Circuits, Systems, and Signal Processing
https://doi.org/10.1007/s00034-020-01630-4

Image Super-Resolution Based on the Down-Sampling Iterative Module and Deep CNN

Xin Yang[1] · Yifan Zhang[1] · Tao Li[1] · Yingqing Guo[1] · Dake Zhou[1]

Received: 18 June 2019 / Revised: 5 December 2020 / Accepted: 11 December 2020

Abstract

Most deep learning-based image SR algorithms do not apply the down-sampling to the reconstructed process. Given this fact and inspired by the iteration idea, we propose a novel image SR method based on the down-sampling iterative module and deep CNN, which explores a new basic iterative module combining up- and down-sampling processes. Each iteration of the iterative module generates the intermediate LR prediction and the HR image. The final reconstructed result is obtained by the weighted summation of the intermediate predicted images generated by multiple iterations. During the training, we adopt the adaptive loss function to achieve fast convergence and accurate reconstruction. Detailed experimental comparisons and analyses show that our method is superior to some state-of-the-art methods in objective performance evaluation and visual effects.

Keywords Super-resolution · Down-sampling · Deep learning · Iterative module · CNN

✉ Xin Yang
yangxin@nuaa.edu.cn

Yifan Zhang
zyf921275@163.com

Tao Li
autolitao@nuaa.edu.cn

Yingqing Guo
839467487@qq.com

Dake Zhou
dkzhou@nuaa.edu.cn

[1] School of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, People's Republic of China

Published online: 03 January 2021

1 Introduction

Image super-resolution (SR) technology produces a high-resolution (HR) image from one or multiple low-resolution (LR) images with poor quality to provide better visual effects and more image information. The SR tasks are generally divided into two categories: multi-frame image SR and single-image SR (SISR). SISR refers to estimating an HR image from a given single LR image when the original image cannot be acquired. In the SISR algorithm, the deep learning-based method has become the mainstream in the current SR field.

Convolutional neural network (CNN) has critical applications in many fields, such as facial expression recognition, staff line removal, and texture image retrieval [1, 8, 12, 15]. In recent years, the SR method based on CNN has also achieved remarkable results. In 2014, inspired by sparse coding, Dong et al. [5] realize the SR reconstruction by CNN (SRCNN), which is the pioneering work of deep learning for SR. Dong et al. [6] design an accelerated SR convolutional network (FSRCNN) in 2016. By increasing the number of convolution layers, the training effect is improved efficiently. The residual network can effectively solve the problem of gradient disappearance. Bee Lim et al. [21] present an Enhanced Deep Residual Network (EDSR), which achieves substantially improved performance by removing the batch normalization layer from each residual and activating ReLU outside the residual block. Fan et al. [7] propose a balanced two-stage residual networks network (EBTSRN), which achieves very promising results when considering either reconstructive accuracy or calculation speed.

Kim et al. [14] propose a deep recursive convolutional network (DRCN) to recursively add the convolutional layer. By increasing the network depth, it avoids introducing additional parameters. Tai et al. [27] construct a persistent memory network MemNet for image restoration. It constructs a memory block consisting of a gate unit and a recursive unit.

Wang et al. [30] design the SCN scheme combining the merits of sparse coding with the domain knowledge of CNN. Lai et al. [16] device a deep Laplacian pyramid SR network (LapSRN), which employs a novel pyramid framework. Using three different models, it performs SR reconstruction well on the scales of $\times 2, \times 4$, and $\times 8$, respectively.

Tong et al. [29] propose the SR-DenseNet by introducing the dense connection. Inspired by SR-DenseNet, Zhang et al. [36] present the residual dense network (RDN), which combines the residual skip connection with the dense connection. The dense deep back-projection network (D-DBPN) [9] uses the iteration to execute back-projection. It can efficiently learn the feedback error information between LR and HR images.

Ledig et al. [17] propose an SR reconstruction method based on the generative adversarial networks (SRGAN), which fully extracts high-frequency information through the alternate execution of the generative model and the discriminative model. Wang et al. [31] propose an enhanced SR model based on GAN (ESRGAN), which achieves better optical quality than SRGAN on more realistic and natural textures.

The adaptive selection and weight of features/coefficients are utilized to enhance feature extraction capabilities [18, 19]. Yanpeng C et al. [2] propose a novel energy-aware improved deep residual network (EA-IDRN), which uses an energy-aware training loss to adaptively adjust the restoration of high-frequency and low-frequency image regions. This technique significantly improves the reconstructive accuracy.

In addition, there are many excellent models [3, 11, 26, 33], which significantly improve the efficiency and reconstruction effect. Similar to the studies [4, 35], Lu et al. [22] fuse the deep information and the shallow contextual information to achieve excellent performance. U-net [25] introduces the down–up-sample layers in semantic segmentation and achieves excellent performance.

In this paper, we design a basic iterative module combining the up-sampling and down-sampling. The iterative module is iterated continuously, and the predicted HR image can be obtained by calculation. The feasibility and superiority of the proposed algorithm are verified by detailed experimental comparison.

The rest of the paper is organized as follows. In Sect. 2, we briefly introduce the related work of our method. In Sect. 3, we propose a novel image SR method based on the down-sampling iterative module and deep CNN. Section 4 gives detailed experiments to prove the feasibility and superiority of our model. Finally, we provide a conclusion in part 5.

2 Related Works

The iterative method is successfully applied to SR, such as DRCN [14]. DRCN is divided into three parts: the embedding network, the inference network, and the reconstruction network, as shown in Fig. 1.

The embedding network converts the input LR image into a set of feature maps used as the inference network input. The embedding network's function $f_1(x)$ can be expressed as

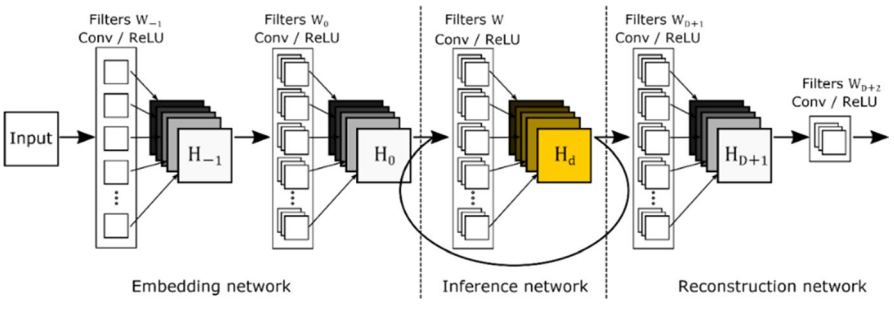

Fig. 1 Unexpanded structure of DRCN

$$\begin{aligned} H_{-1} &= \max(1, W_{-1} * x + b_{-1}) \\ H_0 &= \max(1, W_0 * H_{-1} + b_0) \\ f_1(x) &= H_0, \end{aligned} \tag{1}$$

where * represents the convolutional operation, max represents the activation function ReLU, x denotes the input LR image, W_i $(i=-1, 0)$ represents the weight coefficient, and b_j $(j=-1, 0)$ denotes the bias matrix.

As the main part of DRCN, the inference network iterates a convolution layer 16 times. Its input is H_0, which is the output of the embedding network. The mathematical form of the dth iteration can be represented as

$$H_d = g(H_{d-1}) = \max(0, W * H_{d-1} + b). \tag{2}$$

Inference network iterates the same convolution layer 16 times. Therefore, the weight coefficient W is the same as the bias b. After D iterations, the function f_2 of the inference network can be expressed as

$$f_2(H) = (g \circ g \circ \cdots \circ)g(H) = g^D(H), \tag{3}$$

where g denotes the operation of the convolution layer in a single iteration and g^D is the operation after Dth convolution.

The inputs of the reconstruction network are the output feature map of the first iteration of the inference network and the LR image x, which can be expressed as

$$\hat{y}_d = f_3(x, g_d(f_1(x))). \tag{4}$$

After D iterations $(d = 1, 2, \ldots, D)$, the reconstruction network's function f_3 generates D intermediate predictions of the HR image. The final HR image prediction $\hat{y}$ can be generated by the weighted summing of these all intermediate predictions, which is expressed as

$$\hat{y} = \sum_{d=1}^{D} W_d * \hat{y}_d. \tag{5}$$

After expanding the inference network, the whole structure of DRCN is shown in Fig. 2.

However, multiple iterations of a single convolution layer have no apparent physical meaning except reducing the weight and the bias. It is substantially

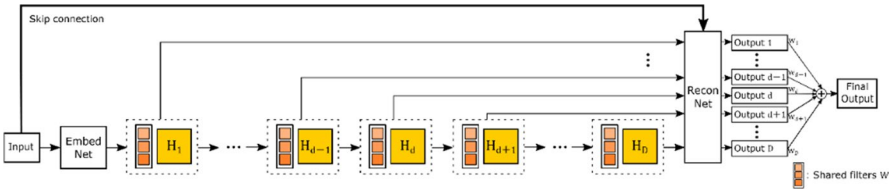

Fig. 2 Expansion structure flowchart of DRCN

equivalent to the convolution layer in VDSR [13] being replaced by multiple iterations of a single convolution layer. Therefore, we adopt the iteration idea, but redesign a novel iterative module.

DRCN uses iterative convolution to extract and reconstruct features and achieves good results. In this paper, we try to add the iterative convolution to the up-sampling and down-sampling modules innovatively. Detailed experiments show that our iterative up- and down-sampling process achieves a good result.

3 The Proposed Algorithms

3.1 Basic Idea

SISR task based on deep learning usually tries to construct an end-to-end network to map LR image to HR image. Generally, a series of feature maps are extracted from the LR image through the feature extraction network. Then, the spatial resolution of the LR image can be improved by one or more up-sampling layers. The mapping from the HR image to the LR image is the inverse process of the reconstruction. Most SR algorithms only consider the mapping from LR image to HR image and ignore the inverse mapping in the reconstruction process. If an SR algorithm can map an LR image to an HR image accurately, its inverse mapping should also be able to map an HR image to an LR image accurately. Considering the process of down-sampling, inverse mapping helps obtain the accurate mapping of the LR image to the HR image in training.

In the early SR method, iterative back-projection (IBP) iteratively calculates the reconstructed error and feeds it back to the reconstruction process to enhance the generated HR image's quality. It has been proved to be effective in improving the quality of the reconstructed images. However, the reconstruction results of IBP are greatly affected by ringing artifacts and checkerboard artifacts. In this paper, we adopt the idea of iteration in the IBP method and complete the mapping from the LR image to the HR image and its inverse mapping by deep learning, efficiently avoiding the influence of ringing artifacts and checkerboard artifacts.

Unlike a priori knowledge of the degradation model used in machine learning, the down-sampling based on deep learning is equivalent to adding an approaching target to the model's training process. In this paper, the predicted LR image is generated and added to the loss function. In this way, the HR image generated by the network model is not only close to the real HR image, but also as close as possible to the actual LR image after down-sampling.

4 The Iterative Module Structure

Inspired by the iteration to a single convolution layer in DRCN's inference network, we construct a basic iterative module with the up-sampling and down-sampling processes. The iterative module is divided into two parts: the up-sampling part and the down-sampling part.

The input of the up-sampling part is an image with LR space size. After extracting the features through convolution layers, the up-sampling operator is used to obtain the prediction of the HR image. The design of the down-sampling part is similar to that of the up-sampling part. The difference is that the input of down-sampling is the HR image and the output is the LR image.

In the iteration, intermediate prediction of LR and HR images is used as the input of the subsequent iteration process and also as a part of the loss function in the training process. It has two advantages as follows:

1. The LR image is added as another label besides the HR image for training. Compared with the loss function using only the HR image as the approaching the target, it means that the LR image obtained from down-sampling of the predicted HR image also needs to be close to the real LR image, which is conducive to the prediction of the HR image close to the real HR image.
2. When the gradient is back-propagated during the training, the gradient information can be directly transferred from the loss function to the early iteration, which improves the convergence performance.

The structure of the iterative module is shown in Fig. 3. From the figure, we can find that the iterative module with only one iteration is still deep when the parameters i_{up} and i_{down} are too large. It is not conducive to the back-propagation of gradients. Therefore, introducing the LR image into the loss function can dramatically improve the back-propagation performance of the gradient of the whole network.

The iterative module is divided into two parts: the up-sampling part and the down-sampling part. The up-sampling part takes the LR image or the intermediate LR image's prediction of the previous iterative module as input. After extracting

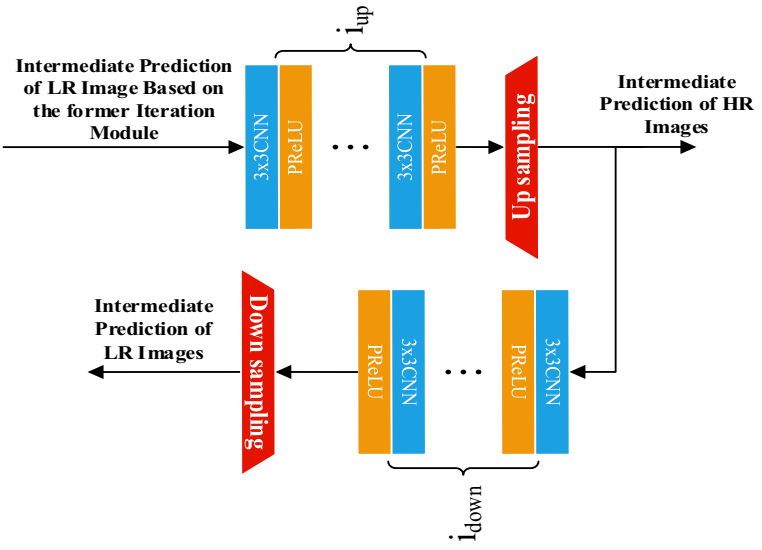

Fig. 3 Structure of the iterative module

features from i_{up} convolution layers, the up-sampling operator generates the iterative module's intermediate predicted HR image. The first convolution layer of the up-sampling part in the iterative module can be expressed as

$$H_{1,d,up} = g(\hat{y}_{LR,d-1}) = \sigma(0, W_{1,d,up} * \hat{y}_{LR,d-1} + b_{1,d,up}), \tag{6}$$

where $\hat{y}_{LR,d-1}$ is the input of the iterative module, i.e., the intermediate predicted LR image of the previous iterative module. The convolution layer after the up-sampling part is expressed as

$$H_{i,d,up} = g(H_{i-1,d,up}) = \sigma(0, W_{i,d,up} * H_{i-1,d} + b_{i,d,up}), \tag{7}$$

where the subscripts i of the feature map H, the weight W, and the bias b of the convolutional layer output represent the ith convolutional layer, the subscript d represents the dth iteration block, and the subscript up represents the up-sampling part of the iterative module. The activation function σ of convolution layer is PReLU, which is defined as

$$F(y) = \max(0, y) + \alpha(\min(0, y)). \tag{8}$$

The up-sampling operator uses the deconvolution layer to complete up-sampling of spatial resolution so that the spatial size of the HR image is obtained. In the down-sampling part, the intermediate HR prediction generated by the up-sampling part of the iterative module is used as the input to generate the intermediate LR prediction of the iterative module. The principle is similar to that of the up-sampling part. The difference is that the down-sampling operator at the end of the down-sampling part sets different strides of convolution layer according to different SR scales to form a trainable down-sampling operator.

4.1 Analysis of the Iterative Module

The whole model of the network adopts the iterative modules as its main component. By running the iterative module iteratively, the intermediate predicted LR image and HR image are generated. Then the ultimate HR prediction can be obtained by weighted summation of the intermediate predicted HR image. The overall structure of our network model is shown in Fig. 4, which is the iterative module's unexpanded structure.

As shown in Fig. 4, the iterative module is the main component in which the convolution layers are included. The LR and HR images are predicted by up- and

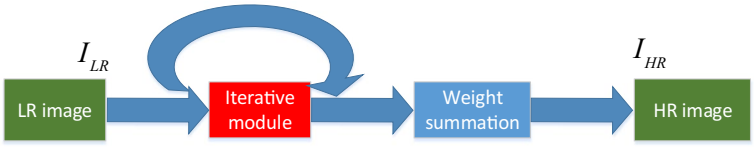

Fig. 4 Unexpanded structure of the iterative module

down-sampling iteratively. The intermediate predictions are output to the end of the network. Then the final predicted HR image is generated by weighted summing the predictions. An important advantage of outputting the intermediate predictions to the end of the network is that the gradient of the loss function can be passed directly to the early iterative module through the skip connection during the training process.

Expanding the iteration part, we get the detailed structure of the iterative module, as shown in Fig. 5. As can be seen from the figure, the whole model includes the iterative module and the weighted summation operations. The mathematical form of the iterative module is represented as

$$(\hat{y}_{LR,d}, \hat{y}_{HR,d}) = f_{iter}(\hat{y}_{LR,d}), \tag{9}$$

where $\hat{y}_{LR,d}$ is the input of the next iteration. $\hat{y}_{LR,d}$ is weighted and then summed to generate the final reconstructed HR image, which can be expressed as

$$\hat{y} = \sum_{d=1}^{D} W_d . \hat{y}_{HR,d}, \tag{10}$$

where W_d is the weights of the intermediate predicted HR image generated by the dth iteration.

4.2 Training of the Iterative Module

Due to a large number of convolutional layers in the model, it is easy to cause the gradient disappearance or the gradient explosion during the training process. Therefore, we connect all the intermediate predicted HR images of the iterative module to the weight to make the model easier to converge. Even if the model uses the mean square error of HR image and real HR image as the loss function, the gradient can reach the early iteration module only through the weighted summation. Therefore, the network convergence in the training process can be guaranteed

For the SR task, the input LR image and the output HR image are highly correlated. The network needs to send most information of the LR image to the end of the network via the convolutional layer. It is inefficient for the very deep network. Due to the difficulty of the back-propagation of gradient, it is complicated to accurately learn the mapping relationship between the input LR image and the output HR in the case of multiple iterations of the iterative modules.

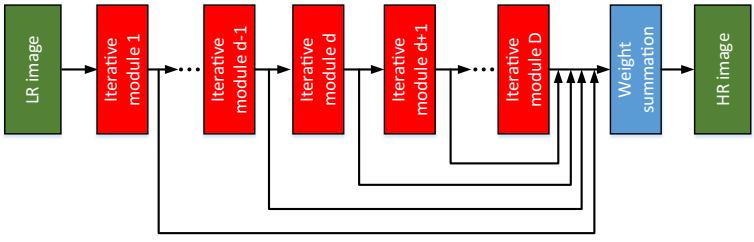

Fig. 5 Expanded structure of the iterative module

In the iterative module, a skip connection is used to directly connect the input LR image to the up-sampling part of the iterative modules. The skip connection has several advantages. Firstly, the network does not need to save the information of the input LR image. Secondly, the information of the input LR image itself can be used when the iterative module generates the intermediate HR prediction. The skip connection is easy and very effective. The LR image is very similar to the HR image in SR reconstruction. In most regions, the difference does not even exist. Therefore, some SR reconstruction methods only predict the details of the image.

The training set can be expressed as $\{x^{(i)}, y^{(i)}\}_{i=1}^{N}$. The goal of SR is to train a network model f that produces accurate predictions of the HR image. Mean square error (MSE) is a widely used loss function in SR. Due to its natural negative correlation with PSNR, minimizing the MSE can achieve better PSNR. The iterative module generates D LR images and an intermediate HR prediction during the iterative process. Therefore, the iterative process can be considered in the design of the loss function. For the Dth iteration, a total of $2 * D + 1$ goals need to be minimized, which are D intermediate LR predictions, D intermediate HR prediction and the final output HR image of the network, respectively. If $l_1(\theta)$ is defined as the loss function of the intermediate predicted stage, we have

$$l_1(\theta) = \sum_{d=1}^{D} \sum_{i=1}^{N} \frac{1}{2DN} \left(\left\| y_{LR}^{(i)} - \hat{y}_{LR,d}^{(i)} \right\|^2 + \left\| y_{HR}^{(i)} - \hat{y}_{HR,d}^{(i)} \right\|^2 \right) y_{HR}^{(i)}, \tag{11}$$

where θ represents the parameter set of the model, $\hat{y}_{LR,d}^{(i)}$ is the intermediate predicted LR image generated by the iterative module in the dth iteration, $\hat{y}_{HR,d}^{(i)}$ represents the intermediate predicted HR image generated by the iterative module in the dth iteration, $y_{LR}^{(i)}$ denotes the real LR image, and $y_{HR}^{(i)}$ represents the real HR image. We define $l_2(\theta)$ as the loss function of the final output HR image of the network, which is given as

$$l_2(\theta) = \sum_{i=1}^{N} \frac{1}{2N} \left\| y_{HR}^{(i)} - \sum_{d=1}^{D} W_d * \hat{y}_{HR,d}^{(i)} \right\|^2. \tag{12}$$

Then, we define the final loss as

$$L(\theta) = \alpha l_1(\theta) + (1 - \alpha) l_2(\theta) + \beta \|\theta\|^2, \tag{13}$$

where α represents the weight of the intermediate predicted LR image and HR image in the process of minimizing the final loss function, and β represents the regularization coefficient of the parameter set θ. At the beginning of the training, α is set to 1 to ensure the easy convergence of the iterative module in the early iterations. The value of α decreases with the training progress to ensure the accuracy of the predicted HR image of the final model.

5 Experiments

5.1 Dataset and Preprocessing

To train the whole system, 91 images in Yang91 [32] and 200 images in B100 [24] are selected as the training set. Given that the richness of data is a crucial factor affecting the model performance, we enhance the dataset to obtain a richer training set. Since learning-based SR is insensitive to the direction of the image, the training dataset is tripled by horizontal and vertical flipping of training images. In training, each image is cropped into a set of small blocks of 41×41 spatial size. Then the whole clipped dataset is grouped into a small batch image set. Each set includes 64 small block images.

In test stage, we select public SET5 [23] and SET14 [34] as the test sets to measure the reconstruction accuracy of our model. We also adopt the B100 and Urban 100 dataset, which contain 100 natural images and 100 real-world images, respectively.

Network initialization: The up- and down-sampling processes in the iterative module use seven convolutional layers to extract the feature map; that is, the parameters i_{up} and i_{down} are both set to 7. The initial value of PReLU is set to zero. The convolutional layer initialization settings in the iterative module are provided in Table 1.

The initial learning rate is set to 0.002. The loss function is minimized through the ADAM approach. When the value of loss functions does not decrease after five consecutive Epochs, the learning rate is reset to one-half of the previous value. When the learning rate is less than 0.00002, the training is terminated. A large number of experiments show that it takes about 7 h to train on a server with a GeForce GTX 1080Ti graphics card.

5.2 Analysis of the Convolutional Layer Number and the Iteration Number

The convolutional layer's number and the iteration number in the iterative module are two critical hyper-parameters impacting on the reconstruction results. By setting the number of convolution layers and the number of iterations, we analyze the influence of these two hyper-parameters on reconstruction results.

The iterative module contains a complete up-sampling process. It can be considered that the up-sampling part of an iterative module is a complete SR reconstruction algorithm. Extracting the features of the input LR image through the convolution layer i_{up}, the predicted HR image is generated by the up-sampling operator. In

Table 1 Setting of the convolution layer in the iterative module

	Kernel_Size	Num_Output
CNN_up_1	$3 \times 3 \times 3$	64
CNN_up_2–7	$3 \times 3 \times 64$	64
CNN_down_1	$3 \times 3 \times 3$	64
CNN_down_2–7	$3 \times 3 \times 64$	64

this paper, we regard the HR image generated in the iterative process as an intermediate prediction. The ultimate HR image is obtained by the weighted summation of all intermediate predictions. Simultaneously, the weights in the iterative module are updated in each iteration. Theoretically, using more convolutional layers in the iterative module to extract features during the up- and down-sampling process can ensure the reconstruction accuracy.

Figure 6 shows the effect of the convolutional layers on PSNR. We perform the experiments on all images in the dataset. Then, we take the average as the final result. It can be seen from the figure that when the number of the iteration is fixed at 6 and i_{up} and i_{down} in the iterative module are less than or equal to 2, the reconstruction accuracy is poor. It is because the number of convolutional layers in the up- and down-sampling process in the iterative module is only 2, which makes the feature extraction ability weak. It can only extract features in a small range so that ignore some areas that are sensitive to local details. In addition, due to the small number of the convolutional layer, the nonlinear mapping ability in the up- and down-sampling process is too weak to learn the mapping and the inverse mapping of the LR image to the HR image.

As the number of convolutional layers continues to increase, the PSNR achieves a consistent improvement. When i_{up} and i_{down} are 7, we can obtain the best PSNR 37.15. If the number of convolutional layers is more than 8, the reconstruction performance has a slight drop because the iterative module does not use dense connections. However, it effectively reduces the amount of computation.

To study the influence of the iteration number on the reconstruction result, the parameters i_{up} and i_{down} of the iterative module are set to 7. The experimental result is given in Fig. 7

From the figure, we can find that with the iteration number increases, the PSNR increases. Then there is a certain decrease when the number of iteration is greater than 6 due to the distortion of the intermediate predicted HR and LR images in the iterative up- and down-sampling processes. The distortion is amplified with the increase in iterations.

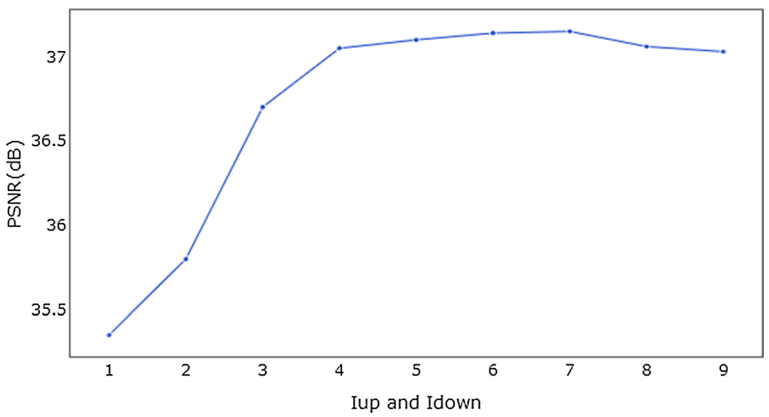

Fig. 6 Effect of the number of convolutional layers on PSNR (the iterative number is 6)

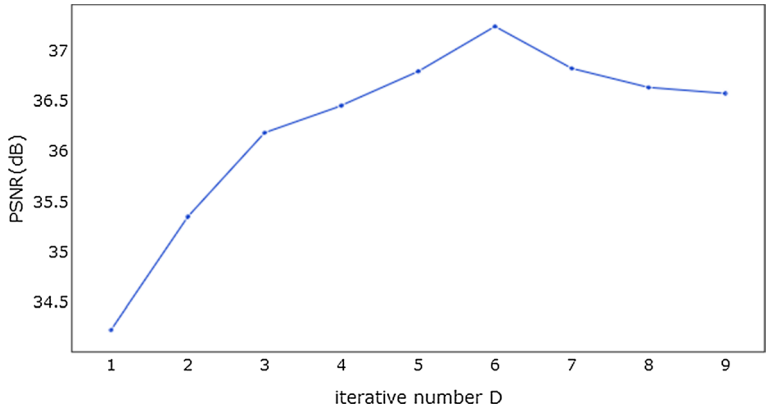

Fig. 7 Effect of iterative number on PSNR

Therefore, the iteration number D is set to 6, and i_{up} and i_{down} are set to 7. The training process of the network is shown in Fig. 8.

Taking SET5 as the test set, we get the convergence curve shown in Fig. 8 when the up-sampling coefficient is 2. It can be seen that the model has excellent convergence performance. Since α in loss function formula (13) is initialized to 1, $l_1(\theta)$ accounts for a large proportion of the loss function in the early stage of training. It dominates the convergence of early training. Therefore, the model quickly converges after the training starts. As the training process progresses, α begins to decline. $l_2(\theta)$ occupies a more significant proportion. The training curve of the model shows some oscillations. However, the PSNR still maintains a gradual improvement. Finally, α is reduced to 0. The loss function only considers the final HR image, and the PSNR of the model keeps a small improvement until the ultimate convergence.

From the whole training process, it can quickly converge and achieve objective reconstruction accuracy. It is because the model uses the intermediate predicted

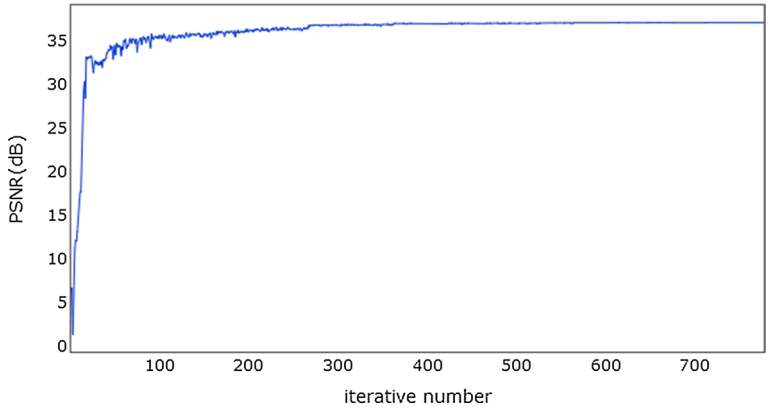

Fig. 8 Convergence of our method

loss function. In some cases where the requirement of the accuracy is not too high, the training can be terminated early by setting different training stop conditions. Besides, when the SR task does not require high accuracy, we can reduce the number of the iterative module or the convolution layer number in the iterative module. It allows the model to update weights online in the application process.

6 Experimental Results and Analysis

To demonstrate the reconstructive accuracy, we compare our method with some classical and state-of-the-art SR methods such as bicubic interpolation, Aplus [28], SelfExSR [10], SRCNN, LapSRN, DRCN, DRRN, VDSR, MemNet, TSCN, DBPN [9] and SRFBN [20] in PSNR and SSIM on the SET5, SET14, B100, and Urban100.

As shown in Table 2, we compare our SR model with some classical or state-of-the-art algorithms on four different datasets. The experimental results on the scale factor of $\times 2, \times 3$, and $\times 4$ are given, respectively. In most cases, our method achieves the best PSNR and SSIM. According to the experimental results, we analyze the reasons for the excellent effect of our model as follows: The deep learning-based SR task is an end-to-end process. By building a neural network for feature extraction and then using an up-sampling layer for amplification, we can map from an LR image to an HR image. Because the amplification layer is almost the same in the SR task, most people focus on the design of the feature extraction layer, while few apply up- and down-sampling layer to the feature extraction layer. In this paper, the up-sampling and down-sampling layers are applied to feature extraction by means of sampling iteration. By generating LR and HR images of intermediate prediction, more supervision information can be added in the training. Gradient information can be directly transferred to each iteration process by the loss function. It makes the reconstructed image closer to the real image.

To verify the effectiveness in visual effects, we select "baby," "bird," and "zebra" images from SET5 and SET14 as the test data. These three images include humans and animals, and their spatial resolution is 512×512, 288×288, and 586×390, respectively. The rich texture in these images can well demonstrate the SR ability to recover the details in generated HR images. Figures 9, 10, and 11 show the comparison results when the magnification factor is set to 2, 3, and 4, respectively. In each figure, the original image, bicubic interpolation, SRCNN, DRCN, VDSR, and the proposed network are arranged from left to right.

It can be seen from the figures that the HR image reconstructed by bicubic interpolation is too smooth, which leads to blurred visual effects. The reconstruction effect of SRCNN is better than that of the bicubic interpolation. However, it produces over-smoothing in the eyelash region. The HR image reconstructed by our model is superior to that reconstructed by DRCN and VDSR in texture details. It demonstrates our algorithm performs as well as some state-of-the-art algorithms. From the criterion PSNR, our algorithm is similar to DRCN and VDSR. It is because that the LR image is a part of the loss function in the iterative module. It is not conducive to achieve a high PSNR. In a word, the HR image reconstructed by our model has more precise details in the visual effect.

Table 2 PSNR and SSIM comparison results of several state-of-the-art SR methods

Scale	Method	Set5 PSNR/SSIM	Set14 PSNR/SSIM	B100 PSNR/SSIM	Urban100 PSNR/SSIM
×2	Bicubic	33.66/0.9299	30.24/0.8688	29.56/0.8431	26.88/0.8403
	Aplus [28]	36.54/0.9544	32.28/0.9056	31.21/0.8863	29.20/0.8938
	SelfExSR [10]	36.50/0.9536	32.22/0.9034	31.17/0.8853	29.52/0.8965
	SRCNN [5]	29.52/0.8965	32.45/0.9067	31.36/0.8879	29.51/0.8946
	LapSRN [16]	37.44/0.9581	32.96/0.9117	31.78/0.8941	30.39/0.9093
	DRCN [14]	37.63/0.9588	33.04/0.9118	31.85/0.8942	30.75/0.9133
	DRRN [26]	37.74/0.9591	33.23/0.9136	32.05/0.8973	31.23/0.9188
	VDSR [13]	37.53/0.9587	33.03/0.9124	31.90/0.8960	30.76/0.9140
	MemNet [27]	37.78/0.9597	33.28/0.9142	32.08/0.8978	31.31/0.9195
	TSCN [11]	37.88/0.9602	33.28/0.9147	32.09/0.8985	31.29/0.9198
	DBPN [9]	38.09/0.9600	32.85/0.9190	32.27/0.9000	32.55/0.9324
	SRFBN [20]	38.11/0.9609	33.83/0.9196	32.29/**0.9010**	**32.62**/0.9328
	Ours	**38.21/0.9628**	**33.30/0.9242**	**32.32**/0.8997	32.56/**0.9334**
×3	Bicubic	30.39/0.8682	27.55/0.7742	27.21/0.7385	24.46/0.7349
	Aplus [28]	32.58/0.9088	29.13/0.8188	28.29/0.7835	26.03/0.7973
	SelfExSR [10]	32.64/0.9097	29.15/0.8196	28.29/0.7840	26.46/0.8090
	SRCNN [5]	32.75/0.9090	29.29/0.8215	28.41/0.7863	26.24/0.7991
	LapSRN [17]	–/–	–/–	–/–	–/–
	DRCN [23]	33.82/0.9226	29.76/0.8311	28.80/0.7963	27.15/0.8276
	DRRN [26]	34.03/0.9244	29.96/0.8349	28.95/0.8004	27.53/0.8378
	VDSR [13]	33.66/0.9213	29.77/0.8314	28.82/0.7976	27.14/0.8279
	MemNet [27]	34.09/0.9248	30.00/0.8350	28.96/0.8001	27.56/0.8376
	TSCN [11]	34.18/0.9256	29.99/0.8351	28.95/0.8012	27.46/0.8362
	DBPN [9]	–/–	–/–	–/–	–/–
	SRFBN [12]	34.70/0.9292	30.51/0.8461	**29.24**/0.8084	28.73/0.8641
	Ours	**34.86/0.9297**	**30.67/0.8463**	29.12/**0.8097**	**28.77/0.8662**
×4	Bicubic	28.42/0.8104	26.00/0.7027	25.96/0.6675	23.14/0.6577
	Aplus [28]	30.28/0.8603	27.32/0.7491	26.82/0.7087	24.32/0.7183
	SelfExSR [10]	30.30/0.8620	27.38/0.7516	26.84/0.7106	24.80/0.7377
	SRCNN [5]	30.48/0.8628	27.50/0.7513	26.90/0.7103	24.52/0.7226
	LapSRN [17]	31.52/0.8854	28.08/0.7687	27.31/0.7255	25.21/0.7545
	DRCN [14]	31.53/0.8854	28.03/0.7673	27.24/0.7233	25.14/0.7511
	DRRN [26]	31.68/0.8888	28.21/0.7721	27.38/0.7284	25.44/0.7638
	VDSR [13]	31.35/0.8838	28.02/0.7678	27.29/0.7252	25.18/0.7525
	MemNet [27]	31.74/0.8893	28.26/0.7723	27.40/0.7281	25.50/0.7630
	TSCN [11]	31.82/0.8907	28.28/0.7734	27.42/0.7301	25.44/0.7644
	DBPN [9]	32.47/0.8980	28.82/0.7860	27.72/0.7400	26.38/0.7946
	SRFBN [20]	32.47/0.8983	28.81/0.7868	27.72/0.7409	26.60/0.8015
	Ours	**32.54/0.8992**	**28.87/0.7871**	**27.89/0.7421**	**26.71/0.8027**

The scale factors are ×2, ×3, and ×4, and the datasets are Set5, Set14, B100, and Urban100

The bold indicates the best performance

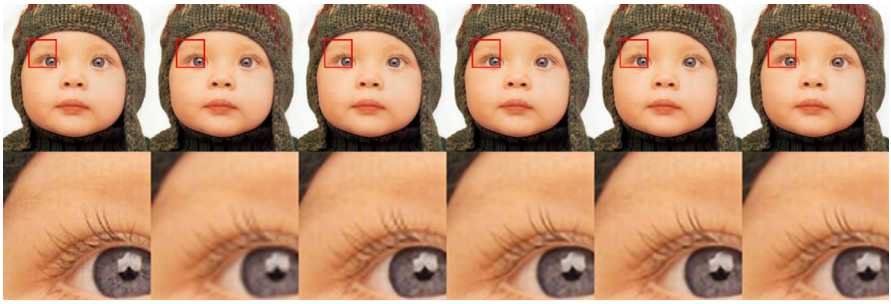

Fig. 9 Reconstruction results of "baby" images under the magnification factor × 2

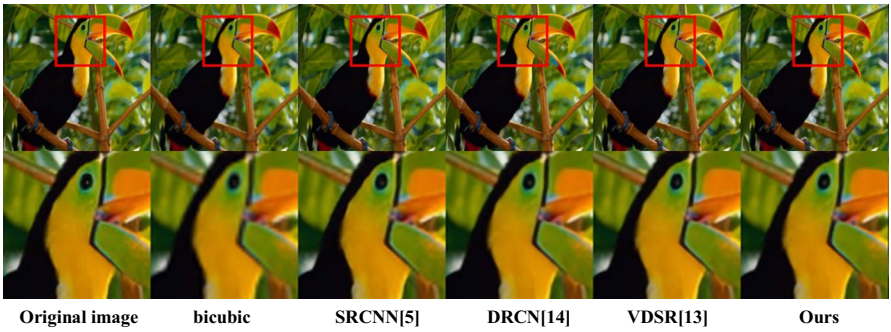

Fig. 10 Reconstruction results of "bird" images under the magnification factor × 3

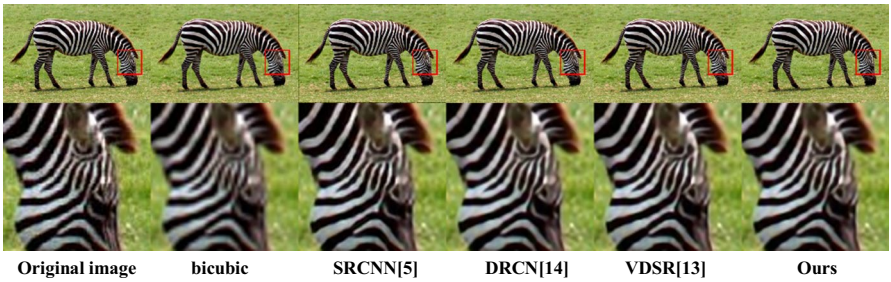

Fig. 11 Reconstruction results of "zebra" images under the magnification factor × 4

Table 3 shows the average running time of different SR algorithms for processing 100 input images with the resolutions of 480 × 360, 640 × 480, and 1280 × 720. The GPU is NVIDIA RTX2080ti (11 GB memory). As can be seen from the table, our network's speed is satisfactory although it is not always the fastest.

Table 3 Average running time results of 7 state-of-the-art methods for scale factors × 4

Dataset	Resolution	VDSR [13]	LapSRN [17]	DRRN [26]	MemNet [27]	TSCN [11]	Ms-LapSRN [17]	Ours
Time (s)	480 × 360	0.0390	**0.0205**	4.0358	7.7708	0.0246	0.0439	0.0218
	640 × 480	0.0739	0.0392	9.4381	13.9254	0.0517	0.0689	**0.0381**
	1280 × 720	0.2028	**0.0904**	22.0019	33.1294	0.1209	0.1846	0.1028

The bold indicates the best performance

7 Conclusions

In this paper, we discuss the iteration idea and introduce the down-sampling to the SR CNN model. Inspired by the concept of iteration, we design a basic iterative module combining up- and down-sampling. The intermediate predicted LR image and the HR image are generated for each iteration of the module. The eventual reconstructed result can be obtained by the weighted summation of intermediate predictions generated by multiple iterations. During the training, deep monitoring is performed according to the iteration process, and the gradient is transferred directly from the loss function to each iteration process. Detailed experimental comparisons and analyses show that our method is similar to or even superior to some state-of-the-art methods in objective evaluation and visual effects.

Acknowledgements This research was supported by the National Natural Science Foundation of China (61573182), and by the Fundamental Research Funds for the Central Universities (NS2020025).

Data availability All data generated or analyzed during this study are included in this published article.

References

1. A.K. Bhunia, S.R.K. Perla, P. Mukherjee, A. Das, P.P. Roy, Texture synthesis guided deep hashing for texture image retrieval, in *2019 IEEE Winter Conference on Applications of Computer Vision* (2019), pp. 609–618
2. Y.P. Cao, Z.W. He, X. Li, Y.L. Cao, J.X. Yang, Fast and accurate single image super-resolution via an energy-aware improved deep residual network. Sig. Process. **162**, 115–125 (2019)
3. X. Cheng, X. Li, J. Yang, Y. Tai, SESR: single image super resolution with recursive squeeze and excitation networks, in *2018 24th International Conference on Pattern Recognition* (2018), pp. 147–152
4. J. Chu, Z.X. Guo, L. Leng, Object detection based on multi-layer convolution feature fusion and online hard example mining. IEEE Access **6**, 19959–19967 (2018)
5. C. Dong, C.C. Loy, K.M. He, X.O. Tang, Image super-resolution using deep convolutional networks. IEEE Trans. Pattern Anal. Mach. Intell. **38**(2), 295–307 (2016)
6. C. Dong, C.C. Loy, X.O. Tang, Accelerating the super-resolution convolutional neural network, in *Computer Vision – ECCV* (2016), pp. 391–407
7. Y.C. Fan, H.H. Shi, J.H. Yu, D. Liu, W. Han, H.C. Yu, Z.Y. Wang, X.C. Wang, T.S. Huang, Balanced two-stage residual networks for image super-resolution, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (2017), pp. 161–168
8. S. Gupta, P.P. Roy, D.P. Dogra, B. Kim, Retrieval of colour and texture images using local directional peak valley binary pattern. Pattern Anal. Appl. **23**, 1569–1585 (2020)
9. M. Haris, G. Shakhnarovich, N. Ukita, Deep back-projection networks for single image super-resolution, eprint arXiv (2019). arXiv:1904.05677
10. J.B. Huang, A. Singh, N. Ahuja, Single image super-resolution from transformed self-exemplars, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 5197–5206
11. Z. Hui, X.M. Wang, X.B. Gao, Two-stage convolutional network for image super-resolution, in *2018 24th International Conference on Pattern Recognition* (2018), pp. 2670–2675
12. J. Kim, B. Kim, P.P. Roy, D. Jeong, Efficient facial expression recognition algorithm based on hierarchical deep neural network structure. IEEE Access **7**, 41273–41285 (2019)
13. J. 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* (2016), pp. 1646–1654

14. J. 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* (2016), pp. 1637–1645
15. A. Konwer, A.K. Bhunia, A. Bhowmick, A.K. Bhunia, P. Banerjee, P.P. Roy, U. Pal, Staff line removal using generative adversarial networks, in *2018 24th International Conference on Pattern Recognition* (2018), pp. 1103–1108
16. W.S. Lai, J.B. Huang, N. Ahuja, M.H. Yang, Deep Laplacian pyramid networks for fast and accurate super-resolution, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 624–632
17. C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z.H. Wang, W.Z. Shi, Photo-realistic single image super-resolution using a generative adversarial network, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 4681–4690
18. L. Leng, M. Li, C. Kim, X. Bi, Dual-source discrimination power analysis for multi-instance contactless palmprint recognition. Multimed. Tools Appl. **76**, 333–354 (2017)
19. L. Leng, J.S. Zhang, J. Xu, M.K. Khan, K. Alghathbar, Dynamic weighted discrimination power analysis in DCT domain for face and palmprint recognition, in *2010 International Conference on Information and Communication Technology Convergence* (2010), pp. 17–19
20. Z. Li, J.L. Yang, Z. Liu, X.M. Yang, G. Jeon, W. Wu, Feedback network for image super-resolution, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 3867–3876
21. B. Lim, S. Son, H. Kim, S. Nah, K.M. Lee, Enhanced deep residual networks for single image super-resolution, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (2017), pp. 136–144
22. Y. Lu, Y. Zhou, Z.Q. Jiang, X.Q. Guo, Z.X. Yang, Channel attention and multi-level features fusion for single image super-resolution, in *2018 IEEE Visual Communications and Image Processing* (2018), pp. 1–4
23. B. Marco, R. Aline, G. Christine, A. Marie, Low-complexity single-image super-resolution based on nonnegative neighbor embedding, in *Proceedings of the 23rd British Machine Vision Conference* (2012), pp. 135.1–135.10
24. A. Mittal, P.P. Roy, P. Singh, B. Raman, Rotation and script independent text detection from video frames using sub pixel mapping. J. Vis. Commun. Image Represent. **46**, 187–198 (2017)
25. O. Ronneberger, P. Fischer, T. Brox, U-net: convolutional networks for biomedical image segmentation, in *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2015), pp. 234–241
26. Y. Tai, J. Yang, X.M. Liu, Image super-resolution via deep recursive residual network, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), pp. 3147–3155
27. Y. Tai, J. Yang, X.M. Liu, C.Y. Xu, MemNet: a persistent memory network for image restoration, in *Proceedings of the IEEE International Conference on Computer Vision* (2017), pp. 4539–4547
28. R. Timofte, V.D. Smet, L.V. Gool, A+: adjusted anchored neighborhood regression for fast super-resolution, in *Asian Conference on Computer Vision* (2014), pp. 111–126
29. T. Tong, G. Li, X.J. Liu, Q.Q. Gao, Image super-resolution using dense skip connections, in *Proceedings of the IEEE International Conference on Computer Vision* (2017), pp. 4799–4807
30. Z.W. Wang, D. Liu, J.C. Yang, W. Han, T. Huang, Deep networks for image super-resolution with sparse prior, in *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 370–378
31. X.T. Wang, K. Yu, S.X. Wu, J.J. Gu, Y.H. Liu, C. Dong, Y. Qiao, C.C. Loy, ESRGAN: enhanced super-resolution generative adversarial networks, in *Proceedings of the European Conference on Computer Vision Workshops* (2018)
32. J.C. Yang, J. Wright, T. Huang, Y. Ma, Image super-resolution as sparse representation of raw image patches, in *2008 IEEE Conference on Computer Vision and Pattern Recognition* (2008), pp. 1–8
33. Y. Yuan, S.Y. Liu, J.W. Zhang, Y.B. Zhang, C. Dong, L. Lin, Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (2018), pp. 701–710

34. R. Zeyde, M. Elad, M. Protter, On single image scale-up using sparse-representations, in *International Conference on Curves and Surfaces* (2010), pp. 711–730
35. Y.Q. Zhang, J. Chu, L. Leng, J. Miao, Mask-refined R-CNN: a network for refining object details in instance segmentation. Sensors **20**(4), 1010 (2020)
36. Y.L. Zhang, Y.P. Tian, Y. Kong, B.N. Zhong, Y. Fu, Residual dense network for image super-resolution, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 2472–2481