Neurocomputing 405 (2020) 186–199

Contents lists available at ScienceDirect

Neurocomputing

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

Deep residual networks with a fully connected reconstruction layer for single image super-resolution

Yongliang Tang[a,*], Jiashui Huang[a], Faen Zhang[a], Weiguo Gong[b]

[a] AInnovation Co. Ltd, Beijing 100089, China
[b] Key Lab of Optoelectronic Technology & Systems of Education Ministry, Chongqing University, Chongqing 400044, China

ARTICLE INFO

Article history:
Received 17 May 2019
Revised 3 April 2020
Accepted 5 April 2020
Available online 15 May 2020

Communicated by Dr. C Chen

Keywords:
Single image super-resolution
Deep neural networks
Fully connected reconstruction layer
Edge difference constraint

ABSTRACT

Recently, deep neural networks have achieved impressive performance in terms of both reconstruction accuracy and efficiency for single image super-resolution (SISR). However, the network model of these methods is a fully convolutional neural network, which is limited to exploit the differentiated contextual information over the global region in the input image because of the weight sharing in convolution height and width extent. In this paper, we discuss a new SISR method where features are extracted in the low-resolution (LR) space, and then we use a fully connected layer which learns an array of upsampling weights to reconstruct the desired high-resolution (HR) image from the final obtained LR features. By doing so, we effectively exploit the differentiated contextual information over the input image, whilst maintaining the low computational complexity for the overall SR operations. In addition, we introduce an edge difference constraint into our loss function to preserve edges and restore textures. Extensive experiments validate that our method outperforms the existing state-of-the-art SISR methods.

1. Introduction

Single image super-resolution (SISR), which aims at recovering the visually pleasing high-resolution (HR) image from a single low-resolution (LR) image generated by the low-cost imaging system and the limited environment condition, has gained increasing attention for decades in computer vision. Since the obtained HR images preserve important details and critical information for later image processing, analysis and interpretation, SISR is widely applied to various field such as video surveillance [1], medical imaging [2], face recognition [3], satellite imaging [4] and etc.

SISR problem usually assumes the observed LR image is a non-invertible low-pass filtered, downsampled and noisy version of HR image. Due to the loss of high-frequency information during the degradation of HR images, SISR is a highly ill-posed problem. To handle the ill-posed nature in SR reconstruction, a variety of methods have been developed in computer vision community. Early SR methods include interpolation and reconstruction-based methods. Interpolation methods such as bicubic interpolation [5], edge-guided interpolation [6] and nearest neighbor interpolation [7], usually perform well in smooth areas, while they generate ringing and jagged artifacts in the regions of high frequency. Although reconstruction-based methods are effective to preserve sharp edges and suppress ringing artifacts by introducing appropriate image priors such as edge-directed priors [8], gradient profile priors [9], Bayesian priors [10], and nonlocal self-similarity priors [11], they fail to add sufficient novel details to the reconstructed HR images.

Currently, learning methods are widely applied to learn the mapping between LR and HR image spaces from millions of co-occurrence LR-HR example image pair, including local linear regression [12], sparse dictionary learning [13], random forest [14], to name a few. Dong et al. [15] show that the convolutional neural networks (CNN) can be used to image SR and obtain an excellent performance. After that, CNN-based SR methods have drawn considerable attention due to the simple architecture and the impressive performance. However, CNN-based SR methods also exhibit limitations in architecture optimality. First, the network model of these methods is a fully convolutional neural network, which is limited to exploit the differentiated contextual information over global image region. Although the methods in literatures [16–18] have improved reconstruction quality by stacking the more convolution layers to exploit contextual information over a larger image region, they also increase the computation cost and memory usage. Thus, they exhibit limitations in terms of balancing the reconstruction accuracy and efficiency. Furthermore, these methods usually use convolution as the reconstruction layer to obtain the final HR image, which is limited to utilize the extracted

* Corresponding author.
E-mail address: 20150801013@cqu.edu.cn (Y. Tang).

https://doi.org/10.1016/j.neucom.2020.04.030
0925-2312/

feature information differentially to reconstruct the desired HR images because of the weight sharing of convolution in the height and width extent. Recently, the more network architectures [19–29] and Datasets [30] have been used to solve SR reconstruction problems. Methods in literatures [19],[20] try to use transposed convolution or sub-pixel convolution to reconstruct the final HR images. Shocher et al. [29] propose "Zero-Shot" SR using deep internal earning, which does not rely on prior training. However, these limitations still exist in the use of the extracted features to reconstruct HR images. Second, most existing SR algorithms [15],[17],[19],[20] optimize the network models with L2 loss and thus inevitably generate blurred edges and textures in the reconstructed HR images. Several algorithms [21],[31],[33] have focused on improving the loss function to achieve the impressive measures and make the reconstructed HR images close to human visual perception on natural images. However, the blurring problem of sharp edges and texture structures still exists in reconstructed HR images.

To address the above-mentioned drawbacks, we propose a new image SR method based on the deep neural networks. Our method takes an LR image as input and trains a cascade of convolutional blocks inspired by deep Residual Networks used for ImageNet classification [36] to extract features in the LR space. Then, we use a fully connected layer which learns an array of upsampling weights to predict residual image (the differences between the upsampled image by bicubic interpolation and the ground truth HR image) from the extracted LR features. Finally, the desired HR image is obtained by adding the predicted residual image to the upsampled image using the bicubic interpolation. In addition, considering that L2 loss function used for most SR methods always leads to the blurring of texture details and edge structures, we introduce an edge difference constraint into the loss function of our proposed network to preserve edges and texture structures.

Overall, the contributions of this paper are mainly in three aspects:

(1) By optimizing a fully connected upsampling layer to differentially exploit the contextual information over the global image region, our network can reduce the undesired visual artifacts effectively and obtain promising performance in computation time and memory usage.
(2) Since all convolution layers can be shared by the networks of the different upscaling factors, our method could facilitate fast training and testing across the different upscaling factors.
(3) We propose a new loss function with an edge difference constraint to optimize our proposed networks for making the reconstructed HR images with sharp edges and textures.

2. Related work

Numerous methods have been proposed to solve SR problem. In this section, we focus our discussion on the SR methods that based on deep neural networks.

2.1. Deep neural networks for SR

In general, the observed LR image is a degraded product of HR image, which can be generally formulated as,

$$\boldsymbol{y} = \boldsymbol{DHx} + \boldsymbol{v} \tag{1}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ represent the original HR and observed LR image respectively, $\boldsymbol{D}$ is the downsampling operator, $\boldsymbol{H}$ is the blurring filter, and $\boldsymbol{v}$ represents the additive noise. In view of the above, it is a typical multi-output regression problem to reconstruct the desired HR image $\boldsymbol{x}$ from an observed LR image $\boldsymbol{y}$. Inspired by the promising performance of deep neural networks in regression tasks, Dong et al. [15] propose a new SR architecture, namely Super-Resolution Convolutional Neural Network (SRCNN). In SRCNN, the mapping F used for reconstructing the desired HR image $\boldsymbol{x}$ consists of three convolution layers and is trained by minimizing the following function,

$$\mathrm{L}(\Theta) = \frac{1}{N}\sum_{i=1}^{N} F(\boldsymbol{y}_i; \Theta) - \boldsymbol{x}_{i2}^2 \tag{2}$$

where $\Theta = \{\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3, \boldsymbol{B}_1, \boldsymbol{B}_2, \boldsymbol{B}_3\}$ is the filter and bias of convolution layers in SRCNN, $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ represent the HR and LR image patch respectively and N is the number of training samples in each batch.

Since the model of SRCNN is shallow network (only including patch extraction/representation, non-linear mapping and reconstruction layer), the prediction of HR images relies on the contextual information of a small image region. To exploit the contextual information over a large image region, Kim et al. [17] propose a deep convolutional neural network for image SR problem (VDSR) by cascading a small convolution layer many times. Although VDSR significantly improves the reconstruction accuracy, the computation time and memory usage also increase with the depth of network. To reduce the computational cost, Dong et al. [20] use a transposed convolution to upscale the features to HR space in the last layer of network model. Lai et al. [31] adopt a similar idea and propose a deeper convolutional network within Laplacian pyramid framework (LapSRN) to progressively reconstruct the HR images. By doing so, LapSRN improves accuracy without increasing computational burden. However, these SR methods have one limitation: the prediction of each pixel of the desired HR images relies on the context information of a local region of the input images since the model of these methods consists of convolutional layers only. In order to exploit contextual information over the larger region and improve the reconstruction quality, we need to cascade more convolution layers in the networks, which means to increase computational cost and memory usage. In this paper, we discuss a new SR method to resolve the dilemma between the reconstruction accuracy and efficiency. The proposed method not only improve the quality of the reconstructed HR images by exploiting the contextual information over the global region of the input image but also reduce the computational cost by using the simplified residual blocks to extract features in the LR space.

2.2. Loss function

As in most image restoration tasks, mean squared error (MSE) or L2 loss is widely used to optimize the network models of SR methods. Since L2 loss is the major performance metrics peak-to-noise-ratio (PNSR) and structural similarity (SSIM), the trained models usually have impressive performance in terms of objective measures. However, there is a blurring problem of texture details and edge structures in reconstructed HR images. Several studies have focused on the loss function to improve the capability of SR models and restore finer texture details and sharp edges. Inspired by the report that training with L2 loss cannot guarantee better performance compared to other loss functions in terms of PSNR and SSIM [37], Lim et al. [21] use L1 loss to optimize their network models for achieving improved performance. Lai et al. [31] propose a robust Charbonnier loss for the deep convolutional network within Laplacian pyramid framework. At each pyramid level, LapSRN has corresponding loss function to reduce the difference between the output reconstructed image and the label image downsampled from ground-truth HR image with bicubic interpolation. Accordingly, the overall loss of LapSRN is defined as,

$$\mathrm{L}(\Theta) = \frac{1}{N}\sum_{i=1}^{N}\sum_{s}^{L} \rho\left(\left(F\left(\boldsymbol{y}^i; \Theta\right) + \boldsymbol{y}_s^i\right) - \boldsymbol{x}_s^i\right) \tag{3}$$

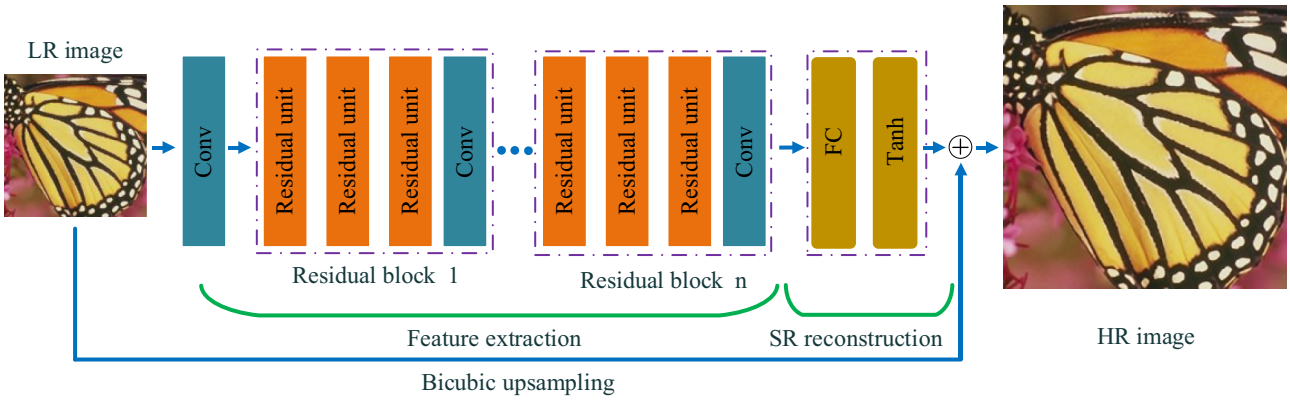


Fig. 1. The architecture of our proposed SISR network. Our network consists of feature extraction and SR reconstruction.

where $\rho(\cdot)$ is the Charbonnier penalty function, L is the number of pyramid level, $\boldsymbol{y}_s^i$ is the upsampled image from the input LR image $\boldsymbol{y}$ in the pyramid level s, and $\boldsymbol{x}_s^i$ is the ground-truth images downsampled from HR image $\boldsymbol{x}$. Due to the deep supervision of multi-loss structure and the robustness of Charbonnier penalty function, Charbonnier loss improves the stability of networks training and the reconstruction quality. Christian et al. [33] propose a perceptual loss function which consists of a content loss and an adversarial loss to reconstruct plausible-looking natural HR images with high perceptual quality.

$$\mathrm{L}(\Theta) = \frac{1}{N}\sum_{i=1}^{N} \phi(G_\theta(\boldsymbol{y}_i)) - \phi(\boldsymbol{x}_i)_2^2 - \gamma \log D_\theta(G_\theta(\boldsymbol{y}_i)) \tag{4}$$

where γ is the weight for the adversarial loss, $\phi(\cdot)$ is the feature representations of VGG network described in Simonyan and Zisserman [34] , $G_\theta(\cdot)$ is the generative model that generate the predicted HR images and $D_\theta(G_\theta(\boldsymbol{y}_i))$ is the estimated probability that the reconstructed HR image is a natural image. Although the researches for the loss function of CNN-based SR methods have improved the quality of reconstructed HR images, the restoration of finer texture details and sharp edges is still a challenging problem. Accordingly, we propose a new loss function with the edge difference constraint to alleviate the blurring problem of edges and textures.

3. Proposed method

In this section, we describe the methodology of the proposed network, the loss functions with the edge difference constraint, and the details for training.

3.1. Network architecture

As shown in Fig. 1, our network can be decomposed into two parts – feature extraction and SR reconstruction. The part of feature extraction takes an observed LR image $\boldsymbol{y}$ as input and uses the cascaded residual blocks to extract features in LR space. SR reconstruction part is a fully connected layer, which upsamples and aggregates the previous features with an array of trainable weights to reconstruct the desired HR images. In the following sections, we first describe the residual units of our networks, and then we suggest the single upscaling model that handles a specific SR scale and the multi-upscaling strategy that quickly trains models for reconstructing various upscaling of HR images.

Residual units. Deep residual networks [35] have emerged as a family of extremely deep architectures showing compelling accuracy and nice convergence behaviors in computer vision, machine translation, speech synthesis, speech recognition. Although the deep residual architecture has been successfully applied to the image SR problem and exhibited excellent performance [21],[31] , we further improve the performance by employing a new residual unit which makes training easiness and reduces training error.

In Fig. 2, we show the residual units of each network from SRResNet [31], EDSR [21], and ours. Although all the skip connections and after-addition activation functions for the residual units of these networks are the identity mapping for creating a direct path to propagate information - not only within a residual unit, but through the entire network, only our residual units adopt a rearranging the after-addition activation method to directly propagate information from one unit to any other units in both forward and backward passes. Since backward information can be propagated directly to the whole unit without through any convolution and activation layers, these residual units, such as Fig. 2(b)–(d), can alleviate the vanishing gradient problem and improve accuracy in the deep networks. SRResNet removes the after-addition activation to create an identity mapping. EDSR improves SRResNet by removing the Batch Normalization (BN) layers to enhance the flexibility of the network and reduce GPU memory usage. Inspired by [36], our residual units recast the after-addition activation as the pre-activation of the next residual units, which means that the activation only affects the residual function. By re-arranging the after-addition activation, we not only reduce the difficulty of network optimization because of the identity mapping, but improves regularization of the models since we don't remove activation function layers from our networks. At the same time, we remove BN layers from our networks since they consume the same amount of memory as the preceding convolution layers.

Furthermore, we use the Parametric Rectified Liner Unit (PReLU) to instead of Rectified Liner Unit (ReLU) as the activation function of our convolution layers. Since PReLU has a learnable coefficient for the negative part of features, it can void the "dead features" cause by zero gradients in ReLU. Accordingly, we can make full use of all parameters to obtain the maximum capacity of our networks.

Single upscaling model. In the convolutional networks, model performance can be enhanced by cascading multiple small filters in a deep network structure. Thus, we further improve our residual unit (Fig. 2(d)) with bottleneck [36] architecture and use it to construct the feature extraction part of our single upscaling model. A bottleneck residual unit consists of a 1×1 convolution layer for reducing dimension, a 3×3 layer, and a 1×1 convolution layer for restoring dimension. As designed in [36], its computational complexity is similar to the residual unit including two 3×3 convolution layers (Fig. 2(c)). However, the model with the bottleneck residual units has improved performance due to the increase of network depth. Therefore, we can maximize our model capacity considering the limited computational resources.

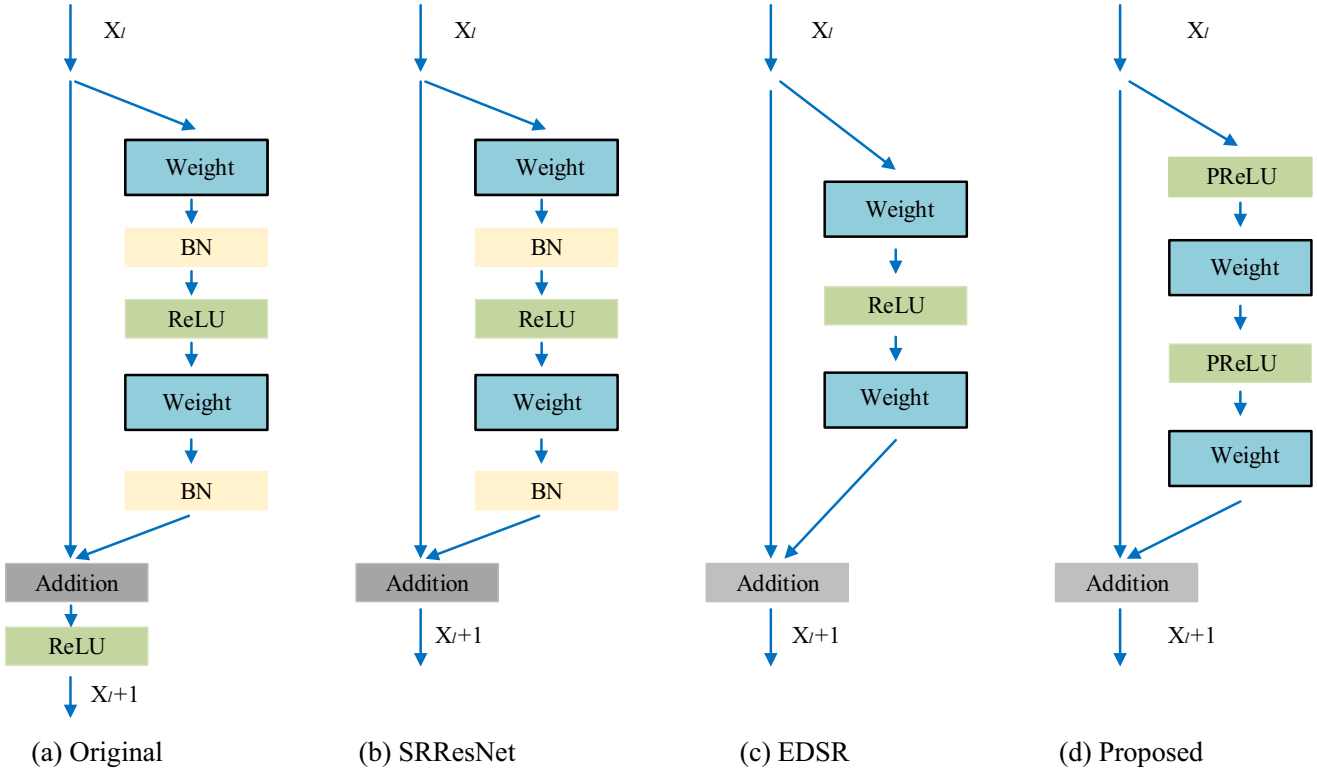


Fig. 2. Comparison of residual units in Original, SRResNet, EDSR, and ours.

For original residual networks, the convolution shortcuts [36] are used to reduce the feature map size and increase dimensions. Since the convolution shortcut is not an identity mapping, the direct path for propagating information is limited to a local region with the same feature map size. Although our network can create a direct path for propagating information over the entire network since it takes LR images as inputs and extracts all the features in LR space, we found that stacking the number of residual units above a certain level would make the training procedure numerically unstable. We resolve this issue by incorporating the 3×3 convolution layers into the cascaded residual units to construct local paths for propagating information directly. As shown in Fig. 1, we use three residual units and one 3×3 convolution layer to form one residual block which has a local propagating path. In addition, we remove the activation (PReLU) of the incorporated convolution layers from residual blocks to void the "dead features" in the identity path, as illustrated in Fig. 1.

Since the existing HR reconstruction layers (the transposed convolution [20] or sub-pixel convolution [19]) only use the feature information in a very small local region to predict each pixel in reconstructed HR images and also share the weights when predicting all the pixels of the reconstructed HR images, the reconstruction results usually contain undesired artifacts. To make effective use of the extracted features and improve reconstruction accuracy, we propose a fully connected layer to differentially utilize the extracted LR features. In our reconstruction layer, we apply the following linear transformation to the extracted LR feature for reconstructing the HR image,

$$\boldsymbol{y_H} = \boldsymbol{x_L}\boldsymbol{A}^T + \boldsymbol{b} \tag{5}$$

where $\boldsymbol{x}_L$ is the vector of input LR features, $\boldsymbol{y}_H$ is the vector of the reconstructed residual HR image, $\boldsymbol{A}^T$ and $\boldsymbol{b}$ are the weights and bias of our reconstruction layer, respectively. Since we can set the shape of $\boldsymbol{A}^T$ and $\boldsymbol{b}$, our reconstruction layer can obtain the HR images with any resolution size. During the reconstruction process, we first reshape the input LR features into the vector $\boldsymbol{x}_L$. Then, we apply the linear transformation of Eq. (5) to the vector $\boldsymbol{x}_L$ for obtaining the vector of the reconstructed residual image $\boldsymbol{y}_H$. Third, we use Tanh activation function to limit the value of vector $\boldsymbol{y}_H$ between -1 and 1, and improve the stability of training. Finally, we reshape the limited vector of $\boldsymbol{y}_H$ and add it to the bicubic image upsampled from the input LR image for obtaining the desired HR image. Although, our reconstruction layer can effectively utilize the extracted LR features to improve the reconstruction accuracy, the parameters of our reconstruction layer would be significantly increased since the fully connected layer construct the trainable weights between each predicted HR pixel and all the extracted LR features. We resolve the issue by reducing the dimension of the extracted feature map by using the incorporated 3×3 convolution layer in the final residual blocks. In addition, considering that our reconstruction layer can only process SR reconstruction with a fixed image size, we adopt a sliding window method with one overlapping pixel to solve SR reconstruction of different resolution images.

Other implementation details of our network are as follows. The dimension of feature maps in the identity path is set to be 128. In residual units, we reduce the dimension to 64 for bottleneck architecture. Our final dimension of the extracted feature maps will shrink to 8 for reducing the total parameters of our networks and improving the training and testing efficiency. For the convolution layers with kernel size 3×3, zero-padding and one-stride strategy are used to keep feature size fixed. The number of residual blocks will detail in Section 4.1

Different upscaling factors. In reality SR applications, we usually need to reconstruct various upscaling factors of HR images. Thus, we expect the proposed method could achieve fast training and testing across different upscaling factors. Since all convolution layers on the whole act like a complex feature extractor of the LR image, and only the last reconstruction layer contains the

information of the upscaling factor, we can transfer the convolution filters for fast training and testing.

In practice, we train a model for an upscaling factor in advance. Then, we only fine-tune the reconstruction layer for another upscaling factor and leave the convolution layers unchanged. The fine-tuning is fast, and the performance is as good as training from scratch (see Section 4.1). During testing, we perform the convolution operations once, and upsample an image to different sizes with the corresponding reconstruction layer. Furthermore, our method can reconstruct HR images with an arbitrary resolution (non-integer upscaling factor) by fine-tuning the fully connected reconstruction layer.

3.2. Loss function

For most of CNN-based SR methods, L2 is the most widely used loss function. As the illustrations in literature [31] since L2 loss function struggles to handle the uncertainty inherent relationship in recovering lost high-frequency details such as small scale structures and texture details, it encourages finding pixel-wise averages of plausible solutions which are typically overly-smooth and thus have poor perceptual quality. In order to resolve this problem and generate more sharp edges and texture details, we propose a new loss function for our network. Similar to [38], we proposed a new loss by utilizing the following edge difference constraint to preserve edges and texture structures,

$$\boldsymbol{E}_d = \mathrm{E}(\boldsymbol{x}_h) - E(\boldsymbol{x})_p \tag{6}$$

where $E(\cdot)$ is the edges and textures extraction operation, $\boldsymbol{x}_h$ is reconstructed HR images, and p is the norm of the edge difference constraint. Considering that the one-dimensional (1-D) processing of images can provide effective edge and texture information [39], we use 1-D processing as the edges and textures extraction operator. Actually, for a given image $\boldsymbol{x}$, the extraction of edges and textures information can be formulated as,

$$E(\boldsymbol{x}) = \sqrt{H(\boldsymbol{x})^2 + V(\boldsymbol{x})^2} \tag{7}$$

where $H(\boldsymbol{x})$ and $V(\boldsymbol{x})$ are the horizontal and vertical edge and texture information of the image $\boldsymbol{x}$ respectively. 1-D processing first use the Gaussian operator to smooth horizon direction, and then the first derivative of Gaussian operator is applied along the orthogonal direction for obtaining the vertical edges and textures $V(\boldsymbol{x})$. By repeating this procedure, we can obtain the corresponding horizon edge and texture information $H(\boldsymbol{x})$. Since the smoothing is done along a direction orthogonal to the direction of the edge extraction, 1-D processing can effectively handle outliers and extract image edges and textures. Thus, the trained network with edge difference constraint can generate sharp edges and finer texture details in the reconstructed HR images. In addition, considering that the models trained with L2 loss can restore most information of the desired HR image, we also keep L2 in our loss function and use the edge difference constrain to reconstruct more textures and edges. By incorporating the edge difference constraint into L2 loss function, we can obtain the following loss function for our network models,

$$\mathrm{L}(\Theta) = \sum_{i=1}^{N} \left(F\left(\boldsymbol{y}^i;\Theta\right) + \boldsymbol{x}_b^i\right) - \boldsymbol{x}^i{}_2 + \beta \mathrm{E}\left(\left(F\left(\boldsymbol{y}^i;\Theta\right) + \boldsymbol{x}_b^i\right)\right) - E\left(\boldsymbol{x}^i\right)_p \tag{8}$$

where β is the weight for the edge difference constraint, $\boldsymbol{x}_b^i$ is upsampled image from the input LR images $\boldsymbol{y}^i$ using bicubic interpolation. Since the researchers in [22],[24] report that training with L1 norm can achieve improved performance compared with the training with L2, we empirically set p to 1 for our edge difference constraint. Furthermore, considering that the edge difference constraint reduces stability at the beginning of model training because of the more constraints between the predicted HR images and ground-truth HR training images, we set the weight β to 0.01 for the first 10 epochs. Then, the weight β will be increased every 10 epochs by using a factor of 10 until it reaches to 1.

3.3. Training details

Training dataset: For fair comparison with most state-of-the-art methods, we first use 91 images from Yang et al.[13]. and 200 images from the training set of BSD500 [17] as the original images to train our SR models. In addition, considering that big data can push a deep model to the best performance, we also use images from DIV2K [34] datasets to optimize our models and compare with the other state-of-the-art SR models trained with the same datasets. We adopt the following ways to augment the training images: (1) Scaling: each HR image is downsampled by bicubic interpolation with the scaling factor 0.9, 0.8, 0.7, and 0.6. (2) Rotation: each image is rotated with the degree of 90, 180 and 270. (3) Flipping: each image is flipped with horizontal and vertical. Thus, we obtain $5 \times 4 \times 3 = 60$ times data to form the final ground-truth HR images $\{\boldsymbol{X}\}$. In order to prepare the training data, we first downsample the HR images $\{\boldsymbol{X}\}$ with the desired upscaling factor n to form the corresponding LR images $\{\boldsymbol{Y}\}$. Then, we crop the LR images into a set of LR image patches $\{\boldsymbol{y}^i\}_{i=1}^N$ with a stride k. The corresponding HR image patches $\{\boldsymbol{x}^i\}_{i=1}^N$ are also cropped with a stride $n \times k$ from the HR images. Actually, the cropped LR/HR image patch pairs $\{(\boldsymbol{y}^i, \boldsymbol{x}^i)\}_{i=1}^N$ are the training data for our proposed networks. Since all the convolutional layers can be shared by the networks of different upscaling factors, it is necessary to employ the LR image patches with the same size to all the networks of our SR method. Thus, for $\times$ 2, $\times$ 3 and $\times$ 4 networks, the size of LR/HR image patches are set to be $32^2/64^2$, $32^2/96^2$ and $32^2/128^2$, respectively.

Training Parameters: For the proposed SR method, we use Caffe package [40] with stochastic gradient descent algorithm to train our networks. For the models trained from scratch, we use a learning rate of 0.1 for the convolution layers and 0.01 for the fully connected reconstruction layer. The learning rate will be decayed every 10 epochs using a factor of 10. Since we adopt an extremely high learning rates (0.1) to accelerate the convergence, the gradient clipping is set to be 1 and then is increased by a factor of 10 every 10 epochs. For weights initialization, all the filters of the convolution and the fully connected reconstruction layer are initialized with the method described in [41].

During the fine-tuning of another upscaling networks, the learning rate for all layers is set to be 0.001 and decayed by an exponential rate of 0.90 each epoch. The training of scratch network uses momentum with a decay of 0.9, while our fine-tuned models are achieved using RMSProp with decay of 0.9 and $\epsilon = 1.0$. The batches of size and weight decay are set to 256 and 0.0001 for all the network training, respectively.

4. Experimental results and discussions

In this section, we first analyze the contributions of the different components of our network model. Then, we compare the proposed method with the existing state-of-the-art SR algorithms on the representative image datasets. Finally, we discuss the application of our models on the real-world images and the efficiency of SR reconstruction.

In our experiments, we follow the publicly available evaluation framework of Timofte et al. [12]. It enables the comparison of the proposed method with many state-of-the-art SR methods in the same setting. The framework only applies the SR algorithm

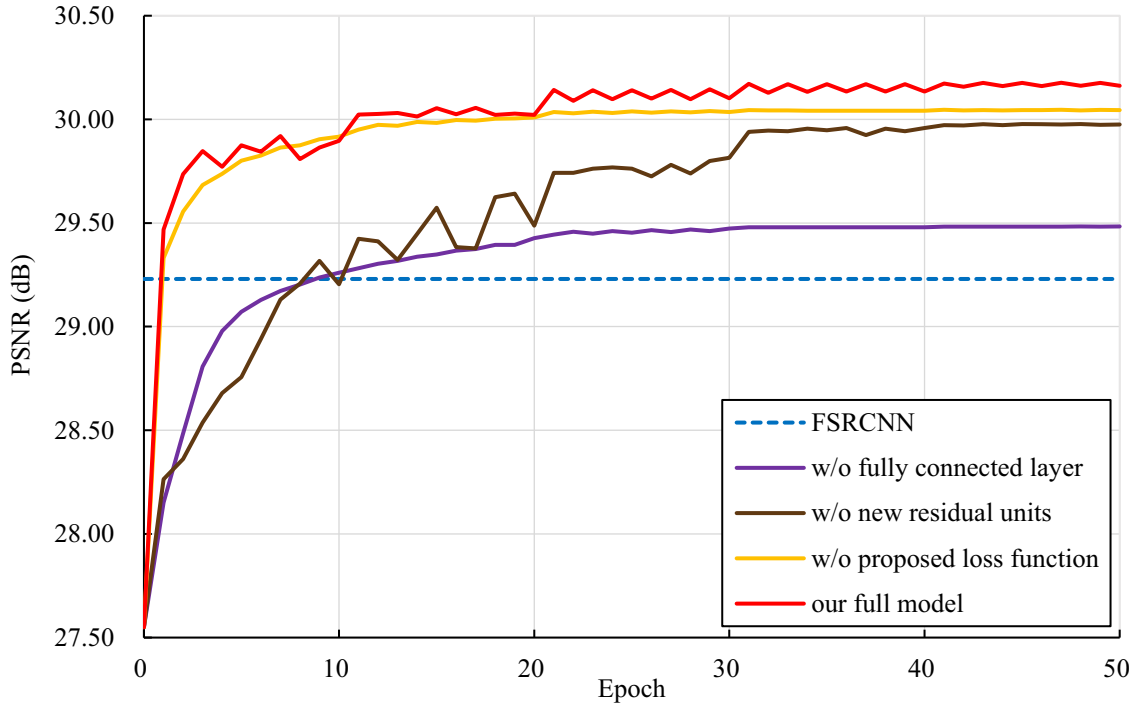


Fig. 3. Convergence analysis on the different components of the proposed network. The results are obtained on all the images in Set14 with the upscaling factor of × 3. All models are trained by using the images from 291-image dataset.

on the luminance channel and directly upscales the chrominance (Cb and Cr) channels to the desired resolution using bicubic interpolation. In addition, although researchers [42–45] have proposed many new SR performance metrics, there are some instabilities for evaluating the HR images reconstructed by the different SR methods [15], Accordingly, we use PSNR, SSIM and information fidelity criterion (IFC) [46] to evaluate the objective quality of reconstructed HR images, considering that we use a large number of test images and comparison methods to verify the effectiveness of our method.

4.1. Investigation of the proposed network model

In this section, we perform experiments to analyze the property of the proposed network and confirm the contributions of the different components of our network for the accuracy of SR reconstruction.

Fully connected reconstruction layer. To demonstrate the effect of our reconstruction layer, we remove the fully connected layer and use the transposed convolution as the reconstruction layer of our network. For the convenience of the next comparisons, we use the residual units of EDSR [21] to construct the fully connected network and the transposed convolutional network. Moreover, both models are optimized by L2 loss function from the same scratch. Fig. 3 shows the convergence curves in terms of PSNR on the Set14 for the upscaling factor of 3. The performance of the transposed convolutional network (purple curve) is significantly worse than the network (coffee curve) with a fully connected reconstruction layer. This is mainly because our reconstruction layer can effectively use the extracted LR features to reconstruct each pixel of the HR image. As shown in Fig. 4, the degraded network reconstructs

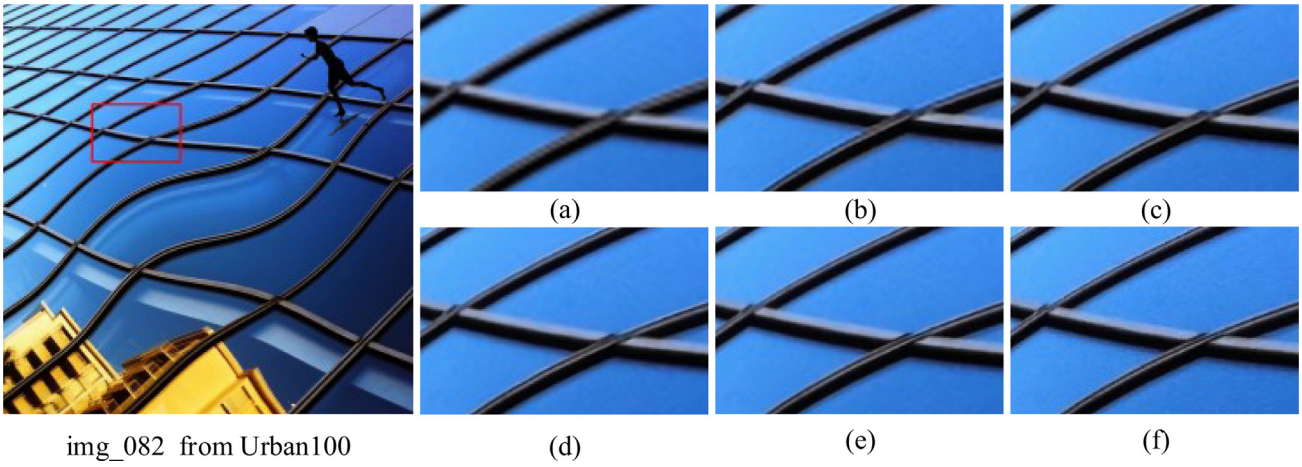


Fig. 4. Visual comparison for the contribution of the different components in our proposed network. All the models are trained by using the images for 291-image dataset with the upscaling factor of × 3. (a) Bicubic. (b) the proposed method without our reconstruction layer, residual units and loss function. (c) the proposed method without the improved residual units and loss function. (d) the proposed method without our loss function. (e) full model. (f) ground-truth HR image.

HR image with the blurring edges and details. In contrast, the reconstructed HR image provided by the proposed network contains the clean edges and visual details. In view of the above, the network with a fully connected layer is more capable of reconstructing HR images.

Residual units. For our proposed models, we use a new residual unit to improve the SR accuracy. Therefore, we verify the effectiveness of our residual unit in the section. To this end, we remove our residual units and use the residual units of EDSR [21] to construct the SR model. Since the SR model with the residual units of EDSR [21] have be trained in above section, we only train the SR model constructed by our residual units in this section. Furthermore, we also use L2 loss function to optimize the SR models from the scratch initialized with the same method for a fair comparison. As illustrated in Fig. 3, the SR model constructed with the residual of EDSR [21] (coffee curve) fluctuates obviously and converges to a worse result slowly. The SR model with our residual units (yellow curve) has better convergence stability and accuracy. Furthermore, we show the reconstructed HR image by the model with our residual units in Fig. 4(d). As shown in the Fig. 4(d), our residual units provide clearer details in the "straight window frame" area and further improves the accuracy of the model. This is because our residual units use the re-arranging of the after-addition activation to reduce the difficulty of networks optimization and improve the regularization of our SR models.

Loss function. In our method, we present a new loss function to preserve edges and texture structures. Here we verify the effectiveness of the proposed loss function. For comparison, we use L2 loss function to optimize our network in the same training parameters and initialization method. As illustrated in Fig. 3, the network optimized with L2 loss (yellow curve) converges smoothly, but has high training loss. Although our final model (red curve) fluctuates "significantly", it can obtain the improved performance in terms of reconstruction accuracy. In addition, Fig. 4 shows the SR results reconstructed by the models trained with our loss function and the L2 loss function, respectively. As shown in Fig. 4(d), the HR images reconstructed by the network trained with L2 loss function contain blurring details. In contrast, the SR model trained with our proposed loss function shows promising performance in preserving sharpen edges and reconstructing visual details. This is mainly because our loss function makes our networks pay more attention to textures and edges, since the edge and texture regions have higher loss value in the process of model optimization.

Table 1
Trade-off between performance and execution time for the different number of residual blocks L in our network (upscaling factor $\times$ 3). All models are trained with the images from 291-image dataset.

L	Set5		Set14	
	PSNR	Seconds	PSNR	Seconds
2	33.78	0.1405	29.82	0.7149
3	34.02	0.1861	29.98	0.9596
4	34.16	0.2430	30.11	1.1811
5	34.22	0.2291	30.18	1.3521
6	34.24	0.2707	30.21	1.6308
7	34.25	0.3129	30.23	1.8832
8	34.26	0.3457	30.24	2.0385

Network depth. To demonstrate the effect of network depth, we train the proposed networks with different depth and show the trade-off between super-resolving performance and execution time in Table 1. Since our network depth is decided by the number of residual blocks L in the feature extraction part, we train the proposed network with different $L = 2, 3, 4, 5, 6, 7, 8$ to validate the effect of network depth. Table 1 illustrates the average PSNR values and running time on the Set5 and Set14 with the upscaling factors of $\times$ 3. In general, deep networks perform better than shallow ones at the expense of the increased time cost. However, PSNR results increase slowly when L is larger than 6. Accordingly, we choose $L = 5$ for our $\times$ 2, $\times$ 3 and $\times$ 4 SR networks to strike a balance between the super-resolving performance and efficiency.

Different upscaling factors. In this section, we demonstrate the flexibility of our network for fast training and testing across different upscaling factors. In our experiment, we first obtain a well-trained model with the upscaling factor of $\times$ 3, then

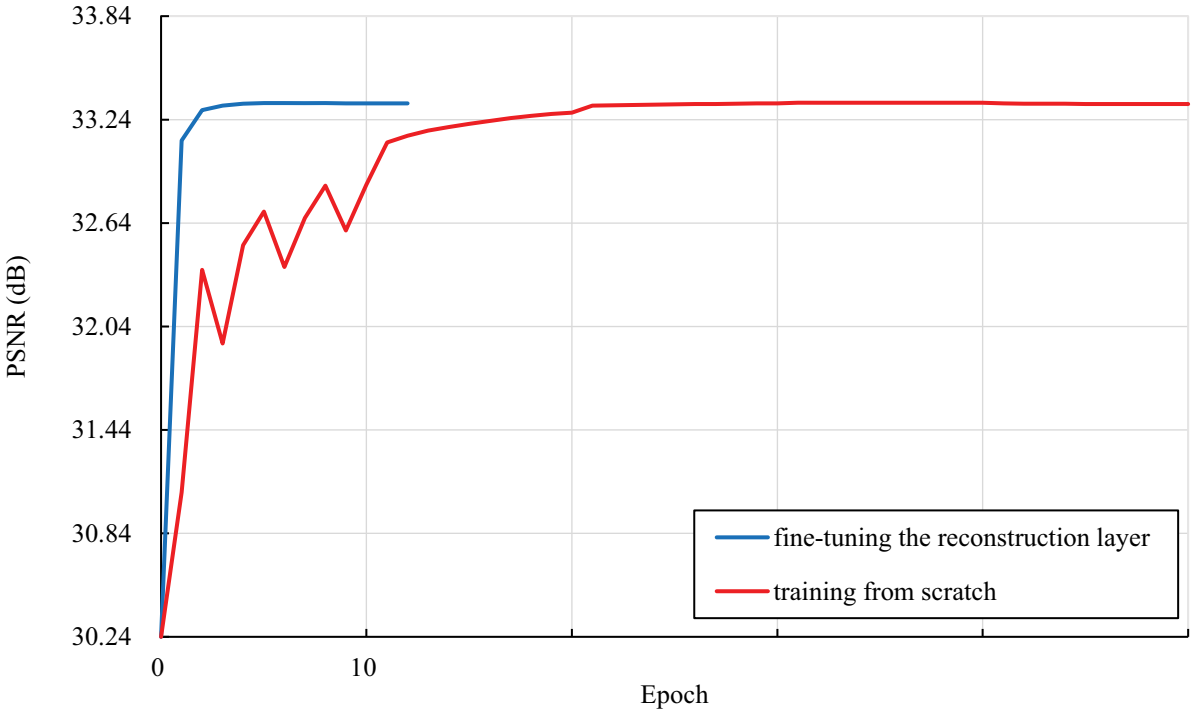


Fig. 5. Convergence curves of different training strategies. The results are obtained on Set14 with the upscaling factor of $\times$ 2. All models are trained by using the images from 291-image dataset.

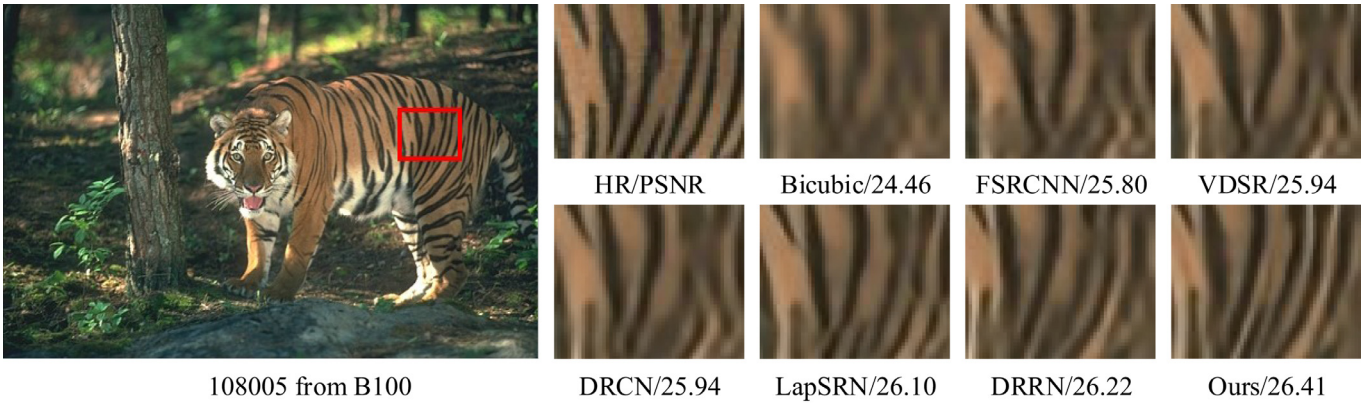


Fig. 6. Visual comparison of our method and the compared methods on image "108,005" for the upscaling factor of × 4. All models are trained by using the images from 291-image dataset.

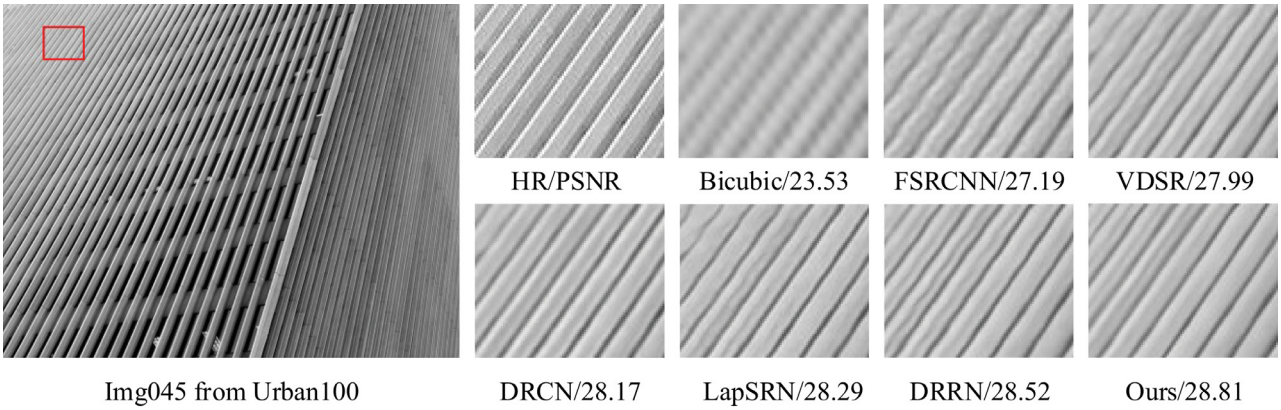


Fig. 7. Visual comparison of our method and the compared methods on image "Img045" for the upscaling factor of × 4. All models are trained by using the images from 291-image dataset.

we train the network for × 2 on the basis of that for × 3. During training, we only fine-tune the fully connected reconstruction layer on the training datasets of × 2 since the parameters of all convolution filters in the well-trained model are transferred to the new network. For comparison, we also train another network for × 2 but from scratch. The convergence curves of these two networks are shown in Fig. 5. Obviously, with the transferred parameters, the network converges very fast (only a few epochs) with the same good performance as that training from scratch. In the following experiments, we only train the networks of × 3 from scratch, and fine-tune the corresponding reconstruction layers for × 2 and × 4.

4.2. Comparisons with the state-of-the-arts methods

To validate the performance of the proposed method, the image SR experiments of different scaling factors (× 2, × 3 and × 4) are performed on all the images in the five representative image datasets Set5, Set14, B100, Urban100 and Manga109 [31]. Among these datasets, Set5, Set14 and B100 consist of natural scenes images; Manga109 and Urban100 include challenging images with details in different frequency bands. Then, we compare the proposed method with the state-of-the-art SR algorithms. All the compared results are reproduced by the corresponding public codes under the same setting with our experiments.

Fig. 8. Visual comparison of our method and the compared methods on image "DualJustice" for the upscaling factor of × 4. All models are trained by using the images from 291-image dataset.

Table 2
Average PSNR/SSIM of the upscaling factors × 2, × 3 and × 4 on datasets Set5, Set14, B100, Urban100 and Manga109. All models are trained by using the images from 291-image dataset. Bold value indicates the best performance and italic value indicates the second-best performance.

Dataset	Scale	Set5 PSNR/SSIM/IFC	Set14 PSNR/SSIM/IFC	B100 PSNR/SSIM/IFC	Urban 100 PSNR/SSIM/IFC	Manga109 PSNR/SSIM/IFC
Bicubic	x2	33.66/0.9299/6.077	30.25/0.8692/6.029	29.56/0.8439/5.594	26.88/0.8409/6.191	30.85/0.9354/6.083
	x3	30.39/0.8678/3.577	27.55/0.7749/3.456	27.21/0.7399/3.130	24.46/0.7359/3.604	26.98/0.8578/3.474
	x4	28.42/0.8100/2.328	26.00/0.7036/2.232	25.96/0.6695/1.976	23.15/0.6592/2.355	24.92/0.7897/2.263
FSRCNN	x2	37.00/0.9557/8.199	32.63/0.9087/7.840	31.50/0.8909/7.180	29.85/0.9011/8.131	36.56/0.9704/8.587
	x3	33.16/0.9132/4.968	29.43/0.8245/4.569	28.52/0.7900/4.061	26.42/0.8070/4.878	31.12/0.9202/4.912
	x4	30.71/0.8647/2.993	27.59/0.7540/2.772	26.97/0.7140/2.370	24.60/0.7267/2.916	27.89/0.8590/2.950
VDSR	x2	37.53/0.9587/8.190	32.97/0.9127/7.878	31.89/0.8958/7.169	30.77/0.9141/8.270	37.16/0.9738/9.120
	x3	33.66/0.9213/5.088	29.77/0.8314/4.602	28.82/0.7976/4.041	27.14/0.8279/5.042	32.01/0.9329/5.381
	x4	31.35/0.8838/3.496	28.03/0.7678/3.071	27.29/0.7252/2.627	25.18/0.7525/3.402	28.88/0.8854/3.664
LapSRN	x2	37.52/0.9591/*9.010*	33.08/0.9130/*8.501*	31.78/0.8941/7.713	30.41/0.9093/8.906	37.27/0.9731/9.479
	x3	33.82/0.9213/5.191	29.87/0.8320/4.662	28.82/0.7973/4.056	27.07/0.8272/5.168	32.19/0.9334/5.401
	x4	31.35/0.8838/3.548	28.19/0.7720/3.146	27.32/0.7280/2.677	25.21/0.7545/3.528	29.09/0.8893/3.728
DRRN	x2	37.74/0.9591/8.671	33.23/0.9136/8.320	32.05/0.8973/7.613	31.23/0.9188/8.917	37.92/0.9756/9.266
	x3	34.03/0.9244/5.397	29.96/0.8349/4.878	28.95/0.8004/4.269	27.53/0.8378/5.456	32.74/0.9389/5.569
	x4	31.68/0.8888/3.703	28.21/0.7721/3.252	27.38/0.7284/*2.760*	25.44/0.7638/3.700	29.46/0.8964/3.878
MemNet	x2	*37.78/0.9597*/8.732	*33.38/0.9142*/8.409	*32.08/0.8978/7.724*	*31.31/0.9195/8.928*	*37.72/0.9740/9.481*
	x3	*34.09/0.9248/5.428*	*30.00/0.8350/4.892*	*28.96/0.8001/4.301*	*27.56/0.8376/5.461*	*32.51/0.9369/5.571*
	x4	*31.74/0.8893/3.802*	*28.26/0.7723/3.254*	*27.40/0.7281*/2.758	*25.50/0.7630/3.701*	*29.42/0.8942/3.891*
Proposed	x2	**37.89/0.9602/9.020**	**33.39/0.9149/8.602**	**32.17/0.8996/7.802**	**31.53/0.9236/9.012**	**38.15/0.9756/9.710**
	x3	**34.22/0.9267/5.684**	**30.18/0.8382/4.921**	**29.09/0.8083/4.468**	**27.82/0.8422/5.523**	**32.95/0.9395/5.678**
	x4	**31.93/0.8917/3.947**	**28.47/0.7799/3.262**	**27.50/0.7317/2.764**	**25.71/0.7784/3.728**	**29.74/0.8965/3.954**

Table 3
Average PSNR/SSIM of the upscaling factors × 2, × 3 and × 4 on datasets Set5, Set14, B100, Urban100 and Manga109. All models are trained with the images from DIV2K, Flickr and ImageNet datasets. Bold value indicates the best performance and italic value indicates the second-best performance.

Dataset	Scale	Set5 PSNR/SSIM/IFC	Set14 PSNR/SSIM/IFC	B100 PSNR/SSIM/IFC	Urban 100 PSNR/SSIM/IFC	Manga109 PSNR/SSIM/IFC
Bicubic	x2	33.66/0.9299/6.077	30.25/0.8692/6.029	29.56/0.8439/5.594	26.88/0.8409/6.191	30.85/0.9354/6.083
	x3	30.39/0.8678/3.577	27.55/0.7749/3.456	27.21/0.7399/3.130	24.46/0.7359/3.604	26.98/0.8578/3.474
	x4	28.42/0.8100/2.328	26.00/0.7036/2.232	25.96/0.6695/1.976	23.15/0.6592/2.355	24.92/0.7897/2.263
EDSR	x2	38.11/0.9602/8.477	*33.92*/0.9195/8.179	*32.32*/0.9013/7.263	*32.93/0.9351*/9.210	39.10/0.9773/8.886
	x3	34.65/0.9280/5.448	30.52/0.8462/4.899	29.25/0.8093/4.170	28.80/0.8653/5.855	34.17/0.9467/5.719
	x4	32.46/0.8968/3.878	28.80/*0.7876*/3.359	27.71/0.7420/2.776	*26.64/0.8033*/4.139	31.02/0.9148/4.137
D-	x2	38.09/0.9600/8.311	33.85/0.9190/8.121	32.35/0.9014/7.189	32.55/0.9324/9.171	38.89/0.9774/9.030
DBPN	x3	-/-	-/-	-/-	-/-	-/-
	x4	32.47/0.8980/3.959	28.82/0.7860/**3.485**	27.86/0.7453/**2.816**	26.38/0.7946/**4.381**	30.91/0.9137/4.560
RDN	x2	*38.16*/0.9605/8.489	33.74/0.9182/8.183	*32.32*/0.9012/**7.296**	32.85/0.9346/*9.260*	39.09/0.9772/**9.115**
	x3	34.67/0.9284/5.493	30.38/0.8437/4.932	29.24/0.8090/4.192	28.78/0.8648/5.913	34.13/0.9484/**5.918**
	x4	32.44/0.8972/3.896	28.65/0.7840/3.375	27.71/*0.7419*/2.785	26.61/0.8026/4.170	31.00/0.9151/**4.303**
RCAN	x2	**38.27/0.9614**/*8.494*	**34.12/0.9216/8.214**	**32.41/0.9027**/7.276	**33.34/0.9384/9.276**	**39.44/0.9786**/8.969
	x3	**34.74/0.9299/5.516**	**30.65/0.8482**/*4.949*	**29.32/0.8111**/*4.193*	**29.09/0.8702/5.941**	**34.44/0.9499**/5.814
	x4	**32.63/0.9002**/*3.946*	**28.87/0.7889**/3.382	*27.77*/**0.7436**/2.785	**26.82/0.8087**/4.199	**31.22/0.9173**/4.222
SRFBN	x2	38.11/*0.9609*/8.374	33.82/*0.9196*/8.066	32.29/0.9010/7.209	32.62/0.9328/8.946	39.08/*0.9779*/8.761
	x3	34.70/*0.9292*/5.410	30.51/0.8461/4.865	29.24/0.8084/4.140	28.73/0.8641/5.739	34.18/0.9481/5.668
	x4	*32.47/0.8983*/3.828	*28.81*/0.7868/3.351	27.72/0.7409/2.760	26.60/0.8015/4.061	*31.15/0.9160*/4.137
Proposed	x2	38.10/0.9606/**8.522**	33.86/0.9189/*8.210*	32.27/*0.9016/7.292*	32.83/0.9336/9.250	*39.15*/**0.9786**/*9.021*
	x3	*34.71*/0.9287/5.503	*30.53/0.8464*/**4.951**	*29.29/0.8095*/**4.223**	*28.82/0.8658/5.929*	*34.27/0.9485/5.913*
	x4	32.40/0.8972/**3.947**	28.79/0.7867/*3.423*	**27.80**/0.7417/*2.805*	26.61/0.8024/*4.216*	31.14/0.9155/*4.231*

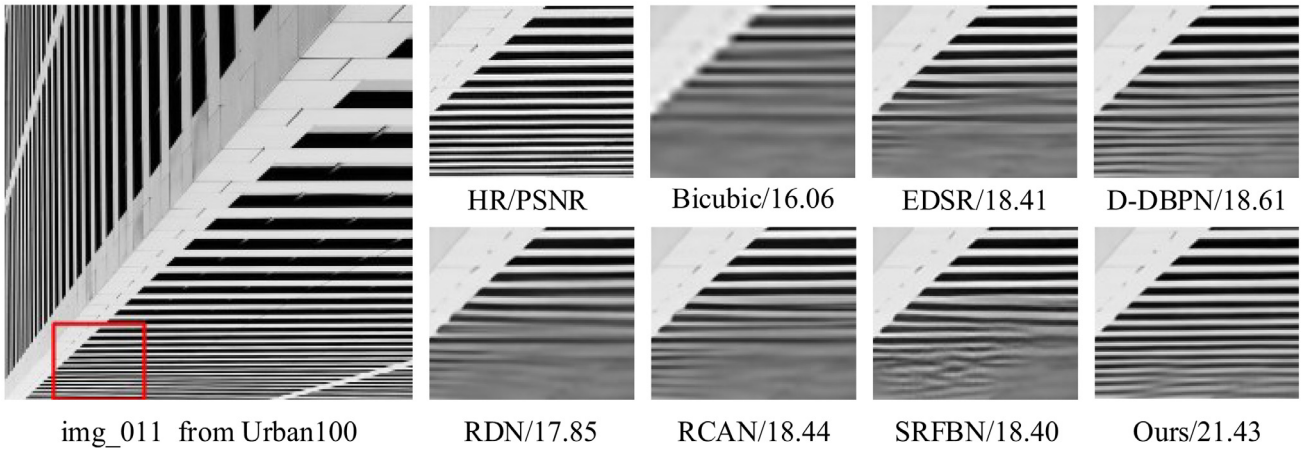


Fig. 9. Visual comparison of our method and the compared methods on image "img_011" for the upscaling factor of × 4. All models are trained by using the images from DIV2K, Flickr and ImageNet datasets.

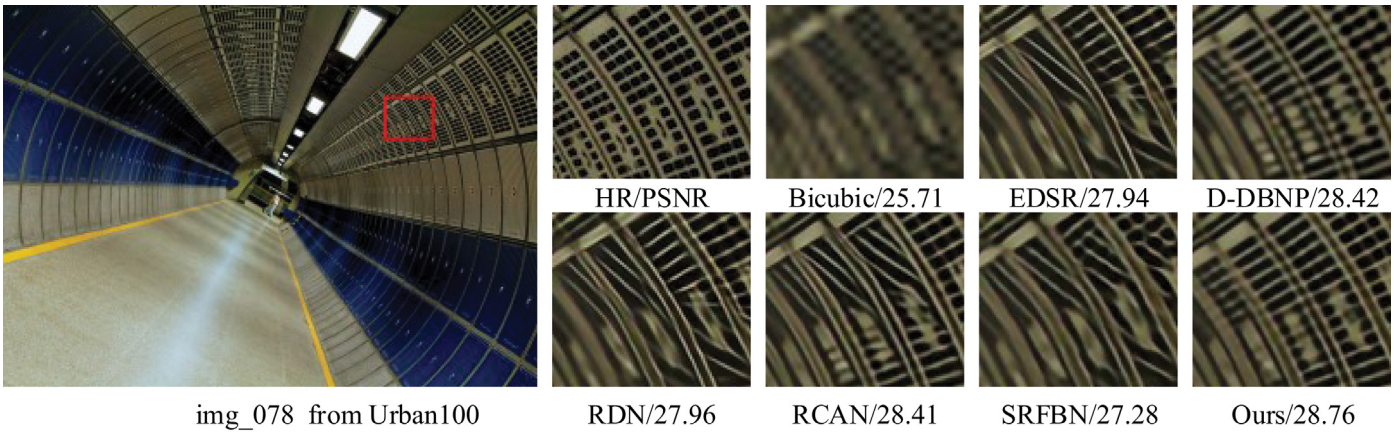


Fig. 10. Visual comparison of our method and the compared methods on image "img_078" for the upscaling factor of × 4. All models are trained by using the images from DIV2K, Flickr and ImageNet datasets.

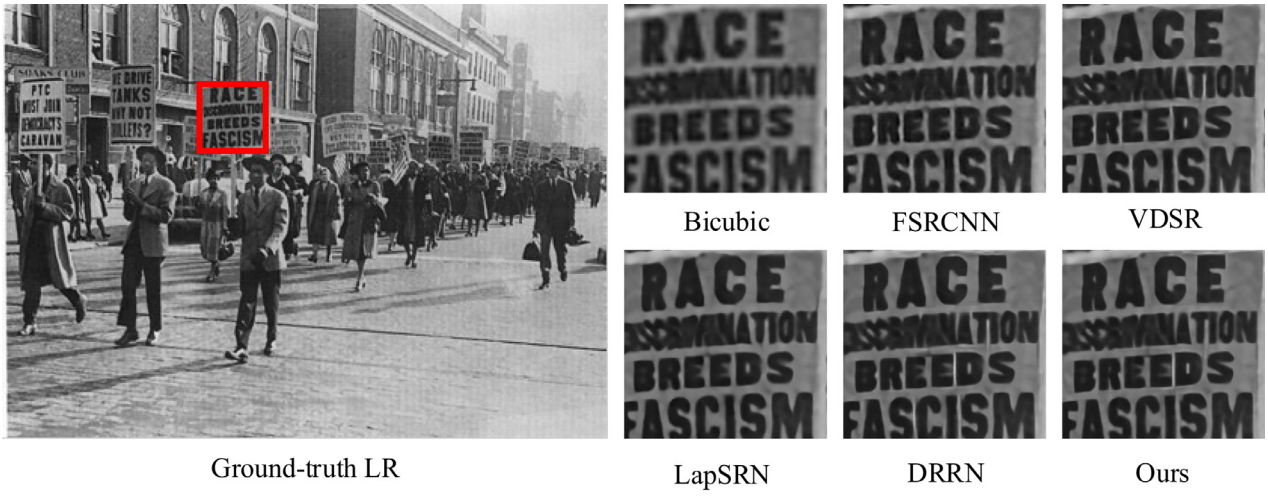


Fig. 11. Visual comparison for the upscaling factor of × 4 on real-world historical images. All models are trained by using the images from 291-image dataset.

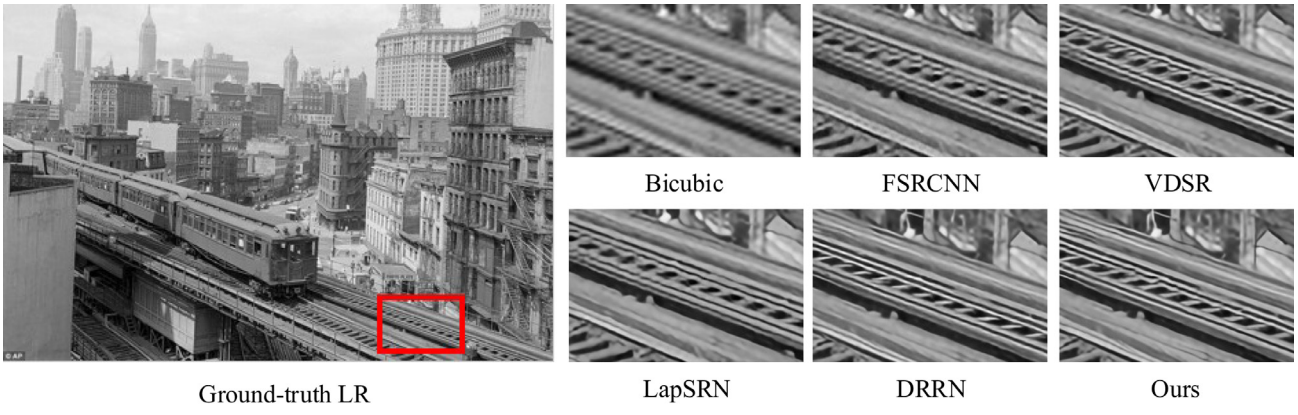
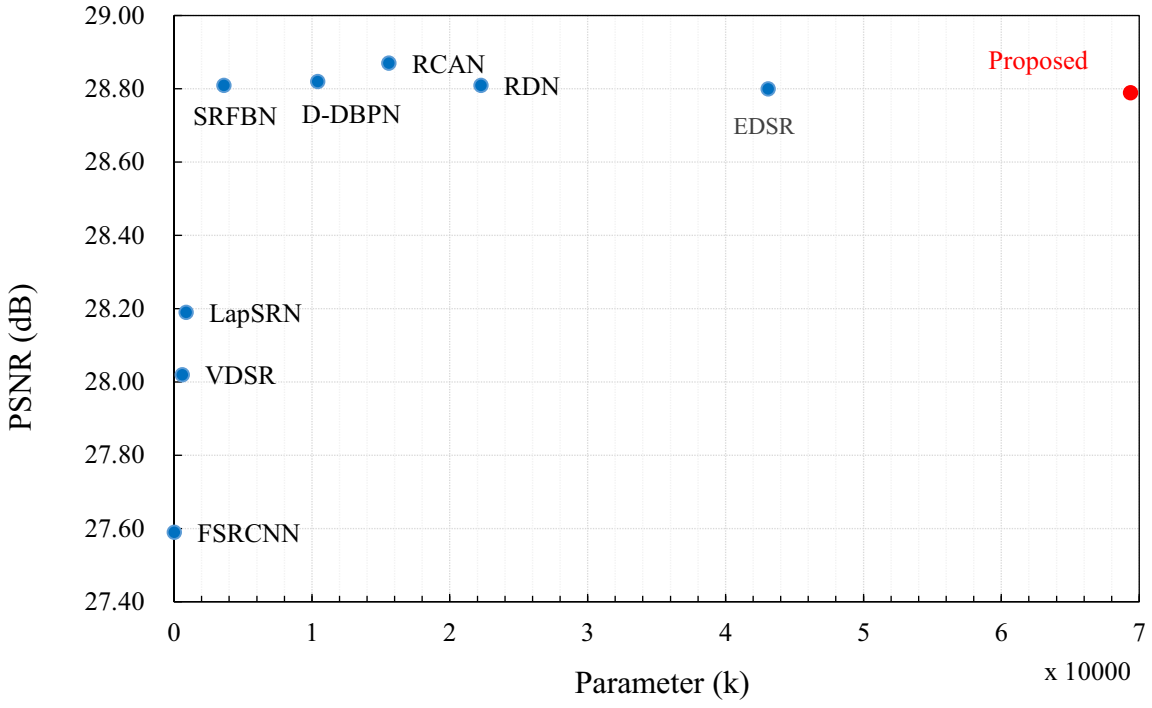


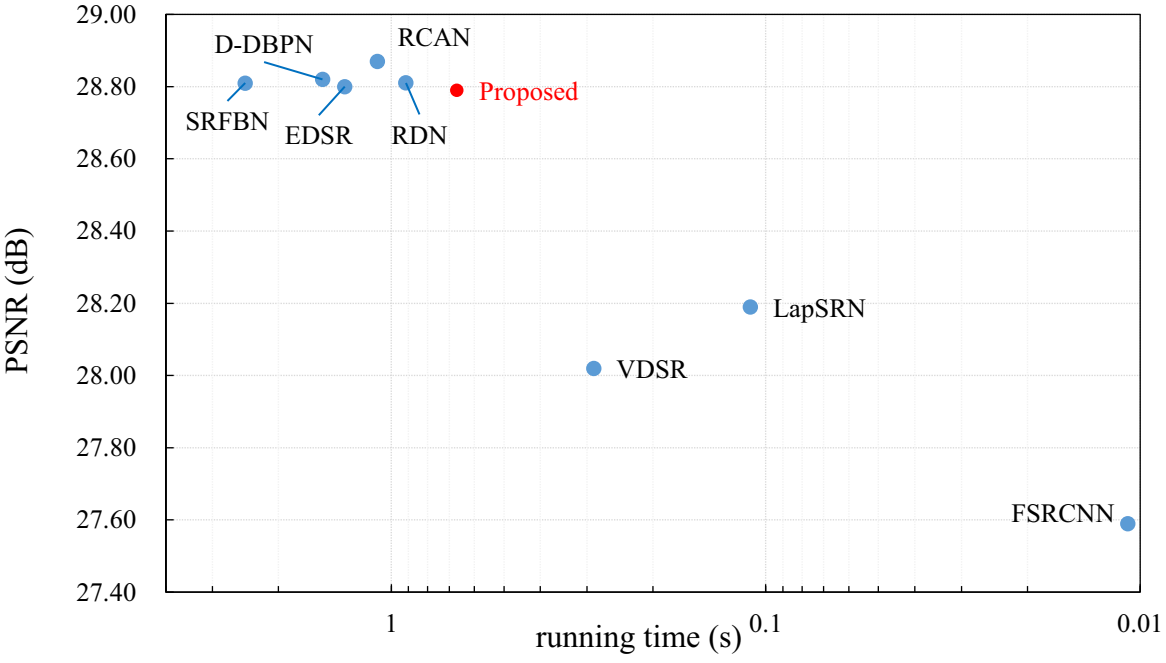
Fig. 12. Visual comparison for the upscaling factor of × 4 on real-world historical images. All models are trained by using by using the images from 291-image dataset.

Quantitative results. We first train our SR models using the images from 91-images dataset and compare with the state-of-the-art SR methods(FSRCNN [20], VDSR [17], DRRN [22], LapSRN [31] and MemNet [32]) that also are trained by the 291-images. Table 2 shows the average PSNR, SSIM and IFC results of reconstructed HR images on the five representative image datasets for the different upscaling factors (× 2, × 3 and × 4). From Table 2, we can see that the proposed method achieves the consistent performance on all the datasets. Due to the limitations of the training images, the performance of all methods tends to decline with the increase of the test images. However, the proposed method still performs better than all the compared methods in both PSNR and SSIM. These experimental results indicate that the proposed method can effectively improve the quality of reconstructed HR images.

Furthermore, we use more images from DIV2K [30] datasets to push our SR models to the best performance. The final quantitative results are shown in Table 3. From the Table 3, our



(a) Performance and number of parameters



(b) Performance and execution time

Fig. 13. Performance, number of parameters and execution time of the proposed and existing methods. The results are evaluated on all the images in Set14 with the upscaling factor of 4. Our method still performs competitively with the existing methods in terms of balancing the reconstruction accuracy and efficiency.

SR models achieve an improved performance with the increase of training images and have comparable PSNR and SSIM results with the comparative SR methods: EDSR [21], RCAN [23], D-DBPN [24] , SRFBN [25] and RDN [26]. Compared with our SR method, EDSR methods uses much more filters (256 vs. 128) in each convolution layer to construct SR models, and D-DBPN and SRFBN employ more training images (DIV2K [30] + Flickr2K [21] + ImageNet [47] and DIV2K [30] + Flickr2K [21] vs. DIV2K [30]) to optimize their SR models. However, our SR method still can earn competitive results with these state-of-the-art SR methods.

Visual results. In Figs. 6-8, we show visual comparisons on the images, drawn from B100, Urban100 and Manga109, with the upscaling factors of × 4. As shown in Figs. 6-8, our SR model can re-

construct the desired HR images more accurately. For the '108,005' image from B100 dataset, FSRCNN, VDSR and DRCN fail to recover the clear stripes on tiger in the reconstructed HR images. Although DRRN and LapSRN provide more clear stripes, the results are much smoother. Our SR models can produce clear and sharp SR images which are very close to the ground-truth HR images.

In addition, we also show the visual comparisons of the model trained with the images from DIV2K [34] datasets in Fig 9-10. For challenging details and edges in images "img_011" and "img_078", most compared methods suffer from heavy blurring problem of texture details and edge structures. RCAN and D-DBPN alleviate the blurring artifacts to some degree and provide more details. In contrasts, our proposed method obtains much better HR images by provides more details and sharp edges. These comparisons indicate that our networks can extract more sophisticated features from the LR spaces and can effectively utilize the extracted LR features to reconstruct the desired HR image.

4.3. Super-resolving on real-world images

In this section, we further validate the super-resolving performance of the proposed method on the historical photographs with JPEG compression artifacts. Because neither the ground-truth HR images nor the downsampling kernels are available, our experiment can demonstrate the super-resolving performance of the proposed and compared methods on the real-word images. Fig. 11-12 show the super-resolved historical images of the upscaling factor $\times$ 4. As shown in Fig. 11-12, the proposed method can provide clearer details and sharper edges in the reconstructed HR images than the compared methods.

4.4. Computation complexity

In this section, we discuss the computation complexity of the proposed method. To this end, we show the comparisons about model parameters, execution time and performance in Fig. 13. As shown in Fig. 13(a), the proposed method has more trainable parameters than all the compared methods, because we use a fully connected layer to reconstruct the final HR image. However, as the illustrations in literature [48], the reducing of the trainable parameters does not necessarily improve the reconstruction efficiency of the desired HR images due to the influence of the other factors such as network structure, memory access cost and platform characteristics. Thus, we further conduct the proposed and compared methods on the same platform with Ubuntu 16.04 operating system, 3.6 GHz AMD 2600x CPU, 32 GB memory and NVIDIA 2080Ti GPU, and show trade-offs between the execution time and PSNR value on Set14 dataset for the upscaling factor of $\times$ 4. As shown in Fig. 13(b), although the model of the proposed method has the most trainable parameters, our method still performs favorably against RDN, RCAN, EDSR, D-DBPN and SRFBN in terms of the reconstruction efficiency. This is mainly because our reconstruction layer provides more than 80% trainable parameters of our model. Although the reconstruction layer with convolution can significantly reduce the parameters by sharing the weights in height and width extent, it does not save FLOPs in the testing phase. Therefore, our proposed method still performs well in terms of reconstruction efficiency. In addition, since the proposed method has complex network structure and applies a lot of convolutional layers to extract LR features, the time complexity of our network is also increasing with respect to the improvement of accuracy. However, our method still performs competitively with the existing methods in terms of balancing the reconstruction accuracy and efficiency.

5. Conclusions

In this work, we propose a new SR method based on deep neural networks to reconstruct the desired HR images. By cascading the improved residual blocks to extract features in LR space and jointly optimizing a fully connected reconstruction layer to exploit the differentiated contextual information over the global region of the input LR image, the proposed network effectively alleviates the issues of the existing SR methods and obtains the visually pleasing HR images with the low computational cost. In addition, we propose a new loss function with the edge difference constraint to preserve the sharp edges and restore texture structures. Comparison to the existing SR methods has shown that our method achieves the state-of-the-art performance in terms of balancing the reconstruction accuracy and efficiency. Yet, there are lots of parts that can be improved from our work such as network architecture, zero-shot learning strategies, and multiscale model, and we leave these as future works.

Author contributions

This manuscript was performed in collaboration between the authors. Yongliang proposed the new SISR method based on deep neural networks. Jiashui Huang, Faen Zhang and Weiguo Gong were involved in the writing and argumentation of the manuscript. All authors discussed and approved the final manuscript.

CRediT authorship contribution statement

Yongliang Tang: Conceptualization, Methodology, Software, Investigation, Writing - original draft, Visualization. **Jiashui Huang:** Data curation, Writing - review & editing. **Faen Zhang:** Supervision, Project administration. **Weiguo Gong:** Funding acquisition.

Acknowledgments

This work was supported by Key Projects of Science and Technology Agency of Guangxi province, China (Guike AA 17129002) and the Municipal Science and Technology Project of CQMMC, China (2017030502)

The authors would like to thank the editors and reviewers for their valuable comments and suggestions.

References

[1] L. Zhang, H. Zhang, H. Shen, P. Li, A super-resolution reconstruction algorithm for surveillance images, Signal Process. 90 (2010) 848–859.
[2] S. Peled, Y. Yeshurun, Super-resolution in MRI: application to human white matter fiber tract visualization by diffusion tensor imaging, Off. J. Soc. Magn. Reson. Med. 45 (2001) 29–35.
[3] B.K. Gunturk, A.U. Batur, Y. Altunbasak, et al., Eigen face-domain super-resolution for face recognition, IEEE Trans. Image Process. 12 (5) (2003) 597–606.
[4] M.W. Thornton, P.M. Atkinson, D. a. Holland, Sub-pixel mapping of rural land cover objects from fine spatial resolution satellite sensor imagery using super-resolution pixel-swapping, Int. J. Remote Sens. 27 (3) (2006) 473–491.
[5] R. Keys, Cubic convolution interpolation for digital image processing, IEEE Trans. Acoust., Speech, Signal Process. 29 (06) (1981) 1153–1160.
[6] L. Zhang, X. Wu, An edge-guided image interpolation algorithm via directional filtering and data fusion, IEEE Trans. Image Process. 15 (08) (2006) 2226–2238.
[7] R. Fattal, Image up-sampling via imposed edge statistics, ACM Trans. Graph. 26 (03) (2007) 095–103.
[8] Y. Tai, S. Liu, M. Brown, S. Lin, Super resolution using edge prior and single image detail synthesis, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Francisc , California, 13-18 June 2010, pp. 2400–2407.
[9] L. Wang, S. Xiang, G. Meng, et al., Edge-directed single-image super-resolution via adaptive gradient magnitude self-interpolation, IEEE Trans. Circuits Syst. Video Technol. 23 (8) (2013) 1289–1299.
[10] N. Akhtar, F. Shafait, A. Mian, Bayesian sparse representation for hyperspectral image super resolution, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, Massachusetts, 8-10 June 2015, pp. 3631–3640.
[11] K. Zhang, X. Gao, D. Tao, X. Li, Image super-resolution via nonlocal steering kernel regression regularization, in: IEEE International Conference on Image Processing, Australia, Sept. 2013, pp. 943–946.
[12] R. Timofte, V. De Smet, L. Van Gool, A+: adjusted anchored neighborhood regression for fast super-resolution, in: Asian Conference on Computer Vision, Singapore, Nov. 2014, pp. 111–126.

[13] J. Yang, J. Wright, T.S. Huang, Y. Ma, Image super-resolution via sparse representation, IEEE Trans. Image Process. 19 (11) (2010) 2861–2873.
[14] 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, Boston, Massachusetts, 8-10 June 2015, pp. 3791–3799.
[15] C. Dong, C.C. Loy, K. He, X. Tang, Image super-resolution using deep convolutional networks, IEEE Trans. Pattern Anal. Mach. Intell. 38 (2) (2016) 295–307.
[16] Z. Wang, D. Liu, J. Yang, W. Han, T. Huang, Deep networks for image super-resolution with sparse prior, in: IEEE International Conference on Computer Vision, Santiago, Chile, 13-16 December 2015, pp. 370–378.
[17] Jiwon Kim, J.K. Lee, K.M. Lee, Accurate image super-resolution using very deep convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, Nevada, 27-30 June 2016, pp. 1646–1654.
[18] Jiwon Kim, J.K. Lee, K.M. Lee, Deeply-recursive convolutional network for image super-resolution, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, Nevada, 27-30 June 2016, pp. 1637–1645.
[19] W. Shi, J. Caballer, et al., Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, Nevada, 27-30 June 2016, pp. 1874–1883.
[20] C. Dong, C.C. Loy, X. Tang, Accelerating the super resolution convolutional neural network, in: European Conference on Computer Vision, Amsterdam, Netherlands, 8-16 October 2016, pp. 391–407.
[21] B. Lim, S. Son, H. Kim, S. Nah, K. Lee, Enhanced deep residual networks for single image super-resolution, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, Hawaii, 21-26 July 2017, pp. 1132–1140.
[22] Y. Tai, J. Yang, X. Liu, Image super-resolution via deep recursive residual network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, Hawaii, 21-26 July 2017, pp. 2790–2798.
[23] Y. Zhang, K. Li, L. Wang, Image super-resolution using very deep residual channel attention networks, in: European Conference on Computer Vision, Munich, Germany, 8-14 September 2018, pp. 286–301.
[24] M. Haris, G. Shakhnarovich, N. Ukita, Deep back-projection networks for super-resolution, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, Utah, 18-22 June 2018, pp. 1664–1673.
[25] Z. Li, J. Yang, Z. Liu, X. Yang, Jeon G, W. Wu, Feedback network for image super-resolution, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, 16-20 June 2019.
[26] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, Y. Fu, Residual dense network for image super-resolution, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, Utah, 18-22 June 2018, pp. 2472–2481.
[27] Y. Wang, L. Wang, H. Wang, P. Li, End-to-end image super-resolution via deep and shallow convolutional networks, IEEE Access 7 (2019) 31959–31970.
[28] Z. Hui, X. Wang, X. Gao, Fast and Accurate single image super-resolution via information distillation network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, Utah, 18-22 June 2018, pp. 3791–3799.
[29] A. Shocher, N. Cohen, M. Irani, “Zero-Shot” Super-resolution using deep internal learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, Hawaii, 21-26 July 2017, pp. 3118–3126.
[30] R. Timofte, E. Agustsson, L. Gool, M. Yang, L. Zhang, et al., NTIRE 2017 challenge on single image super resolution: methods and results, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, Hawaii, 21-26 July 2017, pp. 1110–1121.
[31] W. Lai, J. Huang, N. Ahuja, M. 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. 5835–5843.
[32] Y. Tai, J. Yang, X. Liu, C. Xu. MemNet, A persistent memory network for image restoration, in: International Conference on Computer Vision, Venice, Italy, October 22-29, 2017, pp. 4549–4557.
[33] L. Christian, T. Lucas, H. Rerenc, 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. 105–114.
[34] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, International Conference on Learning Representations, 7-9 May 2015.
[35] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, Nevada, 27-30 June 2016, pp. 770–778.
[36] He K., X. Zhang, S. Ren, J. Sun, Identity mappings in deep residual networks, in: European Conference on Computer Vision, Amsterdam, Netherlands, 8-16 October 2016, pp. 630–645.
[37] Zhao H., O. Gallo, I. Frosio, J. Kautz, Loss Functions for Image Restoration With Neural Networks, IEEE Transactions on Computational Imaging 3 (1) (2017) 47–57.
[38] W. Gong, Y. Tang, X. Chen, Q. Yi, W. Li, Combining edge difference with nonlocal self-similarity constraints for single image super-resolution, Neurocomputing 249 (2) (2017) 157–170.
[39] A. Sao, B. Yegnanarayana, B. Kumar, Significance of image representation for face verification, Signal, Image and Video Process. 1 (3) (2007) 225–237.
[40] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, "Caffe: convolutional architecture for fast feature embedding", arXiv preprint, arXiv: 1408. 5093, 2014.
[41] 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, Santiago, Chile, 13-16 December 2015, pp. 1026–1034.
[42] D. Zhang, FSIM: a feature similarity index for image quality assessment, IEEE Trans. Image Process. 20 (8) (2011) 2378–2386.
[43] A. Liu, W. Lin, M. Narwaria, Image quality assessment based on gradient similarity, IEEE Trans. Image Process. 21 (4) (2012) 1500–1512.
[44] W. Xue, L. Zhang, X. Mou, A.C. Bovik, Gradient magnitude similarity deviation: a highly efficient perceptual image quality index, IEEE Trans. Image Process. 23 (2) (2014) 684–695.
[45] W. Wan, J. Wu, G. Shi, Y. Li, W. Dong, Super-resolution quality assessment: subjective evaluation database and quality index based on perceptual structure measurement, IEEE Interantional Conference on Multi Expo, 23-23 July 2018.
[46] H.R. Sheikh, A.C. Bovik, D. Veciana, An information fidelity criterion for image quality assessment using natural scene statistics, IEEE Trans. Image Process. 14 (12) (2005) 2117–2128.
[47] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, ImageNet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (3) (2015) 211–252.
[48] N. Ma, X. Zhang, H. Zheng, J. Sun, Shufflenet v2: practical guidelines for efficient CNN architecture design, in: European Conference on Computer Vision, Munich, Germany, 8-14 September 2018, pp. 122–138.

Yongliang Tang received his PhD in College of Opto-Electronic Engineering from Chongqing University in 2018. Now he is a senior engineer in AInnovation Co. Ltd, China. His-research interests include machine learning and image processing.

Jiashui Huang received the M.S. and Ph.D. degrees in Computer Science from the Zhejiang University, Zhejiang, China, in 2006 and 2010 respectively. He is currently a senior researcher at AInnovation Co. Ltd. His-research interests include computer vision and machine learning, with focus on face recognition and deep learning.

Faen Zhang received the M.S. degree in computer science and theory from Institute of Software, Chinese Academy of Sciences, Beijing, China. From 2008 to 2015, he served as a senior R&D engineer in Microsoft and Google. In 2015, he joined Baidu and served as the chairman of Baidu Cloud Technology Committee, and the chief architect of big data and artificial intelligence of Baidu Cloud. Now, he is the CTO of AInnovation, the Chief Architect of the AI Institute at Sinovation Ventures and Honorary Professor of Ningbo Nottingham University. His-current interests include high performance computing, power efficiency, deep learning and computer vision.

Weiguo Gong received his doctoral degree in computer science from the Tokyo Institute of Technology of Japan in March 1996 as a scholarship gainer of Japan Government. From April 1996 to March 2002, he served as a researcher or senior researcher in NEC Labs of Japan. Now he is a professor of Chongqing University, China. He has published over 120 research papers in international journals and conferences and two books as an author or co-author. His-current research interests are in the areas of pattern recognition and image processing.