JPEG Image Super-Resolution via Deep Residual Network

Fengchi Xu[1], Zifei Yan[1(✉)], Gang Xiao[2], Kai Zhang[1], and Wangmeng Zuo[1]

[1] Harbin Institute of Technology, 92 West Dazhi Street, Harbin 150001, China
cszfyan@gmail.com
[2] No. 211 Hospital of PLA, 45 Xue Fu Street, Harbin 150080, China

Abstract. In many practical scenarios, the images to be super-resolved are not only of low resolution (LR) but also JPEG compressed, while most of the existing super-resolution methods assume compression free LR image inputs. As a result, the JPEG compression artifacts (e.g., blocking artifacts) are often exacerbated in the super-resolved images, leading to unpleasant visual results. In this paper, we address this problem via learning a deep residual convolutional neural network (CNN) that exploits a skips-in-skip connection. More specifically, by increasing the network depth to 31 layers with receptive field of 63 by 63, we train a single CNN model which is able to handle JPEG image super-resolution with various combinations of scale and quality factors, as well as the extreme cases, i.e., image super-resolution with multiple scale factors, and JPEG image deblocking with different quality factors. Our extensive experimental results demonstrate that the proposed deep model can not only yield high resolution (HR) images that are visually more pleasant than those state-of-the-art deblocking and super-resolution methods in a cascaded manner, but also deliver very competitive results with the state-of-the-art super-resolution methods and JPEG deblocking methods in terms of quantitative and qualitative measures.

Keywords: JPEG image Super-Resolution · Deep residual network
Skips-in-skip connection

1 Introduction

Single image super-resolution (SR), with the goal of generating a visually pleasant high-resolution (HR) image from a given low-resolution (LR) input, has been attracting intensive research attentions in the field of image modeling and computer vision. Since SR is a highly ill-posed problem as the number of pixels to be estimated in the HR image is usually much larger than that in the given LR input, the prior knowledge is widely adopted to impose constraint on the solution space. Such prior knowledge can be either predefined or learned from training data. In this sense, state-of-the-art SR methods can be generally divided into two categories. The first one comes from the maximum a posteriori estimation under the Bayesian framework where the likelihood is known while the prior is specified. Popular natural image priors include non-local self-similarity prior [1] and sparsity prior [2]. The second one concerns the discriminative

D.-S. Huang et al. (Eds.): ICIC 2018, LNAI 10956, pp. 472–483, 2018.
https://doi.org/10.1007/978-3-319-95957-3_50

learning based methods which try to learn the relationship between LR and HR space from an abundant of LR and HR exemplar pairs [3, 4].

While significant advances have been achieved with sophisticated image priors, existing SR methods generally assume the observed LR image is also clean, neglecting the potential artifacts along with the observed image. Nowadays, the down-scaled and lossy compressed images are widely transmitted over the internet due to limited network bandwidth and storage capability [5, 6]. While the downscaling scheme reduces the spatial information from the HR image, the subsequent compression scheme would lead to visually unpleasant visual artifacts, mainly known as blocking artifacts. To be clear, JPEG compression scheme divides an image into 8×8 pixel blocks and applies block discrete cosine transformation (DCT) on each block individually. Quantization which is determined by a quality factor is then applied on the DCT coefficients to reduce the storage, thus introducing the blocking artifacts. Here, the quality factor has a range from 0 to 100, and it controls the JPEG compression ratio. The larger the quality factor is, the smaller the compression ratio is and the better the compressed image (which is referred to as JPEG image in this paper) is.

Directly super-resolving the JPEG image by the plain SR methods without preprocessing would concurrently exacerbate the blocking artifacts, leading to more unpleasant visual results. The reason comes from the fact that existing SR methods only consider that an input LR image is only degraded by down-scaling or at most an additional blurring operation which is an impractical assumption under internet environment. Similar findings for noisy image super-resolution have been presented in the work by Singh et al. [7]. More formally, a general assumption for JPEG image super-resolution in the internet environment is that an input LR image is down-scaled and then compressed with a certain quality factor. Apparently, compared to the plain SR, JPEG image super-resolution is a more complex problem since it involves an additional blocking artifacts removal procedure.

An alternative solution for JPEG image super-resolution is to preprocess the image with a deblocking method, followed by applying a plain SR method. However, such an easy pipeline with existing deblocking and SR methods tend to encounter an inevitable drawback, that is, one needs to know or estimates the JPEG quality factor beforehand. Meanwhile, there are few attempts for directly super-resolving the JPEG image. Xiong et al. [5] designed a SR method which is specially designed for JPEG compressed web images. Kang et al. [6] proposed a joint SR and deblocking method for highly compressed image super-resolution via dictionary based sparse reconstruction and morphological component analysis based image decomposition. Nevertheless, those methods still suffer from the unknown JPEG image quality factor induced drawback, thereby having a limitation for practical applications. As a result, there is a demand for designing a JPEG image super-resolution method which is robust to quality factor.

Recent progresses in deep learning have shown the effectiveness and efficiency of convolutional neural networks (CNN) [8, 9] in machine vision field including high-level and low-level vision problems. For low-level vision problems, CNN has been successfully applied in several tasks such as single image deblurring [10], image super-resolution [4, 11], JPEG image deblocking [12] and image segmentation [13]. It is interesting to investigate whether we can learn a CNN model for robust JPEG image super-resolution without consideration of the quality factor. In particular, it would be

very attractive if the trained model can handle the general cases with various combinations of scale and quality factors, as well as the two extreme cases, i.e., image super-resolution with multiple scale factors, and JPEG image deblocking with different quality factors. The question is whether it is possible to train such an integrated CNN model. In fact, there exist several frameworks that are capable of handling the JPEG image deblocking and single image super-resolution tasks individually. Timofte et al. [3] proposed an anchored neighborhood regression framework for single image super-resolution and then used the framework for image deblocking [14]. Kwon et al. [15] proposed a common solution to image super-resolution and compression artifact removal by using Gaussian Processes under a semi-local approximation. Dong et al. [4] proposed a CNN model for single image super-resolution and then modified the network for compression artifacts removal [12]. Chen and Pock [16] proposed a trainable nonlinear reaction diffusion framework for several image restoration problems such as single image super-resolution and JPEG image deblocking. More recently, Kim et al. [17] trained a single convolutional network for multiple scale factors super-resolution. All the above methods indicate that it is possible to learn a single CNN model for JPEG image super-resolution.

In this paper, we propose a single CNN model with a depth of up to 31 layers for JPEG image super-resolution. We refer to our proposed JPEG image super-resolution network as JSCNN. The most distinguished feature of JSCNN is that, instead of handling a certain combination of scale and quality factors by a specifically trained model, it can not only deal with the task of JPEG image super-resolution with various combinations of scale factors and quality factors, but also can suffice the extreme cases, i.e., image super-resolution with several scale factors and JPEG image deblocking with different quality factors. Specially, to enable the network for fast training, several techniques including batch normalization [18], residual learning [19], skips-in-skip connection and gradient clipping-enabled large learning rate [17] are utilized. Our extensive experiments show that our model exhibits highly competitive quantitative and visual results with state-of-the-art JPEG image super-resolution solution within a cascaded manner, while it can be efficiently implemented with GPU.

2 Proposed JSCNN Model for JPEG Image Super-Resolution

Recent years have witnessed the great success of convolutional neural networks in a wide variety of challenging applications. Such a big success can be attributed to the advances in effective and fast learning of deep CNN models. The representative achievements include the proposals of Rectified Linear Unit (ReLU) [9], batch normalization [18] and residual learning [19, 20], and viewpoint on filter size and model depth [21, 22]. Other factors such as the efficient training implementation on modern powerful GPUs and the easy access to large scale dataset are also very important. In general, training a deep CNN for a specific task involves two steps: network architecture design and network learning from training data. In the following, we will illustrate our JSCNN model from those two aspects.

2.1 The Network Architecture

Since the work by Simonyan and Zisserman [21], increasing network depth with 3×3 convolution filters becomes a basic rule of thumb for most of CNN models. Networks with such design possess several properties, e.g., increasing the depth would also increase the receptive field and nonlinearity capacity. Specially, the receptive field with depth of d and 3×3 filters would be $(2d + 1) \times (2d + 1)$ which means, for image super-resolution problem, each output pixel is connected with a patch of size $(2d + 1) \times (2d + 1)$. Thus, the increase of the depth can enable the network use more information to predict the final pixel value. We set the network depth to 31 with a receptive field of 63×63.

Figure 1 shows the architecture of the proposed JSCNN network. In the first layer, we use 64 filters of size 3×3 to generate 64 feature maps, while in the last layer, we use one filter of size $3 \times 3 \times 64$. In the middle layers, we use 64 filters of size $3 \times 3 \times 64$. As in [17], we pad zeros before each convolution to make sure that each feature map of the middle layers has the same size of the input image. We use one big skip connection and 14 small skip connections which in combination form a skips-in-skip connection. The big skip connection enforces the network to predict the residual image of the input while each small skip connection corresponds to a residual unit as in [20]. Besides residual learning strategy, another important technique called batch normalization can accelerate the training and boost the performance. In the rest of this subsection, we illustrate the three key elements of our JSCNN, i.e., batch normalization, residual learning and skips-in-skip connection.

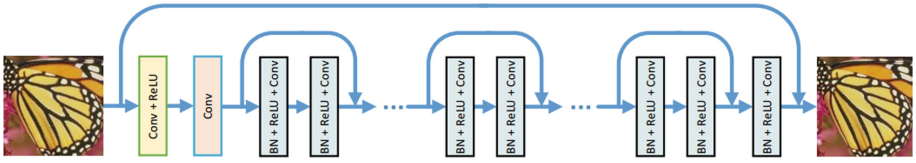


Fig. 1. The architecture of the proposed JSCNN network.

Batch Normalization. It is well-known that one key difficulty in training deep CNNs comes from the problem of vanishing/exploding gradients. This dilemma is caused by the large change in the distributions of internal nodes of a deep network during training (referred to as internal covariate shift [18]). To alleviate the internal covariate shift, the batch normalization was proposed to stabilize the distribution of nonlinearity inputs during the network training [18]. Accomplished by a normalization step that fixes the means and variances of layer inputs and a subsequent scale and shift step before the nonlinearity, batch normalization enjoys several merits. First, it enables large learning rates and thus can speed up the training. Second, it makes the model insensitive to parameter initialization. Third, it plays the role of regularizing the model, and tends to deliver a better performance.

Residual Learning. The residual learning for CNN was originally proposed to solve the performance degradation problem, i.e., performance gets saturated and then

degrades rapidly along with the increase of network depth [19]. The main idea of residual learning lies in that instead of hoping each few stacked layers directly fit a desired underlying mapping, explicitly letting these layers fit a residual mapping would alleviate the degradation problem. In the more recent work by He et al. [20], it has been demonstrated that using identity mappings as the skip connections and after-addition activation can facilitate the forward and backward signals propagation from one block to any other block. As shown in Fig. 1, we follow this strategy to design the small skip connections.

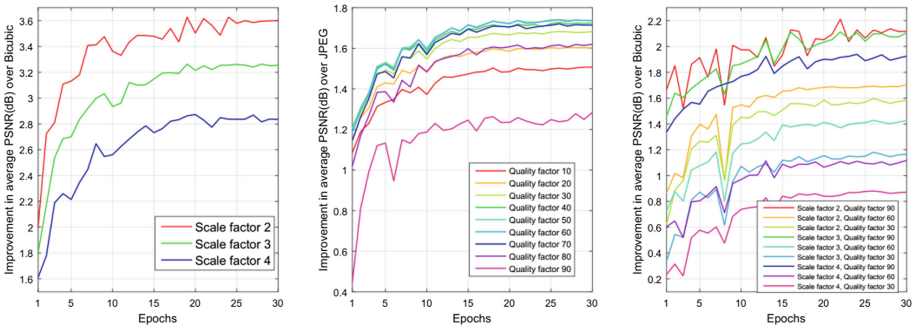
Skips-in-Skip Connection. In most of single image super-resolution works, the main goal lies in predicting the high frequency information (or residual image) of the bicubically interpolated LR image. After that, the final super-resolved image can be obtained by simple addition of the interpolated LR image and the residual image. By learning the residual image rather than the clean image, Kim et al. [17] trained a deep CNN model for single image super-resolution with very promising results. Actually, this is a special case of residual learning, and it has been pointed out in [19] that if the original mapping (i.e., predicting the HR image) is more like identity mapping, then the residual learning (i.e., predicting the residual image) tends to be favorable. According to the above discussion, we can conclude that predicting the residual image can be treated as domain knowledge and it would be beneficial to CNN-based super-resolution. We implement the residual image learning by the big skip connection. As a result, the big and small connections are different levels of residual learning and they both share the merits of residual learning. More precisely, the skips-in-skip connection can result in faster training speed and better performance.

2.2 The Network Learning

Given the network architecture, the next step is to learn the parameters of the network from training data. We first generate an abundant of input and desired output patch pairs and then use the stochastic gradient descent method to minimize the averaged mean squared error loss function. To obtain a fast convergence, the most direct way is to use large learning rate. However, large learning rate is prone to cause exploding gradients problem in the beginning. Usually, there are two strategies to handle the exploding gradients problem. One is to first warm up the training by using a smaller learning rate and then use the large learning rate [20]. The other one is to limit the range of the gradients so that the effective gradient (gradient multiplied by learning rate) is small. In this paper, we adopt the following gradient clipping method proposed in [17] to enable large learning rate.

Gradient Clipping-Enabled Large Learning Rate. By using gradient clipping, Kim et al. [17] trained a CNN model with a very large learning rate of 0.1 for the first 20 epochs. The main idea is to narrow the gradients range when the learning rate is high and free the gradients range when the learning rate is low. Formally, the gradients are clipped into an adjustable interval $[-\tau/\alpha, \tau/\alpha]$ during training, where τ is a predefined threshold and α is the current learning rate. Figure 2 shows the performance of our JSCNN with respect to epochs, here the τ is set to 0.005 and the learning rate was decayed exponentially from 0.1 to 0.0001 for 30 epochs. It can be seen that our

network enjoys a fast convergence. Note that even the batch normalization, fine parameter initialization and residual learning are utilized, the network collapses for convergence without above gradient clipping strategy in a few mini-batch iterations due to exploding gradients problem.



(a) Single Image Super-Resolution (b) JPEG Image Deblocking (c) JPEG Image Super-Resolution

Fig. 2. (a) Average PSNR improvement over Bicubic method for single image super-resolution with respect to epochs on Set5 datasets. (b) Average PSNR improvement over JPEG for image deblocking with respect to epochs on LIVE1 dataset. (c) Average PSNR improvement over Bicubic method for JPEG image super-resolution with respect to epochs on Set5 dataset.

3 Experimental Results

3.1 Training Data

It is widely acknowledged that convolutional neural networks generally benefit from the availability of large training data, thus, as in [17], we use a rather large dataset which consists of 91 images in [2] and 200 images from Berkeley Segmentation Dataset for training. To generate the corresponding LR JPEG image of an HR one, we first downscale the HR image with a certain scale factor to generate the clean LP image, then we compress the clean LR image into JPEG image with a certain quality factor by MATLAB JPEG encoder. Finally, we bicubically interpolated the JPEG image by the same scale factor to obtain an interpolated image such that the interpolated JPEG image and its HR image have pixel-wise correspondence. Note that the two extreme cases, i.e., single image SR and JPEG image deblocking, corresponds to the scale factor with $\times 1$ and quality factor with 100, respectively. Also note that, for color images, we convert the original images into the YCbCr color space and treat the luminance component Y as HR image. Considering the fact that the patch size should be larger than the respective field size (i.e., 63×63) in order to capture enough spatial information for better restoration, we set the training patch size to 64×64. We generate $3{,}000 \times 120$ patch pairs for training, rotation or flip based data augmentation is used during min-batch learning.

3.2 Parameter Setting

We initialize the weights by the method in [23] and use stochastic gradient descent (SGD) with weight decay of 0.0005, a momentum of 0.9 and a mini-batch size of 120. We trained 30 epochs for our model. The learning rate was decayed exponentially from 0.1 to 0.0001 for the 30 epochs. We use the MatConvNet package to train the proposed JSCNN model. All the experiments are carried out in the Matlab (R2015b) environment running on a PC with Intel(R) Core(TM) i7-5820 K CPU 3.30 GHz and an Nvidia Titan X GPU. By benefiting the merits of batch normalization, residual learning, skips-in-skip connection and gradients clipping-enabled large learning rate, our model can converges very fast and it takes about one day to train the model.

3.3 Compared Methods and Testing Datasets

To thoroughly demonstrate the effectiveness of our proposed JSCNN model, we consider three specific image restoration tasks, i.e., single image super-resolution, JPEG image deblocking and the general case for JPEG image super-resolution. For this reason, we use different methods and testing datasets for each task as follows.

For single image super-resolution, we use four state-of-the-art single image SR methods, including three external learning based methods named A+ [3], SRCNN [24] and VDSR [17], and one internal learning based method SelfEx [1]. To give a rather fair comparison, we followed the evaluation code as in [1] where the LR (color) images are provided. Thus, it is normal if the calculated metric values are slightly different from original papers. All implementation codes except [17] are downloaded from the authors' websites and we use the default parameter settings. For testing datasets, we adopt the widely used Set5 and Set14. In addition, B100 which contains 100 natural images from Berkeley Segmentation Dataset and Urban100 which is composed of 100 HR images with transformation-based repetitive structures are also used.

For JPEG image deblocking comparisons, two recently proposed JPEG image deblocking methods, i.e., AR-CNN [12] and TNRD [16], are used. The AR-CNN method trained four specific models for the JPEG quality factors Q = 10, 20, 30 and 40, while TNRD trained three models for the JPEG quality factors Q = 10, 20 and 30, respectively. Following [12], we use the Classic5 and LIVE1 as testing datasets. Classic5 contains 5 classic images and LIVE1 consists of 29 high quality images.

For the general case of JPEG image super-resolution, we use the cascaded model (denoted by AR-SRCNN) which consists of AR-CNN deblocking and a subsequent SRCNN super-resolution for main comparison. Because our model can handle the deblocking task and single image super-resolution task, similar to the above cascaded model, our model can also super-resolve the JPEG image in a cascaded manner. The cascaded model with our JSCNN is denoted as DB-SR-JSCNN. To show the effect of the plain SR method for super-resolving JPEG image, the SRCNN method is also included for comparison. We adopt the Set5 and LIVE1 as testing datasets.

3.4 Quantitative and Qualitative Evaluation

Since the three specific image restoration tasks aim to not only reconstruct the HR images but also generate visually pleasant output, we adopt two image quality metrics, i.e., PSNR and SSIM unless otherwise specified.

Single Image Super-Resolution. Table 1 shows the average PSNR and SSIM results of different single image super-resolution methods on four datasets. Since there is no available source code of VDSR, we just report the results from the original paper. As one can see, A+, SelfEx and SRCNN have very similar results. Even though our JSCNN is proposed for JPEG image super-resolution, it can achieve competing results with VDSR. Specially, our JSCNN and VDSR significantly outperform A+, SelfEx and SRCNN by a large margin. Figure 3 illustrates the super-resolved images by different methods. As can be seen, A+, SelfEx and SRCNN produce more visually pleasant results than the bicubic interpolation method; however, they tend to generate smoothed and unnatural edges. In comparison, our proposed JSCNN yields very appealing visual results as it can preserve image sharpness and naturalness.

Table 1. Average PSNR (dB)/SSIM results for *single image super-resolution* with scale factors ×2, ×3, ×4 on datasets Set3, Set14, B100 and Urban 100.

Dataset	Scale	Bicubic PSNR/SSIM	A+ PSNR/SSIM	SelfEx PSNR/SSIM	SRCNN PSNR/SSIM	VDSR PSNR/SSIM	JSCNN (Ours) PSNR/SSIM
Set5	× 2	33.64/0.9292	36.51/0.9536	36.46/0.9534	36.62/0.9534	37.53/0.9587	37.32/0.9564
	× 3	30.39/0.8678	32.58/0.9082	32.58/0.9090	32.74/0.9084	33.66/0.9213	33.71/0.9199
	× 4	28.42/0.8101	30.28/0.8599	30.32/0.8625	30.48/0.8628	31.35/0.8838	31.35/0.8822
Set14	× 2	30.22/0.8683	32.26/0.9050	32.21/0.9033	32.42/0.9061	33.03/0.9124	32.98/0.9105
	× 3	27.53/0.7737	29.12/0.8185	29.16/0.8196	29.27/0.8210	29.77/0.8314	29.76/0.8304
	× 4	25.99/0.7023	27.32/0.7492	27.39/0.7517	27.48/0.7510	28.01/0.7674	28.02/0.7666
B100	× 2	29.55/0.8425	31.21/0.8856	31.17/0.8853	31.34/0.8872	31.90/0.8960	31.86/0.8938
	× 3	27.20/0.7382	28.29/0.7831	28.29/0.7840	28.40/0.7857	28.82/0.7976	28.82/0.7960
	× 4	25.96/0.6672	26.82/0.7086	26.84/0.7086	26.90/0.7100	27.29/0.7251	27.29/0.7235
Urban100	× 2	26.66/0.8408	28.90/0.8970	29.31/0.9022	29.08/0.8966	30.76/0.9140	30.40/0.9161
	× 4	23.14/0.6573	24.33/0.7185	24.80/0.7376	24.52/0.7224	25.18/0.7524	25.27/0.7532

JPEG Image Deblocking. Table 2 shows average PSNR, SSIM and PSNR-B results for JPEG image deblocking with quality factors Q = 10, 20, 30 and 40 on datasets Classic5 and LIVE1. Note that PSNR-B is a specially designed metric for deblocked image quality assessment. It can be seen that our proposed JSCNN has an average PSNR and PSNR-B improvement larger than 0.3 dB over the state-of-the-art AR-CNN method on all the quality factors. Also, our proposed JSCNN achieves better results than TNRD. Figure 4 shows the JPEG image deblocking results of "Carnivaldolls" (LIVE1) with quality factor Q = 10. We can see that our proposed JSCNN can generate shaper edges than AR-CNN and TNRD.

Table 2. Average PSNR (dB)/SSIM/PSNR-B(dB) results for *JPEG image deblocking* with quality factors Q = 10, 20, 30 and 40 on datasets Classic5 and LIVE1.

Dataset	Quality	JPEG PSNR/SSIM/PSNR-B	AR-CNN PSNR/SSIM/PSNR-B	TNRD PSNR/SSIM/PSNR-B	JSCNN (Ours) PSNR/SSIM/PSNR-B
Classic5	10	27.82/0.7800/25.21	29.04/0.8111/28.75	29.28/0.8170/29.03	29.43/0.8202/29.04
	20	30.12/0.8541/27.50	31.16/0.8694/30.60	31.47/0.8744/31.05	31.69/0.8779/31.20
	30	31.48/0.8844/28.94	32.52/0.8967/31.99	32.78/0.8989/32.24	32.98/0.9018/32.37
	40	32.43/0.9011/29.92	33.34/0.9101/32.80	–	33.82/0.9146/33.17
LIVE1	10	27.77/0.7905/25.33	28.96/0.8217/28.68	29.15/0.8251/28.88	29.26/0.8262/28.83
	20	30.07/0.8683/27.57	31.29/0.8871/30.76	31.46/0.8906/31.03	31.66/0.8941/30.99
	30	31.41/0.9000/28.92	32.69/0.9166/32.15	32.83/0.9176/32.28	33.07/0.9206/32.26
	40	32.35/0.9173/29.96	33.63/0.9306/33.12	–	34.05/0.9346/33.22

JPEG Image Super-Resolution. Table 3 shows the average PSNR and SSIM results for JPEG image super-resolution with quality factor Q = 40, and scale factor $\times 2$, $\times 3$, $\times 4$ on datasets Set5 and LIVE1. Since the bicubic interpolation method can even have better PSNR and SSIM results than SRCNN with quality factor Q = 40. Thus, we use Table 3 as a small example to show the effectiveness of our proposed JSCNN for JPEG image super-resolution. We can have the following observations. First, the plain single image super-resolution method SRCNN loses its effectiveness when the quality factor is set to 40. Second, compared to SRCNN, the cascaded model AR-SRCNN can greatly improve the results. Third, our cascaded DB-SR-JSCNN method can surpass the AR-SRCNN method, and our JSCNN can have a comparable performance with the cascaded DB-SR-JSCNN method. Figure 5 shows the visual results of different methods, we can see that SRCNN tends to exacerbate the blocking artifacts, and the super-resolved image by AR-SRCNN still contains the blocking-like artifacts. In comparison, our proposed method can generate more visually pleasant result. Figure 7 shows the average PSNR (dB) improvement of our proposed JSCNN over Bicubic method with different scale factors and JPEG quality factors on Set5 and LIVE1. It can be seen that our proposed JSCNN has a consistent improvement over Bicubic method, indicating than our JSCNN can handle the JPEG image super-resolution with various combination of scale factors and JPEG quality factors. Figure 6 gives an additional example to show the capacity of our JSCNN model. We can see that our JSCNN can produce visually pleasant output result even the input image is corrupted with different distortions in different parts. Actually, other methods such as AR-CNN or SRCNN cannot handle this case well. Thus, our JSCNN is more appealing for real applications.

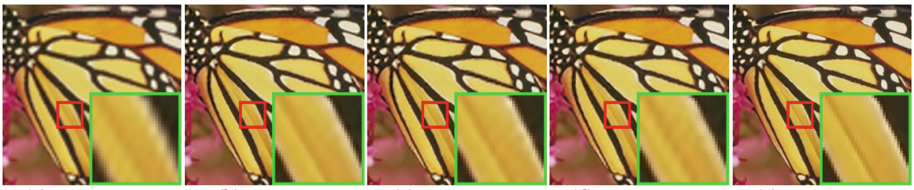


Fig. 3. Super-resolution results of "butterfly" (Set5) with scale factor $\times 3$.

Fig. 4. JPEG image deblocking results of "Carnivaldolls" (LIVE1) with quality factor Q = 10.

Table 3. Average PSNR (dB)/SSIM results for *JPEG image super-resolution* with quality factors Q = 40 and scale factors ×2, ×3, ×4 on datasets Set5 and LIVE1.

Dataset	(Quality, Scale)	Bicubic PSNR/SSIM	SRCNN PSNR/SSIM	AR-SRCNN PSNR/SSIM	DB-SR-JSCNN (Ours) PSNR/SSIM	JSCNN (Ours) PSNR/SSIM
Set5	(40, ×2)	29.98/0.8559	29.57/0.8408	31.14/0.8834	31.64/0.8931	31.68/0.8941
	(40, ×3)	27.64/0.7818	27.39/0.7670	28.60/0.8196	28.95/0.8328	28.96/0.8332
	(40, ×4)	26.09/0.7137	25.87/0.7006	26.86/0.7575	27.16/0.7741	27.12/0.7700
LIVE1	(40, ×2)	27.11/0.7715	27.03/0.7675	27.78/0.7891	28.05/0.7969	28.07/0.7986
	(40, ×3)	25.39/0.6752	25.34/0.6715	25.88/0.6961	26.07/0.7039	26.05/0.7050
	(40, ×4)	24.42/0.6112	24.36/0.6073	24.81/0.6332	24.97/0.6409	24.93/0.6395

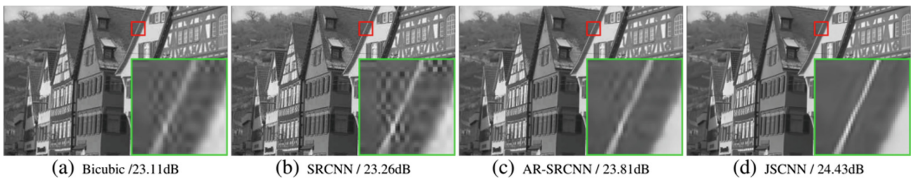


Fig. 5. JPEG image super-resolution results of "Buildings" (LIVE1) with quality factor Q = 40 and scale factor ×2.

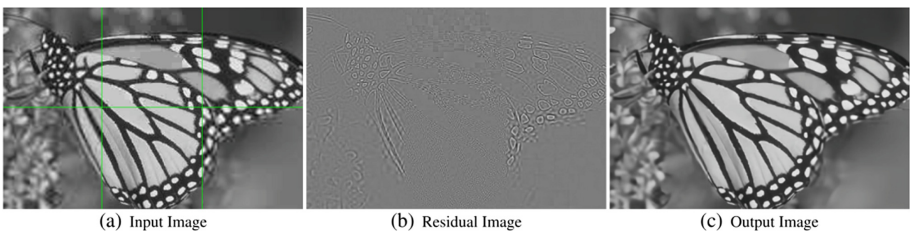


Fig. 6. An example to show the capacity of our proposed JSCNN. The input image is composed by bicubically interpolated LR images with scale factor ×2 (upper left) and scale factor ×4 (lower left), JPEG images with quality factor 10 (upper middle) and quality factor 40 (lower middle), and bicubically interpolated LR JPEG image with scale factor 2, quality factor 40 (upper right) and scale factor 3, quality factor 60 (lower right). Note that the green line in the input image is just used for distinguish the six parts, and the residual image is normalized into the range of [0, 1] for visualization. Even the input image is corrupted with different distortions in different parts, the output image looks natural and does not have obvious artifacts.

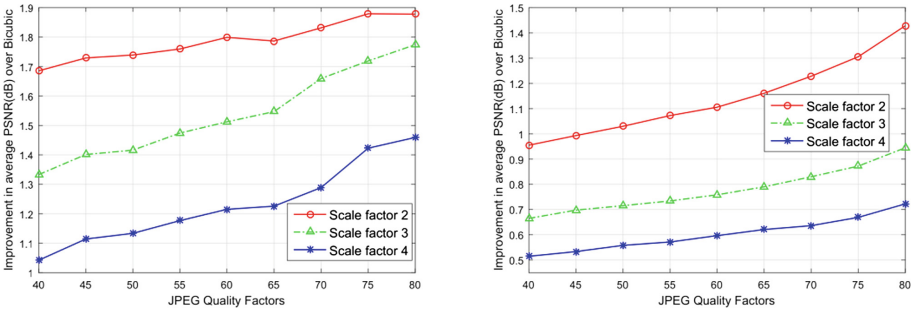


Fig. 7. The average PSNR (dB) improvement of our proposed JSCNN over Bicubic method with different scale factors and JPEG quality factors on Set5 (left) and LIVE1 (right).

Running Time. In addition to visual quality, another important aspect for practical applications is the testing speed. We use the Nvidia cuDNN-v5 deep learning library to accelerate the GPU computation of the proposed JSCNN. We do not count the memory transfer time between CPU and GPU. Our JSCNN can process an image of size 1024×1024 in 0.5 s.

4 Conclusions

We proposed a single deep residual network for JPEG image super-resolution. Different from conventional image super-resolution and deblocking methods which learn a specific model with a specific scaling factor for super-resolution or a specified quality factor for JPEG deblocking, our model can deal with the task of JPEG image super-resolution with various combinations of scaling factors and quality factors. Our model exhibits highly competitive quantitative and qualitative performance with state-of-the-art JPEG image super-resolution solutions, while it can be efficiently conducted with GPU. Furthermore, our model enjoys a fast convergence to a good solution by benefiting from various techniques including batch normalization, residual learning, skips-in-skip connection and gradient clipping-enabled large learning rate. Our model can be readily extended to other low-level vision problems such as simultaneous super-resolution, denoising and deconvolution.

References

1. Huang, J.B., Singh, A., Ahuja, N.: Single image super-resolution from transformed self-exemplars. In: CVPR, pp. 5197–5206. IEEE (2015)
2. Yang, J., Wright, J., Huang, T.S., Ma, Y.: Image super-resolution via sparse representation. IEEE TIP **19**, 2861–2873 (2010)
3. Timofte, R., De Smet, V., Van Gool, L.: A+: adjusted anchored neighborhood regression for fast super-resolution. In: Cremers, D., Reid, I., Saito, H., Yang, M.-H. (eds.) ACCV 2014. LNCS, vol. 9006, pp. 111–126. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16817-3_8

4. Dong, C., Loy, C.C., He, K., Tang, X.: Learning a deep convolutional network for image super-resolution. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8692, pp. 184–199. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10593-2_13
5. Xiong, Z., Sun, X., Wu, F.: Robust web image/video super-resolution. IEEE TIP **19**, 2017–2028 (2010)
6. Kang, L.W., Hsu, C.-C., Zhuang, B., Lin, C.-W., Yeh, C.-H.: Learning-based joint super-resolution and deblocking for a highly compressed image. IEEE Trans. Multimed. **17**, 921–934 (2015)
7. Singh, A., Porikli, F., Ahuja, N.: Super-resolving noisy images. In: CVPR, pp. 2846–2853 (2014)
8. LeCun, Y., Kavukcuoglu, K., Farabet, C., et al.: Convolutional networks and applications in vision. In: ISCAS, pp. 253–256 (2010)
9. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS, pp. 1097–1105 (2012)
10. Xu, L., Ren, J.S., Liu, C., Jia, J.: Deep convolutional neural network for image deconvolution. In: NIPS, pp. 1790–1798 (2014)
11. Bruna, J., Sprechmann, P., LeCun, Y.: Super-resolution with deep convolutional sufficient statistics, arXiv preprint arXiv:1511.05666 (2015)
12. Dong, C., Deng, Y., Loy, C.C., Tang, X.: Compression artifacts reduction by a deep convolutional network. In: ICCV, pp. 576–584 (2015)
13. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR, pp. 3431–3440 (2015)
14. Rothe, R., Timofte, R., Van Gool, L.: Efficient regression priors for reducing image compression artifacts. In: ICIP, pp. 1543–1547 (2015)
15. Kwon, Y., Kim, K.I., Tompkin, J., Kim, J.H., Theobalt, C.: Efficient learning of image super-resolution and compression artifact removal with semi-local gaussian processes. IEEE TPAMI **37**, 1792–1805 (2015)
16. Chen, Y., Pock, T.: Trainable nonlinear reaction diffusion: a flexible framework for fast and effective image restoration, arXiv preprint arXiv:1508.02848 (2015)
17. Kim, J., Lee, J.K., Lee, J.K.: Accurate image super-resolution using very deep convolutional networks, arXiv preprint arXiv:1511.04587 (2015)
18. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167 (2015)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition, arXiv preprint arXiv:1512.03385 (2015)
20. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks, arXiv preprint arXiv:1603.05027 (2016)
21. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014)
22. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR, pp. 1–9 (2015)
23. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: ICCV, pp. 1026–1034 (2015)
24. Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks. IEEE TPAMI **38**, 295–307 (2016)