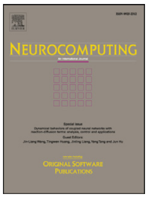
Neurocomputing 398 (2020) 399–407

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom



SCRSR: An efficient recursive convolutional neural network for fast and accurate image super-resolution

Daoyu Lin[a,b,*], Guangluan Xu[b], Wenjia Xu[a,b], Yang Wang[b], Xian Sun[b], Kun Fu[a,b]

[a] Department of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100190, China
[b] The Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, Institute of Electronics, Chinese Academy of Sciences, Beijing 100190, China

ARTICLE INFO

Article history:
Received 4 June 2018
Revised 29 January 2019
Accepted 8 February 2019
Available online 19 October 2019

Keywords:
Super-resolution
Recursive convolutional neural networks
Efficient model

ABSTRACT

Convolutional neural networks have recently demonstrated high-quality reconstruction for single image super-resolution (SR). These CNN networks effectively recover a high-resolution (HR) image from a low-resolution (LR) image, at the cost of enormous parameters and heavy computational burden. In this work, we propose a recursive efficient deep convolutional network for fast and accurate single-image SR with only 0.28M parameters. A Split-Concatenate-Residual (SCR) block is proposed to reduce computation and parameters. With downsampling block and upsampling block, we significantly reduce computational complexity and enlarge the size of the receptive field. Specifically, two-level recursive learning is proposed which can improve accuracy by increasing depth without adding any weight parameters. We also employ local, semi-global and global residual techniques to train our very deep network steadily and improve its performance. Extensive experiments indicate that our proposed method Split-Concatenate-Residual Super Resolution (SCRSR) yields promising SR performance while maintaining shorter running time and fewer parameters.

1. Introduction

Single Image Super-Resolution (SR), which aims to recover a high-resolution (HR) image from a low-resolution (LR) image, is a widespread problem in image processing. High resolution with higher pixel density contains more details, thus it plays an essential part in some applications. Recently, deep neural networks [1–3] provide significantly improved performance in the SR problem. However, those methods usually suffer from computational cost and memory footprint. To carry out SR jobs on real-time platforms such as mobile phone, we aim to explore an efficient network to reconstruct HR image accurately. Table 1 shows some typical CNN based SR algorithms. Those methods can be categorized according to network size or the type of input image.

Interpolated low-resolution (ILR) Image as Input. Methods such as SRCNN [1] and VDSR [2] take bicubic interpolation of LR images (ILR) as input and high-resolution image (HR) as output. Those methods learn an end-to-end mapping between ILR and HR. By doing so, VDSR can handle multi-scale SR problem jointly in a single network. However, big input image leads to heavier computation burden and memory footprint.

LR image as input. Methods such as FSRCNN [4], ESPCN [5] and LapSRN [3] take LR image as input. In those methods, the bicubic interpolation is replaced by an upsampling block at the end of the network. Those methods reduce the computational cost and memory usage. However, when dealing with multi-scale problems, many scale-specific networks must be trained independently.

Shallow networks. We also category networks according to network depth. Methods with less than 10 layers are Shallow networks, such as SRCNN, FSRCNN and ESPCN shown in Table 1. Though shallow networks are more efficient concerning memory and speed, due to limited network capability, they cannot perform well on complicated mappings.

Deep Networks. Methods with very deep networks [2,3,9] are proposed to optimize performance. VDSR uses 20 convolutional layers to improve performance. To accelerate the convergence speed and avoid gradient explosion problem, VDSR adopted residual learning and very high learning rate. LapSRN [3] uses a robust Charbonnier loss function to train a 27-layer convolutional network. A deep network can utilize more contextual information in an image and usually achieves better performance than shallow ones. DRCN [6] uses a deeply recursive convolutional network

[*] Corresponding author at: Department of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100190, China.

E-mail addresses: lindaoyu15@mails.ucas.ac.cn (D. Lin), gluanxu@mail.ie.ac.cn (G. Xu), xuwenjia16@mails.ucas.ac.cn (W. Xu), ywang1@mail.ie.ac.cn (Y. Wang), sunxian@mail.ie.ac.cn (X. Sun), fukun@mail.ie.ac.cn (K. Fu).

https://doi.org/10.1016/j.neucom.2019.02.067
0925-2312/© 2019 Elsevier B.V. All rights reserved.

Table 1
Comparison of typical CNN based SR algorithms. One level recursive learning represents sharing weights among different convolutional layers or residual units. We apply two level recursive learning, sharing weights among both SCR blocks and SubNets.

Method	Input	Layers	Down-sample	Up-sample	Residual Learning	Recursive Learning
SRCNN [1]	ILR	3	No	No	No	No
FSRCNN [4]	LR	8	No	Yes	No	No
ESPCN [5]	LR	3	No	Yes	No	No
VDSR [2]	ILR	20	No	No	Global	No
DRCN [6]	ILR	20	No	No	Global	One level
LapSRN [3]	LR	27	No	Yes	Global	No
DRRN [7]	ILR	52	No	No	Local+Global	One level
EDSR [8]	LR	69	No	Yes	Local+Global	No
SCRSR (ours)	ILR	58	Yes	Yes	Local+Semi+Global	Two level

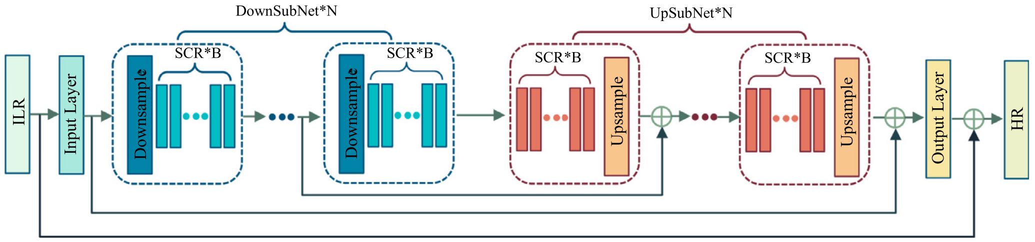


Fig. 1. Our SCRSR model structure is symmetrical as a whole, which contains N downsampling subnetworks (DownSubNet) and N upsampling subnetworks (UpSubNet). Each DownSubNet contains one Downsampling block, B recursive SCR blocks. ⊕ represents the element-wise addition.

for SR. The author also apply skip connections to ease the difficulty of training. Inspired by DRCN, DRRN [7] adopts both residual learning and recursive learning. In order to activate the negative part of the neurons and to preserve the sparsity of activation function, Zhiming [10] propose a paired ReLUs activation scheme: one of the ReLUs is for positive activation and the other is for negative activation. SRDenseNet [11] introduces the basic dense block from DenseNet, providing an effective way to combine the low-level features and high-level features to boost the reconstruction performance. Following SRDenseNet, RDN [12] introduce contiguous memory (CM) mechanism, which allows the state of preceding residual dense block have direct access to each layer of the current block.

Despite achieving better results, deep networks need more parameters, leading to heavier computational cost and memory usage. For a deep model, the training process is hard to converge while the testing process is time-consuming. Its hard to carry out those SR jobs on real-time platforms such as mobile phone.

To address these issues, we propose a fast and accurate super-resolution model called Split-Concate-Residual Super Resolution (SCRSR), shown in Fig. 1. We use a very deep (58 layers) and powerful network, significantly enlarge the receptive field. A large receptive field can provide more context for predicting image details. Table 1 shows a comparison of popular SR methods. *Compared to other state-of-the-art methods, such as VDSR, DRCN, DRRN, our model uses the deepest network, and we can achieve comparable accuracy with the fastest speed and least parameters.* SCRSR has three major novelties:

(1) *Efficient SCR blocks.* We propose an efficient Split-Concatenate-Residual (SCR) block (shown in Fig.) to ease the computational burden. In most SR works based on CNN such as LapSRN [3] and EDSR [8], residual blocks with filter size 3×3 are widely used. Specifically, to reduce computation and parameters, we shrink some filter size to 1×1 in SCR block. Experiment result shows that with SCR block, our method can achieve real-time speed and is faster than several CNN based super-resolution models while keeping accurate SR results. What's more, we apply downsampling layer in DownSubNets to decrease the size of the feature map, which significantly reduces memory footprint. Experiment shows that our method saves approximately 49% memory usage during training, compared to networks without downsampling layer.

(2) *Recursive SCR blocks and SubNets.* We adopt two level recursive learning to improve accuracy by increasing depth without adding any weight parameters, shown in Fig. 2. Level one: SCRSR adopts recursive learning in each subnetwork, which consists of several SCR blocks with same weights. Level two: different downsampling subnetwork (*DownSubNets*) or upsampling subnetwork (*UpSubNets*) also share the same parameters. In this way, no matter SCRSR contains how many SCR blocks and subnetworks, the recursion depth will increase without introducing any new parameters for additional convolutions.

(3) *Effective Residual Learning* is proposed in SCRSR to accelerate training process and prevents vanishing gradients problem. In SCRSR, three kinds of residual learning are adopted.
   - Global residual learning: skip connection between the input and output of the network.
   - Semi-Global residual learning: skip connections between the feature maps pairs before downsample and after upsample.
   - Local residual learning: skip connections in SCR blocks.

With global residual, SCRSR only needs to predict high frequency residual information instead of predicting the whole image. Since downsampling layer shrinks the feature map, we may lose information during training. Semi-local residual helps us save lost information effectively. Inspired by the Residual Unit proposed in ResNetV2 [13], we use local residual in SCR blocks and add ReLU before convolution to achieve fast error reduction.

2. Related works

Recently, with the development of deep learning, convolutional neural networks (CNN) have boosted the performance of SR. SRCNN [1] first applies Convolutional Neural Network to SR problems.

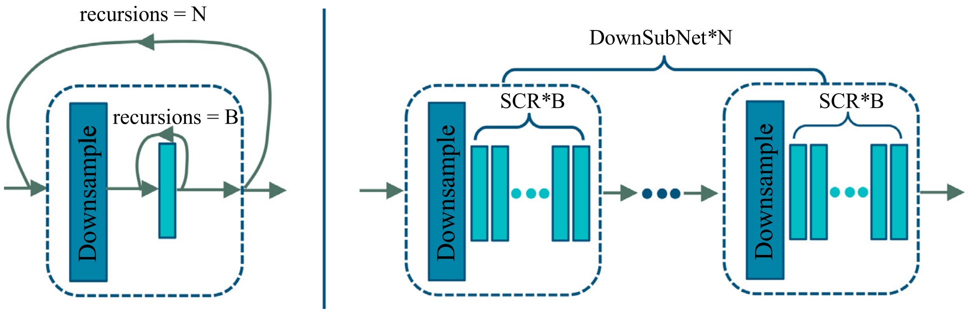
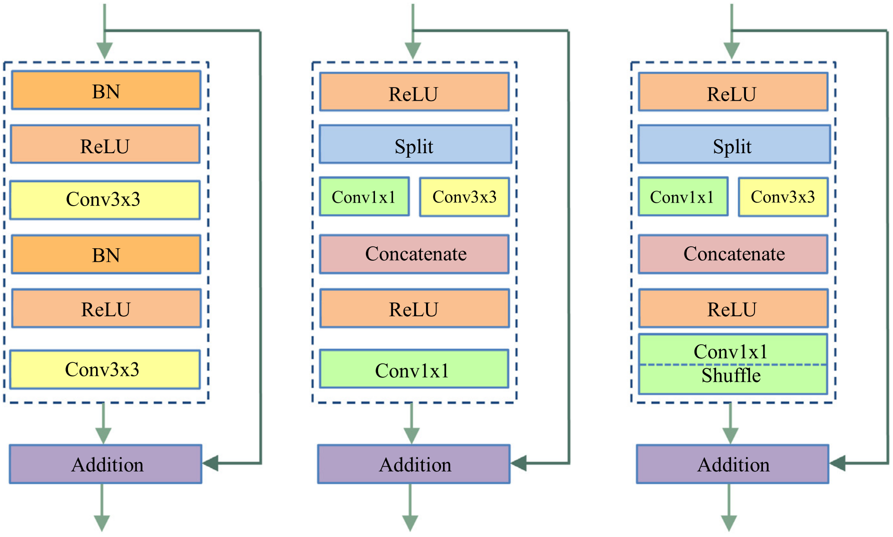


Fig. 2. Two level recursive learning. **Left:** recursive SCR blocks and recursive DownSubNet. **Right:** unfolded structure, every DownSubNet and recursive SCR block with same color shares same parameters. No matter how N and B are chosen, the parameters of the network remain unchanged.



(a) Original Block (b) SCR Block (c) Upsample Block

Fig. 3. The structure comparison of a) Original residual block [13], (b) Split-Concatenate-Residual (SCR) block proposed in our work, (c) The skip connection represents the local residual-learning in SCR block. Every SCR block includes two convolutional layers.

In SRCNN, the input image is a bicubic interpolation of the original LR image. In FSRCNN [4] and LapSRN [3], rather than using the interpolated image as input, a deconvolutional layer is applied at the end of the network for upsampling.

To achieve better accuracy, many networks apply deeper convolutional networks, such as VDSR [2] (20 layers) and LapSRN [3] (27 layers). To speed up the training process, residual-learning and extremely high learning rates are adopted by VDSR. Methods such as DRCN [6] and DRRN [7] apply the recursive convolutional network to SR. They utilize the same convolutional layer or residual unit many times to reduce parameters. EDSR [8], a very deep network, achieves good results on a newly provided DIV2K dataset [14].

SRGAN [15] proposes a GAN-based network, which augments the content loss function with an adversarial loss by training a GAN. The adversarial loss is applied to predict compelling, high-frequency detail. Despite the fact that SRGAN achieves more photo-realistic results than previous works, it cannot achieve high SR accuracy.

3. Proposed methods

In this section, we describe the technical parts of our proposed SCRSR network. The configuration of the network is displayed in Fig. 1. SCRSR can be decomposed into four parts, which is symmetrical as a whole: the input layer, the downsampling subnetworks (DownSubNet), the upsampling subnetworks (UpSubNet) and the output layer. We highlight the differences in the network structures between SCRSR and related models in Table 1. Now, we will describe more details of our model.

3.1. SCR Block

He et al. [13] propose a “pre-activation” residual block (shown in Fig. 3a) with batch normalization (BN) [16] and activation before the weight layers. Compared to “post-activation” [17], pre-activation is easier to train and has better performance. Though BN has achieved satisfactory performance on some high-level computer vision task, it may not suit SR problems and causes memory burden [8]. Thus our SR model adopts residual block without BN (shown in Fig. 3b).

Before introducing details of SCR block, we first investigate the time complexity (Eq. 1) of a convolutional network.

$$O\left(\sum_{l=1}^{d} c_l \cdot f_l^2 \cdot n_l \cdot m_l^2\right) \tag{1}$$

The convolutional layer is denoted as $Conv(n_l, f_l, c_l)$, where the variables n_l, f_l, c_l represent the number of filters in the convolutional layer l, the spatial size of the filter, and the number of input channels. m_l is the spatial size of the output feature map. The input layer has 128 filters of size 3*3, which can be denoted as Conv(1, 3, 128). The output layer contains 1 filter of size 3*3, which can be denoted as Conv(128, 3, 1).

With same output feature map, the computational complexity of one convolutional layer is proportional to the filter size, input channel and the number of filters. To further reduce parameters, we propose an SCR block (Fig. 3b) to modify the 3×3 filter in original residual blocks. According to this, we replace *Conv*(128, 3, 128) with the combination of *Conv*(64, 1, 64) and *Conv*(64, 3, 64). Firstly, we split 128 input channels into two groups, 64 input channels for each. Secondly, 64 filters with the size of 1×1 and 64 filters with the size of 3×3 are applied to two groups separately.

However, simply concatenating two groups together after convolution will cause one side effect: output channels are derived from two groups without communication. In that case, information between two channel groups will be blocked. To avoid this, we concatenate two output channels groups, then merge information with a convolutional layer *Conv*(128, 1, 128), which makes SCR block more powerful and stable. The mathematical formulas of concatenation is as follows,

$$x_o = Conv_{128,1,128}(H([x_0, x_1])), \tag{2}$$

where $H(\cdot)$ denotes a ReLu non-linear transformation, the output of the convolutional layer Conv(128, 1, 128) is denoted as x_o. [x_0, x_1] refers to the concatenation of the feature-maps of Conv(64, 1, 64) and Conv(64, 3, 64).

SCR block significantly reduces computation cost. With the same size of output feature maps, the time complexity ratio of an SCR block to an original residual block is 7: 36 according to Eq. 1. The SCR block can also reduce parameters. The parameter ratio of an SCR block to an original residual block is also 7: 36.

3.2. Recursive SCR blocks and subnets

As is shown in Eq. 1, the spatial size of output feature maps would significantly influence computation complexity. To ease the computational burden, we propose N DownSubNet to shrink the size of output feature maps. For upscaling, N UpSubNet are applied after DownSubNet.

As is shown in Fig. 2, every DownSubNet contains a downsampling block, which is also a SCR block (Fig. 3b) with $stride = 2$ in the first $Conv(1 \times 1)$ and $Conv(3 \times 3)$ layer. With the help of downsampling blocks, we can significantly reduce computation complexity and memory usage. There are B recursive SCR blocks following the downsampling block, which can utilize very large context without adding new weight parameters.

Every UpSubNet also contains B recursive SCR blocks and an upsampling block whose upscale factor is 2, which also shares the same structure with SCR block but the second $Conv(1 \times 1)$ layer is a sub-pixel convolution layer proposed in [5]. Compared to deconvolutional layer, the sub-pixel convolution layer is faster.

Except for recursive SCR blocks, SCRSR also propose recursive SubNets by sharing parameters between different DownSubNets.

3.3. Three level residual learning

With the increase of network depth, high-frequency information gets saturated and degrades rapidly, which indicates that merely increasing the network depth may not be a good solution for super-resolution. Inspired by ResNet [17], we apply three kinds of residual learning to combine deep network output with previous information.

Instead of learning the mappings from ILR image to HR image directly, global residual-learning predicts high-frequency image information.

Neuron science study indicates that short-term memory can be consolidated to long-term memory after rehearsal. The residual information at different stage denotes the short-term memory in this network, providing low-level image details for reconstruction. Thus, we apply Semi-Global Residual to transfer information between downsampling and upsampling blocks.

Specifically, we also adopt local residual in SCR blocks, as is shown in Fig. 3b.

Residual learning adds neither extra parameter nor computation. Besides, it will back-propagate gradient to the bottom layers, accelerating the training process.

3.4. Mathematical formulation

Our work takes an interpolated LR image (ILR) as input $\mathbf{x}$ and produce an high resolution image (HR) $\mathbf{y}$. Our goal is to learn a model f to predict $\hat{\mathbf{y}} = f(\mathbf{x})$ which matches $\mathbf{y}$ as much as possible. Let $f_\text{I}, f_\text{D}, f_\text{U}, f_\text{O}$ denote the functions of **Input Layer**, one **DownSubNet**, one **UpSubNet** and **Output Layer**.

Input Layer $f_\text{I}(\mathbf{x})$ takes the ILR image $\mathbf{x}$ and computes the output H_0. The formula for Input Layer is as follows:

$$H_0 = f_\text{I}(\mathbf{x}) = max(0, W_\text{I} * \mathbf{x} + b_\text{I}), \tag{3}$$

where the operator * denotes a convolution and $max(0, \cdot)$ denotes the ReLU function. W_I and b_I are weights and bias of the Input Layer.

DownSubNet f_D consists of one DownSampling block (represented as d_{W_D} with weights W_D) and one recursive SCR block (represented as $d_{W_{DR}}$ with weights W_{DR}). Let D_{in}, D_{out} denote the input and output, thus one DownSubnet can be formulated as:

$$\begin{aligned} D_{out} &= f_\text{D}(D_{in}) \\ &= d_{W_{DR}}(d_{W_{DR}}(\ldots d_{W_{DR}}(d_{W_D}(D_{in}))\ldots)). \end{aligned} \tag{4}$$

Similar to DownSubNet, one UpSubNet f_U can be formulated as:

$$\begin{aligned} U_{out} &= f_\text{U}(U_{in}) \\ &= d_{W_{UR}}(d_{W_{UR}}(\ldots d_{W_{UR}}(d_{W_U}(U_{in}))\ldots)), \end{aligned} \tag{5}$$

where d_{W_U} denotes the Upsampling block with weights W_U, and $d_{W_{UR}}$ denotes one recursive SCR block with weights W_{UR}. U_{in}, U_{out} are the input and output of one DownSubNet.

Different from the Input Layer, an Output Layer f_O without ReLU can be represented as:

$$H_{out} = f_\text{O}(\mathbf{x}) = W_\text{O} * H_{in} + b_\text{O}, \tag{6}$$

where H_{in}, H_{out} denote the input and output of Output Layer and W_O and b_O are weights and bias of Output Layer.

SCRSR has two key parameters: the number of DownSubNets or UpSubNets N and the number of SCR blocks B. Given diffenrent N and B, we can train SCRSR in different depths. Specifically, there are two convolutional layers in every SCR block, thus the depth of SCRSR is as follows:

$$d = 4\text{N} \times (\text{B} + 1) + 2, \tag{7}$$

where N is the number of DownSubNets or UpSubNets and B is the number of SCR blocks. 1 represents the downsampling block or upsampling block, and 2 represents the input layer and output layer.

It is noteworthy that no matter SCRSR contains how many recursive SCR blocks and subnetworks, our model only include spercific weights W_I, W_D, W_DR, W_U, W_UR and W_O as mentioned above. The whole model of SCRSR with global resudial learning can be formulated as:

$$\hat{\mathbf{y}} = f_O(f_\text{U}\cdots(f_\text{U}(f_\text{D}\cdots(f_\text{D}(f_\text{I}(\mathbf{x})))))) + \mathbf{x}. \tag{8}$$

We now describe the loss function to minimize for our network. Given a training set $\mathbf{x}_i$, $\mathbf{y}_i$, where $\mathbf{y}_i$ denotes the ground truth HR image of the ILR image $\mathbf{x}_i$. Our goal is to find the best model f that accurately predicts values $\hat{\mathbf{y}}_i = f(\mathbf{x}_i)$. The loss function of SCRSR is

$$L(\Theta) = \frac{1}{2n}\sum_{i=1}^{n} ||\mathbf{y}_i - \hat{\mathbf{y}}_i||^2, \tag{9}$$

where n represents the number of training samples.

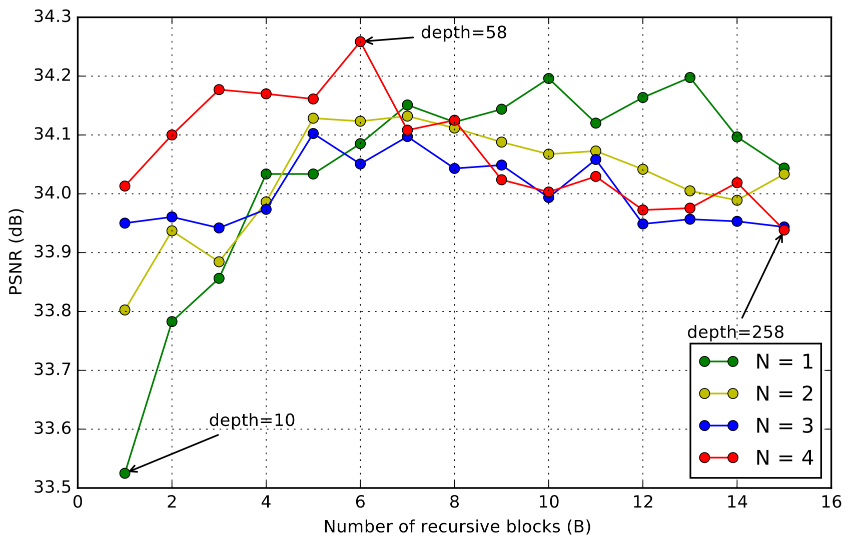


Fig. 4. Average PSNR (Set5 and Set14, × 2) of different SCRSRs at N and B combinations. The equation of SCRSR network depth is $d = 4\mathrm{N} \times (\mathrm{B}+1) + 2$ (Eq. 7). The network depth of SCRSR ranges from 10 (B=1, N=1) to 258 (B=15, N=4), while these networks contains same amount of parameters because of proposed two level recursive learning. Considering both the performance and speed, we choose B = 6, N = 2 (depth=58) as our best model.

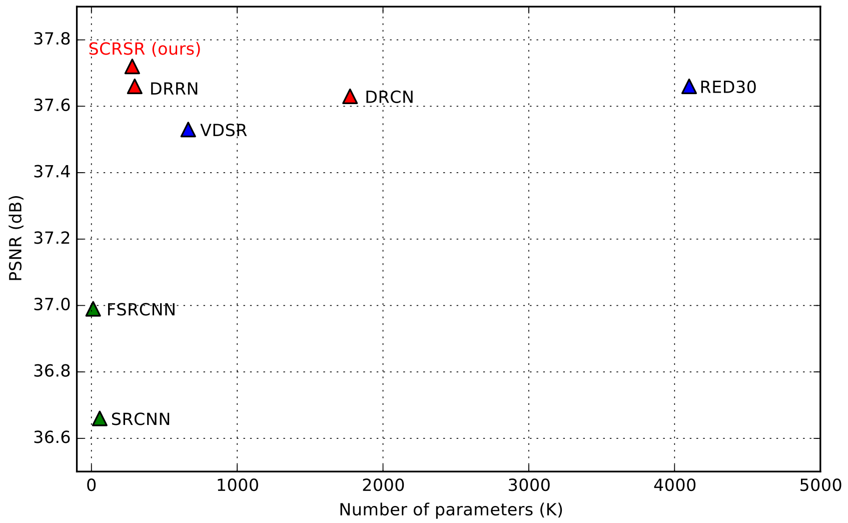


Fig. 5. PSNR and model size of recent CNN methods for scale factor × 2 on Set5. The green △ denotes shallow models with less than 10 layers. The blue △ represents deep networks. The red △ denotes recursive networks. SCRSR has the deepest network depth while the least parameters.

4. Experiment results

4.1. Traning

Dataset. For training, We use 91 images from [18] and 200 images from [19]. Following previous works, we only consider the luminance channel in YCbCr color space, because humans are more sensitive to luminance changes. For benchmark, we use four datasets: Urban100 [20], Set5 [21], Set14 [22] and BSD100 [23].

Implementation details. The training images are split into 48 × 48 sub-images with no overlap. Every feature map has the same size as the input image by padding zeros around the boundary. We train the proposed network with multiple scale factors (× 2, × 3, × 4). Every training batch consists of 64 sub-images with different scales. To augment the training data, we make two operations on them: (1) Flipping: flip images horizontally or vertically with a probability of 0.5. (2) Rotation: randomly rotate images by 90°, 180°, or 270°. The total number of training sub-images is 389622.

We use the initialization scheme described in [24] for all layers. Initial weights are randomly picked from a uniform distribution ranging from $[-1/\sqrt{c}, 1/\sqrt{c}]$, where c is the number of input channels. We train our model with ADAM optimizer [25] by setting $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^8$, $weight\_decay = 10^{-4}$. The learning rate is initialized as 10^{-4} and decreased every ten epochs by a factor of 10. The experiments were performed on an Intel Core i7-6700K 4.0-GHz CPU, 16-GB random access memory, and a Nvidia 1080 GPU. The proposed SCRSR is conducted by the Pytorch framework.

4.2. Study of N and B

We build different SCRSR models with a various combination of N and B to study their influence on network performance. In Fig. 4, network depth ranges from 10 (B = 1, N = 1) to 258 (B = 15, N = 4). Since the recursive learnings we adopt, the parameter numbers of SCRSR keep the same no matter what B and N we take. Fig. 4 shows that, before B reaches 6, the model performance grows up with B. Since the deeper network has large receptive field, providing more context to predict image details. When B is larger than 6, the model performance declines gradually. Under

Table 2
Quantitative evaluation of state-of-the-art SR methods. We evaluated average PSNR/SSIM for scale factor × 2, × 3 and × 4 on datasets Set5, Set14, BSD100, and Urban 100.

Dataset	Scale	Bicubic PSNR/SSIM	SRCNN [1] PSNR/SSIM	VDSR [2] PSNR/SSIM	LapSRN [3] PSNR/SSIM	DRRN [7] PSNR/SSIM	Ours PSNR/SSIM
Set5	× 2	33.66/0.9299	36.66/0.9542	37.53/0.9587	37.52/0.9592	37.74/0.9591	37.78/0.9593
	× 3	30.39/0.8682	32.75/0.9090	33.66/0.9213	33.78/0.9214	34.03/0.9244	33.95/0.9233
	× 4	28.42/0.8104	30.48/0.8628	31.35/0.8838	31.54/0.8850	31.68/0.8888	31.56/0.8844
Set14	× 2	30.24/0.8688	32.42/0.9063	33.03/0.9124	33.08/0.9132	33.23/0.9136	33.17/0.9133
	× 3	27.55/0.7742	29.28/0.8209	29.77/0.8314	29.87/0.8333	29.26/0.8349	29.93/0.8334
	× 4	26.00/0.7027	27.49/0.7503	28.01/0.7674	28.19/0.7720	28.21/0.7720	28.18/0.7672
BSD100	× 2	29.56/0.8431	31.36/0.8879	31.90/0.8960	31.80/0.8958	32.05/0.8973	32.06/0.8998
	× 3	27.21/0.7385	28.41/0.7863	28.82/0.7976	28.81/0.7973	28.95/0.8004	28.89/0.8017
	× 4	25.96/0.6675	26.90/0.7101	27.29/0.7251	27.32/0.7281	27.38/0.7284	27.30/0.7233
Urban100	× 2	26.88/0.8403	29.50/0.8946	30.76/0.9140	30.41/0.9101	31.23/0.9188	31.08/0.9194
	× 3	24.46/0.7349	26.44/0.8088	27.14/0.8279	27.06/0.827	27.53/0.8378	27.26/0.8367
	× 4	23.14/0.6577	24.79/0.7374	25.18/0.7524	25.21/0.7563	25.44/0.7638	25.22/0.7620

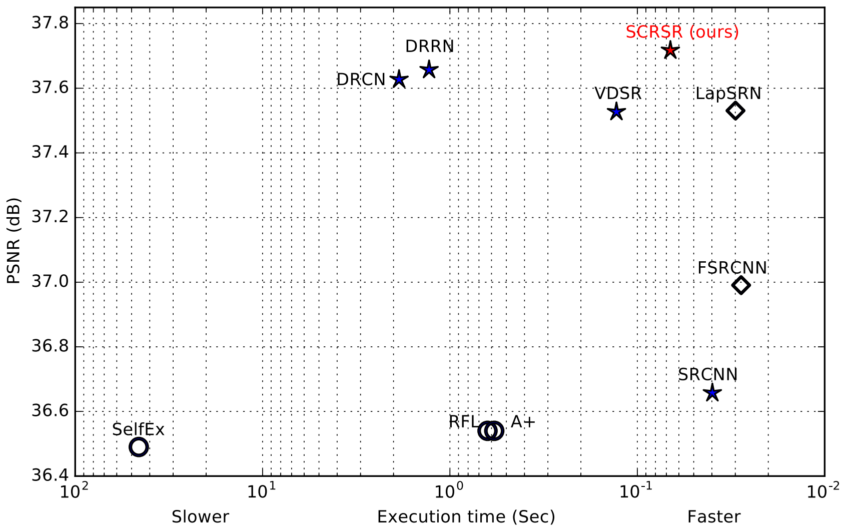


Fig. 6. Speed and accuracy trade-off of recent SR models for scale factor × 2 on Set5. Red ⋆ is our model, ⋆ means methods using ILR images as input; ◇ means to methods using IR images as input; ○ means traditional SR methods. With deeper layers but fewer parameters, SCRSR achieves better performance than the state-of-the-art methods.

same training epochs (50), when the network exceeds a certain depth, the performance of deeper network declined respectively. Considering both the performance and speed, we choose $B = 6$, $N = 2$ ($depth = 58$) as our final model. Among deep SR networks such as VDSR ($depth = 20$, $parameters = 664k$), DRCN ($depth = 20$, $parameters = 1774k$) and DRRN ($depth = 52$, $parameters = 297k$), our model SCRSR ($depth = 58$, $parameters = 280k$) is the deepest network and contains the least parameters, which is shown in Fig. 5.

4.3. Comparisons with the state-of-the-arts

Our proposed work is compared with 7 state-of-the-art SR algorithms: A+ [26], RFL [27], SelfExSR [20], SRCNN, FSRCNN, VDSR and LapSRN. Table 2 summarizes quantitative results on the four testing datasets. SCRSR outperforms those methods in these datasets.

Fig. 6 shows the trade-off between accuracy and speed when performing SR with the scale factor of 2. The results present the mean PSNR and running time over the images from Set5. The speed of SCRSR is faster than most existing methods.

In Fig. 7, we compare the results of SCRSR with some state-of-the-art methods. SCRSR reconstructs detailed textures and sharper edges in the HR images, providing noticeable improvements compared to other works.

5. Model analysis

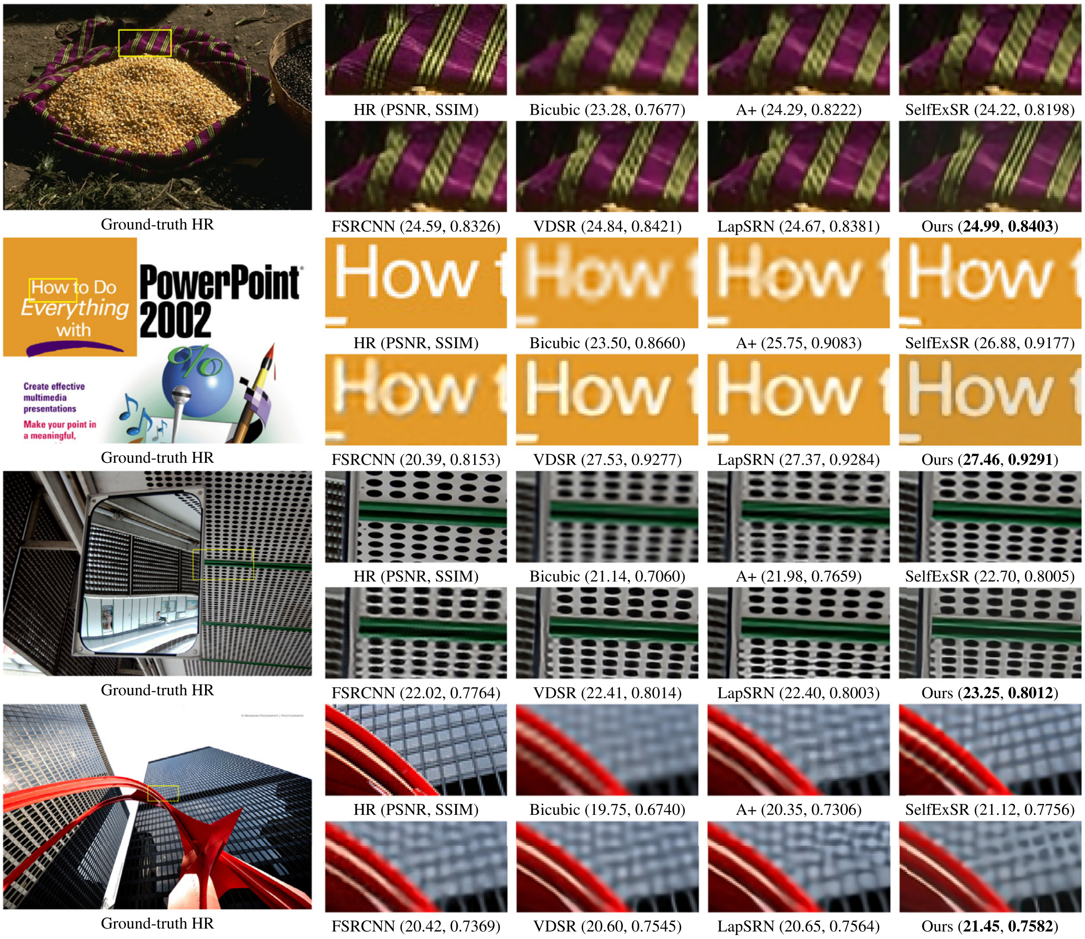
Residual Learning. To investigate the effect of three different skip connections in SCRSR, we remove them in turn. Fig. 8 shows the PSNR convergence curves of our network on Set5 for × 2 SR. The network without local residual-learning performs worst and converges slowly (cyan curve). The network without semi-global residual-learning converges slowly and fluctuates significantly (Khaki curve). Our method (blue curve) achieves the best performance.

Multi-Scale. Theoretically, our work can achieve SR problems at any scale factor. That is meaningful in real-world applications, *e.g*, when enlarging pictures on the mobile phone, we might want any scale. With our deep and powerful network, multi-scale factor super-resolution is fulfilled.

Fig. 9 shows the results of SR with various scale factors. The result of bicubic interpolation is on the first row while ourâs is on the second row. In our result, the zebra stripes are clear and vivid for arbitrary scale factors, even for fractions.

Downsample and Upsample. To evaluate the effect of downsampling block, we disable it by setting $stride = 1$, and disable the upsampling block by setting $upscale = 1$. Table 3 shows the performance of networks with or without downsampling blocks.

Compared to the network without downsampling block, the memory usage in our work reduced by 49.23%, and the testing time on dataset Set5 reduced by 61.49%. The accuracy of these



HR (PSNR, SSIM) Bicubic (23.28, 0.7677) A+ (24.29, 0.8222) SelfExSR (24.22, 0.8198)

Ground-truth HR FSRCNN (24.59, 0.8326) VDSR (24.84, 0.8421) LapSRN (24.67, 0.8381) Ours (**24.99**, **0.8403**)

HR (PSNR, SSIM) Bicubic (23.50, 0.8660) A+ (25.75, 0.9083) SelfExSR (26.88, 0.9177)

Ground-truth HR FSRCNN (20.39, 0.8153) VDSR (27.53, 0.9277) LapSRN (27.37, 0.9284) Ours (**27.46**, **0.9291**)

HR (PSNR, SSIM) Bicubic (21.14, 0.7060) A+ (21.98, 0.7659) SelfExSR (22.70, 0.8005)

Ground-truth HR FSRCNN (22.02, 0.7764) VDSR (22.41, 0.8014) LapSRN (22.40, 0.8003) Ours (**23.25**, **0.8012**)

HR (PSNR, SSIM) Bicubic (19.75, 0.6740) A+ (20.35, 0.7306) SelfExSR (21.12, 0.7756)

Ground-truth HR FSRCNN (20.42, 0.7369) VDSR (20.60, 0.7545) LapSRN (20.65, 0.7564) Ours (**21.45**, **0.7582**)

Fig. 7. "58060" (BSD100, × 2): a three-line stripe in ground truth is also observed in our result, while it is not clear in other results. "ppt3" (Set14, × 3): there are sharper edges between "H", "o", and "w" in our result. While in other work, texts are blurry. "img004" (Urban100, × 4): our image has clear boundaries, straight edges, and appropriate contrast. "img062" (Urban100, × 4): Contrast to other results, the edge of the red bridge in our result is sharp, and the color is vivid.

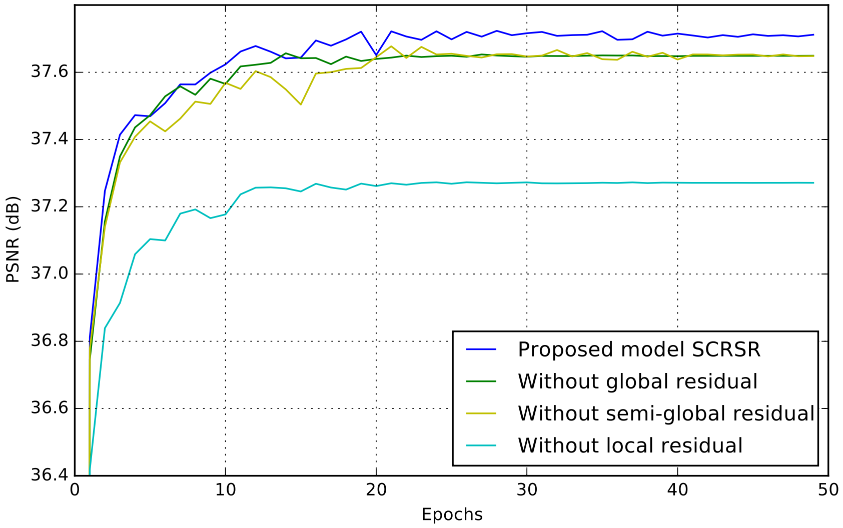


Fig. 8. Convergence analysis of residual-learning (Set5, × 2). Our SCRSR with local, semi-global, and global residual-learning converges faster and achieves improved performance.

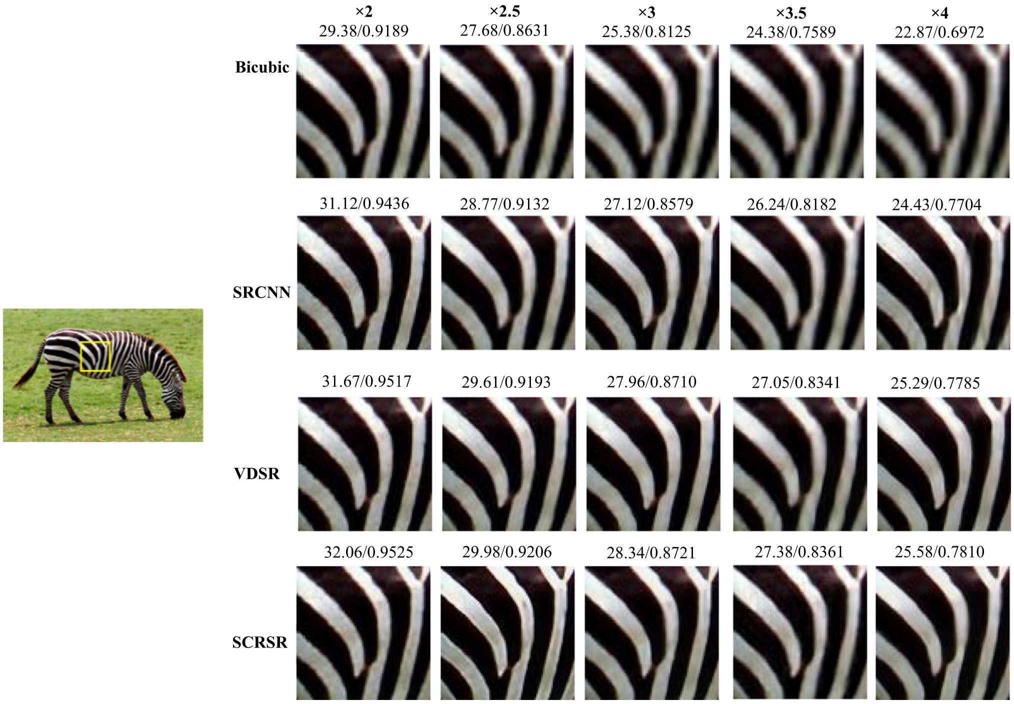


Fig. 9. SR with arbitrary scale factors compare to other methods.

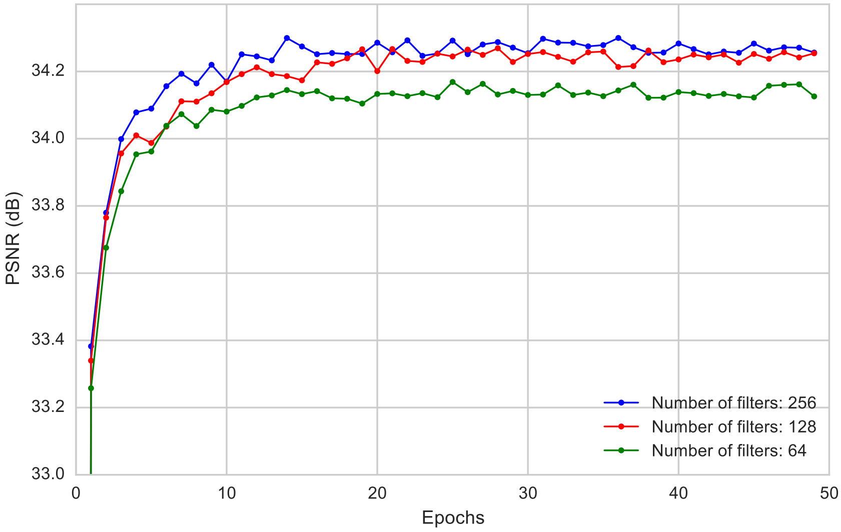


Fig. 10. Average PSNR (Set5 and Set14, × 2) of networks with different filter number.

Table 3
Experiment on downsampling blocks. Networks with or without downsampling blocks are evaluated. Dis_Down indicates the model with downsampling blocks disabled. (Set5, × 2).

Model Type	Memory Usage (MB)	PSNR (dB)	Times (Sec)
Dis_Down	2821	37.73	0.144
SCRSR	1431	37.72	0.037

networks is not much different. Overall, downsampling blocks can significantly reduce memory footprint and improve speed while maintaining same performance.

Model width. To evaluate the effect of the width of the proposed model, we set filters number n with 64, 128 and 256 and train three networks. Fig. 10 shows the performance of these networks with different width. Considering the trade-off between parameter, speed and good performance, we adopt n = 128 as the network depth of SCRSR.

6. Conclusion

In this work, we propose a deep convolutional network (SCRSR) for fast and accurate single-image super-resolution. Our method is designed to handle multi-scale SR problem. By replacing the original residual convolutional block with the SCR block, the proposed network maintains fewer parameters and higher speed. We utilize downsampling block and upsampling block to reduce computational complexity and memory footprint. Our work utilizes two level recursive learning to further reduce parameters, which achieves an effective result. With local, semi-global and global residual techniques, our network can learn image details steadily and effectively. Extensive evaluations of benchmark datasets suggest that SCRSR performs favorably against the state-of-the-art methods in terms of visual quality and run time. Except for super-resolution, SCRSR can also be applied to other low-level vision problems, such as image deblurring and denoising.

Declaration of Competing Interest

None.

References

[1] C. Dong, C.C. Loy, K. He, X. Tang, Learning a deep convolutional network for image super-resolution, in: European Conference on Computer Vision, Springer, 2014, pp. 184–199.

[2] J. Kim, J. Kwon Lee, K. Mu Lee, Accurate image super-resolution using very deep convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1646–1654.

[3] W.-S. Lai, J.-B. Huang, N. Ahuja, M.-H. Yang, Deep laplacian pyramid networks for fast and accurate super-resolution, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 624–632.

[4] C. Dong, C.C. Loy, X. Tang, Accelerating the super-resolution convolutional neural network, in: European Conference on Computer Vision, Springer, 2016, pp. 391–407.

[5] W. Shi, J. Caballero, F. Huszár, J. Totz, A.P. Aitken, R. Bishop, D. Rueckert, Z. 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, 2016, pp. 1874–1883.

[6] J. Kim, J. Kwon Lee, K. Mu Lee, Deeply-recursive convolutional network for image super-resolution, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1637–1645.

[7] Y. Tai, J. Yang, X. Liu, Image super-resolution via deep recursive residual network, Proceeding of IEEE Computer Vision and Pattern Recognition, Honolulu, HI, 2017.

[8] B. Lim, S. Son, H. Kim, S. Nah, K.M. Lee, Enhanced deep residual networks for single image super-resolution, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, 2017, pp. 1132–1140.

[9] X.-J. Mao, C. Shen, Y.-B. Yang, Image Restoration Using Very Deep Convolutional Encoder-Decoder Networks with Symmetric Skip Connections, Curran Associates Inc, 2016, pp. 2810–2818.

[10] Z. Tang, L. Luo, H. Peng, S. Li, A joint residual network with paired relus activation for image super-resolution, Neurocomputing 273 (2018) 37–46.

[11] T. Tong, G. Li, X. Liu, Q. Gao, Image super-resolution using dense skip connections, in: 2017 IEEE International Conference on Computer Vision (ICCV), IEEE, 2017, pp. 4809–4817.

[12] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, Y. Fu, Residual dense network for image super-resolution, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[13] K. He, X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in: European Conference on Computer Vision, 2016, pp. 630–645.

[14] E. Agustsson, R. Timofte, Ntire 2017 challenge on single image super-resolution: dataset and study, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2017.

[15] L Christian, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, et al., Photo-realistic single image super-resolution using a generative adversarial network, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4681–4690.

[16] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, in: International Conference on Machine Learning, 2015, pp. 448–456.

[17] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[18] J. Yang, J. Wright, T.S. Huang, Y. Ma, Image super-resolution via sparse representation, IEEE Trans. Image Process. Publ. IEEE Signal Process. Soc. 19 (11) (2010) 2861.

[19] D. Martin, C. Fowlkes, D. Tal, J. Malik, A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, in: Proceedings of Eighth IEEE International Conference on Computer Vision, 2001. ICCV 2001, 2, 2002, pp. 416–423.

[20] J.-B. Huang, A. Singh, N. Ahuja, Single image super-resolution from transformed self-exemplars, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5197–5206.

[21] M. Bevilacqua, A. Roumy, C. Guillemot, A. Morel, Low-complexity single-image super-resolution based on nonnegative neighbor embedding, BMVC, 2012.

[22] R. Zeyde, M. Elad, M. Protter, On single image scale-up using sparse-representations, in: International Conference on Curves and Surfaces, 2010, pp. 711–730.

[23] P. Arbelaez, M. Maire, C. Fowlkes, J. Malik, Contour detection and hierarchical image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 33 (5) (2011) 898–916.

[24] G.B. Orr, K.-R. Müller, Neural Networks: Tricks of the Trade, Springer, 2003.

[25] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014 arXiv preprint arXiv:1412.6980.

[26] R. Timofte, V. De Smet, L. Van Gool, A+: adjusted anchored neighborhood regression for fast super-resolution, in: Asian Conference on Computer Vision, Springer, 2014, pp. 111–126.

[27] S. Schulter, C. Leistner, H. Bischof, Fast and accurate image upscaling with super-resolution forests, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3791–3799.

Daoyu Lin received the bachelor's degree from Beijing University of Posts and Telecommunications. He is currently a graduate student in Institute of Electronics, University of Chinese Academic of Sciences, working computer vision and image processing.

Guangluan Xu received the B.Sc. degree from the Beijing Information Science and Technology University, Beijing, China, in 2000, and the M.Sc. and Ph.D. degrees from the Institute of Electronics, Chinese Academy of Sciences, Beijing, in 2005. He is currently an Professor with the Institute of Electronics, Chinese Academy of Sciences. His research interests include computer vision and remote sensing image understanding.

Wenjia Xu received the bachelor's degree with distinction from Beijing Institute of Technology. She is currently a Ph.D. candidate in Institute of Electronics, University of Chinese Academic of Sciences, working on machine learning and computer vision.

Yang Wang received the B.E. degree from Beihang University, Beijing, China, and the Ph.D. degree from the Peking University, Beijing, China. She is currently an Assistant Professor with the Institute of Electronics, Chinese Academy of Sciences, Beijing. Her research interests include the geospatial data organization and visualization.

Xian Sun received the B.Sc. degree from the Beihang University, Beijing, China, in 2004, and the M.Sc. and Ph.D. degrees from the Institute of Electronics, Chinese Academy of Sciences, Beijing, in 2009. He is currently an Associate Professor with the Institute of Electronics, Chinese Academy of Sciences. His research interests include computer vision and remote sensing image understanding.

Kun Fu received the B.Sc. and Ph.D. degrees from the National University of Defense Technology, Changsha, China, in 1995 and 2002, respectively. He is currently the Head and a Professor with the Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, Institute of Electronics, Chinese Academy of Sciences, Beijing, China. He has authored two books and over 60 refereed publications. His research interests include geospatial data organization and visualization, computer vision, and remote sensing image interpretation.