Multimedia Tools and Applications (2020) 79:9019–9035
https://doi.org/10.1007/s11042-018-7017-y

Clustering based multiple branches deep networks for single image super-resolution

Zhen Li[1] · Qilei Li[1] · Wei Wu[1] · Zongjun Wu[1] · Lu Lu[1] · Xiaomin Yang[1]

Received: 31 July 2018 / Revised: 12 October 2018 / Accepted: 30 November 2018 /
Published online: 14 December 2018

Abstract

Since the limitation of optical sensors, it's often hard to obtain an image with the ideal resolution. Image super-resolution (SR) technology can generate a high-resolution image from the corresponding low-resolution image. Recently, deep learning (DL) based SR methods draw much attention due to their satisfying reconstruction results. However, these methods often neglect the diversity of image patches. Therefore, the reconstruction effect is limited. To fully exploit the texture variability across different image patches, we propose a universal, flexible, and effective framework. The proposed framework can be adopted to any DL based methods. It can significantly improve the SR accuracy while maintaining the running time. In the proposed framework, K-means is employed to cluster image patches into different categories. Multiple CNN branches are designed for these different categories to reconstruct the SR image. Each branch is weighted in accordance with the Euclidean distance to the cluster centers. Experimental results demonstrate that by applying the proposed framework, performance of the DL based SR method can be significantly improved.

Keywords Image super-resolution · Deep learning · K-means clustering · Multiple CNN branches

✉ Wei Wu
wuwei@scu.edu.cn

Zhen Li
2016222055165@stu.scu.edu.cn

Qilei Li
qilei.li@outlook.com

Zongjun Wu
zongjunwu@foxmail.com

Lu Lu
lulu19900303@126.com

Xiaomin Yang
arielyang@scu.edu.cn

[1] College of Electronics and Information Engineering, Sichuan University, Chengdu, Sichuan, 610064, China

1 Introduction

An image with high-resolution facilitates human perception and computer analysis [28]. However, due to the limitation of optical sensors, it's often difficult to obtain an image with the desired resolution. Besides, for some critical engineering applications, such as geographic information system [16] and surveillance system [39], it is even harder to obtain an HR image because of the imaging speed requirements. Though the HR image can be obtained by setting up a more sophisticated optical sensor, this not increase the cost, but also harmful to real-time imaging. To address the problem, single image super-resolution (SR) [15], which is a classical task in computer vision, aims to reconstruct a high-resolution (HR) image from a corresponding low-resolution (LR) image. Due to an LR image may be corrupted via multiple degradation operations, SR is usually classified as a type of ill-posed problem. To date, various SR methods have been proposed, and they can be generally divided into three categories: interpolation based methods, statistic based methods, and learning based methods.

Interpolation [43] based methods estimate the pixels of HR image by using a base function or an interpolation kernel. Among the interpolation based methods, linear interpolation method (Linear) and bicubic interpolation method (Bicubic) are two most original and most representative SR methods. To further optimize the SR results, lanczos filter [5] has been applied to calculate the relationship between LR image and HR image. Besides, some parametric [13] and non-parametric [22, 23, 48] interpolation methods also have been adopted to estimate HR images from the LR images. These interpolation based methods are efficient in time due to the algorithms are not complex. However, these methods perform poorly on modeling the complex mapping from LR images to HR images, causing the blurring and jaggy artifacts near the edges of the HR images. Moreover, the HR images obtained via the interpolation based methods often suffer overly-smooth regions in the complex areas.

Different from the form, statistic based methods calculate the SR image by using the statistical edge information of corresponding LR image. Raanan et al. [7] took the advantage of distinctive edge dependency between LR and HR to upsample the LR image. Sun et al. [33] learned prior knowledge of the gradient profiles [17] to estimate the HR image. These methods mainly based on priors of edge statistical information and they can achieve good SR results when the upscaling factor is small. However, the SR results obtained by the statistic based methods often lose much high-frequency detail information when the upscaling factor is large.

To date, example based methods [9] are the most successful SR approaches. These methods aim to learn the mapping between the HR image and corresponding LR image by a large number of example pairs. According to the different reference images, these methods can be roughly divided into two categories: internal-example based methods [12–44], and external-example based methods [20–43].

Internal-example based methods set the original image as the HR image, and the corresponding down-sampled images are set to the LR image. Then, the mapping can be learned from these example pairs. In the testing phase, the SR images can be obtained by applying the mapping to the given LR images. Yang et al. [44] exploit the self-similarity of the reference to learn the mapping. Based on this work, Wu et al. [41] combine self-similarity with generalized nonlocal mean to further improve the quality of the SR image. Freedman et al. [8] apply the internal-example based method to video SR. Moreover, Huang et al. [14] use the self-dictionaries to handle the geometric transformations. These methods are simple and effective. However, these internal-example based methods only work well when the image

contains a lot of repeated structures. When the LR image consists of rich textures and rich structures, these methods often fail to generate a satisfying HR image.

External-example based methods utilize example pairs from an external dataset to learn the mapping. A large number of representative example pairs are sampled from the external dataset Then, these example pairs are utilized to learn the universal mapping, which would be used to calculate the SR images. Usually, these representative pairs can be embodied by one or more pre-trained dictionary. For instance, the dictionary learned via sparse coding is widely applied in SR [42, 45, 46]. Yang et al. [45] train two coupled dictionaries, namely LR dictionary and HR dictionary, from LR and HR pairs, respectively. Considering the internal relationship, the SR image can be obtained by applying the sparse coefficients, which is calculated via the LR dictionary. The SR images obtained via these sparse coding based methods contain rich details and sharpened edges. However, to calculate the dictionaries, a lot of computational costs are required. Thus, these methods are limited due to high computational cost and running time. To address this problem, Timofte et al. [34, 35] apply neighbor embedding to SR tasks. In their method, the SR image is calculated via optimizing a least square with l_2 norm regularization. Compared with [45], the computational cost is efficiently reduced. Random forests [2] is also applied to generate the SR image [29, 30]. Generally, these methods require less running time compared with the sparse coding based methods. The biggest problem of these methods is that the forest model is too huge.

Recently, deep learning (DL) has achieved great success in many computer vision tasks. Dong et al. [3] (SRCNN) first introduce convolutional neural network (CNN) to SR. In their method, a CNN with 3 layers is used to model the complex mapping between LR and HR. Liu et al. [25] propose a robust SR method by using deep networks with sparse prior. Inspired by VGG-net [32], Kim et al. [19] (VDSR) propose an SR architecture, in which a lot of small filters are cascaded to calculate the residual component. Then, the SR image can be obtained by combining the LR image and the residual component. The SR images obtained via VDSR demonstrate the outstanding capacity of deep network models. Besides, Kim et al. [20] also design an SR architecture (DRCN), in which 16 recursive layers are used. It solves gradient exploding/vanishing by using recursive-supervision and skip-connection. Though these deep learning based methods can generate satisfying SR results, a lot of computational costs and running time are required. To accelerate the SR process, Shi et al. [31] propose a real-time SR method by using a compact convolutional network model with up-sample filter, which is applied in the last layer to up-sample the output image into the ideal size. By doing so, the computational cost is efficiently reduced. Similarly, based on [3], Dong et al. [4] adopt deconvolution layers with small convolutional kernel size to accelerate the SR process. Inspired by ResNet [12], Lim et al. [24] propose an SR method (EDSR) by using an enhanced deep ResNet, and the excellence of EDSR is proved by winning the NTIRE2017 [36] challenge. Considering the dependencies between LR and HR images, Haris et al. [11] propose an SR method based on deep back-projection networks (DBPN). In their method, a feedback mechanism is utilized to learn the projection errors, which are used to calculate the SR image. Generative adversary network (GAN) [10], which is consist of a discriminator (D) and a generator (G), is widely applied in many computer vision tasks. GAN also achieves promising results in SR. Ledig et al. [21] first introduce GAN to SR. In their method, with the zero-sum game between D and G, the generated SR image can recover photo-realistic textures. Wang et al. [40] use GAN with the spatial feature transform layer to generate the SR image with realistic and visually pleasing textures. However, these methods often neglect the diversity of image patches. Therefore, the reconstruction effect is limited.

Loss function is a key factor for these DL based SR methods. The mean square error, l_2, is widely employed in many image restoration tasks [3, 4, 37]. It is utilized to calculate the l_2 distance between the SR image and the corresponding HR image. The structural similarity index loss (SSIM) is another popular reference-based measure. It is based on that human vision system (HVS) is sensitive to local structural changes. Johnson et al. [18] propose the perceptual loss, in which the correlations among high-level features are considered. Based on this, Ledig et al. [21] propose the perceptual loss for GAN, in which the adversarial loss is also considered.

The contexts across the image patches reflect the diversity. To fully use the diversity of the image patches, we proposed a universal, flexible, and effective framework. The proposed framework can be adopted to any DL based methods. It can significantly improve the SR accuracy while maintaining the running time. In the proposed framework, K-means is employed to cluster the different texture characteristics. This makes full use of the intrinsic characteristics of the training images. To train the convolutional neural network, we propose a new weighted loss function, in which the coefficients are calculated adaptively according to the distances to the cluster centers. To sum up, there are mainly three contributions:

1. We propose a generic SR framework, which fully exploits the texture variability across different image patches. Our framework provides a cogent strategy to improve the SR performance without extra time consumption over existing DL based methods with a single branch. Experimental results show the superiority of our framework.
2. We propose a clustering based multiple branch networks for our framework. The coefficients, which is calculated via the relationship between the cluster center and the input patch, induce multiple branches to learn the diversity of different image patch. During the reconstruction stage, the outputs from multiple branches are combined in accordance with the coefficient maps.
3. We introduce a new weighted loss function to our framework. This loss function computes the Euclidian distance between the HR image and the weighted sum of the output from multiple branches. It enforces a prior knowledge based on cluster information to each branch in our network.

To validate the effectiveness of the proposed framework, experiments are performed on VDSR with applying the proposed framework (VDSR-K). By experiments, we demonstrate that the VDSR-K outperforms the original VDSR.

The rest of the paper is organized as follow. Section 2 briefly reviews the VDSR and Kmeans. Section 3 describes the proposed framework in detail. Section 4 analyses the experimental results. Finally, Section 5 concludes the paper.

2 VDSR

VDSR is a representative deep learning based SR approach. Its network structure is shown in Fig. 1.

Network depth is a key factor for SR accuracy. The deeper network can bring a better performance. Out of this motive, Kim et al. deepen the network by cascading small filters (i.e. 3×3) for 20 times. By doing so, the receptive field is effectively increased to 41×41, while the receptive field for SRCNN is 13×13. That is to say, more image context can be taken into account.

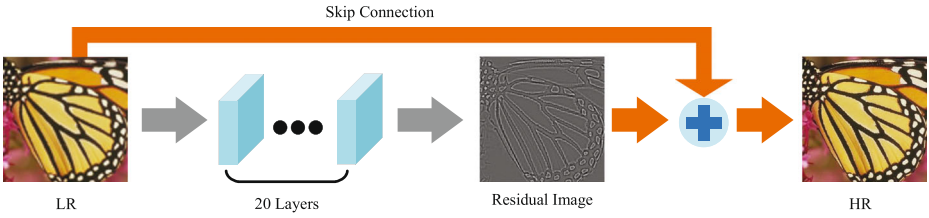


Fig. 1 Structure of the VDSR

In general, deeper networks require more time to train. This not only consumes huge computational costs, but also leads to an adverse effect to the real-time performance [6]. To address this problem, VDSR employs four novel solutions:

1. **Residual learning**: Instead of learning all the information to generate the SR image, VDSR employs a residual learning mechanism, in which only the residual between the HR image and LR image is learned. Compared to other methods, the required computational cost is efficiently reduced.
2. **High learning rate**: Learning rate has a significant impact for network convergence, especially when the network is deep. To accelerate convergence, VDSR initially set the learning rate to 0.1, and it would be decreased by a factor of 10 after every 20 epochs.
3. **Adjustable gradient clipping**: To avoid exploding gradients caused by high learning rate, the gradient is clipped to $[-\frac{\theta}{\gamma}, \frac{\theta}{\gamma}]$, where γ denotes a predefined value, θ denotes the current learning rate, respectively.
4. **Multi-scale**: VDSR utilizes a larger training set to train a multi-scale model. Such a model not only reduces training time, more importantly, it also improves the performance for large scales.

By adopting these strategies, VDSR significantly improves SR accuracy and reduces the required time compared with other methods.

3 Proposed method

3.1 Network architecture

As shown in Fig. 2, our network architecture mainly consists of 2 parts, namely feature extraction and reconstruction. Give an input LR image upsampled to the original size by

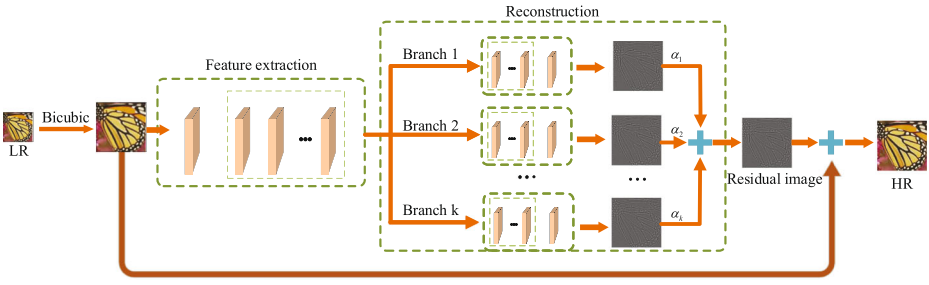


Fig. 2 Network architecture of our VDSR-K

Table 1 Comparison of our network structures

Phase	Layers No.	Branch(es) in VDSR-K	Branch(es) in VDSR	Filter number	Filter size	Activation function
Feature extraction	1	1	1	64	$3 \times 3 \times c_{in}$	ReLu
	2-15	1	1	64	$3 \times 3 \times 64$	ReLu
Reconstruction	16-19	k	1	64	$3 \times 3 \times 64$	ReLu
	20	k	1	c_{out}	$3 \times 3 \times 64$	None

bicubic interpretation, the network aims to learn an output SR image, which is supposed to be close to the ground truth image HR.

It's a rule of thumb that the deeper neural network can better extract the representative feature. Since SR is an ill-posed problem, an LR image may be obtained via different HR images. To recovery the most possible HR image, these representative features play an import role. In this paper, we employ 15 convolutional layers to extract the features. The filter size of the convolutional layer is denoted as $w \times h \times c$, where w is the width, h is the hight, and c is the depth of the channel. In the first layer, we use 64 filters with the size of $3 \times 3 \times c_{in}$, where c_{in} is the color channel. Specifically, experiments are performed on Y channel of YCbCr space, so c_{in} is equal to 1. For each of the other 14 layers, 64 filters with the size of $3 \times 3 \times 64$ are employed. To make full use of the diversity of the image, multiple CNN branches is utilized to reconstruct the SR image. For each branch, 5 CNN layers are used to reconstruction the residual component. In detail, for the first 4 layers, 64 filters with the size of $3 \times 3 \times 64$ are employed. For the last layer, c_{out} filters with the size of $3 \times 3 \times 64$ are employed. c_{out} denotes the color channel of the output image. In this paper, same as c_{in}, c_{out} is equal to 1. The activation function for all the layers except the last layer is $ReLu$. The detail of the network structure is shown in Table 1. We formulate that k is the number of clusters. Thus, the number of convolutional layers in VDSR-K is $15 + k \times 5$. To generate a better residual image, we use no bias for all the CNN layers. The final output of our network is obtained via the weighted sum of all the branches. The weights for different branches is calculated via measuring the distance between the LR image and pre-calculated cluster centers. Moreover, to train the neural networks toward different branches, a new weighted loss function based on l_2 loss is proposed. More detail is explained in the following subsections (Fig. 3).

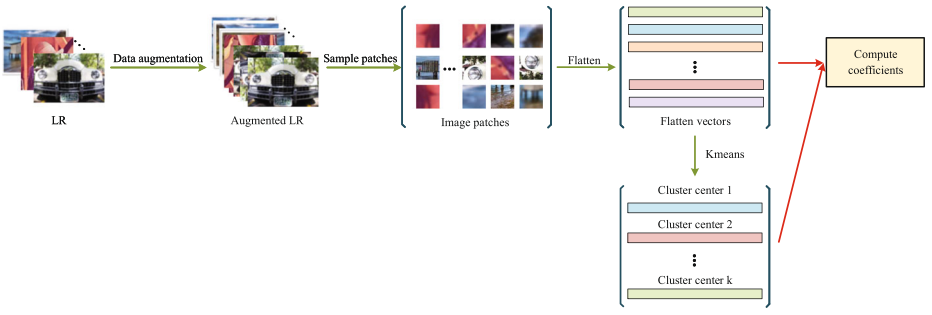


Fig. 3 The pre-process of dataset and calculating the coefficients. Flatten is to reshape an image patch with the size of $m \times n$ into a vector with $m * n$ elements

3.2 Dataset K-means clustering

By applying data augmentation, and cropping from the augmented dataset, a large number of LR patches can be obtained. To make a full use of the diversity of these patches, we use K-means to cluster the LR patch dataset. Given a LR patch dataset $P = (p_1, p_2, \cdots, p_{n-1}, p_n)$, where n is the number of the patches. Every element $p_i, i = (1, 2, \cdots n-1, n)$ is a matrix with the size of $w \times h$. We first flatten it into a column vector with the size of $1 \times w * h$. The flatten dataset $X = (x_1, x_2, \cdots, x_{n-1}, x_n)$ would be used to calculate the cluster centers. This process can be explained as (1).

$$C = Kmeans(X, k) \tag{1}$$

where $Kmeans(\bullet)$ denotes the K-means cluster operation. k is the number of clusters. C, which is a $k \times n$ matrix, denotes the cluster centers.

To calculate the coefficients for different reconstruction branches, we calculate the Euclidean distance from the sample to the cluster centers by solving (2):

$$D = distance(X, C) \tag{2}$$

where $distance(\bullet)$ represents the process of calculating the Euclidean distance. D, which is a $n \times k$ matrix, denotes the Euclidean distance from the sample to the cluster centers. For each column vector d_i consists of k elements in D, the t_{th} element d_i^t denote the distance from x_t to the i_{th} cluster centers.

With the distance D, coefficients α_i^t for the t_{th} sample to the i_{th} reconstruction branch can be easily obtained via solving (3) and (4):

$$w_i^t = \frac{\prod_{j=1}^{k} d_j^t}{d_i^t} \tag{3}$$

$$\alpha_i^t = \frac{w_i^t}{\sum_{n=1}^{k} w_n^t} \tag{4}$$

Equations (3) and (4) reflect the similarity between each sample and a certain cluster.

As shown in Fig. 2, in the reconstruction phase, multiple CNN branches are adopted. The number of the branches is same as the number of clusters. The weighted coefficient is assigned to each convolution branch, and the outputs are summed up to generate the SR residual. Finally, the SR image can be obtained by combining the LR image and the learned residual component. The reconstruction process can be explained in (5).

$$\begin{aligned} F &= \sum_{j=1}^{k} \alpha_j \times channel_j(input) \\ SR &= LR + F \end{aligned} \tag{5}$$

3.3 Loss function

Our framework aims to learn the residual component R of HR image and LR image. Thus, to optimize the neural network, we proposed a weighted loss function based on l_2 loss. Our loss function is shown in (6).

$$L = \sum_{i=1}^{n} \left\| R_i - \sum_{j=1}^{k} \alpha_j F_j(x_i) \right\|_2^2 \tag{6}$$

where $R_i = \mathrm{HR}_i - \mathrm{LR}_i$ denotes the i_{th} residual component, F_j denotes the output from j_{th} branch of construction branches.

With the learned residual component F, the SR image can be obtained via solving (7).

$$SR = LR + F \tag{7}$$

3.4 Reconstruction stage

Our framework employs fully convolution networks (FCN). That is to say, it can accept the input image with arbitrary size in the test phase. Since k CNN branches are adopted, we obtain k outputs, namely $F_1, \cdots, F_k$ in total. We assign the coefficient maps S_i to F_i as its weights (see Fig. 4). To generate the coefficient map S_i, we sample patches from the test image pixel by pixel with the same size in training phase, i.e. 41×41, then the coefficients α for each patch is calculated via solving (3) and (4). We assign α_i to its corresponding area in S_i. Then, all the overlapped elements in S_i are averaged. Thus, the final residual component can be obtained by applying S_i to F_i. This can be expressed as (8).

$$F = \sum_{i=1}^{k} F_i \otimes S_i \tag{8}$$

where $\otimes$ denotes the element-wise product operation.

4 Experiments

4.1 Settings

Following VDSR, we use 291 image dataset, which is consist of 291 nature images, to train our network. We also perform data augmentation by rotating 90°, 180°, 270°, flipping horizontally and vertically. Thus the number of the images used to train the network

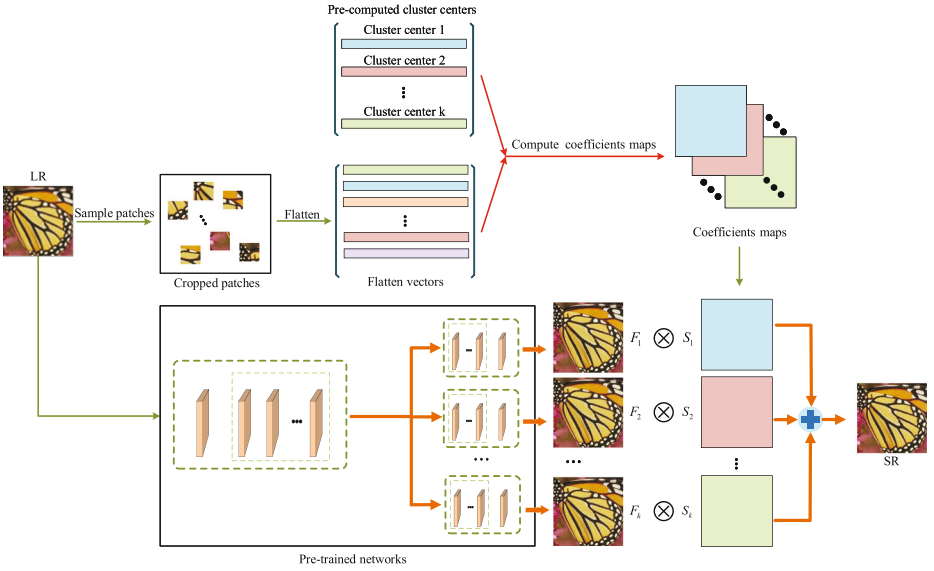


Fig. 4 Reconstruction stage

is $291 \times 5 = 1455$. The degradation operator of the HR image is bicubic interpretation. Then, the LR image is up-sampled to its original size. Experiments are performed on Y channel of YCbCr space. A sliding window, whose size of 41×41, is adopted to sample patch pairs from top left to top right. The stride of the sliding window is set to 41. We obtain 1111616 patch pairs totally. To test the performance of our network, three standard benchmark datasets, namely Set5 [1] with 5 images, Set14 [47] with 14 images, and B100 [26] with 100 images. Moreover, the experimental results are evaluated by SSIM [38] and PSNR. For a fair comparison, we also perform shave operation as VDSR et al. do. As explained in Section 3.1, the network mainly consists of feature extraction part and reconstruction part, the detailed structure is shown in Table 1. The optimizer for our model is stochastic gradient descent (SGD). The learning rate is initialized to 0.1, and divided by 10 after very 20 epochs. Momentum and weight decay are set to 0.9 and 0.0001, respectively. To avoid gradient exploding, we also perform gradient clipping same as VDSR. Our network is trained on 80 epochs with batch size of 128. Every epoch takes about 45 minutes. We use PyTorch [27] with one NVIDIA 980 Ti GPU to implement our model. The source code is available at https://github.com/Paper99/K-means_based_SR/. The proposed method is compared with some state-of-the-art methods, including Zeyde [47], ANR [34], A+ [35], SRCNN [3], FSRCNN [4], and VDSR [19]. We re-measure the performance of these contrast methods by using the relevant public codes. The parameters for these contrast experiments are set in accordance with the corresponding publications.

4.2 Influence of the number of clusters

By applying the sample to the cluster centers, the coefficients α can be obtained. Each coefficient α_i denotes the weight for the i_{th} branch in the reconstruction part.

To investigate the influence of the number of clusters, we evaluate our framework VDSR-K_C2 and VDSR-K_C3 with group dataset into 2 clusters and 3 clusters, respectively. In this phase, experiments are performed on Set5, Set14, and B100 with up-sacle of 2, 3, and 4. The subjective evaluation metrics are shown in Table 2. It can be seen that with the increase of the number of the clusters, the performance gets better. That is because of more clusters can make more detailed distinctions between the texture characteristics of the image set. At the reconstruction part of VDSR-K, more branches can effectively encode more representative features of different texture categories. Thus the performance can be efficiently improved.

4.3 Comparison with the state-of-the-arts

In this section, we compare our VDSR-K with some state of the art methods, namely Zeyde, ANR, A+, SRCNN, FSRCNN, and VDSR. We use VDSR-K_C3 for an objective comparison, VDSR-K_C2 for subjective comparison The quantitative evaluation on Set5, Set14, and B100 with upsacle of 2, 3, and 4 is shown in Table 4. For both PSNR and SSIM, the higher

Table 2 Comparison of different cluster numbers in average PSNR

Model	Set5			Set14			B100		
Upscale	×2	×3	×4	×2	×3	×4	×2	×3	×4
VDSR-K_C2	37.51	33.72	31.38	**33.03**	29.79	28.00	31.90	28.83	27.28
VDSR-K_C3	**37.52**	**33.76**	**31.41**	32.97	**29.80**	**28.04**	**31.91**	**28.85**	**27.29**

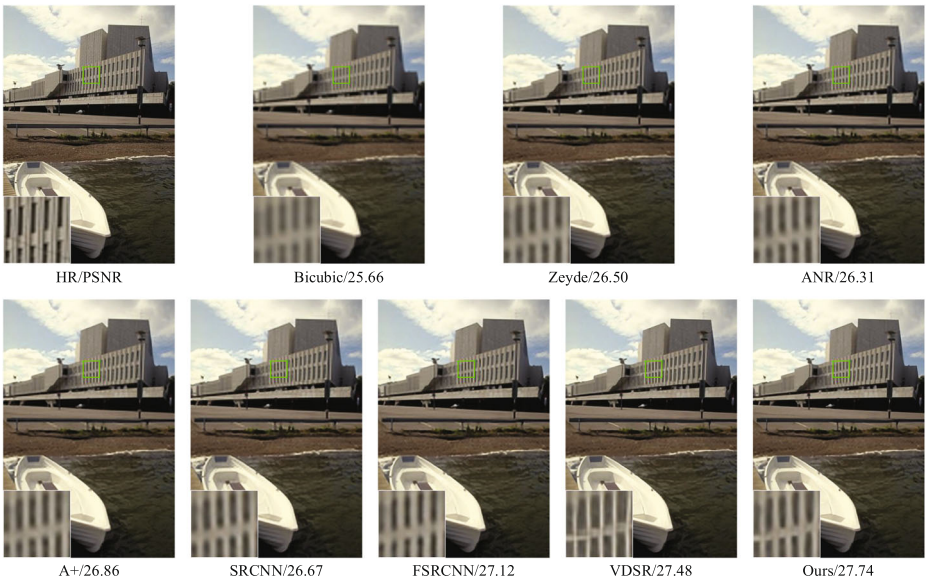


Fig. 5 "78004" image from B100 with scale factor ×3

value denotes a better reconsider effect. The highest is shown in bold. It can be seen that by utilizing our framework, VDSR-K outperforms almost all the contrast methods in all the datasets with all the scale factors. In detail, our method obtains the best result in PSNR for all except 2 items. That is to say, the SR results obtained via our network is the closest to the corresponding HR image. Besides, we obtain the highest results in SSIM except for 1 item. That means our method is more suitable for HVS. This demonstrates the effectiveness of our framework from the perspective of objective indicators.

SR results of the VDSR-K_C2 and other contrast methods are shown in Figs. 5, 6, 7 and 8. It can be seen our method achieve the highest PSNR of these images. First, we show the SR results of "78004" in B100 dataset. From Fig. 5, we can see that our method can generate a more faithful SR image which can recover most sharp lines. From the enlarged image, we can see that the outline of the windows is the closest to the HR image, and no any other extra structure is introduced. However, other contrast methods fail to recover the detail information. What worse, VDSR produces obvious artifacts around the boundary of

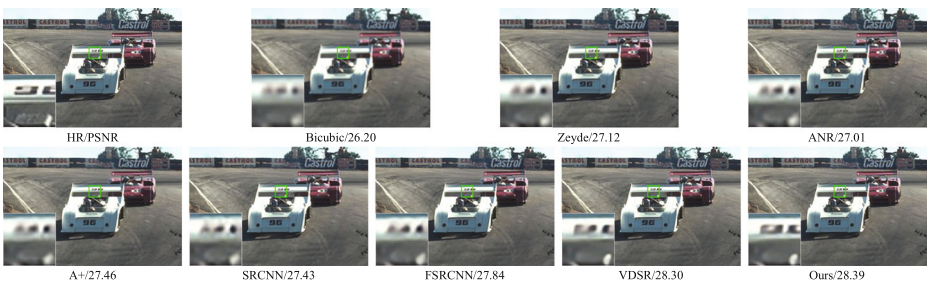


Fig. 6 "21077" image from B100 with scale factor ×3

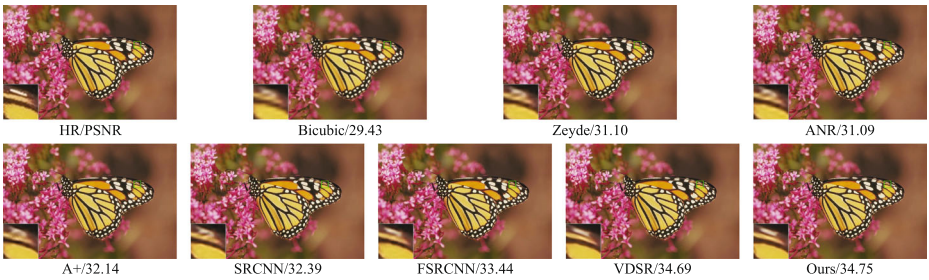


Fig. 7 "Monarch" image from Set14 with scale factor ×3

Fig. 8 "Woman" image from Set5 with scale factor ×3

Table 3 Comparison of running time and the number of parameters

Model	Running time (s)	The number of parameters
VDSR	0.0012	0.6M
VDSR-K_C2	0.0015	0.8M
VDSR-K_C3	0.0019	0.9M

Table 4 Average PSNR/SSIM for scale factor ×2, ×3, ×4 on Set5, Set14, B100 dataset of VDSR-3 with other contrast methods

Datasets	Scale	Bicubic PSNR/SSIM	Zeyde PSNR/SSIM	ANR PSNR/SSIM	A+ PSNR/SSIM	SRCNN PSNR/SSIM	FSRCNN PSNR/SSIM	VDSR PSNR/SSIM	Ours(VDSR-K_C3) PSNR/SSIM
Set5	×2	33.66/0.9382	35.78/0.9563	35.83/0.9567	36.55/0.9611	36.34/0.9590	37.00/0.9623	**37.53**/0.9651	37.52/**0.9654**
	×3	30.39/0.8802	31.90/0.9075	31.92/0.9074	32.59/0.9193	32.39/0.9141	33.16/0.9244	33.66/0.9315	**33.76/0.9328**
	×4	28.41/0.8246	29.69/0.8565	29.69/0.8557	30.28/0.8737	30.09/0.8669	30.70/0.8796	31.35/0.8968	**31.41/0.8975**
Set14	×2	30.24/0.9415	31.81/0.9611	31.80/0.9624	32.28/0.9649	32.27/0.9638	32.72/0.9664	**33.03/0.9681**	32.97/0.9675
	×3	27.55/0.8587	28.67/0.8859	28.65/0.8881	29.13/0.8940	29.10/0.8913	29.52/0.8979	29.77/0.9028	**29.80/0.9032**
	×4	26.00/0.7838	26.88/0.8159	26.85/0.8175	27.32/0.8281	27.30/0.8216	27.69/0.8321	28.01/0.8419	**28.04/0.8430**
B100	×2	29.56/0.8431	30.78/0.8773	30.82/0.8801	31.21/0.8863	31.14/0.8847	31.50/0.8906	31.90/0.8960	**31.91/0.8964**
	×3	27.21/0.7385	27.97/0.7717	27.97/0.7747	28.29/0.7835	28.21/0.7800	28.52/0.7893	28.82/0.7976	**28.85/0.7983**
	×4	25.96/0.6675	26.55/0.6967	26.54/0.6989	26.82/0.7087	26.71/0.7022	26.96/0.7128	**27.29**/0.7251	**27.29/0.7258**

the windows. The SR results of "Monarch" in Set14 is shown in Fig. 7. It can be seen that our method accurately recovers the spots on the butterfly wings. While the SR results of Zeyde, ANR, A+, and SRCNN glue yellow spots and white spots, which are separated in the corresponding HR image. Figure 6 shows the SR images of "21077" in B100 dataset. The SR result of our method is sharp on the line and the numbers printed on the car are clearer. From Fig. 8 which shows the SR results for "woman" in Set5, our method better restored the outline of the hollow on the scarf compared with all the contrast methods. Combining the objective quantitative assessments and the subjective visual effect, the superiority of our method is demonstrated.

We compare the running time and the number of parameters with VDSR, the results are shown in Table 3. It can be seen that the required running time of VDSR-K_C2 and VDSR-K_C3 is roughly the same as that of VDSR. The number of parameters in both VDSR-K_C2 and VDSR-K_C3 are not increased significantly compared with VDSR. However, our framework shows obvious advantages in both subjective visual effects and objective indicators (Table 4).

5 Conclusion

Deep learning based SR methods often neglect the diversity of image patches. To fully exploit the texture variability across different image patches, a universal, flexible, and effective framework is proposed. In the proposed framework, K-means is employed to group images into different clusters. Multiple branches are adopted to reconstruct the SR image. Each branch is weighted in accordance with Euclidean distance to the cluster centers. To train the neural network, a weighted loss function based on l_2 loss is proposed. To demonstrate the effectiveness of our framework, we apply it to the VDSR. Experimental results illustrate that VDSR-K performance the original VDSR and some state-of-the-art SR approaches.

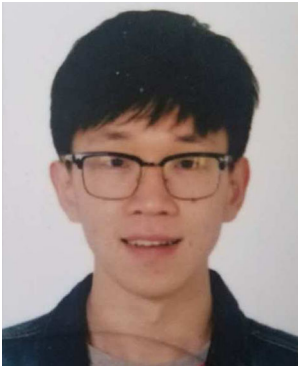
Acknowledgments The research in our paper is sponsored by National Natural Science Foundation of China (No.61711540303, No.61701327), Science Foundation of Sichuan Science and Technology Department(No. 2018GZ0178).

References

1. Bevilacqua M, Roumy A, Guillemot C, Alberi-Morel ML (2012) Low-complexity single-image super-resolution based on nonnegative neighbor embedding
2. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
3. Dong C, Loy CC, He K, Tang X (2014) Learning a deep convolutional network for image super-resolution. In: European conference on computer vision. Springer, pp 184–199
4. Dong C, Loy CC, Tang X (2016) Accelerating the super-resolution convolutional neural network. In: European conference on computer vision. Springer, pp 391–407
5. Duchon CE (1979) Lanczos filtering in one and two dimensions. Japplmeteor 18(8):1016–1022
6. Farina R, Cuomo S, De Michele P, Piccialli F (2013) A smart gpu implementation of an elliptic kernel for an ocean global circulation model. Appl Math Sci 7(61-64):3007–3021
7. Fattal R (2007) Upsampling via imposed edges statistics. ACM Trans Graph (Proceedings of SIGGRAPH 2007), 26(3):to appear

8. Freedman G, Fattal R (2011) Image and video upscaling from local self-examples. Acm Trans Graph 30(2):1–11
9. Freeman WT, Jones TR, Pasztor EC (2002) Example-based super-resolution. IEEE Comput Graph Appl 22(2):56–65
10. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Advances in neural information processing systems, pp 2672–2680
11. Haris M, Shakhnarovich G, Ukita N (2018) Deep backprojection networks for super-resolution. In: Conference on computer vision and pattern recognition
12. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
13. Hou H, Andrews H (1978) Cubic splines for image interpolation and digital filtering. IEEE Trans Acoust Speech Signal Process 26(6):508–517
14. Huang JB, Singh A, Ahuja N (2015) Single image super-resolution from transformed self-exemplars. In: Computer vision and pattern recognition, pp 5197–5206
15. Irani M, Peleg S (1991) Improving resolution by image registration. CVGIP: Graph models image process 53(3):231–239
16. Jeon G, Anisetti M, Lee J, Bellandi V, Damiani E, Jeong J (2009) Concept of linguistic variable-based fuzzy ensemble approach: application to interlaced hdtv sequences. IEEE Trans Fuzzy Syst 17(6):1245–1258
17. Jeon G, Anisetti M, Wang L, Damiani E (2016) Locally estimated heterogeneity property and its fuzzy filter application for deinterlacing. Inform Sci 354:112–130
18. Johnson J, Alahi A, Fei-Fei L (2016) Perceptual losses for real-time style transfer and super-resolution. In: European conference on computer vision. Springer, pp 694–711
19. Kim J, Kwon Lee J, Mu Lee K (2016) Accurate image super-resolution using very deep convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1646–1654
20. Kim J, Kwon Lee J, Mu Lee K (2016) Deeply-recursive convolutional network for image super-resolution. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1637–1645
21. Ledig C, Theis L, Huszár F, Caballero J, Cunningham A, Acosta A, Aitken A, Tejani A, Totz J, Wang Z et al (2017) Photo-realistic single image super-resolution using a generative adversarial network. arXiv preprint
22. Li M, Nguyen TQ (2008) Markov random field model-based edge-directed image interpolation. IEEE Trans Image Process 17(7):1121–1128
23. Li X, Orchard MT (2001) New edge-directed interpolation. IEEE Trans Image Process 10(10):1521–1527
24. Lim B, Son S, Kim H, Nah S, Lee KM (2017) Enhanced deep residual networks for single image super-resolution. In: The IEEE conference on computer vision and pattern recognition (CVPR) workshops, vol 1, p 4
25. Liu D, Wang Z, Wen B, Yang J, Han W, Huang TS (2016) Robust single image super-resolution via deep networks with sparse prior. IEEE Trans Image Process 25(7):3194–3207
26. Martin D, Fowlkes C, Tal D, Malik J (2001) A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Eighth IEEE International conference on computer vision, 2001. ICCV 2001. Proceedings, vol 2. IEEE, pp 416–423
27. Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, Lin Z, Desmaison A, Antiga L, Lerer A (2017) Automatic differentiation in pytorch
28. Piccialli F, Cuomo S, De Michele P (2013) A regularized mri image reconstruction based on hessian penalty term on cpu/gpu systems. Procedia Comput Sci 18:2643–2646
29. Salvador J, Perez-Pellitero E (2015) Naive Bayes super-resolution forest. In: Proceedings of the IEEE International conference on computer vision, pp 325–333
30. Schulter S, Leistner C, Bischof H (2015) Fast and accurate image upscaling with super-resolution forests. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3791–3799
31. Shi W, Caballero J, Huszár F, Totz J, Aitken AP, Bishop R, Rueckert D, Wang Z (2016) 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, pp 1874–1883
32. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. Computer Science
33. Sun J, Sun J, Xu Z, Shum HY (2011) Gradient profile prior and its applications in image super-resolution and enhancement. IEEE Trans Image Process 20(6):1529–1542

34. Timofte R, De Smet V, Van Gool L (2013) Anchored neighborhood regression for fast example-based super-resolution. In: Proceedings of the IEEE international conference on computer vision, pp 1920–1927
35. Timofte R, De Smet V, Van Gool L (2014) A+: Adjusted anchored neighborhood regression for fast super-resolution. In: Asian Conference on computer vision. Springer, pp 111–126
36. Timofte R, Agustsson E, Van Gool L, Yang MH, Zhang L, Lim B, Son S, Kim H, Nah S, Lee KM et al (2017) Ntire 2017 challenge on single image super-resolution: methods and results. In: 2017 IEEE Conference on computer vision and pattern recognition workshops (CVPRW). IEEE, pp 1110–1121
37. Wang YQ (2014) A multilayer neural network for image demosaicking. In: 2014 IEEE International conference on image processing (ICIP). IEEE, pp 1852–1856
38. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612
39. Wang J, Wu J, Wu Z, Anisetti M, Jeon G (2018) Bayesian method application for color demosaicking. Opt Eng 57(5):053102
40. Wang X, Yu K, Dong C, Loy CC (2018) Recovering realistic texture in image super-resolution by deep spatial feature transform. arXiv:180402815
41. Wu W, Zheng C (2013) Single image super-resolution using self-similarity and generalized nonlocal mean. In: TENCON 2013-2013 IEEE Region 10 conference (31194). IEEE, pp 1–4
42. Wu W, Yang X, Liu K, Liu Y, Yan B et al (2016) A new framework for remote sensing image super-resolution: sparse representation-based method by processing dictionaries with multi-type features. J Syst Archit 64:63–75
43. Wu J, Anisetti M, Wu W, Damiani E, Jeon G (2016) Bayer demosaicking with polynomial interpolation. IEEE Trans Image Process 25(11):5369–5382
44. Yang CY, Huang JB, Yang MH (2010) Exploiting self-similarities for single frame super-resolution. In: Asian conference on computer vision, pp 497–510
45. Yang J, Wang Z, Lin Z, Cohen S, Huang T (2012) Coupled dictionary training for image super-resolution. IEEE Trans Image Process 21(8):3467–3478
46. Yang X, Wu W, Liu K, Chen W, Zhang P, Zhou Z (2017) Multi-sensor image super-resolution with fuzzy cluster by using multi-scale and multi-view sparse coding for infrared image. Multimed Tools Appl 76(23):24871–24902
47. Zeyde R, Elad M, Protter M (2010) On single image scale-up using sparse-representations. In: International conference on curves and surfaces. Springer, pp 711–730
48. Zhang L, Wu X (2006) An edge-guided image interpolation algorithm via directional filtering and data fusion. IEEE Trans Image Process 15(8):2226–2238



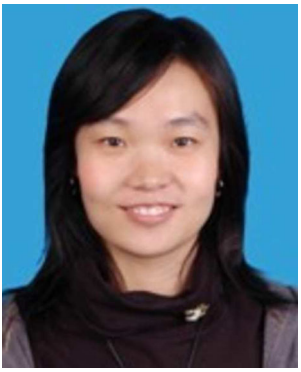
Zhen Li is currently pursuing the M.S. degree in college of electronics and information engineering from Sichuan University, Chengdu, China. His research interests are image restoration and deep learning.

Qilei Li is currently pursuing the M.S. degree in college of electronics and information engineering from Sichuan University, Chengdu, China. He received his BS degrees in electronic information engineering from Shandong University of Technology. His research interests are image processing and deep learning.

Wei Wu is currently a professor in College of Electronics and Information Engineering, Sichuan University. He received his BS degree from Tianjin University, and received his MS and PhD degrees in communication and information system from Sichuan University. He worked in a National Research Council Canada as a post doctorate for one year. His research interests are image processing and pattern recognition.

Zongjun Wu is currently pursuing the M.S. degree in college of electronics and information engineering from Sichuan University, Chengdu, China. He received his BS degrees in measurement and control technology and instruments from Taiyuan University of Technology. His research interests are image processing and deep learning.

Lu Lu was born in Chengdu, China, in 1990. He received the Ph.D. degree in the field of signal and information processing at the School of Electrical Engineering, Southwest Jiaotong University, Chengdu, China, in 2018. From 2017 to 2018, he was a visiting Ph.D. student with the Electrical and Computer Engineering at McGill University, Montreal, QC, Canada. He is currently a Postdoctoral Fellow with the College of Electronics and Information Engineering, Sichuan University, Chengdu, China. His research interests include adaptive filtering, kernel methods and distributed estimation.



Xiaomin Yang is currently an associate professor in College of Electronics and Information Engineering, Sichuan University. She received her BS degree from Sichuan University, and received her PhD degree in communication and information system from Sichuan University. She worked in University of Adelaide as a post doctorate for one year. Her research interests are image processing and pattern recognition.