Embedded Block Residual Network: A Recursive Restoration Model for Single-Image Super-Resolution

Yajun Qiu ∗
Yunnan University
qyjyun@gmail.com

Ruxin Wang ∗
Union Vision Innovation
rosinwang@gmail.com

Dapeng Tao †
Yunnan University
dapeng.tao@gmail.com

Jun Cheng
SIAT, Chinese Academy of Sciences
jun.cheng@siat.ac.cn

Abstract

Single-image super-resolution restores the lost structures and textures from low-resolved images, which has achieved extensive attention from the research community. The top performers in this field include deep or wide convolutional neural networks, or recurrent neural networks. However, the methods enforce a single model to process all kinds of textures and structures. A typical operation is that a certain layer restores the textures based on the ones recovered by the preceding layers, ignoring the characteristics of image textures. In this paper, we believe that the lower-frequency and higher-frequency information in images have different levels of complexity and should be restored by models of different representational capacity. Inspired by this, we propose a novel embedded block residual network (EBRN) which is an incremental recovering progress for texture super-resolution. Specifically, different modules in the model restores information of different frequencies. For lower-frequency information, we use shallower modules of the network to recover; for higher-frequency information, we use deeper modules to restore. Extensive experiments indicate that the proposed EBRN model achieves superior performance and visual improvements against the state-of-the-arts.

1. Introduction

Single-image super-resolution (SISR) has attracted extensive attention in both academia and industry. This technique aims at recovering a high-resolved (HR) image from a single low-resolved (LR) one, which offers an opportunity of overcoming resolution limitations in various computer vision applications, such as security, medical imaging [37], and object recognition [3]. The problem of SISR is ill-posed since there exist multiple HR solutions for any LR input. To overcome this issue, most methods such as those based on deep convolutional neural networks constrain the solution space by learning a mapping function from external low- and high-resolution exemplar pairs or by involving a priori knowledge on the HR feature space.

Learning-based methods are placed in the top performers in literatures, especially deep or wide convolutional neural networks because of their high representational capacity. With extensive parameters and a good learning process, the models have the ability of fitting on a large number of training data and that of exploiting the underlying structures of natural images. Most methods advocate to design an end-to-end learning process that facilitates both training and inference. The performance improvement in such a design comes from an increase of the parameter number and the elaboration of neural connection. However, the resultant complex models usually raise high consumption of computation and memory, which hinders their real-world applications.

The reason of the above issue is that the deep model-based methods fail to consider the frequency characteristics of images which is, however, widely used in conventional image processing techniques. The characteristics state

Note:

∗These authors contributed equally to the work.
†D. Tao is the corresponding author.
that natural images consist of different frequency bands of information, with each band containing structures and textures of different complexity. Hence, different bands of information are extracted by using different base functions, as in wavelet analysis [6]. In image restoration tasks including SISR, the recovery of each band information requires a specific restoring function. Considering that the feature distribution varies across different frequency bands, lower-frequency information is composed of simpler structures and textures where simpler functions are needed for restoration; higher-frequency information consists of complex structures and textures where more complex restoring functions are expected.

At this point, the existing deep model-based methods do not distinguish the image frequency. The task of each layer in those models is to recover all information based on the features of the preceding layer. For shallow layers, the parameters may fit on the low-frequency information (which has simple textures), but underfit on the high-frequency information (which has complex textures). For deep layers, the parameters may fit on the high-frequency information, but overfit on the low-frequency information. The inconsistency between the model complexity and the frequency is a key issue that limits the performance of those deep CNN-based methods. While residual connection provides a way to split the information as those recovered and those not-recovered, the residual architectures have no correlation with the frequency-splitting principle. Instead, they advocate that the residual connection transfers the information of shallow layers to deep layers in a dense and direct way. To bridge the connection between the model architecture and the frequency bands, elaboration of the residual idea is required. We illustrate an example in Figure 1, from which we see that the textures on the book cannot be well restored by EDSR [29]. That method takes advantage of residual connection which, however, fails to recover the simple textures by using a very deep architecture. Instead, complex curves appear in the result. Our result exhibits better visual properties. This comparison validates the drawback of EDSR [29] that deep layers are easily over-fitted on the low-frequency information of the image.

Based on the above analyses, in this paper, we propose an embedded block residual network (EBRN) for single image super-resolution that restores the textures of different frequency by using sub-networks of different complexity. Specifically, the block residual module (BRM) is the basic module in our model, which splits the data flow as a super-resolution flow and a back-projection flow. The former flow restores most structures and textures of lower frequency, while the latter flow calculates the information of higher frequency which is remained to be recovered by deeper layers. The whole model is an embedding of multiple BRMs. Each BRM is stacked on the back-projection flow of its antecedent BRM. In this way, a BRM is responsible for the recovery of information at lower frequency, passing the information of higher frequency to deeper BRMs. To fuse the outputs of all BRMs, we also propose a recurrent fusion technique that stabilizes the feature flow and the gradient flow in training and encourages a faster convergence rate of training. Extensive experiments on multiple SISR datasets illustrate the state-of-the-art performance of the proposed method and validate the correlation of the model complexity and the image frequency as discussed above. In summary, the main contributions of this work are as follows:

1. We propose a motivation that the information of different frequency in images should be restored by the models of different complexity. In a bad case, the information of lower frequency could be over-recovered by a deeper model while the information of higher frequency would be under-recovered by a shallower model.

2. We propose a block residual module (BRM) that tries to restore the image structures and textures while passing the hard-to-recovered information to deeper modules. This allows each BRM to focus on the information of proper frequency, which is important for ensuring the correlation of model complexity and image frequency.

3. We propose a novel technique for embedding multiple BRMs, which can effectively improve the final reconstruction quality based on the outputs of each module. We also empirically demonstrate that the proposed model is superior over the state-of-the-arts.

2. Related work

SISR is an active research field and has a long history. Existing literatures could be grouped into three categories: the interpolation-based methods [20, 9], the reconstruction-based methods [5, 38], and the learning-based methods. While the conventional methods have a long list, here we review the top performers, especially the deep learning-based methods, due to the limit of page length.

Dong et al. [7] introduced CNN [26] into the SR task and proposed the SRCNN model that was composed of a three-layer network to learn the mapping from LR images to HR images. This model achieved much better performance compared with the traditional algorithms. Kim et al. [21] proposed the VDSR model that used a very deep network with 20 layers which produced improved performance compared with SRCNN. A main contribution of this method is to employ residual learning which encourages a fast convergence rate in the training process. Lai et al. [25] proposed the lapSRN method that took the original LR images as input and progressively reconstructed the sub-band residuals.
of HR images. Kim et al. [22] proposed the DRCN method which was the first to involve recursive learning into SISR. To reuse the features of each layer in CNN, Tong et al. [42] developed the DenseNet [14] by increasing dense connections among the convolutional layers. Lim et al. [29] designed the EDSR model by removing unnecessary modules in conventional residual networks, which achieved the champion of the NTIRE2017 SR Challenge [40]. Sajjadi et al. [35] compared the performance of different combinations of loss functions in EnhanceNet, and empirically draw the conclusion that the combination of perceptual loss, texture matching loss, and anti-loss worked best. Li et al. [39] proposed MenNet to address the issue that a deep network lacks long-term memory. This method introduced a memory block which consisted of a recursive unit and a gate unit, to explicitly mine persistent memory through an adaptive learning process. In order to reduce model parameters and model practicability, Ahn et al. [1] proposed a cascading residual network (CARN) that achieved good performance by using a cascading mechanism with few parameters. Haris et al. [11] developed a novel architecture which was named as DBPN. This model exploited iterative up- and down-sampling layers, providing an error feedback mechanism for project errors at each stage. DBPN also improved super-resolution performance, yielding superior results and in particular establishing new state-of-the-art performance for large scale factors such as \(\times 8\) on multiple datasets. The enhanced version D-DBPN achieved the best performance in \(\times 8\) enlargement in NTIRE2018 [41] and won the championship of NTIRE2018 SR Challenge. Zhang et al. [48] proposed the RDN model which differed from other CNN models, i.e., this model did not make full use of hierarchical features of LR images. Hui et al. [16] proposed the information distillation network (IDN) with lightweight architecture and low computational complexity. Zhang et al. [47] stated that previous SR models treated each channel equally, hindering the representational capacity of CNN. They proposed RCAN to solve the problem by introducing channel attention mechanism. Li et al. [28] proposed the MSRN model to explore the multi-scale information of LR images.

The above deep learning-based methods were proposed to improve the PSNR/SSIM indexes of the restored images. However, existing studies indicate that the solution of the L2 objective function is an averaged version of multiple real HR solutions. Regarding this, the perceptual loss [19] was investigated to recover visually pleasing results for textures. For example, Ledig et al. [27] proposed SRGAN which inferred photo-realistic natural images. The results did not yield a high PSNR value, but produced realistic visual effects by using a perceptual loss consisting of an adversarial loss and a content loss. Wang et al. [43] proposed the SFTGAN model which involved a spatial feature modulation layer that integrated a priori of semantic categories into the network, generating more realistic and visually pleasing textures. Park et al. [32] proposed the SRFeat model to alleviate the issue that GAN-based approaches tend to include less meaningful high-frequency noise which is irrelevant to the input image. This model involved an additional discriminator on the feature domain. Wang et al. [44] developed ESRGAN to remove the artificial artifacts in the results of SRGAN by compensating an improvement sub-network.

The above literature review reveals that a significant improvement on SISR has been achieved by deep learning-based methods, especially CNNs and GANs. While the performance in the cases of scale factors \(\times 2\), \(\times 3\), and \(\times 4\) may reach a bottleneck, the restoration of \(\times 8\) becomes a main interest in recent publications. With the increase of the scale factor, we find that existing models have no concern about image frequency and model complexity, resulting in over-restoration of simple textures by using complex models, and under-restoration of complex textures by using simple models. Therefore, to alleviate this issue, this work starts from a different viewpoint, developing a proper architecture that associates the information of a frequency range with the model of appropriate complexity.

3. Proposed Method

In this section, we introduce the details of the proposed EBRN model and analyze how the information of different frequency is processed by the network. The architecture is illustrated in Figure 2, where the basic module is BRM which is presented in the following.

3.1. Block Residual Module

The block residual module (BRM) aims at restoring parts of the HR information while passing the remained signals to deeper modules for restoration. At this regard, the module contains two data flow: the super-resolution flow and the back-projection flow.

The super-resolution flow is a basic deconvolution network which takes the LR feature maps \(I_x\) as input and processes by using a stack of a deconvolutional layer (also
known as transposed convolution) and three convolutional layers. The output of this flow is the super-resolved feature maps $O_x$, where $x$ is the index of BRM in the model. An alternative choice of the deconvolutional layer is the sub-pixel convolutional layer [36] which could improve the performance but yielding more parameters. Considering the tradeoff between performance and model efficiency, the deconvolutional layer is selected for up-scaling.

To compute the information that the super-resolution flow has not recovered, the back-projection flow employs an operation which first down-samples the deconvolved feature maps to the LR spatial size and then compute a minus between the down-sampled feature maps and the input LR feature maps of this module. The computed residual conveys the information that the super-resolution flow fails to recover. This residual is then processed by a local residual learning stage, outputting a set of encoded features $I_{x+1}$ which forms the input of the next BRM.

The design of BRM is illustrated in Figure 3. All the convolutional layers utilize $3 \times 3 \times 64$ convolutional kernels. The layers except that for down-sampling are set with the stride of $1 \times 1$ and the padding size of $1 \times 1$. The parameters of the down-sampling layer are set according to the up-scaling factor, i.e., the output feature maps have the same spatial size as the input feature maps. The local residual learning stage is to encourage a fast convergence rate of training, as in other residual learning methods. With such a design, we empirically find that the super-resolution flow could restore information of lower frequency, and the information of higher frequency which is difficult to be recovered is passed to later modules.

### 3.2. Embedded Block Residual Network

The embedded block residual network (EBRN) is composed of multiple BRMs, as shown in Figure 2. Before the first BRM, an initial feature extraction module is presented to formulate the shape of the feature maps. In this module, the first convolutional layer produces 256-channel feature maps, followed by which two convolutional layers are stacked with each outputting 64-channel feature maps. The convolutional kernel size in these layers is $3 \times 3$.

The BRMs are composed in an embedding way, instead of a simple stacking way. That is, the first BRM is stacked on the output of the initial feature extraction module, the second BRM is concatenated to the output of the back-projection flow of the first BRM, and so on. Each BRM is responsible for restoring the residual feature maps produced by the back-projection flow of its antecedent BRM. Note that the last BRM only contains the super-resolution flow where the back-projection flow is dropped. In this way, the information of lower frequency is only passed through the shallower BRMs which have low model complexity. The issue of overfitting on this part of information can be avoided. On the other hand, the information of higher frequency is flowed to deeper BRMs which have higher model complexity, where the underfitting problem can be alleviated. Therefore, a deeper BRM always tries to restore what has not been restored by shallower BRMs. This is consistent with our motivation. Another important point is that we associate the information of certain frequencies with a sub-network of proper complexity. It is not required to fit a simple model on complex textures and also not to fit a complex model on simple structures. Hence, the number of the parameters in those sub-networks could be significantly re-
Deconv                     Conv +

We follow the suggestion of et al. [49] which says that a SR model trained with the L2 loss function does not guarantee better PSNR/SSIM performance than with other loss functions. Here, we first select the L1 loss as the training objective of the proposed model, which is shown to speed up the convergence of training compared with the L2 loss. As a second step, we employ the L2 loss to finetune the model, which could result in higher PSNR performance. More details about training can be found in Section 5.2.

4. Discussions

In this section, we mainly discuss the difference between the proposed model and its related methods.

4.1. EBRN vs. Residual Network

Residual networks [13] have recently exhibited excellent performance in various computer vision tasks. In SISR, the first model using the residual learning idea is VDSR [21], which achieved superior performance compared with its competitors. The advantage of residual networks relative to traditional CNN models is that residual learning promotes the transmission of features in the network, and alleviates the gradient vanishing problem, making the network easier to train.

In this work, we exploit the residual learning idea, which is different from the conventional residual networks. For example, as shown in Figure 3, the proposed model does not use the batch normalization (BN) [17] layer since the BN layer limits the range flexibility of the intermediate features during feature normalization [29]. Another important difference comes from how the residual is computed and what the residual conveys. In the residual networks, the residual signal is the difference between the input and the output. In the proposed model, one type of residual signal is the information of a certain frequency range; another type of residual signal is the difference between the original LR features and the back-projected LR features. In each BRM, the second residual signal is important for SR since it explicitly conveys which information is to be recovered by the following BRM.

4.2. EBRN vs. Deep Back-Projection Network

A similar method to the current work is the deep back-projection network (DBPN) proposed by Haris et al. [11]. This method exploits iteratively up- and down-sampling layers, providing an error feedback mechanism for projecting errors at each stage. The errors can effectively improve the restoration by deep layers in the model.

The difference between the two methods comes from two aspects: 1) in each up- and down-projection unit, DBPN directly maps the LR residual to the HR space, whereas the LR residual in our model contains higher frequency information which is fed into deeper sub-networks for restora-
5. Experiments

In this section, we present the experimental details and analyses to validate the effectiveness of the propose model.

5.1. Datasets

Following [29], we use the DIV2K [40] dataset for training, which is a high-quality (2K resolution) image restoration dataset containing 800 training images, 100 validation images, and 100 test images. The up-scaling factors including \(\times 2\), \(\times 4\), and \(\times 8\) are used for training and model evaluation. 5 standard benchmark datasets are employed during testing, among which Set5 [4], Set14 [46], BSD-S100 [2] consist of natural scenes, Urban100 [15] contains urban scenes with large amounts of regular texture patterns, and Manga109 [31] is a dataset of Japanese manga.

5.2. Implementation Details

To prepare the training data, we synthesize the LR images by down-sampling the training HR images using bicubic interpolation. Three training datasets are collected with each corresponding to one of the up-scaling factors, i.e., \(\times 2\), \(\times 4\), and \(\times 8\). Data augmentation techniques are utilized including horizontal, vertical flipping, and \(90^\circ\) rotation. Regarding the training details, the proposed model takes the RGB-channel images as input and output. The LR images are randomly cropped as \(64 \times 64\) patch images which are then fed into the model with the batch size of 32. The sizes of the ground-truth HR patch images are determined by the up-scaling factor. The model weights are initialized using the model proposed in [12] and the biases are initialized as zero. The parametric rectified linear units (PReLU) [12] is used as the activation function. To ensure numeric stability during training, we scale the pixel range of LR and HR images to \([0, 1]\). The Adam [23] optimization algorithm is employed with \(\beta_1 = 0.9\), \(\beta_2 = 0.999\), and \(\epsilon = 10^{-8}\). The learning rate is initially set to \(10^{-4}\) and decreased by a factor of 10 at every 100 epochs. All experiments are implemented using the Pytorch [33] framework and evaluated on the NVIDIA TITAN X GPU devices.

5.3. Model Analyses

In this section, we conduct a series of experiments to validate the proposed motivation and investigate the effects of parameters on model performance.

Recall the motivation that the information of different frequency range should be processed by models of different complexity. To validate this, we illustrate the energy distributions of different-BRM outputs across different frequency bands in Figure 6. The energy distribution across different frequency bands is computed based on the wavelet coefficients of different levels. The result indicates that the outputs of shallower BRMs contain more lower-frequency information while the outputs of deeper BRMs tends to recover more higher-frequency information.

We also investigate the proposed model via ablation studies, including the correlation between the model performance and the feature fusion technique, and the correlation between the performance and the number of BRMs. Table 1 reveals the superiority of the proposed recursive feature fusion technique, compared with a simple summation op-
Table 4. The average performance of the state-of-the-art methods. **Red** font indicates the best performer and **blue** font indicates the second best performer.

<table>
<thead>
<tr>
<th>Method</th>
<th>Scale</th>
<th>PSNR</th>
<th>SSIM</th>
<th>Set5</th>
<th>Set14</th>
<th>BSD100</th>
<th>Urban100</th>
<th>Manga109</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicubic</td>
<td>2</td>
<td>33.66</td>
<td>0.9299</td>
<td>30.24</td>
<td>0.8688</td>
<td>29.56</td>
<td>0.8431</td>
<td>26.88</td>
</tr>
<tr>
<td>SRCNN [7]</td>
<td>2</td>
<td>36.66</td>
<td>0.9542</td>
<td>32.45</td>
<td>0.9067</td>
<td>31.36</td>
<td>0.8879</td>
<td>29.50</td>
</tr>
<tr>
<td>VDSR [21]</td>
<td>2</td>
<td>37.53</td>
<td>0.9590</td>
<td>33.05</td>
<td>0.9130</td>
<td>33.05</td>
<td>0.8960</td>
<td>30.77</td>
</tr>
<tr>
<td>N²Net [34]</td>
<td>2</td>
<td>37.57</td>
<td>-</td>
<td>-</td>
<td>31.98</td>
<td>-</td>
<td>30.80</td>
<td>-</td>
</tr>
<tr>
<td>DRCN [22]</td>
<td>2</td>
<td>37.63</td>
<td>0.9588</td>
<td>33.04</td>
<td>0.9118</td>
<td>31.85</td>
<td>0.8942</td>
<td>30.75</td>
</tr>
<tr>
<td>LapSRN [25]</td>
<td>2</td>
<td>37.52</td>
<td>0.9591</td>
<td>33.08</td>
<td>0.9130</td>
<td>31.08</td>
<td>0.8950</td>
<td>30.41</td>
</tr>
<tr>
<td>MemNet [39]</td>
<td>2</td>
<td>37.78</td>
<td>0.9597</td>
<td>33.28</td>
<td>0.9142</td>
<td>32.08</td>
<td>0.8978</td>
<td>31.31</td>
</tr>
<tr>
<td>EDSR [29]</td>
<td>2</td>
<td>38.11</td>
<td>0.9602</td>
<td>33.92</td>
<td>0.9195</td>
<td>32.32</td>
<td>0.9013</td>
<td>32.93</td>
</tr>
<tr>
<td>D-DBPN [11]</td>
<td>2</td>
<td>38.09</td>
<td>0.9600</td>
<td>33.85</td>
<td>0.9190</td>
<td>32.27</td>
<td>0.9000</td>
<td>32.55</td>
</tr>
<tr>
<td>IDN [16]</td>
<td>2</td>
<td>38.73</td>
<td>0.9603</td>
<td>33.30</td>
<td>0.9148</td>
<td>32.08</td>
<td>0.8985</td>
<td>31.27</td>
</tr>
<tr>
<td>NLRN [30]</td>
<td>2</td>
<td>38.00</td>
<td>0.9603</td>
<td>33.46</td>
<td>0.9159</td>
<td>32.19</td>
<td>0.8992</td>
<td>31.81</td>
</tr>
<tr>
<td>MSRN [28]</td>
<td>2</td>
<td>38.08</td>
<td>0.9605</td>
<td>33.74</td>
<td>0.9170</td>
<td>32.23</td>
<td>0.9015</td>
<td>32.22</td>
</tr>
<tr>
<td>CARN [1]</td>
<td>2</td>
<td>37.76</td>
<td>0.9590</td>
<td>33.52</td>
<td>0.9166</td>
<td>32.09</td>
<td>0.8978</td>
<td>31.92</td>
</tr>
<tr>
<td>RDN [48]</td>
<td>2</td>
<td>38.24</td>
<td>0.9614</td>
<td>34.01</td>
<td>0.9212</td>
<td>32.34</td>
<td>0.9017</td>
<td>32.89</td>
</tr>
<tr>
<td>EBRN(ours)</td>
<td>2</td>
<td>38.37</td>
<td>0.9620</td>
<td>34.24</td>
<td>0.9226</td>
<td>32.47</td>
<td>0.9032</td>
<td>33.52</td>
</tr>
<tr>
<td><strong>Bicubic</strong></td>
<td>4</td>
<td>28.42</td>
<td>0.8104</td>
<td>26.00</td>
<td>0.7027</td>
<td>25.96</td>
<td>0.6675</td>
<td>23.14</td>
</tr>
<tr>
<td><strong>SRCNN [7]</strong></td>
<td>4</td>
<td>30.48</td>
<td>0.8628</td>
<td>27.50</td>
<td>0.7513</td>
<td>26.90</td>
<td>0.7101</td>
<td>24.52</td>
</tr>
<tr>
<td><strong>VDSR [21]</strong></td>
<td>4</td>
<td>31.35</td>
<td>0.8830</td>
<td>28.02</td>
<td>0.7680</td>
<td>27.29</td>
<td>0.0726</td>
<td>25.18</td>
</tr>
<tr>
<td><strong>N²Net [34]</strong></td>
<td>4</td>
<td>31.50</td>
<td>-</td>
<td>-</td>
<td>27.34</td>
<td>-</td>
<td>25.23</td>
<td>-</td>
</tr>
<tr>
<td><strong>DRCN [22]</strong></td>
<td>4</td>
<td>31.53</td>
<td>0.8854</td>
<td>28.02</td>
<td>0.7670</td>
<td>27.23</td>
<td>0.7233</td>
<td>25.14</td>
</tr>
<tr>
<td><strong>LapSRN [25]</strong></td>
<td>4</td>
<td>31.54</td>
<td>0.8850</td>
<td>28.19</td>
<td>0.7720</td>
<td>27.32</td>
<td>0.7270</td>
<td>25.21</td>
</tr>
<tr>
<td><strong>MemNet [39]</strong></td>
<td>4</td>
<td>31.74</td>
<td>0.8893</td>
<td>28.26</td>
<td>0.7723</td>
<td>27.40</td>
<td>0.7281</td>
<td>25.50</td>
</tr>
<tr>
<td><strong>EDSR [29]</strong></td>
<td>4</td>
<td>32.46</td>
<td>0.8968</td>
<td>28.80</td>
<td>0.7876</td>
<td>27.71</td>
<td>0.7426</td>
<td>26.64</td>
</tr>
<tr>
<td><strong>D-DBPN [11]</strong></td>
<td>4</td>
<td>32.47</td>
<td>0.8980</td>
<td>28.82</td>
<td>0.7860</td>
<td>27.72</td>
<td>0.7400</td>
<td>26.38</td>
</tr>
<tr>
<td><strong>IDN [16]</strong></td>
<td>4</td>
<td>31.82</td>
<td>0.8903</td>
<td>28.25</td>
<td>0.7730</td>
<td>27.41</td>
<td>0.7297</td>
<td>25.41</td>
</tr>
<tr>
<td><strong>NLRN [30]</strong></td>
<td>4</td>
<td>31.92</td>
<td>0.8916</td>
<td>28.36</td>
<td>0.7745</td>
<td>27.48</td>
<td>0.7306</td>
<td>25.79</td>
</tr>
<tr>
<td><strong>MSRN [28]</strong></td>
<td>4</td>
<td>32.07</td>
<td>0.8903</td>
<td>28.60</td>
<td>0.7751</td>
<td>27.52</td>
<td>0.7273</td>
<td>26.04</td>
</tr>
<tr>
<td><strong>CARN [1]</strong></td>
<td>4</td>
<td>32.13</td>
<td>0.8937</td>
<td>28.60</td>
<td>0.7806</td>
<td>27.58</td>
<td>0.7349</td>
<td>26.07</td>
</tr>
<tr>
<td><strong>RDN [48]</strong></td>
<td>4</td>
<td>32.47</td>
<td>0.8990</td>
<td>28.81</td>
<td>0.7871</td>
<td>27.72</td>
<td>0.7419</td>
<td>26.61</td>
</tr>
<tr>
<td><strong>PFF [24]</strong></td>
<td>4</td>
<td>32.74</td>
<td>0.9021</td>
<td>28.98</td>
<td>0.7904</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>RCAN [47]</strong></td>
<td>4</td>
<td>32.63</td>
<td>0.9002</td>
<td>28.87</td>
<td>0.7889</td>
<td>27.77</td>
<td>0.7436</td>
<td>26.82</td>
</tr>
<tr>
<td><strong>EBRN(ours)</strong></td>
<td>4</td>
<td>32.79</td>
<td>0.9032</td>
<td>29.01</td>
<td>0.7903</td>
<td>27.85</td>
<td>0.7464</td>
<td>27.03</td>
</tr>
</tbody>
</table>

Table 4. The average performance of the state-of-the-art methods. **Red** font indicates the best performer and **blue** font indicates the second best performer.

The number of model parameters is an important factor for SISR in real applications. As discussed in previous sections, the proposed embedding strategy could significantly reduce the number of parameters, which is validated here by comparing with the state-of-the-arts. As shown in Figure 9, the EBRN model with 10 BRMs exhibits better performance and fewer parameters than MDSR [29], D-DBPN [11], RCAN [47], and EDSR [29] which are the recently published SR methods. EBRN is also superior over the conventional small models including SRDenseNet [42], DRCN [22], LapSRN [25], VDSR [21], FSRCNN [8], and SRCNN [7]. These results indicate that the proposed EBRN performs well with limited amount of parameters, owing to its elaborate architecture.

5.4. Comparison with State-of-the-arts

In this section, we compare the proposed model with the state-of-the-arts including SRCNN [7], VDSR [21], DRCN [22], LapSRN [25], EDSR [29], RDN [48], IDN [16], M-
Figure 8. Visualization of the selected parts restored by different methods. The up-scaling factor is 4.

Figure 9. PSNR vs. the number of parameters. The comparison is conducted on Set5 with the ×4 up-scaling factor.

SRN [28], D-DBPN [11], and RCAN [47]. The peak signal-to-noise ratio (PSNR) [18] and the structural similarity index (SSIM) [45] are employed as the evaluation metrics. Following a common setting and for fair comparison, we use the luminance channel (Y) of the transformed YCbCr space for quality measurement. While the proposed model takes RGB images as input, the Y-channel output is extracted after color conversion. The LR images are synthesized by using bicubic interpolation. Table 4 presents the ×2, ×4, and ×8 performances of different methods, from which we see that the proposed method achieves the best PSNR and SSIM scores in all cases. Regarding the inference time, we use the published codes of the competitors which are evaluated on a server with 4.2GHz Intel i7 CPU, 32GB RAM, and a Nvidia TITANX GPU card. In Table 2, we show the comparison of running time of several efficient methods, indicating that the proposed model fullfills the requirements of real-time applications. We select four examples for visualization, as shown in Figure 7. The details of the examples are zoomed in and visualized in Figure 8, from which it is observed that the proposed method can synthesize more pleasing textures and structures. The competitors produce flawed textures which may be caused by underfitting of the model on complex textures or overfitting of the model on simple areas.

6. Conclusions

In this paper, we are motivated by that information of different frequency should be restored by models of different complexity, and propose an embedded block residual network for single image super-resolution. We advocate that the limitation of existing methods is caused by underfitting of the models on complex textures and overfitting on simple structures. As such, we develop a block residual module that could restore parts of the image information while passing the remained information to deeper layers. The modules are embedded to form a deep architecture. An elaborate sub-network is also designed for effective feature fusion. Using the proposed model, the information of lower frequency is restored by shallower BRMs while the information of higher frequency is recovered by deeper BRMs. Comprehensive experiments demonstrate the effectiveness of the proposed idea.

Acknowledgements This work was supported by National Natural Science Foundation of China under Grant 61772455, 61772508, and U1713213, Yunnan Natural Science Funds under Grant 2018FY001(-013), the Program for Excellent Talents of National Natural Science Foundation of Yunnan University under Grant 2018YDJQ004, Shenzhen Technology Project (JCYJ20170413152535587, JCYJ20180507182610734), and CAS Key Technology Talent Program.
References


4189