Signal Processing 172 (2020) 107567

Contents lists available at ScienceDirect

Signal Processing

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

Accurate single image super-resolution using multi-path wide-activated residual network

Kan Chang[a,*], Minghong Li[a], Pak Lun Kevin Ding[b], Baoxin Li[b]

[a] School of Computer and Electronic Information, Guangxi University, Nanning 530004, China
[b] Department of Computer Science and Engineering, Arizona State University, Tempe 85287, USA

ARTICLE INFO

Article history:
Received 24 October 2019
Revised 8 February 2020
Accepted 3 March 2020
Available online 4 March 2020

Keywords:
Super-resolution
Convolutional neural network
Residual learning
Multi-Scale learning
Channel attention

ABSTRACT

In many recent image super-resolution (SR) methods based on convolutional neural networks (CNNs), the superior performance was achieved by training very large networks, which may not be suitable for real-world applications with limited computing resources. Therefore, it is necessary to develop more compact networks that achieve a better trade-off between the model size and the performance. In this paper, we propose an efficient and effective network called multi-path wide-activated residual network (MWRN). Firstly, as the basic building block of MWRN, the multi-path wide-activated residual block (MWRB) is presented to extract the multi-scale features. MWRB consists of three parallel wide-activated residual paths, where the dilated convolutions with different dilation factors are used to increase the receptive fields. Secondly, the fusional channel attention (FCA) module, which contains a bottleneck layer and a multi-path wide-activated residual channel attention (MWRCA) block, is designed to well exploit the multi-level features in MWRN. In each FCA, the MWRCA block refines the fused features by taking the interdependencies among feature channels into consideration. The experiments demonstrate that, compared with the state-of-the-art methods, the proposed MWRN model is able to provide very competitive performance with a relatively small number of parameters.

1. Introduction

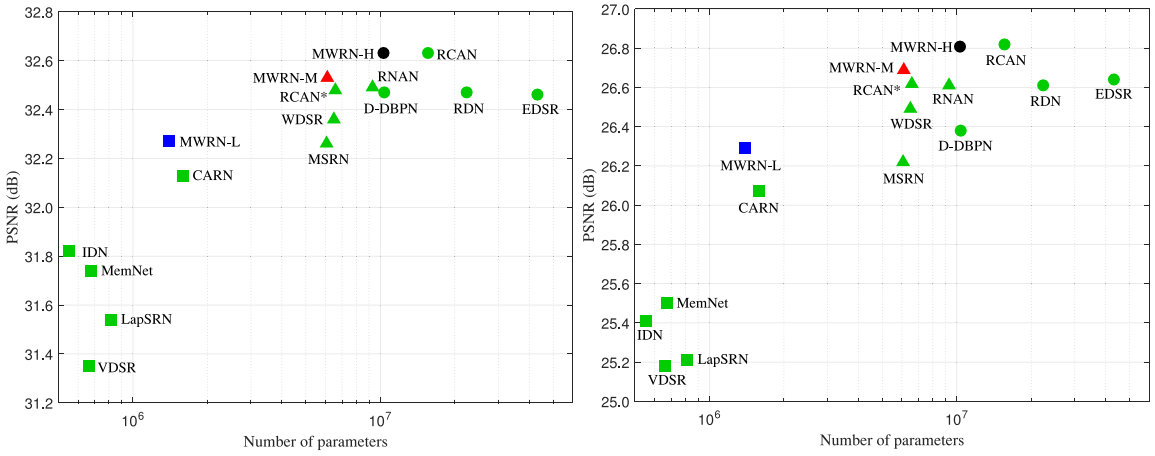
Single image super-resolution (SR) aims to find the corresponding high-resolution (HR) estimation for a given low-resolution (LR) image. This task is challenging due to its ill-posed nature. In the past two decades, many SR methods have been developed [1], and the three traditional types of methods include the interpolation-based methods, the reconstruction-based methods and the learning-based methods.

Among the three traditional types of SR methods, the interpolation-based methods have the lowest computational complexity, but usually cannot well recover the image details. In the reconstruction-based methods [2–9], the SR problem is considered as the maximum a posterior problem [1]. By properly exploring the image priors, this type of methods is able to well preserve the structures in images. However, solving the inverse problem often requires a large amount of computation, which hampers the usage of the reconstruction-based methods in the real-world applications. In the learning-based methods [10–16], the mapping relationship from the LR space to the HR space is learned offline. Since the offline training is carried out on external datasets, the learning-based methods are good at restoring the fine details. Although the above traditional methods have been studied for quite a long time, due to the usage of hand-crafted priors or models, it is still difficult for them to achieve very satisfactory results.

With recent developments in deep learning, the methods based on convolutional neural networks (CNNs) have achieved remarkable performance. In the very beginning, some shallow networks were proposed, such as SRCNN [17], FSRCNN [18] and ESPCN [19]. It is well known that by increasing the depth of CNN, the representation power of the networks can be enhanced. However, training a very deep model is difficult due to the vanishing-gradient and exploding-gradient problems. To alleviate these problems, the strategies of residual learning [20–27] and dense connections [28–32] have been widely used. In addition, many special strategies have also been studied to improve the representation ability of the networks, such as the multi-scale learning [33–38], the attention mechanism [39–43], the non-local networks [43–45], etc.

* Corresponding author.
E-mail addresses: changkan0@gmail.com (K. Chang), minghongli233@gmail.com (M. Li), kevinding@asu.edu (P.L.K. Ding), baoxin.li@asu.edu (B. Li).

https://doi.org/10.1016/j.sigpro.2020.107567
0165-1684/



(a) Dataset: *Set5* (×4) (b) Dataset: *Urban100* (×4)

Fig. 1. Performance comparisons among different SR models. The symbols □, △ and ○ indicate the lightweight, middleweight, and heavyweight models, respectively.

When designing a CNN model, besides the quality of the results, another important consideration is the number of parameters required by the model. Although a deeper and wider CNN often leads to better performance, the number of parameters required by such a network could increase dramatically. In many real-world scenarios, such as super-resolving images on a mobile device, it is impractical to deploy a model that is too large. Therefore, it is always important to be aware of the trade-off between the model size and the performance.

In this paper, an efficient and effective CNN model called multi-path wide-activated residual network (MWRN) is proposed, and three versions of MWRN, including a lightweight version (denoted as MWRN-L), a middleweight version (denoted as MWRN-M) and a heavyweight version (denoted as MWRN-H), are trained in our experiments. In Fig. 1, the average peak signal-to-noise ratio (PSNR) values obtained by different models and the numbers of parameters consumed by these models are shown. In this paper, we define the models with numbers of parameters less than 2 million (M), less than 10 M and more than 10 M as the lightweight, middleweight and heavyweight models, respectively. It can be observed that, compared with the recently proposed lightweight model CARN [25], MWRN-L has significantly better performance, yet requires a smaller number of parameters; with the lowest cost of parameters, MWRN-M achieves the best performance among all the middleweight models; MWRN-H obtains almost the same performance as the state-of-the-art method RCAN [40], but the number of parameters of MWRN-H is less than that of RCAN [40].

In summary, the main contributions of this paper are threefold:

(1) To efficiently and effectively extract multi-scale image features, we propose a basic building block named multi-path wide-activated residual block (MWRB), where the input features are sliced into three parts, each of which will go through one of the three parallel wide-activated residual paths. To reduce the required parameters, on the three wide-activated residual paths, instead of the standard convolutions, the dilated convolutions with different dilation factors are applied to reach large receptive fields. As will be discussed later, compared with other multi-scale learning blocks such as MSRB [36], the proposed MWRB is much more efficient.

(2) A fusional channel attention (FCA) mechanism is presented to fully take advantage of different levels of features within the network. For this purpose, we set up dense connections among different FCA modules, and then fuse the features from different states of MWRN. Unlike other works where the channel attention (CA) unit is integrated into each building block or module [40,43], we only carry out CA on the fused features in the FCA modules.

(3) The MWRN, which is constructed by stacking many feature enhancement groups (FEGs), is built for the single image SR task. Each FEG consists of a cascaded-blocks-based (CB) module and a FCA module, and each CB module contains several cascaded MWRBs. For further study and evaluation of the proposed method, our code will be released at https://github.com/minghongli233/MWRN.

The rest of this paper is organized as follows. Section 2 briefly reviews the related background. The proposed building block MWRB, the structure of MWRN, and the mechanism of FCA are detailed in Section 3. Extensive experiments and discussions are provided in Section 4. Finally, we conclude this paper in Section 5.

2. Related background

2.1. CNN-based SR methods

As the first CNN-based SR method, SRCNN [17] has only three convolution layers, which limits the performance of the network. Later on, by using the skip connection, Kim et al. [20] trained a 20-layer CNN called VDSR, which achieves noticeable improvement over SRCNN. However, both of the SRCNN and VDSR extract features from the bicubic-interpolated LR images, which is not a good choice as the details might be lost and the computational burden is heavy [46].

To address the above problem, many methods directly extract features from the input LR images, and utilize the deconvolution layer (also known as the transpose convolution layer) [18] or the sub-pixel layer [19] to upscale the extracted features. Doing so is beneficial for training a very deep networks for the SR task, as the computational burden is lower and the extracted features are more accurate. Furthermore, by taking advantage of different training tricks and skills (e.g., residual learning [21] and dense connections [28]), many methods trained models of very large scales, so that the superior performance can be achieved. For example, both RDN [31] and RCAN [40] have more than 15 M parameters, and the number of parameters of EDSR [24] even reaches 43 M.

Researchers have noticed that in many real-world scenarios, deploying a huge CNN model is impractical. To reduce the required parameters, the recursive convolutions were adopted in

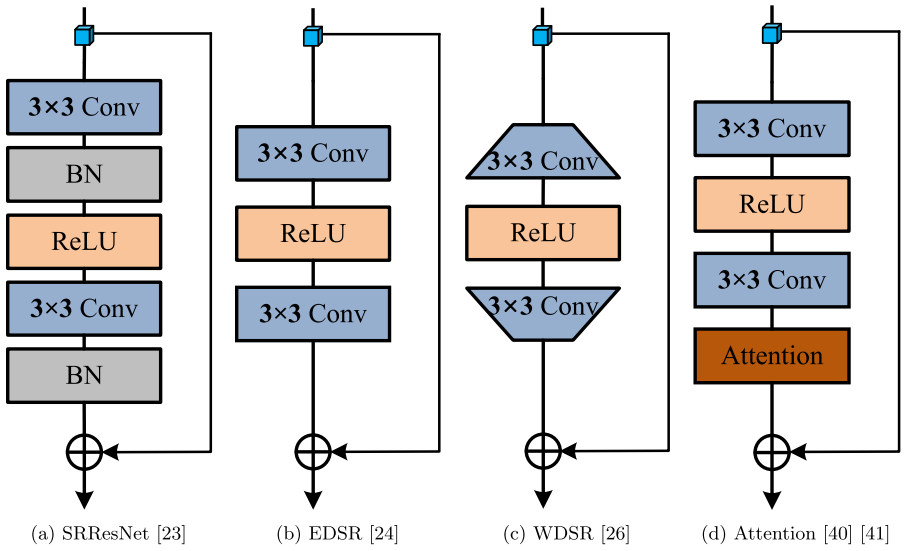


Fig. 2. The structures of different residual learning blocks. 'Conv', 'ReLU', 'BN' and 'Attention' stand for the convolution layer, rectified linear unit, batch normalization and the attention mechanism, respectively.

some networks, such as DRCN [47], DRRN [48], MemNet [29], etc. However, performing recursive convolutions still introduces intensive computation. In the mobile version of CARN (CARN-M) [25], group convolution was adopted instead of the normal convolution, but this operation could impair performance. Besides the above models, other lightweight networks include FSRCNN [18], LapSRN [22], IDN [49], etc.

2.2. Residual learning

Residual learning [21] is one of the most important strategies to ease the difficulty of training large-scale networks. In [20], Kim et al. introduced a global skip connection, so that the network can focus on predicting the residual images. In [21], residual learning was adopted to a few stacked layers, leading to a basic building block of CNN called ResNet. With the short-term skip connection, rich information can easily pass through each ResNet block. Since ResNet was originally established for the image recognition task, many works have modified the structure of this building block, so that it can work well in the SR task. Fig. 2 illustrates several variations of the residual-learning-based building blocks.

Compared with ResNet, the building block of SRResNet [23] does not have the activation layer after the element-wise addition. It was argued in [24] that batch normalization (BN) might not be suitable for the SR task, and thus the two BN layers were removed, leading to the building block of EDSR. Later on, the wide-activated SR (WDSR) block [26,50] was proposed, where the features are expanded before the activation layers, so that more information can easily pass through. In order to reduce the number of parameters and the computational complexity, the second 3×3 convolution layer in WDSR shrinks the expanded features to the original dimension. Due to the fact that the normal building block lacks discriminative learning ability, in [40] and [41], the attention mechanism is further integrated into the residual learning block, and the 'Attention' in Fig. 2 (d) could be channel attention (CA), spatial attention (SA) or their combination.

2.3. Multi-scale learning

The first multi-scale learning architecture is the inception module [33], which is the basic building block of the GoogLeNet network [33] for the image recognition task. In the early version of inception module, along with max-pooling, 1×1, 3×3 and 5×5 convolutions are simultaneously utilized. By doing so, the information at different scales can be well captured. Afterwards, Szegedy et al. [34] further proposed to combine the inception module with skip connections, and showed that this type of variations can significantly improve the performance of image recognition.

Inspired by the inception module and its variations, the strategy of multi-scale learning has also been applied to extract image features for the SR task [35–38]. In Fig. 3 (a), the structure of a typical multi-scale learning block called multi-scale residual block (MSRB) [36] is shown. As can be seen, there are two main paths in MSRB, where one path consists of two stacked 3×3 convolution layers, while the other path contains two stacked 5×5 convolution layers. To let different paths share their information, the output of the first 3×3 or 5×5 convolution layer is additionally connected to the input of the second 5×5 or 3×3 convolution layer. Although image features of different scales can be well detected by MSRB, it is obvious that this structure consumes many parameters and much computation. Due to high complexity, it is impractical to stack many MSRBs to construct a network for the SR task. Therefore, it is necessary to design a more efficient multi-scale-learning-based building block.

3. The proposed method

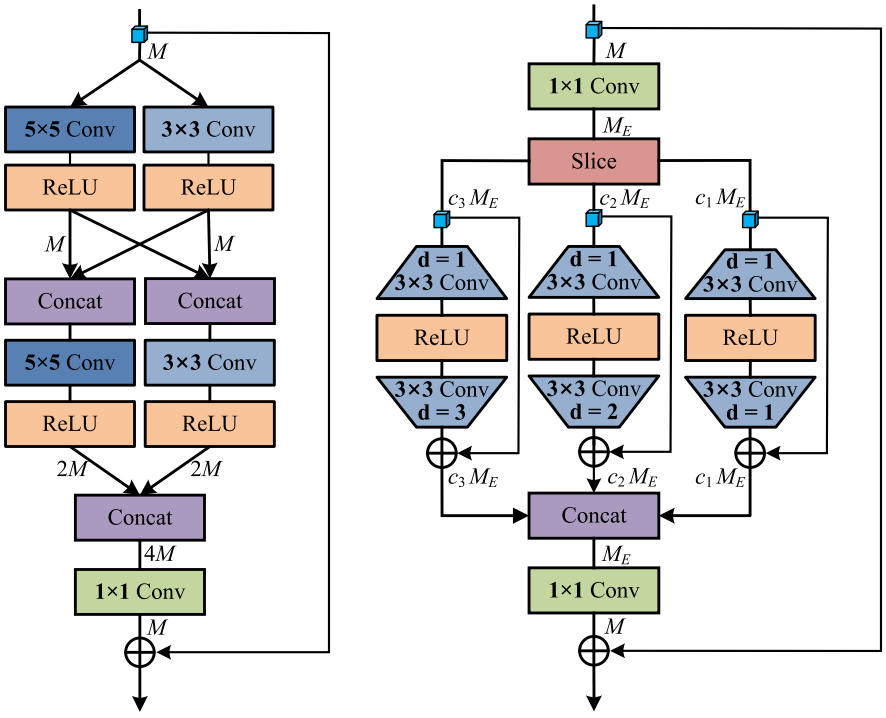
3.1. Multi-path wide-activated residual block (MWRB)

Most advanced deep models are established by stacking multiple building blocks. In this section, an efficient and effective building block called MWRB is presented. With the detailed structure shown in Fig. 3**(b)**, the overall block of MWRB can be represented by

$$\mathbf{X}_t = F_R(F_M(F_E(\mathbf{X}_{t-1}))) + \mathbf{X}_{t-1} \tag{1}$$

where $\mathbf{X}_{t-1}$ and $\mathbf{X}_t$ are the input and the output of the tth MWRB, respectively; $F_E(.)$ and $F_R(.)$ denote the functions of feature extension and reduction by using 1×1 convolution layers, respectively; $F_M(.)$ represents the multi-scale feature extraction function, which is based on three parallel wide-activated residual units.

It was demonstrated in [26] that expanding the features before the activation layers helps to improve the performance. Therefore, we apply $F_E(.)$ and $F_R(.)$ in the front and the back of the MWRB,



(a) MSRB [36] (b) The proposed MWRB

Fig. 3. The structures of different multi-scale learning blocks. 'Concat' and 'Slice' represent the channel-wise concatenation and slice operations, respectively. '3 × 3 Conv' with 'd = 1', '2', and '3' stand for the 3 × 3 dilated convolutions with a dilation factor of 1, 2 and 3, respectively.

respectively. Assuming that the dimension of $\mathbf{X}_{t-1}$ is M, we use a 1 × 1 convolution layer to expand the dimension from M to M_E, and another 1 × 1 convolution layer to reduce the dimension from M_E back to M. As shown in Fig. 3 **(b)**, the output of the first 1 × 1 convolution will be sliced into three parts, and then the signals from different paths will be concatenated before entering the second 1 × 1 convolution. As a result, the first 1 × 1 convolution layer also implicitly learns to pack the features which will be sent to the following three paths. On the other hand, the second 1 × 1 convolution layer works as a bottleneck layer, which is responsible for fusing and compressing the information coming from different paths.

Let us denote $F_E(\mathbf{X}_{t-1})$ by $\mathbf{X}^E_{t-1}$. The major part of our MWRB can be expressed by

$$F_M(\mathbf{X}^E_{t-1}) = C(W_1(\mathbf{P}_1), W_2(\mathbf{P}_2), W_3(\mathbf{P}_3)) \tag{2}$$

where $C(.)$ stands for the channel-wise concatenation; $W_i(.)$ represents the operation of the ith wide-activated residual path, with $i \in \{1, 2, 3\}$; and

$$\mathbf{P}_i = S(\mathbf{X}^E_{t-1}, c_i) \tag{3}$$

where $S(\mathbf{X}^E_{t-1}, c_i)$ means slicing $c_i M_E$ channels of features out from $\mathbf{X}^E_{t-1}$, with the slicing factor $c_i \in (0, 1)$. Obviously, we have $\mathbf{X}^E_{t-1} = C(\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3)$ and $c_1 + c_2 + c_3 = 1$.

To fully explore multi-scale information, we design three different wide-activated residual paths. The structure of the first path is the same as the normal WDSR block [26]. However, the 3 × 3 dilated convolutions are used for feature reduction on the second and third paths, where the dilation factors are respectively chosen as 2 and 3. With much fewer parameters, the 3 × 3 dilated convolutions with the dilation factors of 2 and 3 can reach the same sizes of the receptive fields as the 5 × 5 and 7 × 7 convolutions, respectively. Therefore, it is reasonable to apply different dilated convolutions to detect different scales of features on different paths. However, directly using the dilated convolutions would induce blind spots in the receptive field [51]. In order to alleviate the effects of blind spots on the second and third paths, the standard convolutions, rather than the dilated convolutions, are used for feature expanding.

Compared with MSRB [36], the proposed MWRB has three main advantages:

(1) Due to the usage of dilated convolutions, with a small number of parameters, each MWRB is able to reach a maximum receptive field of 9 × 9, which is the same as that achieved by stacking two standard 5 × 5 convolution layers.
(2) By using the slice operation, the features are divided into different parts, and then delivered to different feature extraction paths in MWRB, which is quite different from the structure of MSRB shown in Fig. 3 **(a)**. As a result, the number of parameters required by the MWRB is largely reduced.
(3) MWRB additionally absorbs the benefits of wide-activated residual learning, resulting in an even stronger representation ability of this building block.

Now let us compare the number of parameters of an MWRB with that of an MSRB [36]. Supposing that the dimension of the input features is M, according to Fig. 3 **(a)**, the 3 × 3 path in an MSRB unit requires $9M^2 + 9 \times (2M)^2 = 45M^2$ parameters, while the 5 × 5 path costs $25M^2 + 25 \times (2M)^2 = 125M^2$ parameters. As a result, the total number of parameters of an MSRB reaches $45M^2 + 125M^2 + 4M^2 = 174M^2$. Since M is set as 64 in [36], a single MSRB has 712.7 kilo (K) parameters. On the other hand, according to the structure in Fig. 3 **(b)**, by setting the expansion factor of each wide-activated residual path as 4, the three paths in an MWRB have a total of $2 \times 9 \times 4 \times \left[(c_1 M_E)^2 + (c_2 M_E)^2 + (c_3 M_E)^2\right]$ parameters. By taking the two 1 × 1 convolution layers into account, the number of parameters required by an MWRB is $72M_E^2(c_1^2 + c_2^2 + c_3^2) + 2MM_E$. According to the settings in Section 4.1, with $M = 32$, $M_E = 48$, $c_1 = 3/6$, $c_2 = 2/6$, $c_3 = 1/6$, an MWRB unit only costs 67.6 K parameters, which is much

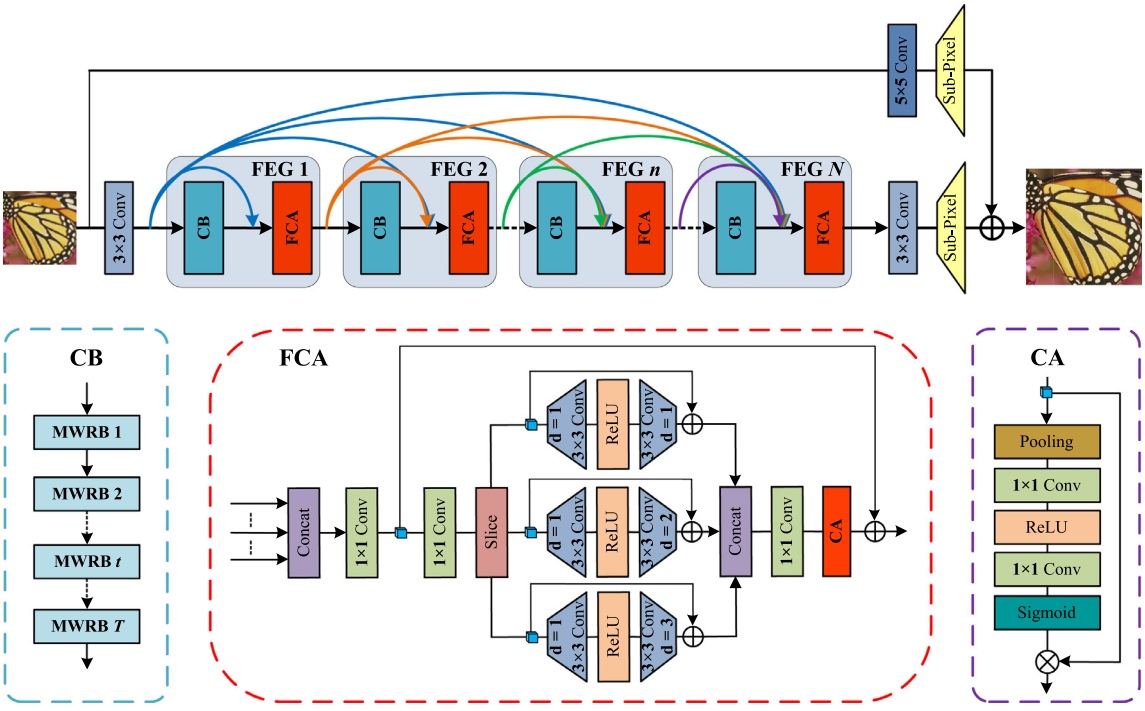


Fig. 4. The structures of the MWRN and the FCA module.

smaller than the number required by a single MSRB. Therefore, it is reasonable to stack many MWRBs to form a deep network, so that more plausible SR results can be obtained.

3.2. Overall network structure

The overall structure of our MWRN is shown in Fig. 4. Similar to other traditional SR models, MWRN can be divided into three stages, including a shallow feature extraction stage, a nonlinear mapping stage and an image reconstruction stage.

At the shallow feature extraction stage, low-level features $\mathbf{X}_0$ are extracted from the LR color image $\mathbf{I}_{LR}$, i.e.,

$$\mathbf{X}_0 = F_S(\mathbf{I}_{LR}) \tag{4}$$

where $F_S(.)$ represents shallow feature extraction by using a 3×3 convolution layer.

At the nonlinear mapping stage, the features are progressively refined by N FEGs. Denoting the output of the nth FEG by $\mathbf{X}_n$, we have

$$\mathbf{X}_n = G_n(\mathbf{X}_{n-1}) = G_n(G_{n-1}(...(G_1(\mathbf{X}_0)))) \tag{5}$$

where $G_n(.)$ is the function of the nth FEG. In each FEG, there is a CB module followed by a FCA module. The CB module, which is formed by stacking T MWRBs, is responsible for extracting deep features. The FCA module, on the other hand, fuses the hierarchical features from the previous states, and then further explores statistics of the fused features. The structure of FCA will be detailed in Section 3.3.

At the image reconstruction stage, the refined features $\mathbf{X}_N$ are upscaled to the HR space. Similar to [26], the final HR color image $\mathbf{I}_{HR}$ can be established by

$$\mathbf{I}_{HR} = F_{FU}(\mathbf{X}_N) + F_{IU}(\mathbf{I}_{LR}) \tag{6}$$

where $F_{FU}(.)$ and $F_{IU}(.)$ denote the upsampling functions for the refined features and the LR color image, respectively. In practice, both $F_{FU}(.)$ and $F_{IU}(.)$ are implemented by cascading a standard convolution layer (3×3 for $F_{FU}(.)$ and 5×5 for $F_{IU}(.)$) and a sub-pixel convolution layer [19].

3.3. Fusional channel attention (FCA) module

Now we present the details of the FCA module. Since the MWRN contains a set of stacked MWRBs, there exists rich multi-level information within the network. Therefore, to make full use of the previous states, dense connections are built among different states in MWRN. As can be seen in Fig. 4, the outputs of the preceding FEGs, as well as the shallow features $\mathbf{X}_0$, are directly connected to the input of the FCA module, where a 1×1 convolution layer is introduced as a bottleneck layer to fuse the input features.

For the FCA module in the nth FEG, the fusion operation can be expressed by

$$\mathbf{X}_n^F = F_B(C(\mathbf{X}_{T,n}, \mathbf{X}_{n-1}, \mathbf{X}_{n-2}, ...\mathbf{X}_1, \mathbf{X}_0)) \tag{7}$$

where $F_B(.)$ denotes the function of the bottleneck layer in the FCA module, and $\mathbf{X}_{T,n}$ is the output of the CB module in the nth FEG.

Since $\mathbf{X}_n^F$ contains multiple levels of information, it might provide more clues for the image SR task. Therefore, it is necessary to pay special attention to $\mathbf{X}_n^F$. To this end, we integrate the CA mechanism [39] into MWRB, and propose a multi-path wide-activated residual channel attention (MWRCA) block to refine $\mathbf{X}_n^F$. As a result, the output of the FCA module in the nth FEG can be achieved by

$$\mathbf{X}_n = F_A(F_R(F_M(F_E(\mathbf{X}_n^F)))) + \mathbf{X}_n^F \tag{8}$$

where $F_A(.)$ stands for the function of the CA unit, which is applied to rescale the channel-wise features. Assuming that $\mathbf{X}_c = F_R(F_M(F_E(\mathbf{X}_n^F)))$, as shown in Fig. 4, $F_A(.)$ can be expressed as

$$F_A(\mathbf{X}_c) = \mathbf{X}_c \odot \mathbf{S}_c \tag{9}$$

where $\odot$ stands for channel-wise multiplication; $\mathbf{S}_c$ is a vector containing the scaling coefficients for the corresponding feature channels in $\mathbf{X}_c$, which is obtained by

$$\mathbf{S}_c = F_G(F_P(\mathbf{X}_c)) \tag{10}$$

where $F_P(.)$ and $F_G(.)$ stand for the functions of the global average pooling and the gating mechanism [39], respectively. Suppose that the input feature $\mathbf{X}_c$ of the CA unit has a dimension of $N_c \times H \times W$,

with $H \times W$ denoting the spatial dimension and N_c being the number of channels. If we write $\mathbf{X}_p = F_P(\mathbf{X}_c)$, then $\mathbf{X}_p$ contains the channel-wise statistics with a dimension of $N_c \times 1 \times 1$. After the global average pooling, $\mathbf{S}_c$ is generated by the gating mechanism [39] as below:

$$F_G(\mathbf{X}_p) = f_s(f_u(f_r(f_d(\mathbf{X}_p)))) \tag{11}$$

where $f_d(.)$ stands for the first 1×1 convolution layer which downscales the aggregated information to a dimension of $\frac{N_c}{r} \times 1 \times 1$, with r being the reduction ratio; $f_r(.)$ represents the function of the ReLU layer, which is responsible for learning the nonlinear interactions among channels; $f_u(.)$ indicates the second 1×1 convolution layer, which upscales the information back to the dimension of $N_c \times 1 \times 1$; $f_s(.)$ is the sigmoid function which generates the final $\mathbf{S}_c$.

Note that, in many works (e.g., [40,41]), the CA unit is integrated into each residual learning block, leading to the architecture shown in Fig. 2 **(d)**. Different from those methods, we only integrate the CA unit into the FCA module, which exists at the end of each FEG. In our MWRN, the cross-stage skip connections not only help the information easily flow within the network, but also deliver multiple levels of information to each FEG. After being fused by the bottleneck layer in FEG, these fused hierarchical features contain rich information for image SR. As the CA mechanism is only applied to the fused hierarchical features, our strategy is obviously more efficient.

4. Experimental results

4.1. Experimental settings

Following the recent works such as [25,31,36], DIV2K [52] is used as the training dataset. Five widely used datasets, including Set5 [53], Set14 [11], BSD100 [13], Urban100 [14] and Manga109 [54], are selected for testing. To generate the LR observations, the original images from the above datasets are downsampled by using the bicubic interpolation. In the following sections, the PSNR and the structural similarity index (SSIM) are utilized to measure the quality of the Y channel of a super-resolved image.

During training, each batch consists of 16 LR RGB patches with a size of 48×48. Data augmentation, such as random rotation and flip, is conducted on 800 training images from the DIV2K dataset. Following EDSR [24] and WDSR [26], we also subtract the mean RGB values from the training images.

We train a lightweight version ($T = 4$, $N = 4$), a middleweight version ($T = 10$, $N = 8$), and a heavyweight version ($T = 14$, $N = 10$) of MWRN, which are respectively denoted as MWRN-L, MWRN-M, and MWRN-H. In each MWRB, the feature dimensions M and M_E are respectively chosen as 32 and 48, and the slicing factors c_1, c_2 and c_3 for three paths are set as 3/6, 2/6 and 1/6, respectively. The expansion factor of the wide-activated residual unit is fixed as 4. The reduction ratio of the CA unit is set as 8.

Denoting the kth LR and HR RGB training patch pair by $\{\mathbf{x}^{(k)}, \mathbf{y}^{(k)}\}$, the objective function for training the MWRN models is written as

$$\mathcal{L}(\mathbf{\Theta}) = \frac{1}{N_p} \sum_{k=1}^{N_p} \| \mathcal{F}_{\mathrm{MWRN}}(\mathbf{x}^{(k)}; \mathbf{\Theta}) - \mathbf{y}^{(k)} \|_1 \tag{12}$$

where $\mathcal{F}_{\mathrm{MWRN}}(\cdot)$ and $\mathbf{\Theta}$ stand for the function and the parameter set of MWRN, respectively; N_p represents the total number of the training patch pairs in each batch, which is fixed as 16 in the following experiments.

For a fair comparison, the settings of the optimizer, the initialization method and other training details of MWRN-L/MWRN-M/MWRN-H are kept the same as that of WDSR [26]. Specifically, ADAM with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$ is chosen as the optimizer, and the initialization method proposed by He et al. [55] is applied. As suggested by [26], instead of batch normalization, weight normalization [56] is used. The total number of epochs is set as 1000. The initial learning rate is 10^{-3}, and it will decrease to half of the previous value after every 200 epochs. Note that all the models with different scale factors are trained from scratch. The proposed networks are implemented with the PyTorch framework and trained by using a GTX 1080Ti GPU. [1]

Table 1
The effects of M_E ($\times$ 2).

M_E	36	42	48	54
Parameters (K)	787	1058	1370	1722
PSNR (dB)	32.30	32.38	32.46	32.45

Table 2
The effects of the expansion factor of the wide-activated residual unit ($\times$ 2).

Expansion Factor	2	3	4	5
Parameters (K)	721	1045	1370	1694
PSNR (dB)	32.22	32.34	32.46	32.47

Table 3
The effects of the slicing factors c_1, c_2 and c_3 ($\times$ 2).

c_1: c_2: c_3	1: 2: 3	1: 1: 2	1: 1: 1	2: 1: 1	3: 2: 1
Parameters (K)	1370	1324	1185	1324	1370
PSNR (dB)	32.37	32.37	32.43	32.39	32.46

4.2. Empirical study of the parameters of MWRB

Since MWRB is the basic building block of the MWRN model, we study the most important parameters of MWRB in this section. To do so, we build a network which consists of 20 MWRBs (thus this network is equivalent to the network $\mathbb{N}_d$ in Section 4.3), and then evaluate the effects of the parameters of MWRB by changing their values. All the results are obtained with a scale factor of 2 and on the Urban100 dataset.

(1) *The Effects of M_E* : M_E is the feature dimension of the output of the first 1×1 convolution layer in MWRB. By keeping the feature dimension of the input of MWRB as $M = 32$, we analyze the effects of M_E, and the results are shown in Table 1. It can be found that as M_E varies from 36 to 48, the average PSNR of the SR results is improving, and the number of parameters is also increasing. However, for the three-wide-activated-path situation, no further improvement is observed when M_E is set as 54. Therefore, it is reasonable to choose $M_E = 48$, which achieves the best performance with acceptable number of parameters.

(2) *The Effects of the Expansion Factor of the Wide-Activated Residual Unit*: The effects of the expansion factor of each wide-activated residual unit in MWRB are listed in Table 2. As can be seen, when the expansion factor varies from 2 to 5, the PSNR value gradually increases. However, the increment becomes very small when the expansion factor is larger than 4. This phenomenon is consistent with the observation in [26]. Since a larger expansion factor leads to a larger number of parameters (and also a heavier computational burden), as a trade-off, the expansion factor is fixed as 4.

(3) *The Effects of the Slicing Factors c_1, c_2 and c_3*: The slicing factors c_1, c_2 and c_3 determine how to allocate the features to

[1] The source code and the trained models will be released at https://github.com/minghongli233/MWRN.

Table 4
The effects of the number of paths (× 2).

	M_E	Slicing Factors	Parameters (K)	PSNR (dB)
Single Path	–	–	1486	32.35
Two Paths	42	$c_1 : c_2 = 2 : 1$	1482	32.39
Three Paths	48	$c_1 : c_2 : c_3 = 3 : 2 : 1$	1370	32.46
Four Paths	56	$c_1 : c_2 : c_3 : c_4 = 3 : 2 : 1 : 1$	1474	32.44

Table 5
Ablation study of different modules on five datasets (× 2).

		Base-L	$\mathbb{N}_a$	$\mathbb{N}_b$	$\mathbb{N}_c$	$\mathbb{N}_d$	$\mathbb{N}_e$
MWRB		×	×	×	×	✓	✓
FCA	DCs	×	✓	×	✓	×	✓
	MWRCA	×	×	✓	✓	×	✓
Parameters (K)		1486	1501	1464	1479	1370	1386
PSNR (dB)		35.03	35.05	35.06	35.08	35.08	**35.11**

different paths in MWRB. The effects of these three slicing factors are evaluated in Table 3. We can see that these slicing factors do not affect the performance as much as M_E and the expansion factor. However, if c_3 is larger than c_1 and c_2, performance decrement can be observed. Since setting $c_1 : c_2 : c_3 = 3 : 2 : 1$ leads to the highest PSNR value, we choose $c_1 = 3/6$, $c_2 = 2/6$ and $c_3 = 1/6$.

(4) *The Effects of the Number of Paths*: The PSNR values achieved by the two-path, three-path and four-path versions of MWRB are listed in Table 4, where the single-path situation refers to the normal WDSR network with 20 WDSR blocks [26]. On the 4th path of the four-path version of MWRB, the dilation factor for the 3 × 3 dilated convolution is set as 4, and the corresponding slicing factor is denoted as c_4. For a fair comparison, the settings of the M_E and the slicing factors are adjusted for the two-path and four-path versions of MWRB, so that all the networks have similar numbers of parameters. From Table 4, we can conclude that: (a) Designing multiple paths does bring PSNR improvement over the single-path situation. (b) Compared with the three-path situation, introducing one more path does not obtain better performance. (c) Since the three-path situation achieves the best performance with the least parameters, the number of paths in MWRB is fixed as three.

4.3. Ablation study

The average PSNR values achieved by different modules on the five testing datasets with a scale factor of 2 are listed in Table 5, where 'DCs' stands for dense connections. In Table 5, the network Base-L refers to the lightweight baseline model, which is the normal WDSR network with 20 WDSR blocks [26]. When only the dense connections are added to network Base-L, we obtain network $\mathbb{N}_a$. If the last block in every five WDSR blocks in network Base-L is replaced by the MWRCA block (i.e., without the dense connections and the 1 × 1 convolution layer for fusing the input features in each FCA), we get network $\mathbb{N}_b$. Utilizing the full FCA modules in network Base-L leads to network $\mathbb{N}_c$. The network $\mathbb{N}_d$ is set up by replacing all the WDSR blocks in network Base-L with the MWRBs. Finally, $\mathbb{N}_e$ represents the full MWRN-L model. From Table 5, we can conclude that:

(1) Since the multi-scale information has been well detected, the proposed building block MWRB is superior to the WDSR block. Moreover, the number of parameters required by an MWRB is smaller than the number required by a WDSR block.

(2) Independently applying dense connections or the MWRCA components does help to improve the performance. Due to the reason that the hierarchical features contain more clues for reconstruction, applying the CA mechanism on the fused features (which leads to the FCA module) can obtain even better performance.

(3) Further improvement can be achieved by jointly using the MWRB and FCA modules, which suggests that these two types of modules are complementary to each other.

4.4. Convergence of the MWRN model

The convergence of the MWRN model is verified in this section. The curves of the L1 loss obtained during training different models with a scale of 2 are shown in Fig. 5, where the L1 loss is calculated by Eq. (12) and measured on the training dataset DIV2K. Base-L and Base-M stand for the normal WDSR networks [26] with 20 and 88 WDSR blocks, respectively. It can be observed from Fig. 5 that:

(1) Compared with the normal WDSR models, the proposed MWRN-L and MWRN-M are able to converge faster and finally reach a lower value of the L1 loss function, which indicates that integrating the proposed MWRB and FCA modules can help the network converge well.

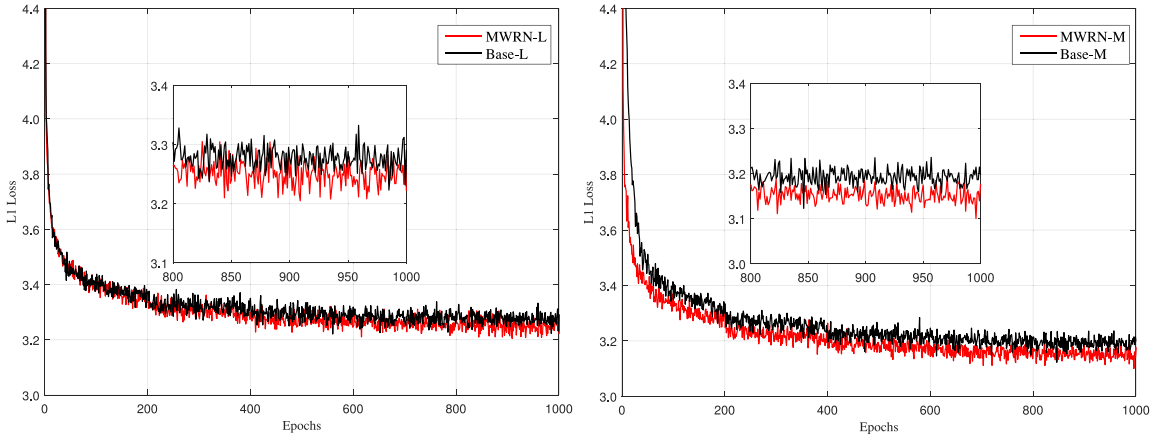
(2) In Fig. 5 **(a)**, the advantages of introducing MWRB and FCA modules are not as obvious as that in Fig. 5 **(b)**. Especially in the first 200 epochs, the curve of MWRN-M decays significantly faster than that of Base-M. This phenomenon suggests that the deeper network benefits more from the proposed two modules.

4.5. Comparison with state-of-the-art methods

To demonstrate the effectiveness of the proposed networks, MWRN-L is compared with five lightweight methods, including VDSR [20], LapSRN [22], MemNet [29] IDN [49] and CARN [25], while MWRN-M and MWRN-H are compared with seven representative middleweight/heavyweight models, including MSRN [36], D-DBPN [32], EDSR [24], RDN [31], RNAN [43], RCAN [40] and WDSR [26] [2]. Note that IDN [49] and CARN [25] are two of the most successful lightweight models; EDSR [24] is the winner of the NTIRE 2017 SR challenge [52], while D-DBPN [32] and WDSR [26] are the winners of Track 1 and Track 2 in the NTIRE 2018 SR challenge [57], respectively; In addition, RCAN [40] achieves the state-of-the-art performance on the SR task. Therefore, these methods are very strong competitors. For a comprehensive comparison, we also re-train an RCAN network with 80 RCAB blocks [40] (denoted as RCAN*), which has a number of parameters close to MWRN-M.

Following [24,31,43], the geometric self-ensemble strategy is used to boost the performance of the proposed methods, and the lightweight and heavyweight MWRN models using this strategy are denoted as MWRN-L$^+$ and MWRN-H$^+$, respectively. The quantitative comparisons for different models are provided in Tables 6 and 7, where the best two results are highlighted.

[2] We re-trained a WDSR network which consists of 88 WDSR blocks. Therefore, the compared WDSR network has the same number of building blocks as our MWRN-M.



(a) Lightweight models (b) Middleweight models

Fig. 5. The L1 loss of different models at different training epochs ($\times$ 2).

Table 6
Quantitative comparison among different lightweight models (PSNR (dB) /SSIM).

Method	Scale	Set5	Set14	BSD100	Urban100	Manga109
VDSR [20]	2	37.53 / 0.9587	33.03 / 0.9124	31.90 / 0.8960	30.76 / 0.9140	37.22 / 0.9729
LapSRN [22]	2	37.52 / 0.9590	33.08 / 0.9130	31.80 / 0.8950	30.41 / 0.9100	37.27 / 0.9740
MemNet [29]	2	37.78 / 0.9597	33.28 / 0.9142	32.08 / 0.8978	31.31 / 0.9195	37.72 / 0.9740
IDN [49]	2	37.83 / 0.9600	33.30 / 0.9148	32.08 / 0.8985	31.27 / 0.9196	38.01 / 0.9749
CARN [25]	2	37.76 / 0.9590	33.52 / 0.9166	32.09 / 0.8978	31.92 / 0.9256	38.36 / 0.9764
MWRN-L	2	**38.12** / **0.9609**	**33.80** / **0.9195**	**32.23** / **0.9002**	**32.46** / **0.9313**	**38.96** / **0.9775**
MWRN-L$^+$	2	**38.18** / **0.9611**	**33.88** / **0.9202**	**32.27** / **0.9007**	**32.60** / **0.9325**	**39.13** / **0.9780**
VDSR [20]	3	33.66 / 0.9213	29.77 / 0.8314	28.82 / 0.7976	27.14 / 0.8279	32.01 / 0.9310
LapSRN [22]	3	33.82 / 0.9227	29.87 / 0.8320	28.82 / 0.7980	27.07 / 0.8280	32.21 / 0.9350
MemNet [29]	3	34.09 / 0.9248	30.01 / 0.8350	28.96 / 0.8001	27.56 / 0.8376	32.51 / 0.9369
IDN [49]	3	34.11 / 0.9253	29.99 / 0.8354	28.95 / 0.8013	27.42 / 0.8359	32.71 / 0.9381
CARN [25]	3	34.29 / 0.9255	30.29 / 0.8407	29.06 / 0.8034	28.06 / 0.8493	33.49 / 0.9440
MWRN-L	3	**34.50** / **0.9278**	**30.45** / **0.8443**	**29.15** / **0.8064**	**28.40** / **0.8569**	**33.90** / **0.9465**
MWRN-L$^+$	3	**34.59** / **0.9285**	**30.52** / **0.8454**	**29.20** / **0.8072**	**28.54** / **0.8591**	**34.14** / **0.9477**
VDSR [20]	4	31.35 / 0.8838	28.01 / 0.7674	27.29 / 0.7251	25.18 / 0.7524	28.83 / 0.8809
LapSRN [22]	4	31.54 / 0.8850	28.19 / 0.7720	27.32 / 0.7270	25.21 / 0.7560	29.09 / 0.8900
MemNet [29]	4	31.74 / 0.8893	28.26 / 0.7723	27.40 / 0.7281	25.50 / 0.7630	29.42 / 0.8942
IDN [49]	4	31.82 / 0.8903	28.25 / 0.7730	27.41 / 0.7297	25.41 / 0.7632	29.41 / 0.8942
CARN [25]	4	32.13 / 0.8937	28.60 / 0.7806	27.58 / 0.7349	26.07 / 0.7837	30.40 / 0.9082
MWRN-L	4	**32.27** / **0.8960**	**28.69** / **0.7845**	**27.63** / **0.7383**	**26.29** / **0.7926**	**30.75** / **0.9118**
MWRN-L$^+$	4	**32.40** / **0.8978**	**28.77** / **0.7860**	**27.68** / **0.7395**	**26.44** / **0.7957**	**31.01** / **0.9143**

As can be seen from Table 6, our MWRN-L and MWRN-L$^+$ obtain the best performance in all the cases. MWRN-L significantly outperforms CARN [25], especially at a scale factor of 2. For example, at such a scale factor, the PSNR improvements of MWRN-L over CARN [25] are 0.36dB on Set5, 0.28dB on Set14, 0.14dB on BSD100, 0.54dB on Urban100 and 0.60dB on Manga109, respectively. However, as will be discussed in Section 4.6, the number of parameters and the number of Mult-Adds required by MWRN-L are both smaller than that required by CARN [25].

It can be found from Table 7 that, although MSRN [36] has the same number of parameters as MWRN-M (please see Table 9), MWRN-M outperforms MSRN [36] by a large margin. This phenomenon indicates that MWRN-M is much more efficient than MSRN due to the usage of the effective MWRB and FCA modules. In our experiment, the re-trained WDSR [26] network has the same number of building blocks as MWRN-M, but the results of MWRN-M are significantly better. Despite achieving the performance close to MWRN-M, due to the usage of non-local blocks, RNAN [43] consumes much larger memory than MWRN-M (during testing, if the RGB LR image has a resolution of 160 $\times$ 160, for a scale factor of 4, RNAN and MWRN-M cost around 8 G and 800 M GPU memory, respectively). Although the results of RCAN [40] are slightly better than that of MWRN-H in some cases, MWRN-H has a significantly smaller number of parameters (please see Table 10). On the other hand, with a reduced number of parameters, the middleweight model RCAN* is inferior to MWRN-M in most cases. Therefore, it can be concluded that the proposed MWRN model is more efficient.

Visual comparisons among different methods are provided in Figs. 6–11. From these figures, we can find that:

(1) Usually, the SR results provided by the lightweight models are more blurry than the middleweight and heavyweight models. In addition, more artifacts occur in the results of the lightweight models.
(2) Compared with other methods, MWRN-H can produce sharper edges (e.g., the fence in image *image024*, the windows in image *image020*), preserve more details (e.g., the characters in image *KoukouNoHitotachi*, the rope on the boat in image *62096*), and generate fewer artifacts (e.g., the windows on the building in image *image062*, the structure of the building in image *image073*). These subjective results suggest that the proposed MWRB and FCA modules are very effective for capturing image features.

Table 7
Quantitative comparison among middleweight and heavyweight models (PSNR (dB)/SSIM).

Method	Scale	Set5	Set14	BSD100	Urban100	Manga109
MSRN [36]	2	38.08 / 0.9607	33.70 / 0.9186	32.23 / 0.9002	32.29 / 0.9303	38.69 / 0.9772
RNAN [43]	2	38.17 / 0.9611	33.87 / 0.9207	32.32 / 0.9014	32.73 / 0.9340	39.23 / 0.9785
WDSR [26]	2	38.14 / 0.9610	33.91 / 0.9207	32.30 / 0.9013	32.73 / 0.9340	38.99 / 0.9778
RCAN* [40]	2	38.20 / 0.9612	33.94 / 0.9201	32.33 / 0.9016	32.96 / 0.9358	39.23 / 0.9778
MWRN-M	2	38.25 / **0.9614**	33.93 / 0.9200	32.37 / 0.9020	33.04 / 0.9363	39.33 / 0.9783
D-DBPN [32]	2	38.09 / 0.9600	33.85 / 0.9190	32.27 / 0.9000	32.55 / 0.9324	38.89 / 0.9775
EDSR [24]	2	38.11 / 0.9602	33.92 / 0.9195	32.32 / 0.9013	32.93 / 0.9351	39.10 / 0.9773
RDN [31]	2	38.24 / **0.9614**	34.01 /0.9212	32.34 / 0.9017	32.89 / 0.9353	39.18 / 0.9780
RCAN [40]	2	**38.27** / **0.9614**	34.11 / 0.9216	**32.41** / **0.9026**	**33.34** / **0.9384**	**39.44** / **0.9786**
MWRN-H	2	38.25 / **0.9614**	**34.14** / **0.9217**	32.38 / **0.9022**	**33.17** / 0.9378	39.38 / 0.9785
MWRN-H$^+$	2	**38.29** / **0.9616**	**34.18** / **0.9224**	**32.42** / **0.9026**	**33.34** / **0.9389**	**39.51** / **0.9788**
MSRN [36]	3	34.46 / 0.9278	30.41 / 0.8437	29.15 / 0.8064	28.33 / 0.8561	33.67 / 0.9456
RNAN [43]	3	34.66 / 0.9290	30.53 / 0.8463	29.26 / 0.8090	28.75 / 0.8646	34.25 / 0.9483
WDSR [26]	3	34.59 / 0.9285	30.46 / 0.8445	29.21 / 0.8080	28.65 / 0.8624	34.09 / 0.9477
RCAN* [40]	3	34.61 / 0.9288	30.54 / 0.8466	29.25 / 0.8090	28.82 / 0.8656	34.12 / 0.9483
MWRN-M	3	34.68 / 0.9296	30.58 / 0.8470	29.27 / 0.8096	28.82 / 0.8657	34.33 / 0.9490
EDSR [24]	3	34.65 / 0.9280	30.52 / 0.8462	29.25 / 0.8093	28.80 / 0.8653	34.17 / 0.9476
RDN [31]	3	34.71 / 0.9296	30.57 / 0.8468	29.26 / 0.8093	28.80 / 0.8653	34.13 / 0.9484
RCAN [40]	3	34.74 / 0.9299	**30.65** / **0.8482**	**29.32** / **0.8111**	**29.09** / **0.8702**	34.44 / **0.9499**
MWRN-H	3	**34.77** / **0.9301**	30.64 / 0.8479	29.29 / 0.8102	28.99 / 0.8688	**34.48** / **0.9499**
MWRN-H$^+$	3	**34.83** / **0.9305**	**30.72** / **0.8492**	**29.34** / **0.8110**	**29.15** / **0.8711**	**34.70** / **0.9509**
MSRN [36]	4	32.26 / 0.8960	28.63 / 0.7836	27.61 / 0.7380	26.22 / 0.7911	30.57 / 0.9103
RNAN [43]	4	32.49 / 0.8982	28.83 / 0.7878	27.72 / 0.7421	26.61 / 0.8023	31.09 / 0.9149
WDSR [26]	4	32.36 / 0.8974	28.77 / 0.7863	27.68 / 0.7403	26.49 / 0.7993	30.94 / 0.9144
RCAN* [40]	4	32.48 / 0.8986	28.80 / 0.7869	27.71 / 0.7408	26.62 / 0.8024	31.03 / 0.9153
MWRN-M	4	32.53 / 0.8993	28.86 / 0.7884	27.75 / 0.7422	26.69 / 0.8049	31.22 / 0.9175
D-DBPN [32]	4	32.47 / 0.8980	28.82 / 0.7860	27.72 / 0.7400	26.38 / 0.7946	30.91 / 0.9137
EDSR [24]	4	32.46 / 0.8968	28.80 / 0.7876	27.71 / 0.7420	26.64 / 0.8033	31.02 / 0.9148
RDN [31]	4	32.47 / 0.8990	28.81 / 0.7871	27.72 / 0.7419	26.61 / 0.8028	31.00 / 0.9151
RCAN [40]	4	**32.63** / 0.9002	**28.87** / **0.7889**	**27.77** / **0.7436**	**26.82** / **0.8087**	31.22 / 0.9173
MWRN-H	4	**32.63** / **0.9005**	28.86 / 0.7888	27.76 / 0.7433	26.81 / 0.8081	**31.32** / **0.9187**
MWRN-H$^+$	4	**32.73** / **0.9014**	**28.97** / **0.7907**	**27.82** / **0.7445**	**27.01** / **0.8123**	**31.62** / **0.9211**

Table 8
The numbers of parameters and Mult-Adds of different lightweight networks ($\times$ 4).

	VDSR [20]	LapSRN [22]	MemNet [29]	IDN [49]	CARN [25]	MWRN-L
Parameters (K)	665	813	677	**553**	1592	1399
Mult-Adds (G)	612.6	149.5	2662.3	**47.6**	90.9	79.7

Table 9
The numbers of parameters and Mult-Adds of middleweight networks ($\times$ 4).

	MSRN [36]	RNAN [43]	WDSR [26]	RCAN* [40]	MWRN-M
Parameters (M)	**6.1**	9.3	6.5	6.6	**6.1**
Mult-Adds (G)	376.8	479.6	374.8	403.3	**346.2**

Table 10
The numbers of parameters and Mult-Adds of heavyweight networks ($\times$ 4).

	D-DBPN [32]	EDSR [24]	RDN [31]	RCAN [40]	MWRN-H
Parameters (M)	10.4	43.1	22.3	15.6	**10.3**
Mult-Adds (G)	5211.4	2894.5	1309.2	917.6	**588.8**

(3) Generally, MWRN-M is superior to MWRN-L, while MWRN-H outperforms MWRN-M. This phenomenon indicates that stacking more MWRB and FCA modules does lead to better performance. An exception is shown in Fig. 9, where MWRN-H has no significant improvement over MWRN-M in the objective result. However, we can still see that in image *62096*, the structure of the sail recovered by MWRN-H is more clear.

4.6. Analysis of network scale and computational complexity

The network scale refers to the maximum width, the depth and the number of parameters of a network. Due to the usage of the wide-activated residual learning, the maximum feature dimension in MWRN is

$$w = f_E(c_1 M_E + c_2 M_E + c_3 M_E) = f_E M_E \tag{13}$$

where f_E is the expansion factor of the wide-activated residual unit. As suggested by [26], we choose $f_E = 4$. In our experiments, $M_E = 48$, thus leading to $w = 192$.

According to the structure of MWRN, the network depth can be calculated by

$$d = (4T + 5)N + 2 \tag{14}$$

where T is the number of MWRBs in a CB module, and N denotes the number of FEG modules in MWRN. Therefore, the depths of MWRN-L ($T = 4, N = 4$), MWRN-M ($T = 10, N = 8$) and MWRN-H ($T = 14, N = 10$) are 86, 362, and 612, respectively.

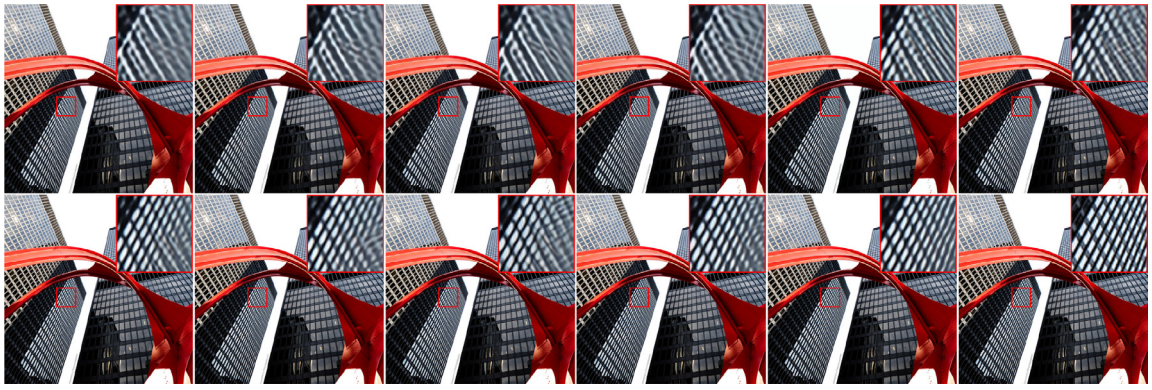


Fig. 6. The visual comparisons on image *image062* from *Urban100* (× 3). The first row (from left to right): VDSR (PSNR: 22.36dB, SSIM: 0.8457), LapSRN (22.38dB, 0.8441), MemNet (22.89dB, 0.8671), IDN (22.74dB, 0.8620), CARN (23.42dB, 0.8864), MWRN-L (24.17dB, 0.9020). The second row: EDSR (25.66dB, 0.9240), RDN (25.68dB, 0.9226), RCAN (26.13dB, 0.9318), MWRN-M (26.13dB, 0.9273), MWRN-H (**26.56dB, 0.9316**), Original Image. Please zoom in for better viewing.

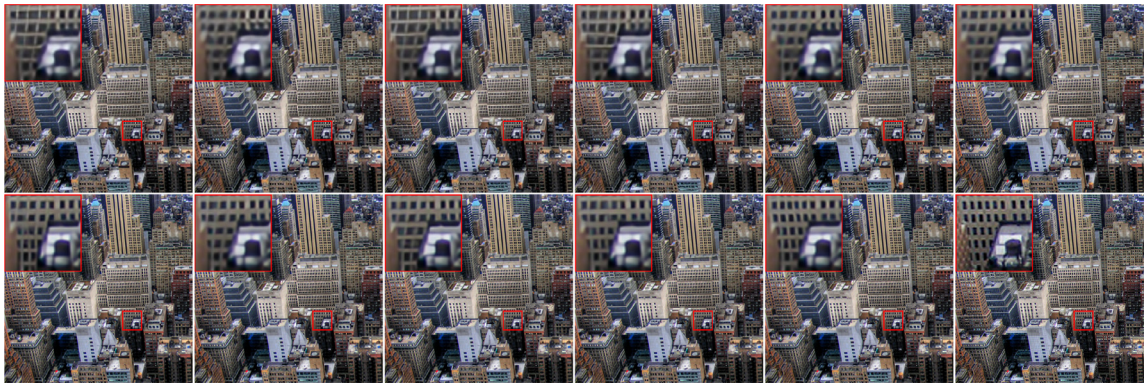


Fig. 7. The visual comparisons on image *image073* from *Urban100* (× 3). The first row (from left to right): VDSR (PSNR: 21.98dB, SSIM: 0.6841), LapSRN (22.09dB, 0.6850), MemNet (22.05dB, 0.6853), IDN (22.12dB, 0.6885), CARN (22.64dB, 0.7188), MWRN-L (22.83dB, 0.7319). The second row: EDSR (23.18dB, 0.7503), RDN (23.18dB, 0.7477), RCAN (23.11dB, 0.7501), MWRN-M (23.12dB, 0.7476), MWRN-H (**23.28dB, 0.7550**), Original Image. Please zoom in for better viewing.

Fig. 8. The visual comparisons on image *image024* from *Urban100* (× 3). The first row (from left to right): VDSR (PSNR: 21.51dB, SSIM: 0.7257), LapSRN (21.37dB, 0.7239), MemNet (21.96dB, 0.7498), IDN (21.72dB, 0.7357), CARN (22.37dB, 0.7633), MWRN-L (23.02dB, 0.7906). The second row: EDSR (23.36dB, 0.8036), RDN (22.93dB, 0.7959), RCAN (23.82dB, 0.8145), MWRN-M (23.46dB, 0.8060), MWRN-H (**23.93dB, 0.8153**), Original Image. Please zoom in for better viewing.

Compared with the wider networks, it is believed that deeper networks are easier to attain better performance [40]. To study the impact of the network depth, we show the PSNR values obtained by different sizes of MWRN in Fig. 12. As can be observed, a deeper network does lead to a higher PSNR value. Therefore, if necessary, one can stack more components to further enhance the performance of MWRN.

As increasing the number of parameters of a model might lead to higher performance, it is important to compare the efficiencies among different models. Fig. 1 shows the performance of all the networks. Apparently, compared with other models, all the three MWRN models have a better trade-off between the model size and the performance. With seemingly long and wide network, we are surprised to see that MWRN-L/MWRN-M/MWRN-H has a relatively small number of parameters. The main reasons are:

(1) Many 1 × 1 convolution layers are applied in the MWRN models to adjust the feature dimension. These 1 × 1 convolution layers make the network deeper, yet contribute less to the number of parameters.

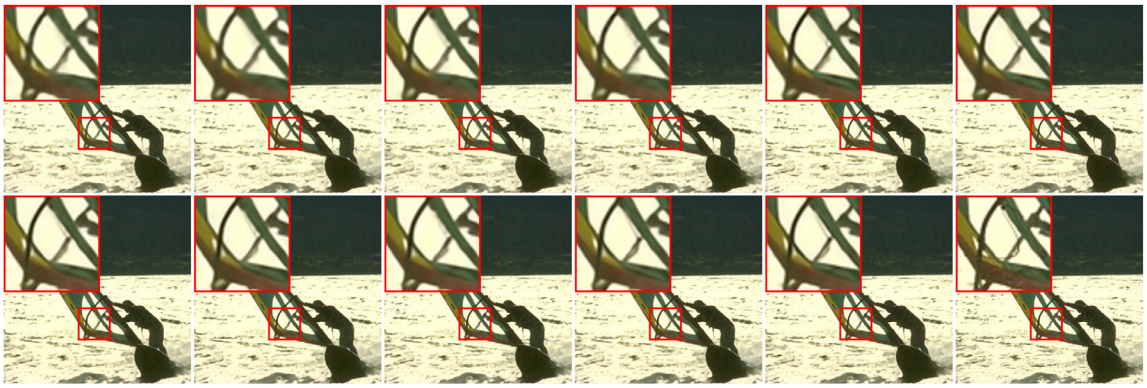


Fig. 9. The visual comparisons on image *62096* from *BSD100* (× 3). The first row (from left to right): VDSR (PSNR: 26.73dB, SSIM: 0.8398), LapSRN (26.69dB, 0.8389), MemNet (26.84dB, 0.8443), IDN (26.83dB, 0.8448), CARN (26.90dB, 0.8520), MWRN-L (26.98dB, 0.8541). The second row: EDSR (26.94dB, 0.8556), RDN (26.89dB, 0.8552), RCAN (26.96dB, **0.8589**), MWRN-M (**27.06dB**, 0.8584), MWRN-H (26.99dB, 0.8586), Original Image. Please zoom in for better viewing.

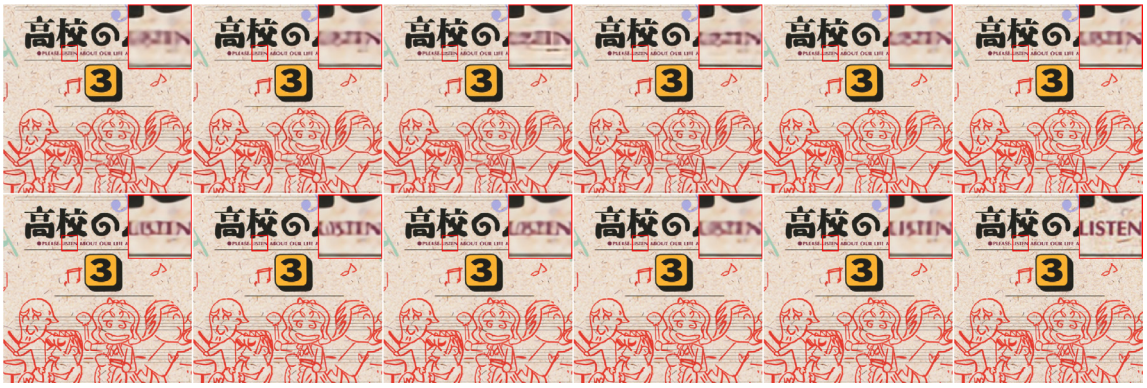


Fig. 10. The visual comparisons on image *KoukouNoHitotachi* from *Manga109* (× 4). The first row (from left to right): VDSR (PSNR: 28.74dB, SSIM: 0.8181), LapSRN (29.16dB, 0.8318), MemNet (29.44dB, 0.8335), IDN (29.23dB, 0.8348), CARN (30.62dB, 0.8597), MWRN-L (30.98dB, 0.8669). The second row: EDSR (30.93dB, 0.8689), RDN (30.76dB, 0.8653), RCAN (**31.60dB**, 0.8753), MWRN-M (31.36dB, 0.8732), MWRN-H (31.58dB, **0.8757**), Original Image. Please zoom in for better viewing.

Fig. 11. The visual comparisons on image *image020* from *Urban100* (× 4). The first row (from left to right): VDSR (PSNR: 21.47dB, SSIM: 0.6798), LapSRN (21.54dB, 0.6853), MemNet (21.60dB, 0.6899), IDN (21.60dB, 0.6911), CARN (22.10dB, 0.7180), MWRN-L (22.26dB, 0.7266). The second row: EDSR (22.36dB, 0.7374), RDN (22.36dB, 0.7377), RCAN (22.18dB, 0.7302), MWRN-M (22.40dB, 0.7374), MWRN-H (**22.44dB, 0.7421**), Original Image. Please zoom in for better viewing.

(2) In MWRB, the features are first expanded before activation and then compressed to the original dimension on each path. By choosing a small dimension of input, the consumed parameters on each path can still be equal to the normal residual learning block [26].

To better evaluate the computational complexity, the number of Mult-Adds [25], which counts every composite operation of multiplying and accumulating, is adopted. Along with the number of parameters, the Mult-Adds[3] required by different methods for super-resolving a single image are shown in Tables 8–10 [4]. We can see that, MWRN-M and MWRN-H have the lowest computational burden among all the middleweight and the heavyweight models, respectively. Among the lightweight models, IDN [49] has the least

[3] The number of Mult-Adds is counted by assuming that the resolution of the HR image is 1280 × 720.

[4] The reported numbers of parameters and Mult-Adds are calculated by using the pytorch tool available at https://github.com/Lyken17/pytorch-OpCounter.

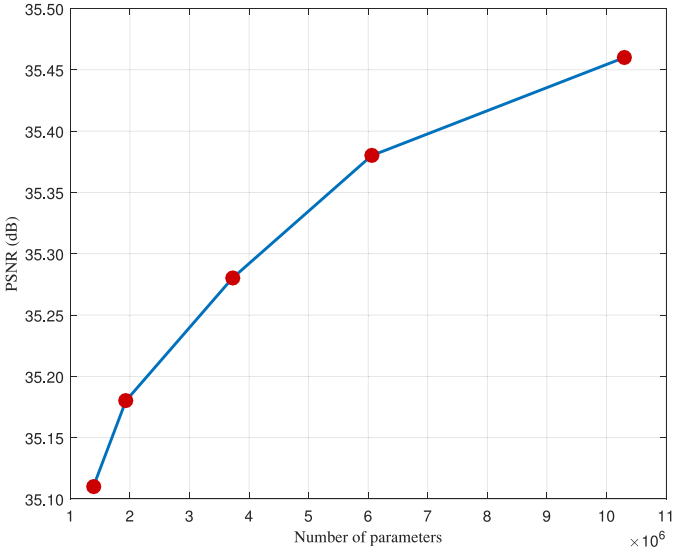


Fig. 12. Average PSNR values obtained by different sizes of MWRN on five datasets with a scale factor of 2. From the smallest model size to the biggest model size, (*T, N*) are selected as (4, 4), (6, 4), (8, 6), (10, 8) and (14, 10) respectively.

parameters and the lowest computational burden. However, the performance of MWRN-L is significantly better than IDN. Although VDSR [20], LapSRN [22] and MemNet [29] require fewer parameters than MWRN-L, they suffer from heavier computational burden. The reasons lie in that:

(1) MWRN-L extracts features in the LR space, thus resulting in less computation than VDSR [20] and LapSRN [22].
(2) To reduce the number of parameters, the recursive convolutions are utilized in MemNet [29]. However, recursively executing convolutions still requires a large amount of computation.

5. Conclusion

In this paper, we proposed a new CNN model called MWRN for the SR task. As the basic building block of MWRN, MWRB uses three wide-activated residual paths to adaptively detect different scales of image features. To reduce the number of parameters, instead of using the standard convolutions with large filter sizes such as 5×5 and 7×7, the dilated convolutions are utilized in MWRB to enlarge the receptive field. Besides MWRB, another important type of components in MWRN is the FCA module, in which multiple levels of information are fused and the channel-wise fused features are rescaled. It was demonstrated through the ablation study that both MWRB and FCA contribute to the improvement of MWRN over the basic WDSR network. Compared with other advanced CNN-based methods, our MWRN is able to achieve higher performance with similar or smaller number of parameters.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Kan Chang: Conceptualization, Methodology, Software, Writing - original draft, Resources, Funding acquisition, Project administration. **Minghong Li:** Methodology, Software, Investigation, Data curation, Visualization. **Pak Lun Kevin Ding:** Validation, Formal analysis, Writing - review & editing. **Baoxin Li:** Writing - review & editing, Supervision.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (NSFC) [grant numbers 61761005, 61761007], in part by the Natural Science Foundation of Guangxi Zhuang Autonomous Region [grant number 2016GXNSFAA380154], and in part by Guangxi Key Laboratory of Multimedia Communications and Network Technology. Part of the experiments were carried out on the High-performance Computing Platform of Guangxi University.

References

[1] L. Yue, H. Shen, J. Li, Q. Yuan, H. Zhang, Image super-resolution: the techniques, applications, and future, Signal Process. 128 (2016) 389–408.
[2] W. Dong, L. Zhang, G. Shi, X. Wu, Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization, IEEE Trans. Image Process. (TIP) 20 (7) (2011) 1838–1857.
[3] W. Dong, L. Zhang, G. Shi, X. Li, Nonlocally centralized sparse representation for image restoration, IEEE Trans. Image Process. (TIP) 22 (4) (2013) 1620–1630.
[4] K. Zhang, X. Gao, D. Tao, X. Li, Single image super-resolution with non-local means and steering kernel regression, IEEE Trans. Image Process. 21 (11) (2012) 4544–4556.
[5] C. Ren, X. He, Y. Pu, T.Q. Nguyen, Enhanced non-local total variation model and multi-directional feature prediction prior for single image super resolution, IEEE Trans. Image Process. 28 (8) (2019) 3778–3793.
[6] H. Chen, X. He, L. Qing, Q. Teng, Single image super-resolution via adaptive transform-based nonlocal self-similarity modeling and learning-based gradient regularization, IEEE Trans. Multimed. 19 (8) (2017) 1702–1717.
[7] K. Chang, P.L.K. Ding, B. Li, Single image super resolution using joint regularization, IEEE Signal Process. Lett. 25 (4) (2018) 596–600.
[8] K. Chang, P.L.K. Ding, B. Li, Single image super-resolution using collaborative representation and non-local self-similarity, Signal Process. 149 (2018) 49–61.
[9] K. Chang, X. Zhang, P.L.K. Ding, B. Li, Data-adaptive low-rank modeling and external gradient prior for single image super-resolution, Signal Process. 161 (2019) 36–49.
[10] J. Yang, J. Wright, T.S. Huang, Y. Ma, Image super-resolution via sparse representation, IEEE Trans. Image Process. (TIP) 19 (11) (2010) 2861–2873.
[11] R. Zeyde, M. Elad, M. Protter, On single image scale-up using sparse-representations, in: International Conference on Curves and Surfaces, Springer, Avignon, France, 2010, pp. 711–730.
[12] R. Timofte, V.D. Smet, L.V. Gool, Anchored neighborhood regression for fast example-based super resolution, in: IEEE International Conference on Computer Vision (ICCV), IEEE, Sydney, Australia, 2013, pp. 1920–1927.
[13] R. Timofte, V.D. Smet, L.V. Gool, A+: adjusted anchored neighborhood regression for fast super-resolution, in: Asian Conference on Computer Vision (ACCV), Springer, Singapore, 2014, pp. 111–126.
[14] J. Huang, A.S.N. Ahuja, Single image super-resolution from transformed self-exemplars, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Boston, USA, 2015, pp. 5197–5206.
[15] K. Zhang, D. Tao, X. Gao, X. Li, J. Li, Coarse-to-fine learning for single-image super-resolution, IEEE Trans. Neural Netw. Learn.Syst. 28 (5) (2017) 1109–1122.
[16] K. Zhang, Z. Wang, J. Li, X. Gao, Z. Xiong, Learning recurrent residual regressors for single image super-resolution, Signal Process. 154 (2019) 324–337.
[17] C. Dong, C.C. Loy, K. He, X. Tang, Image super-resolution using deep convolutional networks, IEEE Trans. Pattern Anal. Mach.Intell. (TPAMI) 38 (2) (2016) 295–307.
[18] C. Dong, C.C. Loy, X. Tang, Accelerating the super-resolution convolutional neural network, in: Proceedings of European Conference on Computer Vision (ECCV), Springer, Amsterdam, The Netherlands, 2016, pp. 391–407.
[19] W. Shi, J. Caballero, F. Huszar, 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: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Las Vegas, USA, 2016, pp. 1874–1883.
[20] J. Kim, J. Lee, K.M. Lee, Accurate image super-resolution using very deep convolutional networks, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Las Vegas, USA, 2016, pp. 1646–1654.
[21] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Las Vegas, USA, 2016, pp. 770–778.
[22] W.S. Lai, J.B. Huang, N. Ahuja, M.H. Yang, Deep Laplacian pyramid networks for fast and accurate super-resolution, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Honolulu, USA, 2017, pp. 5835–5843.
[23] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, Photo-realistic single image super-resolution using a generative adversarial network, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Honolulu, USA, 2017, pp. 105–114.

[24] B. Lim, S. Son, H. Kim, S. Nah, K.M. Lee, Enhanced deep residual networks for single image super-resolution, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, Honolulu, USA, 2017, pp. 1132–1140.
[25] N. Ahn, B. Kang, K.A. Sohn, Fast, accurate, and lightweight super-resolution with cascading residual network, in: European Conference on Computer Vision (ECCV), Springer, Munich, Germany, 2018, pp. 256–272.
[26] J. Yu, Y. Fan, J. Yang, N. Xu, Z. Wang, X. Wang, T. Huang, Wide activation for efficient and accurate image super-resolution, arXiv:1808.08718 (2018).
[27] Y. Cao, Z. He, Z. Ye, X. Li, Y. Cao, J. Yang, Fast and accurate single image super-resolution via an energy-aware improved deep residual network, Signal Process. 162 (2019) 115–125.
[28] G. Huang, Z. Liu, L.V.D. Maaten, K. Weinberger, Densely connected convolutional networks, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Honolulu, USA, 2017, pp. 4700–4708.
[29] Y. Tai, J. Yang, X. Liu, C. Xu, MemNet: a persistent memory network for image restoration, in: IEEE International Conference on Computer Vision (ICCV), IEEE, Venice, Italy, 2017, pp. 4549–4557.
[30] T. Tong, G. Li, X. Liu, Q. Gao, Image super-resolution using dense skip connections, in: IEEE International Conference on Computer Vision (ICCV), IEEE, Venice, Italy, 2017, pp. 4799–4807.
[31] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, Y. Fu, Residual dense network for image super-resolution, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Salt Lake City, USA, 2018, pp. 2472–2481.
[32] M. Haris, G. Shakhnarovich, N. Ukita, Deep back-projection networks for super-resolution, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Salt Lake City, USA, 2018, pp. 1664–1673.
[33] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, Going deeper with convolutions, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, USA, 2015, pp. 1–9.
[34] C. Szegedy, S. Ioffe, V. Vanhoucke, A.A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, in: AAAI Conference on Artificial Intelligence, San Francisco, USA, 2017, pp. 4278–4284.
[35] W. Shi, F. Jiang, D. Zhao, Single image super-resolution with dilated convolution based multi-scale information learning inception module, in: International Conference on Image Processing (ICIP), IEEE, Beijing, China, 2017, pp. 977–981.
[36] J. Li, F. Fang, K. Mei, G. Zhang, Multi-scale residual network for image super-resolution, in: European Conference on Computer Vision (ECCV), Springer, Munich, Germany, 2018, pp. 527–542.
[37] X. Fan, Y. Yang, C. Deng, J. Xu, X. Gao, Compressed multi-scale feature fusion network for single image super-resolution, Signal Process. 146 (2018) 50–60.
[38] W. Yang, W. Wang, X. Zhang, S. Sun, Q. Liao, Lightweight feature fusion network for single image super-resolution, IEEE Signal Process. Lett. 26 (4) (2019) 538–542.
[39] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Salt Lake City, USA, 2018, pp. 7132–7141.
[40] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, Y. Fu, Image super-resolution using very deep residual channel attention networks, in: European Conference on Computer Vision (ECCV), Springer, Munich, Germany, 2018, pp. 294–310.
[41] Y. Hu, J. Li, Y. Huang, X. Gao, Channel-wise and spatial feature modulation network for single image super-resolution, IEEE Trans. Circuits Syst. Video Technol. (TCSVT) (2019), doi:10.1109/TCSVT.2019.2915238.
[42] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, L. Zhang, Second-order attention network for single image super-resolution, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Long Beach, USA, 2019, pp. 11065–11074.
[43] Y. Zhang, K. Li, K. Li, B. Zhong, Y. Fu, Residual non-local attention networks for image restoration, in: International Conference on Learning Representations (ICLR), New Orleans, USA, 2019.
[44] X. Wang, R. Girshick, A. Gupta, K. He, Non-local neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Salt Lake City, USA, 2018, pp. 7794–7803.
[45] D. Liu, Y. Fan, C. Loy, T.S. Huang, Non-local recurrent network for image restoration, in: Annual Conference on Neural Information Processing Systems (NIPS), Montreal, Canada, 2018, pp. 1673–1682.
[46] W. Yang, X. Zhang, Y. Tian, W. Wang, J.-H. Xue, Deep learning for single image super-resolution: a brief review, IEEE Trans. Multimed. (TMM) 21 (12) (2019) 3106–3121.
[47] J. Kim, J. Lee, K.M. Lee, Deeply-recursive convolutional network for image super-resolution, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Las Vegas, USA, 2016, pp. 1637–1645.
[48] Y. Tai, J. Yang, X. Liu, Image super-resolution via deep recursive residual network, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Honolulu, USA, 2017, pp. 3147–3155.
[49] Z. Hui, X. Wang, X. Gao, Fast and accurate single image super-resolution via information distillation network, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Salt Lake City, USA, 2018, pp. 723–731.
[50] Y. Fan, J. Yu, T. Huang, Wide-activated deep residual networks based restoration for BPG-compressed images, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, Salt Lake City, USA, 2018, pp. 2621–2624.
[51] Z. Zhang, X. Wang, C. Jung, DCSR: dilated convolutions for single image super-resolution, IEEE Trans. Image Process. (TIP) 28 (4) (2019) 1625–1635.
[52] E. Agustsson, R. Timofte, NTIRE 2017 challenge on single image super-resolution: dataset and study, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, Honolulu, USA, 2017, pp. 1122–1131.
[53] M. Bevilacqua, A. Roumy, C. Guillemot, A. Morel, Low-complexity single-image super-resolution based on nonnegative neighbor embedding, in: British Machine Vision Conference (BMVC), 2012, pp. 1–12.
[54] Y. Matsui, K. Ito, Y. Aramaki, A. Fujimoto, T. Ogawa, T. Yamasaki, K. Aizawa, Sketch-based manga retrieval using manga109 dataset, Multimed. Tools Appl. 76 (20) (2017) 21811–21838.
[55] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: surpassing human-level performance on imagenet classification, in: IEEE International Conference on Computer Vision (ICCV), IEEE, Santiago, USA, 2015, pp. 1026–1034.
[56] T. Salimans, D.P. Kingma, Weight normalization: a simple reparameterization to accelerate training of deep neural networks, in: Neural Information Processing Systems (NIPS), Barcelona, Spain, 2016, pp. 901–909.
[57] R. Timofte, S. Gu, J. Wu, L.V. Gool, et al., NTIRE 2018 challenge on single image super-resolution: methods and results, in: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, Salt Lake City, USA, 2018, pp. 965–976.

Kan Chang received the B.S. degree in communication engineering, and the Ph.D. degree in communications and information systems from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2005 and 2010, respectively. From Feb. 2014 to Feb. 2015, he was a Visiting Scholar with the Department of Computer Science, Arizona State University (ASU), Tempe, AZ, USA. He is currently an Associate Professor with the School of Computer and Electronic Information, and is also a researcher in the Guangxi Key Laboratory of Multimedia Communications and Network Technology, Guangxi University, Nanning, China. His research interests include image and video processing, compressive sensing, video coding, etc. He has authored and co-authored over 50 scientific articles and has obtained 10 issued Chinese patents.

Minghong Li received the B.S. degree in electronic science and technology from Guangxi University, Nanning, China, in 2017. He is currently pursuing his M.S. degree with School of Computer and Electronic Information, Guangxi University, Nanning, China. His research interests include image denoising and super-resolution.

Pak Lun Kevin Ding received the B.S. degree in computing mathematics from the City University of Hong Kong, Hong Kong, in 2013, and the M.A. degree in mathematics from Arizona State University (ASU), Tempe, AZ, in 2015. He is currently a Ph.D student in Computer Science with ASU and a research assistant in the Visual Representation and Processing Group. His research interests include machine learning and its applications to computer vision and image processing.

Baoxin Li received the Ph.D. degree in electrical engineering from the University of Maryland, College Park, in 2000. He joined Arizona State University in 2004, where he is currently a Professor in Computer Science & Engineering and a Graduate Faculty Endorsed to Chair in the Computer Science, Electrical Engineering, and Computer Engineering programs. From 2000 to 2004, he was a senior researcher with SHARP Laboratories of America, Camas, WA, where he was the technical Lead in developing SHARP's HiIMPACT™ Sports technologies. He holds 16 issued US patents. His current research interests include computer vision and pattern recognition, image/video processing, multimedia, medical image processing, and statistical methods in visual computing. He won the SHARP Laboratories' President Award twice, in 2001 and 2004. He also received the SHARP Laboratories' Inventor of the Year Award in 2002. He received the National Science Foundation's CAREER Award from 2008 to 2009. He is on the Editorial Boards of several journals, including Signal Processing: Image Communication, IEEE Trans. on Circuits and Systems for Video Technology (TCSVT) and IEEE Trans. on Image Processing (TIP).