratory of Multidimensional Information Processing, and Department of Computer Science and Technology, East China Normal University, Shanghai, China
cvjunchengli@gmail.com, {fmfang,gxzhang}@cs.ecnu.edu.cn
[2] School of Computer Science and Information Engineering, Jiangxi Normal University, Nanchang, China
meikangfu@jxnu.edu.cn

Abstract. Recent studies have shown that deep neural networks can significantly improve the quality of single-image super-resolution. Current researches tend to use deeper convolutional neural networks to enhance performance. However, blindly increasing the depth of the network cannot ameliorate the network effectively. Worse still, with the depth of the network increases, more problems occurred in the training process and more training tricks are needed. In this paper, we propose a novel multi-scale residual network (MSRN) to fully exploit the image features, which outperform most of the state-of-the-art methods. Based on the residual block, we introduce convolution kernels of different sizes to adaptively detect the image features in different scales. Meanwhile, we let these features interact with each other to get the most efficacious image information, we call this structure Multi-scale Residual Block (MSRB). Furthermore, the outputs of each MSRB are used as the hierarchical features for global feature fusion. Finally, all these features are sent to the reconstruction module for recovering the high-quality image.

Keywords: Super-resolution · Convolutional neural network
Multi-scale residual network

1 Introduction

Image super-resolution (SR), particularly single-image super-resolution (SISR), has attracted more and more attention in academia and industry. SISR aims to reconstruct a high-resolution (HR) image from a low-resolution (LR) image which is an ill-posed problem since the mapping between LR and HR has multiple solutions. Thence, learning methods are widely used to learn a mapping from LR to HR images via applying large image datasets.

Currently, convolutional neural networks (CNNs) have indicated that they can provide remarkable performance in the SISR problem. In 2014, Dong et al.

V. Ferrari et al. (Eds.): ECCV 2018, LNCS 11212, pp. 527–542, 2018.
https://doi.org/10.1007/978-3-030-01237-3_32

proposed a model for SISR problem termed SRCNN [1], which was the first successful model adopting CNNs to SR problem. SRCNN was an efficient network that could learn a kind of end-to-end mapping between the LR and HR images without requiring any engineered features and reached the most satisfactory performance at that time. Since then, many studies focused on building a more efficient network to learn the mapping between LR and HR images so that a series of CNNs-based SISR models [2–9] were proposed. EDSR [9] was the champion of the NTIRE2017 SR Challenge. It based on SRResNet [8] while enhanced the network by removing the normalization layers as well as using deeper and wider network structures. These models received excellent performance in terms of peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM [10]) in the SISR problem. Nevertheless, all of these models tend to construct deeper and more complex network structures, which means training these models consumes more resources, time, and tricks. In this work, we have reconstructed some classic SR models, such as SRCNN [1], EDSR [9] and SRResNet [8]. During the reconstruction experiments, we find most existing SR models have the following problems:

(a) Hard to Reproduce: The experimental results manifest that most SR models are sensitive to the subtle network architectural changes and some of them are difficult to reach the level of the original paper due to the lack of the network configuration. Also, the same model achieves different performance by using different training tricks, such as weight initialization, gradient truncation, data normalization and so on. This means that the improvement of the performance may not be owing to the change of the model architecture, but the use of some unknown training tricks.

(b) Inadequate of Features Utilization: Most methods blindly increase the depth of the network in order to enhance the performance of the network but ignore taking full use of the LR image features. As the depth of the network increases, the features gradually disappear in the process of transmission. How to make full use of these features is crucial for the network to reconstruct high-quality images.

(c) Poor Scalability: Using the preprocessed LR image as input will add computational complexity and produce visible artifacts. Therefore, recent approaches pay more attention to amplifying LR images directly. As a result, it is difficult to find a simple SR model that can accommodate to any upscaling factors, or can migrate to any upscaling factors with only minor adjustments to the network architecture.

In order to solve the mentioned problems, we propose a novel multi-scale residual network (MSRN) for SISR. In addition, a multi-scale residual block (MSRB) is put forward as the building module for MSRN. Firstly, we use the MSRB to acquire the image features on different scales, which is considered as local multi-scale features. Secondly, the outputs of each MSRB are combined for global feature fusion. Finally, the combination of local multi-scale features and global features can maximize the use of the LR image features and completely solve the problem that features disappear in the transmission process. Besides, we introduce a convolution layer with 1×1 kernel as a bottleneck layer to obtain global feature fusion. Furthermore, we utilize a well-designed reconstruction structure that is simple but efficient, and can easily migrate to any upscaling factors.

We train our models on the DIV2K [11] dataset without special weight initialization method or other training tricks. Our base-model shows superior performance over most state-of-the-art methods on benchmark test-datasets. Besides, the model can achieve more competitive results by increasing the number of MSRB or the size of training images. It is more exciting that our MSRB module can be migrate to other restoration models for feature extraction. Contributions of this paper are as follows:

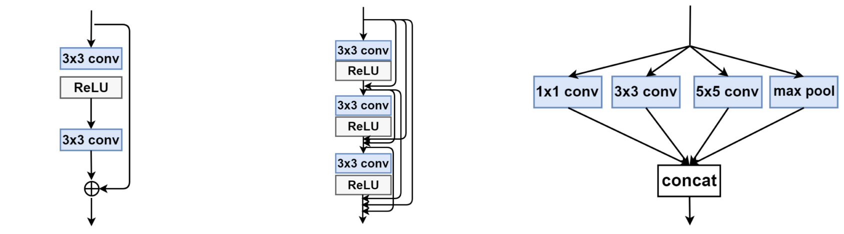
- Different from previous works, we propose a novel multi-scale residual block (MSRB), which can not only adaptively detect the image features, but also achieve feature fusion at different scales. This is the first multi-scale module based on the residual structure. What's more, it is easy to train and outperform the existing modules.
- We extend our work to computer vision tasks and the results exceed those of the state-of-the-art methods in SISR without deep network structure. Besides, MSRB can be used for feature extraction in other restoration tasks which show promising results.
- We propose a simple architecture for hierarchical features fusion (HFFS) and image reconstruction. It can be easily extended to any upscaling factors.

2 Related Works

2.1 Single-Image Super-Resolution

The SISR problem can be divided into three major stages roughly. Early approaches use interpolation techniques based on sampling theory like linear or bicubic. Those methods run fast, but can not rebuild the detailed, realistic textures. Improved works aim to establish complex mapping functions between LR and HR images. Those methods rely on techniques ranging from neighbor embedding to sparse coding.

Recent works tend to build an end-to-end CNNs model to learn mapping functions from LR to HR images by using large training datasets. Since Dong et al. proposed the SRCNN [1] model, various CNNs architectures have been used on SISR problem. Previous work often used pre-processed LR image as input, which was upscaled to HR space via an upsampling operator as bicubic. However, this method has been proved [2] that it will add computational complexity and produce visible artifacts. To avoid this, new methods are proposed, such as Fast Super-Resolution Convolutional Neural Networks (FSRCNN [3]) and Efficient Sub-pixel Convolutional Networks (ESPCN [2]). All of the models mentioned above are shallow networks (less than 5 layers). Kim et al. [12] first introduced the residual architecture for training much deeper network (20 layers) and achieved



(a) Residual block (b) Dense block (c) Inception block

Fig. 1. The structure of feature extraction blocks: (a) residual block, (b) dense block, and (c) inception block.

great performance. After that, many SR models have been proposed, including DRCN [5], DRNN [7], LapSRN [6], SRResNet [8], and EDSR [9]. Unfortunately, these models become more and more deeper and extremely difficult to train.

2.2 Feature Extraction Block

Nowadays, many feature extraction blocks have been proposed. The main idea of the inception block [13] (Fig. 1(c)) is to find out how an optimal local sparse structure works in a convolutional network. However, these different scale features simply concatenate together, which leads to the underutilization of local features. In 2016, Kim et al. [12] proposed a residual learning framework (Fig. 1(a)) to ease the training of networks so that they could achieve more competitive results. After that, Huang et al. introduced the dense block (Fig. 1(b)). Residual block and dense block use a single size of convolutional kernel and the computational complexity of dense blocks increases at a higher growth rate. In order to solve these drawbacks, we propose a multi-scale residual block.

Based on the residual structure, we introduce convolution kernels of different sizes, which designed for adaptively detecting the features of images at different scales. Meanwhile, a skip connection is applied between different scale features so that the features information can be shared and reused with each other. This helps to fully exploit the local features of the image. In addition, a 1×1 convolution layer at the end of the block can be used as a bottleneck layer, which contributes to feature fusion and reduces computation complexity. We will give a more detailed description in Sect. 3.1.

3 Proposed Method

In this work, our intent is to reconstruct a super-resolution image I^{SR} from a low-resolution image I^{LR}. The I^{LR} is the low-resolution version of I^{HR}, which is obtained by the bicubic operation. We convert the image to the YCbCr color space and train only on the Y channel. For an image with C color channels, we describe the I^{LR} with a tensor of size $W \times H \times C$ and denote the I^{HR}, I^{SR}

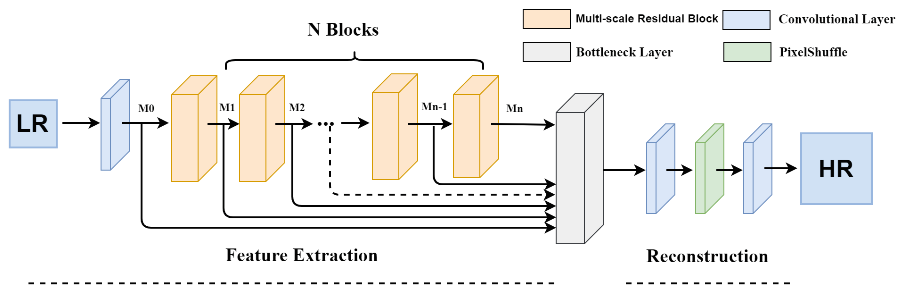


Fig. 2. The complete architecture of our proposed model. The network is divided into feature extraction and reconstruction, different color squares represent different operations, the top-right of the picture gives a specific description.

with $rW \times rH \times C$, where $C = 1$, represents the Y channel and r represents the upscaling factor.

Our ultimate goal is to learn an end-to-end mapping function F between the I^{LR} and the I^{HR}. Given a training dataset $\{I_i^{LR}, I_i^{HR}\}_{i=1}^{N}$, we solve the following problem:

$$\hat{\theta} = arg \min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}^{SR}(F_\theta(I_i^{LR}), I_i^{HR}), \tag{1}$$

where $\theta = \{W_1, W_2, W_3...W_m, b_1, b_2, b_3...b_m\}$, denotes the weights and bias of our m-layer neural network. $\mathcal{L}^{SR}$ is the loss function used to minimize the difference between I_i^{SR} and I_i^{HR}. Recently, researchers also focus on finding a superior loss function to improve the network performance. The most widely-used image objective optimization functions are the MSE function and L2 function. Although these methods can obtain high PSNR/SSIM, solutions for MSE optimization and L2 optimization problems often produce excessively smooth textures. Now, a variety of loss functions have been proposed such as VGG [4] function and Charbonnier Penalty function [6]. On the contrary, we find that their performance improvement is marginal. In order to avoid introducing unnecessary training tricks and reduce computations, we finally choose the L1 function. Thus, the loss function $\mathcal{L}^{SR}$ can be defined as:

$$\mathcal{L}^{SR}(F_\theta(I_i^{LR}), I_i^{HR}) = \left\| F_\theta(I_i^{LR}) - I_i^{HR} \right\|_1 . \tag{2}$$

As shown in Fig. 2, it is the complete architecture of our proposed model. Our model takes the unprocessed LR images as input, which are directly upsampled to high-resolution space via the network. Our model can be divided into two parts: the feature extraction module and the image reconstruction module. The feature extraction module is composed of two structures: multi-scale residual block (MSRB) and hierarchical feature fusion structure (HFFS).

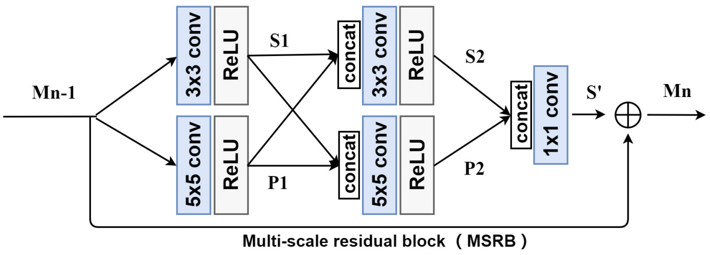


Fig. 3. The structure of multi-scale residual block (MSRB).

3.1 Multi-scale Residual Block (MSRB)

In order to detect the image features at different scales, we propose multi-scale residual block (MSRB). Here we will provide a detailed description of this structure. As shown in Fig. 3, our MSRB contains two parts: multi-scale features fusion and local residual learning.

Multi-scale Features Fusion: Different from previous works, we construct a two-bypass network and different bypass use different convolutional kernel. In this way, the information between those bypass can be shared with each other so that able to detect the image features at different scales. The operation can be defined as:

$$S_1 = \sigma(w^1_{3\times3} * M_{n-1} + b^1), \tag{3}$$

$$P_1 = \sigma(w^1_{5\times5} * M_{n-1} + b^1), \tag{4}$$

$$S_2 = \sigma(w^2_{3\times3} * [S_1, P_1] + b^2), \tag{5}$$

$$P_2 = \sigma(w^2_{5\times5} * [P_1, S_1] + b^2), \tag{6}$$

$$S^{'} = w^3_{1\times1} * [S_2, P_2] + b^3, \tag{7}$$

where w and b represent the weights and bias respectively, and the superscripts represent the number of layers at which they are located, while the subscripts represent the size of the convolutional kernel used in the layer. $\sigma(x) = max(0, x)$ stands for the ReLU function, and $[S_1, p_1]$, $[P_1, S_1]$, $[S_2, P_2]$ denote the concatenation operation.

Let M denote the number of feature maps sent to the MSRB. So the input and output of the first convolutional layer have M feature maps. And the second convolutional layer has 2M feature maps, either input or output. All of these feature maps are concatenated and sent to a 1×1 convolutional layer. This layer reduces the number of these feature maps to M, thus the input and output of our MSRB have the same number of feature maps. The distinctive architecture allows multiple MSRBs to be used together.

Local Residual Learning: In order to make the network more efficient, we adopt residual learning to each MSRB. Formally, we describe a multi-scale residual block (MSRB) as:

$$M_n = S^{'} + M_{n-1}, \tag{8}$$

where M_n and M_{n-1} represent the input and output of the MSRB, respectively. The operation $S^{'}+M_{n-1}$ is performed by a shortcut connection and element-wise addition. It is worth mentioning that the use of local residual learning makes the computational complexity greatly reduced. Simultaneously, the performance of the network is improved.

3.2 Hierarchical Feature Fusion Structure (HFFS)

For SISR problem, input and output images are highly correlated. It is crucial to fully exploit the features of the input image and transfer them to the end of the network for reconstruction. However, as the depth of the network increases these features gradually disappear during transmission. Driven by this problem, various methods have been proposed, among which the skip connection is the most simple and efficient method. All of these methods try to create different connections between different layers. Unfortunately, these methods can't fully utilize the features of the input image, and generate too much redundant information for aimlessness.

In the experiment, we notice that with the growth of depth, the spatial expression ability of the network gradually decreases while the semantic expression ability gradually increases. Additionally, the output of each MSRB contains distinct features. Therefore, how to make full use of these hierarchical features will directly affect the quality of reconstructed images. In this work, a simple hierarchical feature fusion structure is utilized. We send all the output of the MSRB to the end of the network for reconstruction. On the one hand, these feature maps contain a large amount of redundant information. On the other hand, using them directly for reconstruction will greatly increase the computational complexity. In order to adaptively extract useful information from these hierarchical features, we introduce a bottleneck layer which is essential for a convolutional layer with 1×1 kernel. The output of hierarchical feature fusion structure (HFFS) can be formulated as:

$$F_{LR} = w * [M_0, M_1, M_2, ..., M_N] + b, \tag{9}$$

where M_0 is the output of the first convolutional layer, $M_i (i \neq 0)$ represents the output of the i^{th} MSRB, and $[M_0, M_1, M_2, ..., M_N]$ denotes the concatenation operation.

3.3 Image Reconstruction

The previous work paid close attention to learn a mapping function between LR and HR images, where the LR image was upsampled to the same dimensions as HR by bicubic. Yet, this approach introduced redundant information and increased the computational complexity. Inspired by it, recent work tends to use the un-amplified LR as the input image to train a network that can be directly upsampled to HR dimensions. Instead, it is difficult to find an SR model which is able to migrate to any upscaling factors with only minor adjustments

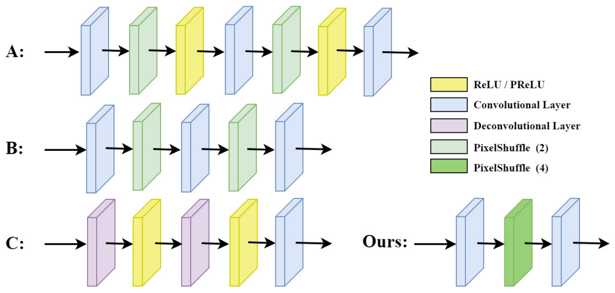


Fig. 4. Comparison of some common image reconstruction structure (×4).

to the network architecture. Moreover, most of these networks tend to be a fixed upscaling factor (x4), with no specific instructions given to migrate to other upscaling factors.

PixelShuffle [2] and deconvolutional layer are widely used in SISR tasks. As shown in Fig. 4, there are several common reconstruction modules. Taking the upscaling factor of ×4 as an example, all of these modules use pixelShuffle or deconvolution operation and the SR image is reconstructed gradually with upscaling factor 2 as the base. However, as the upscaling factor increases (e.g. ×8), the network becomes deeper accompanied with more uncertain training problems. Moreover, these methods does not work on odd upscaling factors, while one might expect a tardy growth in upscaling factor (e.g. ×2, ×3, ×4, ×5) rather than exponential increase.

For this purpose, we put forward a new reconstruction module (Fig. 4(ours)), which is a simple, efficient, and flexible structure. Thanks to pixelshuffle [2], our modules can be migrated to any upscaling factor with minor adjustments. In Table 1. We provide thorough configuration information about the reconstruction structure. In our network, for different upscaling factors, we only need to change the value of M whose change is negligible. Experiments indicate that this structure performs well on different upscaling factors.

4 Experiments

In this section, we evaluate the performance of our model on several benchmark test-datasets. We first introduce the dataset used for training and testing, then we give implementation details. Next, we compare our model with several state-of-the-art methods. Finally, we give a series of qualitative analysis experiments results. In addition, we show some of the results on other low-level computer vision tasks with our MSRB.

4.1 Datasets

The most widely used training dataset in previous studies includes 291 images, of which 91 images are from [14] and the other 200 images are from [15]. And

Table 1. Detailed configuration information about the reconstruction structure. For different upscaling factors, we only need to change the value of M.

Laye_name	Input_channel	Output_channel	Kernel_size
conv_input	64	64 × 2 × 2	3 × 3
PixelShuffle(×2)	64 × 2 × 2	64	/
conv_output	64	1	3 × 3
conv_input	64	64 × 3 × 3	3 × 3
PixelShuffle(×3)	64 × 3 × 3	64	/
conv_output	64	1	3 × 3
conv_input	64	64 × 4 × 4	3 × 3
PixelShuffle(×4)	64 × 4 × 4	64	/
conv_output	64	1	3 × 3
conv_input	64	64 × 8 × 8	3 × 3
PixelShuffle(×8)	64 × 8 × 8	64	/
conv_output	64	1	3 × 3
conv_input	64	64 × M × M	3 × 3
PixelShuffle(×M)	64 × M × M	64	/
conv_output	64	1	3 × 3

some methods take ImageNet [16] as training dataset, since it contains richer samples. In our work, we choose DIV2K [11] as our training dataset, a new high-quality image dataset for image restoration challenge. During testing, we choose five widely used benchmark datasets: Set5 [17], Set14 [18], BSDS100 [19], Urban100 [20] and Manga109 [21]. These datasets contain a wide variety of images that can fully verify our model. Following previous works, all our training and testing are based on luminance channel in YCbCr colour space, and upscaling factors: ×2, ×3, ×4, ×8 are used for training and testing.

4.2 Implementation Details

Following [6], we augment the training data in three ways: (1) scaling (2) rotation (3) flipping. In each training batch, we randomly extract 16 LR patches with the size of 64×64 and an epoch having 1000 iterations of back-propagation. We train our model with ADAM optimizer [22] by setting the learning rate $lr = 0.0001$. In our final model, we use 8 multi-scale residual blocks (MSRB, $N = 8$) and the output of each MSRB has 64 feature maps. Simultaneously, the output of each bottleneck layer (1 × 1 convolutional layer) has 64 feature maps. We implement MSRN with the Pytorch framework and train them using NVIDIA Titan Xp GPU. We do not use a special weight initialization method or other training tricks, and code is available at https://github.com/MIVRC/MSRN-PyTorch.

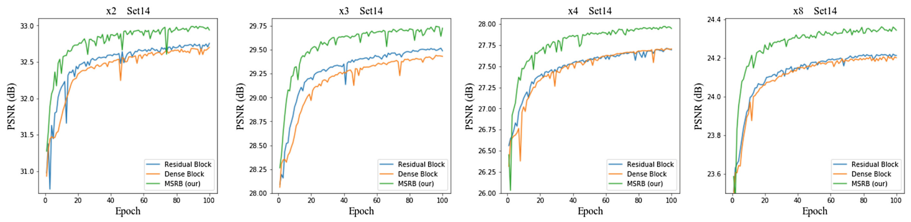


Fig. 5. Quantitative comparison of three different feature extraction blocks (residual block [12], dense block [24], and MSRB(our)) on SISR. The green line represents our model and it achieves the best results under different upscaling factors.

4.3 Comparisons with State-of-the-art Methods

We compare our model with 10 state-of-the-art SR methods, including Bicubic, A+ [23], SelfExSR [20], SRCNN [1], ESPCN [2], FSRCNN [3], VDSR [4], DRCN [5], LapSRN [6] and EDSR [9]. For fair, we retrain most of these models (except for EDSR [9], the results of EDSR provided by their original papers). Taking the equality of comparison into account, we evaluate the SR images with two commonly-used image quality metrics: PSNR and SSIM. Moreover, all the reported PSNR/SSIM measures are calculated on the luminance channel and remove M-pixel from each border (M stands for the upscaling factor).

The evaluation results of the SR method including our model and 10 state-of-art methods are demonstrated in Table 2. Our model outperforms by a large margin on different upscaling factors and test-datasets. It can be seen that our results are slightly lower than EDSR [9]. But it is worth noting that EDSR [9] use RGB channels for training, meanwhile, the data augment methods are different. To better illustrate the difference with EDSR [9], we show a comparison of model specifications in Table 3. EDSR [9] is an outstanding model gained amazing results. However, it is a deep and wide network which contains large quantities of convolutional layers and parameters. In other words, training this model will cost more memory, space and datasets. In contrast, the specifications of our model is much smaller than EDSR [9], which makes it easier to reproduce and promote.

In Figs. 6 and 7 we present visual performance on different datasets with different upscaling factors. Our model can reconstruct sharp and natural images, as well as outperforms other state-of-the-art methods. This is probably owing to the MSRB module can detect the image features at different scales and use them for reconstruction. For better illustration, more SR images reconstructed by our model can be found at https://goo.gl/bGnZ8D.

4.4 Qualitative Analysis

Benefit of MSRB: In this work, we propose an efficient feature extraction structure: multi-scale residual block. This module can adaptively detect image features at different scales and fully exploit the potential features of the image.

Table 2. Quantitative comparisons of state-of-the-art methods. Red text indicates the best performancen and blue text indicate the second best performance. Notice that the EDSR results were not retrained by us, but were provided by their original paper.

Algorithm	Scale	Set5 PSNR/SSIM	Set14 PSNR/SSIM	BSDS100 PSNR/SSIM	Urban100 PSNR/SSIM	Manga109 PSNR/SSIM
Bicubic	x2	33.69/0.9284	30.34/0.8675	29.57/0.8434	26.88/0.8438	30.82/0.9332
A+ [23]	x2	36.60/0.9542	32.42/0.9059	31.24/0.8870	29.25/0.8955	35.37/0.9663
SelfExSR [20]	x2	36.60/0.9537	32.46/0.9051	31.20/0.8863	29.55/0.8983	35.82/0.9671
SRCNN [1]	x2	36.71/0.9536	32.32/0.9052	31.36/0.8880	29.54/0.8962	35.74/0.9661
ESPCN [2]	x2	37.00/0.9559	32.75/0.9098	31.51/0.8939	29.87/0.9065	36.21/0.9694
FSRCNN [3]	x2	37.06/0.9554	32.76/0.9078	31.53/0.8912	29.88/0.9024	36.67/0.9694
VDSR [4]	x2	37.53/0.9583	33.05/0.9107	31.92/0.8965	30.79/0.9157	37.22/0.9729
DRCN [5]	x2	37.63/0.9584	33.06/0.9108	31.85/0.8947	30.76/0.9147	37.63/0.9723
LapSRN [6]	x2	37.52/0.9581	33.08/0.9109	31.80/0.8949	30.41/0.9112	37.27/0.9855
EDSR [9]	x2	38.11/0.9601	33.92/0.9195	32.32/0.9013	-/-	-/-
MSRN(our)	x2	38.08/0.9605	33.74/0.9170	32.23/0.9013	32.22/0.9326	38.82/0.9868
Bicubic	x3	30.41/0.8655	27.64/0.7722	27.21/0.7344	24.46/0.7411	26.96/0.8555
A+ [23]	x3	32.63/0.9085	29.25/0.8194	28.31/0.7828	26.05/0.8019	29.93/0.9089
SelfExSR [20]	x3	32.66/0.9089	29.34/0.8222	28.30/0.7839	26.45/0.8124	27.57/0.7997
SRCNN [1]	x3	32.47/0.9067	29.23/0.8201	28.31/0.7832	26.25/0.8028	30.59/0.9107
ESPCN [2]	x3	33.02/0.9135	29.49/0.8271	28.50/0.7937	26.41/0.8161	30.79/0.9181
FSRCNN [3]	x3	33.20/0.9149	29.54/0.8277	28.55/0.7945	26.48/0.8175	30.98/0.9212
VDSR [4]	x3	33.68/0.9201	29.86/0.8312	28.83/0.7966	27.15/0.8315	32.01/0.9310
DRCN [5]	x3	33.85/0.9215	29.89/0.8317	28.81/0.7954	27.16/0.8311	32.31/0.9328
LapSRN [6]	x3	33.82/0.9207	29.89/0.8304	28.82/0.7950	27.07/0.8298	32.21/0.9318
EDSR [9]	x3	34.65/0.9282	30.52/0.8462	29.25/0.8093	-/-	-/-
MSRN(our)	x3	34.38/0.9262	30.34/0.8395	29.08/0.8041	28.08/0.8554	33.44/0.9427
Bicubic	x4	28.43/0.8022	26.10/0.6936	25.97/0.6517	23.14/0.6599	24.91/0.7826
A+ [23]	x4	30.33/0.8565	27.44/0.7450	26.83/0.6999	24.34/0.7211	27.03/0.8439
SelfExSR [20]	x4	30.34/0.8593	27.55/0.7511	26.84/0.7032	24.83/0.7403	27.83/0.8598
SRCNN [1]	x4	30.50/0.8573	27.62/0.7453	26.91/0.6994	24.53/0.7236	27.66/0.8505
ESPCN [2]	x4	30.66/0.8646	27.71/0.7562	26.98/0.7124	24.60/0.7360	27.70/0.8560
FSRCNN [3]	x4	30.73/0.8601	27.71/0.7488	26.98/0.7029	24.62/0.7272	27.90/0.8517
VDSR [4]	x4	31.36/0.8796	28.11/0.7624	27.29/0.7167	25.18/0.7543	28.83/0.8809
DRCN [5]	x4	31.56/0.8810	28.15/0.7627	27.24/0.7150	25.15/0.7530	28.98/0.8816
LapSRN [6]	x4	31.54/0.8811	28.19/0.7635	27.32/0.7162	25.21/0.7564	29.09/0.8845
EDSR [9]	x4	32.46/0.8968	28.80/0.7876	27.71/0.7420	-/-	-/-
MSRN(our)	x4	32.07/0.8903	28.60/0.7751	27.52/0.7273	26.04/0.7896	30.17/0.9034
Bicubic	x8	24.40/0.6045	23.19/0.5110	23.67/0.4808	20.74/0.4841	21.46/0.6138
A+ [23]	x8	25.53/0.6548	23.99/0.5535	24.21/0.5156	21.37/0.5193	22.39/0.6454
SelfExSR [20]	x8	25.49/0.6733	24.02/0.5650	24.19/0.5146	21.81/0.5536	22.99/0.6907
SRCNN [1]	x8	25.34/0.6471	23.86/0.5443	24.14/0.5043	21.29/0.5133	22.46/0.6606
ESPCN [2]	x8	25.75/0.6738	24.21/0.5109	24.37/0.5277	21.59/0.5420	22.83/0.6715
FSRCNN [3]	x8	25.42/0.6440	23.94/0.5482	24.21/0.5112	21.32/0.5090	22.39/0.6357
VDSR [4]	x8	25.73/0.6743	23.20/0.5110	24.34/0.5169	21.48/0.5289	22.73/0.6688
DRCN [5]	x8	25.93/0.6743	24.25/0.5510	24.49/0.5168	21.71/0.5289	23.20/0.6686
LapSRN [6]	x8	26.15/0.7028	24.45/0.5792	24.54/0.5293	21.81/0.5555	23.39/0.7068
MSRN(our)	x8	26.59/0.7254	24.88/0.5961	24.70/0.5410	22.37/0.5977	24.28/0.7517

Table 3. Specifications comparison (x4). 'RGB' means the model is trained on RGB channels, 'Y' means the model is trained on luminance channel in YCbCr colour space, and 'M' means million.

Algorithm	Feature extraction	Filters	Layers	Depth	Parameters	Updates	Channel
EDSR [9]	**32 blocks**	**256**	**69**	**69**	**43M**	1×10^6	RGB
MSRN (our)	8 blocks	64	44	28	6.3M	4×10^5	Y

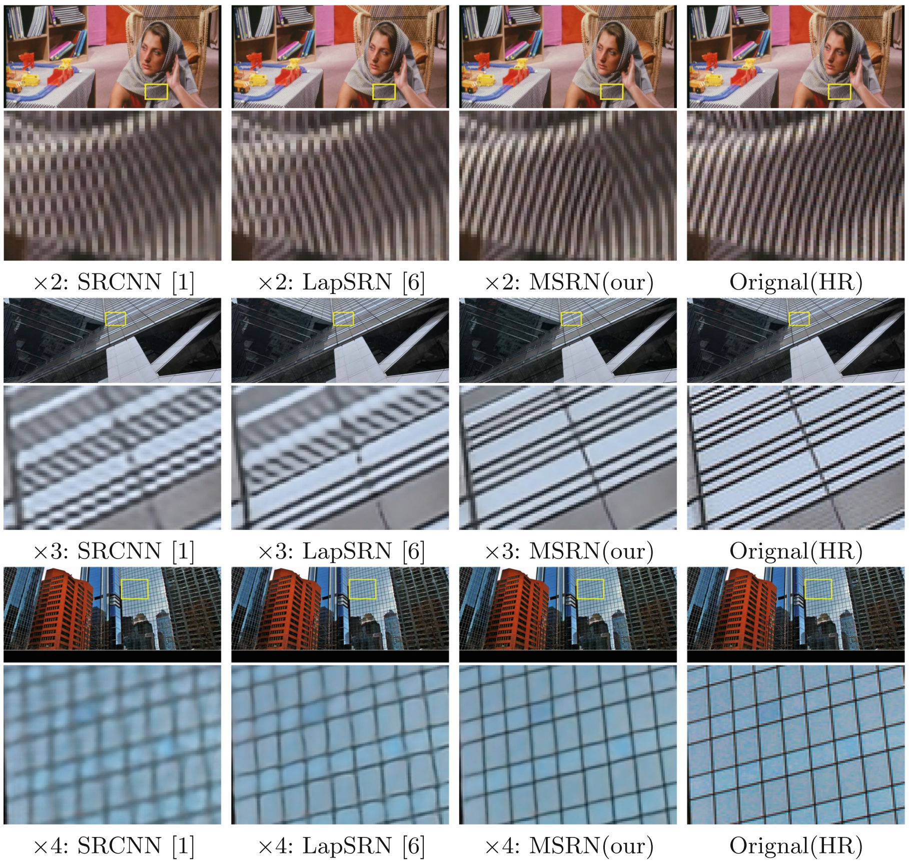


Fig. 6. Visual comparison for ×2, ×3, ×4 SR images. Our MSRN can reconstruct realistic images with sharp edges.

To validate the effectiveness of our module, we design a set of comparative experiments to compare the performance with residual block [12], dense block [24] and MSRB in SISR tasks. Based on the MSRN architecture, we replace the feature extraction block in the network. The three networks contain different feature extraction block, and each network contains only one feature extraction block.

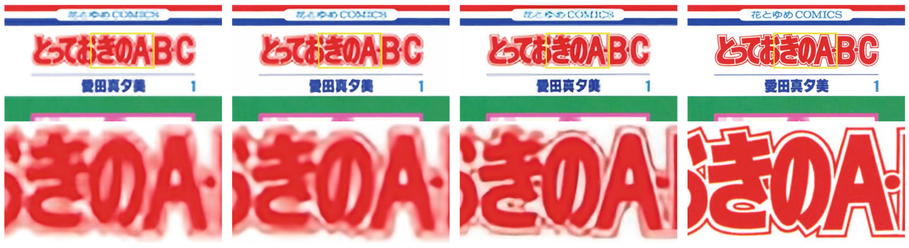


Fig. 7. Visual comparison of MSRN with other SR methods on large-scale (×8) SR task. Obviously, MSRN can reconstruct realistic images with sharp edges.

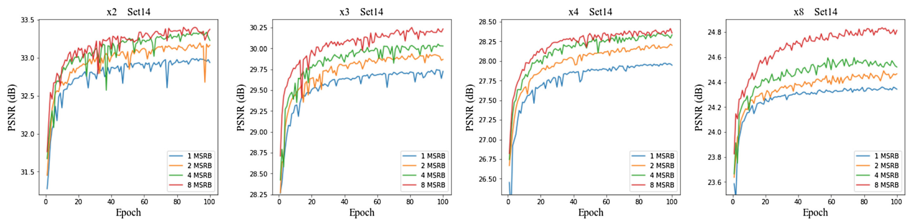
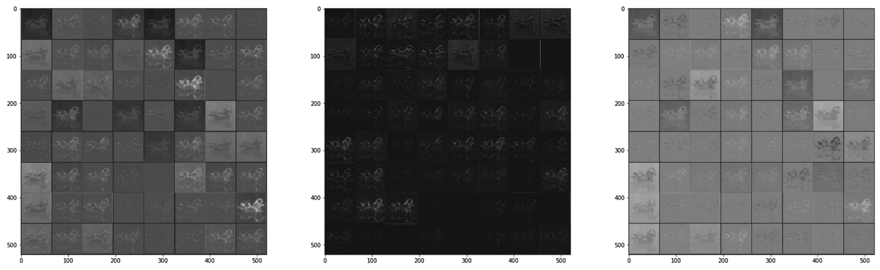


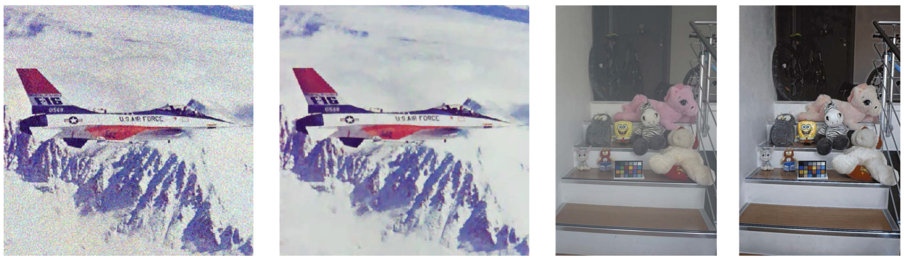
Fig. 8. Performance comparison of MSRN with different number of MSRBs.



(a) Residual block (b) Dense block (c) MSRB (our)

Fig. 9. Feature maps visualization. Represent the output of the residual block [12], the dense block [24], and our MSRB, respectively.

For quick verification, we use a small training dataset in this part, and all these models are trained in the same environment by 10^5 iterations. The results (Fig. 5) show that our MSRB module is superior to other modules at all upsampling factors. As shown in Fig. 9, we visualize the output of these feature extraction blocks. It deserves to notice that the activations are sparse (most values are zero, as the visualization shown in black) and some activation maps may be all zero which indicates dead filters. It is obvious that the output of the MSRB contains more valid activation maps, which further proves the effectiveness of the structure.



(A.1) input (A.2) output (B.1) input (B.2) output

Fig. 10. Application examples for image denoising and image dehazing, respectively.

Benefit of Increasing the Number of MSRB: As is acknowledged, increasing the depth of the network can effectively improve the performance. In this work, adding the number of MSRBs is the simplest way to gain excellent result. In order to verify the impact of the number of MSRBs on network, we design a series of experiments. As shown in Fig. 8, our MSRN performance improves rapidly with the number of MSRBs growth. Although the performance of the network will further enhance by using more MSRB, but this will lead to a more complex network. While weighing the network performance and network complexity, we finally use 8 MSRBs, the result is close to EDSR, but the number of model parameters is only one-seventh of it.

Performance on Other Tasks: In order to further verify the validity of our proposed MSRB module, we apply it to other low-level computer vision tasks for feature extraction. As shown in Fig. 10, we provide the results of image-denoising and image-dehazing, respectively. It is obvious that our model achieves promising results on other low-level computer vision tasks.

5 Discussion and Future Works

Many training tricks have been proposed to make the reconstructed image more realistic in SISR. For example, multi-scale (the scale here represents the upscaling factor) mixed training method is used in [4,9], and geometric selfensemble method is proposed in [9]. We believe that these training tricks can also improve our model performance. However, we are more inclined to explore an efficient model rather than use training tricks. Although our model has shown superior performance, the reconstructed image is still not clear enough under large upscaling factors. In the future work, we will pay more attention to large-scale downsampling image reconstruction.

6 Conclusions

In this paper, we proposed an efficient multi-scale residual block (MSRB), which is used to adaptively detect the image features at different scales. Based on

MSRB, we put forward multi-scale residual network (MSRN). It is a simple and efficient SR model so that we can fully utilize the local multi-scale features and the hierarchical features to obtain accurate SR image. Additionally, we achieved promising results by applying the MSRB module to other computer vision tasks such as image-denoising and image-dehazing.

Acknowledgments. This work are sponsored by the key project of the National Natural Science Foundation of China (No. 61731009), the National Science Foundation of China (No. 61501188), the "Chenguang Program" supported by Shanghai Education Development Foundation and Shanghai Municipal Education Commission (No. 17CG25) and East China Normal University.

References

1. Dong, C., Loy, C.C., He, K., Tang, X.: Learning a deep convolutional network for image super-resolution. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8692, pp. 184–199. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10593-2_13
2. Shi, W. et al.: 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, pp. 1874–1883 (2016)
3. Dong, C., Loy, C.C., Tang, X.: Accelerating the super-resolution convolutional neural network. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9906, pp. 391–407. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_25
4. Kim, J., Kwon Lee, J., Mu Lee, K.: Accurate image super-resolution using very deep convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1646–1654 (2016)
5. Kim, J., Kwon Lee, J., Mu Lee, K.: Deeply-recursive convolutional network for image super-resolution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1637–1645 (2016)
6. Lai, W.S., Huang, J.B., Ahuja, N., Yang, M.H.: Deep laplacian pyramid networks for fast and accurate superresolution. In: IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, p. 5 (2017)
7. Tai, Y., Yang, J., Liu, X.: Image super-resolution via deep recursive residual network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, p. 5 (2017)
8. Ledig, C., et al.: Photo-realistic single image superresolution using a generative adversarial network. In: The IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, p. 4 (2017)
9. Lim, B., Son, S., Kim, H., Nah, S., Lee, K.M.: Enhanced deep residual networks for single image super-resolution. In: The IEEE Conference on Computer Vision and Pattern Recognition Workshops. vol. 1, p. 4 (2017)
10. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
11. Agustsson, E., Timofte, R.: Ntire 2017 challenge on single image super-resolution: dataset and study. In: The IEEE Conference on Computer Vision and Pattern Recognition Workshops, vol. 3, p. 2 (2017)

12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
13. Szegedy, C., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
14. Yang, J., Wright, J., Huang, T.S., Ma, Y.: Image super-resolution via sparse representation. IEEE Trans. Image Process. **19**(11), 2861–2873 (2010)
15. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proceedings Eighth IEEE International Conference on Computer Vision, pp. 416–423 (2001)
16. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. **115**(3), 211–252 (2015)
17. Bevilacqua, M., Roumy, A., Guillemot, C., Alberi-Morel, M.L.: Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In: Proceedings of the 23rd British Machine Vision Conference. (2012)
18. Zeyde, R., Elad, M., Protter, M.: On single image scale-up using sparse-representations. In: Boissonnat, J.-D., Chenin, P., Cohen, A., Gout, C., Lyche, T., Mazure, M.-L., Schumaker, L. (eds.) Curves and Surfaces 2010. LNCS, vol. 6920, pp. 711–730. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27413-8_47
19. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(5), 898–916 (2011)
20. Huang, J.B., Singh, A., Ahuja, N.: Single image super-resolution from transformed self-exemplars. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5197–5206 (2015)
21. Matsui, Y., Ito, K., Aramaki, Y., Fujimoto, A., Ogawa, T., Yamasaki, T., Aizawa, K.: Sketch-based manga retrieval using manga109 dataset. Multimed. Tools Appl. **76**(20), 21811–21838 (2017)
22. Kingma, D.P., Ba, J.L.: Adam: a method for stochastic optimization. In: Proceedings of the 3rd International Conference on Learning Representations (2014)
23. Timofte, R., De Smet, V., Van Gool, L.: A+: adjusted anchored neighborhood regression for fast super-resolution. In: Cremers, D., Reid, I., Saito, H., Yang, M.-H. (eds.) ACCV 2014. LNCS, vol. 9006, pp. 111–126. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16817-3_8
24. Huang, G., Liu, Z., Weinberger, K.Q., van der Maaten, L.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, vol. 1, p. 3 (2017)