Efficient Single-image Super-resolution Using Dual path Connections with Multiple scale Learning

BIN-CHENG YANG and GANGSHAN WU, State Key Laboratory for Novel Software Technology, Nanjing University, China

Deep convolutional neural networks have been demonstrated to be effective for single-image super-resolution in recent years. On the one hand, residual connections and dense connections have been used widely to ease forward information and backward gradient flows to boost performance. However, current methods use residual connections and dense connections separately in most network layers in a sub-optimal way. On the other hand, although various networks and methods have been designed to improve computation efficiency, save parameters, or utilize training data of multiple scale factors for each other to boost performance, they either do super-resolution in high-resolution space to have a high computation cost or cannot share parameters between models of different scale factors to save parameters and inference time. To tackle these challenges, we propose an efficient single-image super-resolution network using dual path connections with multiple scale learning (EMSRDPN). By introducing dual path connections inspired by Dual path Networks into EMSRDPN, it uses residual connections and dense connections in an integrated way in most network layers. Dual path connections have the benefits of both reusing common features of residual connections and exploring new features of dense connections to learn a good representation for single-image super-resolution. To utilize the feature correlation of multiple scale factors, EMSRDPN shares all network units in low-resolution space between different scale factors to learn shared features and only uses a separate reconstruction unit for each scale factor, which can utilize training data of multiple scale factors to help each other to boost performance, meanwhile, which can save parameters and support shared inference for multiple scale factors to improve efficiency. Experiments show EMSRDPN achieves better performance and comparable or even better parameter and inference efficiency over state-of-the-art methods. Code will be available at https://github.com/yangbincheng/EMSRDPN.

CCS Concepts: • Computing methodologies → Neural networks; Computational photography;

Additional Key Words and Phrases: Single-image super-resolution, dual path connection, multiple scale training and inference

ACM Reference format:
Bin-Cheng Yang and Gangshan Wu. 2023. Efficient Single-image Super-resolution Using Dual path Connections with Multiple scale Learning. ACM Trans. Multimedia Comput. Commun. Appl. 19, 3, Article 115 (February 2023), 21 pages.https://doi.org/10.1145/3570164
1 INTRODUCTION

Single-image super-resolution (SISR) [13] is the task of recovering one high-resolution (HR) image from one low-resolution (LR) image. It has tremendous applications such as medical imaging [43], satellite imaging [32], surveillance [64] and object recognition [41], which require a lot of image details to help post-processing. It is an ill-posed problem that has no unique solution, because the LR image can be produced from many different HR images. Due to lacking of the prior knowledge of the HR image, it is difficult to recover the HR image from one single LR image.

Many methods have been proposed to tackle this difficult problem, which can be classed into three categories: interpolation-based methods [2, 30, 59], model-based methods [11, 57], and example-based learning methods [5, 9, 12, 15, 44, 54, 62]. Because of the ineffectiveness of interpolation methods for large scale factors and the computation complexity of optimization processes in the prior dependent model-based methods, example-based learning methods have attracted huge attention to learn a direct mapping from LR image to HR image these years. Especially deep CNN-based methods have been demonstrated to be able to learn the mapping from LR image to HR image effectively.

Since the pioneer work of Dong et al. [9] first uses a CNN (SRCNN) to solve the SISR problem, a lot of deep networks have been proposed to tackle this problem. On the one hand, residual and dense connections are used widely to ease the forward information and backward gradient flows of current deep CNN-based methods, which have shown great progress in the reconstruction quality of SISR task. VDSR [23], DRCN [24], RED [36], DRRN [45], LapSRN [26], SRGAN/SRResNet [28], EDSR/MSDR [34], RCAN [60], and SAN [7] use residual connections global-wise, block-wise, or layer-wise, SRDenseNet [49], MemNet [46], and RDN [62] also use dense connections between layers in one block or between different blocks in a network. Although the success of residual and dense connections in SISR networks, they mostly utilize the residual and dense connections separately in most network layers, in this manner, which leverage residual and dense connections in a sub-optimal way. On the other hand, how to improve computation efficiency, save parameters, or use training data of multiple scale factors for each other to boost performance is also well studied. Some networks, such as VDSR [23], DRCN [24], DRRN [45], and MemNet [46], use interpolated LR image as input while sharing all the parameters between different scale factors to leverage training data of multiple scale factors for each other, but they cannot do super resolution of multiple scale factors in one pass because of different interpolated inputs for different scale factors and have high computation and memory cost, because they do super resolution in interpolated HR space. Other networks such as EDSR [34], RDN [62], and RCAN [60] use LR image as input directly to train separate models for different scale factors to save computation and memory cost, and use transfer learning to fine-tune large scale factor models from a specific small scale factor model to leverage training data of the specific small scale factor for large scale factors, but they cannot share parameters between different scale factors and must do super resolution for different scale factors separately using different models, meanwhile they cannot fully leverage training data of multiple scale factors for each other, especially training data of large scale factors for small scale factors.

To overcome these drawbacks, we propose a novel network design named as EMSRDPN by introducing dual path connections and multiple scale training and inference into a very deep convolutional neural network. Through dual path connections, EMSRDPN uses residual and dense connections in an integrated way in most convolutional layers to leverage both residual connections to reuse common features and dense connections to explore new features to learn a good representation for SISR. Meanwhile, by sharing most parameters between different scale factors to use training data of different scale factors for each other to utilize feature correlation during training and amortize most parameters and computation between different scale factors during
Efficient Single-image Super-resolution Using Dual path Connections

![Fig. 1. Zoomed visual results of “img011” from Urban100 dataset for ×4. Our method reconstructs more high-frequency details and less artifacts.](image)

![Fig. 2. Network architecture of EMSRDPN for scale factors ×2, ×3, and ×4.](image)

Inference, EMSRDPN has benefits of both achieving better performance as shown in Figure 1 and improving parameter efficiency and inference time of network.

EMSRDPN includes four parts as shown in Figure 2, the first part is the feature extraction block (FEB), which extracts low level features from downsampled LR image, the second part is multiple stacked dual path blocks (DPBs) to learn rich hierarchical features from the low level features, each DPB is made of several cascading dual path units (DPUs) and one transition unit, the third part is hierarchical feature integration block (HFIB), which fuses rich hierarchical feature maps output by all the DPBs and FEB both making information forward-propagation direct and alleviating gradient vanishing/exploding problem, the last part is multiple parallel reconstruction blocks (RBs) for different scale factors, respectively. FEB, DPBs and HFIB are shared between different scale factors. EMSRDPN is a multi-task super-resolution learning and inference network for multiple scale factors, which can use training data of super-resolution tasks of different scale factors to utilize feature correlation to help each other to improve performance while saving number of parameters and inference time of the model. The key component of network is the dual path unit, which integrates residual connection and dense connections in one layer seamlessly. As pointed
by Chen et al. [6], this design can both exploit common features and explore new features to learn rich hierarchical features from low level features, which is beneficial to learn a good representation for SISR. To build a deep network, we add a transition unit in every dual path block to fuse residual and dense features to keep computational cost and memory consumption under control.

A preliminary version of this work was presented as a conference paper [52], the model in which is named as SRDPN. We incorporate additional improvements in significant ways in current work:

- We propose more efficient DPUs to utilize dual path connections to learn a good representation for SISR, and improve network architecture to do SISR in LR space to save computation and memory cost, these two improvements benefit the efficiency of network and form a base for multiple scale learning.
- We propose a multiple scale training and inference architecture of network to utilize the feature correlation between multiple scale factors by sharing most of network parameters between different scale factors to make use of training data of multiple scale factors for each other during training and amortize parameters and computation between different scale factors during inference, which has benefits of both achieving better performance and improving parameter efficiency and inference time of network. The improved model in this work is named as EMSRDPN.
- Extensive experiments on standard benchmark datasets demonstrate the effectiveness of our EMSRDPN model, which achieves better performance and comparable or even better parameter efficiency and inference time compared to state-of-the-art methods.

2 RELATED WORK

A lot of methods [2, 5, 11, 12, 15, 30, 44, 47, 48, 50, 53–55, 57–59] have been proposed to solve the classic SISR problem for decades. We focus on recent CNN-based methods for SISR in this section. We first introduce development of CNN-based methods for SISR. Second, we discuss current use of residual and dense connections in CNN-based methods, their limitation and our motivation. Third, we discuss multiple scale training and inference strategies of current CNN-based methods, their limitations and our motivation.

2.1 CNN-based Methods for SISR

The pioneer work of Dong et al. [9] first uses a CNN (SRCNN) to solve the SISR problem, which processes interpolated LR image and learns feature extraction, feature mapping and HR image reconstruction jointly. Furthermore, Dong et al. [10] improve SRCNN by designing a deeper network architecture (FSRCNN), which directly processes LR image and uses a deconvolution layer to do upscaling. To improve the computation efficiency further, Shi et al. [42] introduce an efficient sub-pixel convolution layer to do HR image reconstruction (ESPCN). From a human perception view, Johnson et al. [22] propose a perceptual loss function based on IMAGENET [8] pre-trained features. To use one model to handle SISR for different scale factors, Kim et al. [23] propose a 20-layer convolution network (VDSR) to leverage large context regions, and use residual learning and adjustable gradient clipping to ease training process. Moreover, to improve the model parameter efficiency, they also propose a deeply recursive convolutional network (DRCN) [24] using recursive supervision and skip connections. Afterwards, Tai et al. [45] adopt both global and local residual leaning into recursive network (DRRN) to mitigating training difficulty of network while reducing the model complexity. Also to implement a single model that can process multiple scale factors, Lai et al. [26] propose a laplacian pyramid super-resolution network (LapSRN) to progressively reconstruct sub-bands of the HR image, it can produce HR images of different scale factors in one pass. To leverage relations between SISR tasks of different scale factors, Lim et al. [34] first
propose enhanced deep residual networks (EDSR) for SISR, which remove unnecessary blocks in residual networks and adopt residual scaling, then they use transfer learning from small scale factor models to large scale factor models. Moreover, to exploit inter-task correlation of different scale factors, they design a multi-scale architecture (MDSR) based on EDSR, which uses the common network trunk and different network heads and tails for SISR tasks of different scale factors. From different points of view, Hui et al. [19] design an information distillation network (IDN) to reduce computation cost and memory consumption, Haris et al. [16] propose a deep convolutional neural network (DBPN) to implement iterative back projection algorithm [21], Li et al. [29] design a multi-scale residual network (MSRN) to detect features at different scales and combine them to boost performance, and Qiu et al. propose a novel embedded block residual network (EBRN) [40], which uses different modules to restore information of different frequencies.

Some networks focus on using residual connections and dense connections to boost performance. Mao et al. [36] use a symmetric convolution-deconvolution architecture (RED) and skip connections to recursively learn the residual of HR image and LR image. Then Ledig et al. [28] first use adversarial learning to generate HR image (SRGAN) based on a residual network based generator network (SRResNet), which uses residual blocks and global residual learning to reconstruct HR image. However, Tong et al. [49] introduce dense connections to a very deep network for SISR (SRDenseNet), both enabling an effective way to combine features and alleviating the gradient vanishing/exploding problem. Moreover, Tai et al. [46] propose a persist memory network (MemNet) design to leverage long-term and short-term memories, which uses dense connections among big blocks in a global way. Afterwards, Zhang et al. [62] propose residual dense network (RDN) to better exploit multi-level features in the network, which uses dense connections among the layers in local blocks and residual connections in blocks and globally. And Anwar et al. [3] propose densely connected residual blocks and an attention network (DRLN) to utilize residual connection, dense connection and attention for SISR.

Attention mechanisms and non-local operations have also been introduced into deep CNN-based methods to improve performance further. Zhang et al. further introduce attention mechanism in convolutional network (RCAN) [60] to solve SISR, which uses skip connections in multiple levels. After that, Dai et al. propose a second-order attention network (SAN) [7] with residual connections for more powerful feature expression and feature correlation learning. Then Zhang et al. propose kernel attention network [56] for SISR by dynamically selecting appropriate kernel size to adjust receipt field size. Niu et al. [39] propose HAN to model the holistic interdependencies among layers, channels and positions, Li et al. propose a lightweight dense connection distillation network [31] by incorporating contrast-aware channel attention. Since Liu et al. [35] first introduce non-local operations into a recurrent neural network (NLRN) to capture feature correlation and improve parameter efficiency for image restoration, Zhang et al. [61] propose RNAN to use local and non-local attention blocks to capture long-range dependency and attend to challenging parts and Mei et al. [38] combine non-local operation and sparse representation into deep networks (NLSN) to boost performance and efficiency.

2.2 Residual and Dense Connections Used in SISR Networks

Skip connections, especially residual connections and dense connections, are adopted widely to make use of hierarchical features and ease training of very deep networks. On the one hand, VDSR [23], DRCN [24], DRRN [45], MemNet [46], EDSR [34], LapSRN [26], DBPN [16], RED [36], SRResNet [28], RDN [62], RCAN [60], and SAN [7] all use residual connections to ease forward information flow and backward gradient flow of networks to boost performance. On the other hand, MemNet [46], DBPN [16], SRDenseNet [49], and RDN [62] also use dense connections to
leverage multi-level features and alleviate the gradient vanishing/exploding problem to improve performance.

Although residual connections and dense connections have been used to boost the SR performance of deep CNN-based methods effectively, the current methods only use residual connections and dense connections in a separate way in most network layers. To learn a good representation for single-image super-resolution by using residual connections and dense connections in a better way, we introduce dual path connections inspired by Chen et al. [6] into a very deep neural network, which has the benefits of both residual connections to reuse common features and dense connections to explore new features to learn good features.

2.3 Multiple scale Training and Inference for SISR Networks

Since VDSR [23] first uses the interpolated LR image as input to the network to leverage training data of multiple scale factors for each other and share model parameters between different scale factors, DRCN [24], DRRN [45], and MemNet [46] all follow this strategy. Although this kind of strategy has the benefits of both exploiting correlation between SISR task of different scale factors to boost performance and sharing one model for different scale factors to improve parameter efficiency, it has two limitations. First, this strategy does super resolution in the HR space, which has high computation and memory cost. Second, it cannot do multi-scale super resolution, because the interpolated inputs to the network for different scale factors will be different, leading to multiple passes of inference for different scale factors. On the other hand, EDSR [34], RDN [62] and RCAN [60] use transfer learning to fine-tune model parameters of large scale factors from model parameters of a specific small scale factor. Although this kind of strategy somehow utilizes the knowledge of small scale factor task to boost the performance of large scale factor tasks, it has some limitations. First, the training data of different scale factors cannot be utilized for each other, especially training data of large scale factors for small scale factors. Second, the parameters thereafter models cannot be shared between different scale factors impacting the parameter efficiency, because the parameters are different for different scale factors after fine-tuning. At last, this strategy cannot do multi-scale super resolution either, because different scale factors have different parameters and models although the inputs to different models can be the same. Besides these two main kind of strategies, a variety of EDSR [34] model named MDSR [34] utilizes shared parameters of network trunk between different scale factors and uses training data of different scale factors to help each other, but it has limitation. Although the parameters of network trunk are shared, MDSR has different pre-processing and upsampling modules for different scale factors, leading to which cannot do multi-scale super resolution, because there will be different input feature maps to shared network trunk after pre-processing by different pre-processing modules during inference. LapSRN [26] and MS-LapSRN [27] also use training data of different scale factors to help each other and share parameters between different scale factors, their strategy is different from ours, which need more inference stages therefore more parameter, computation and memory budgets for large scale factors.

To overcome these drawbacks and utilize feature correlation between multiple scale factors, we propose a multiple scale training and inference architecture of network that shares most network parameters and uses only different parameters in network tails for different scale factors, which have benefits of shared parameters and shared inference in most of network to boost efficiency and using training data of different scale factors to train shared features to boost performance.

3 METHOD

In this section, we describe network architecture and components of EMSRDPPN in detail.
3.1 Network Architecture

The proposed EMSRDPN, as shown in Figure 2, consists of four parts: a FEB, multiple stacked DPBs, a HFIB, and finally multiple parallel RBs for different scale factors. Let us denote $x$ as the input and $\hat{y}_s$ as the output of the network for scale factor $s$, which is arbitrary one of the different scale factors of network without loss of generality. First, we use two convolution layers in FEB to extract low-level features from network input $x$,

$$B_0 = f_{\text{FEB}}(x),$$

where $f_{\text{FEB}}$ denotes function of FEB, $B_0$ denotes the extracted features to be sent to the first DPB and $x$ denotes downsampled LR image which will be upscaled. Second, we use multiple stacked DPBs to learn rich hierarchical features from low level features. Suppose $D$ blocks are stacked, we have

$$B_d = f_{\text{DPB},d}(f_{\text{DPB},d-1}(\cdots(f_{\text{DPB},1}(B_0))\cdots)),$$

where $f_{\text{DPB},d}$ denotes the function of $d$th DPB, $B_d$ denote the output feature maps of the $d$th DPB. Afterwards, we integrate rich hierarchical feature maps output by all the DPBs and FEB using HFIB,

$$H = f_{\text{HFIB}}(B_{-1}, [B_1, B_2, \ldots, B_D]),$$

where $B_{-1}$ denotes the output of first convolution layer of FEB, $f_{\text{HFIB}}$ denotes the function of HFIB and $H$ denotes the output of HFIB. Finally, we use integrated features output by HFIB as input to RB specific to scale factor $s$ to reconstruct the HR image $y_s$,

$$\hat{y}_s = g_s(x) = f_{\text{RB},s}(H).$$

where $f_{\text{RB},s}$ denotes the function of the RB for scale factor $s$ and $g_s$ denotes the function of EMSRDPN for scale factor $s$, respectively. (In this article, either $[B_1, B_2, \ldots, B_D]$ denotes the concatenation of feature maps along feature dimension when used on the right-hand side of formulae or as the arguments of functions, or $[B_1, B_2, \ldots, B_D]$ denotes the split of feature maps along feature dimension when used on the left-hand side of formulae.)

Given a set of training image pairs $\{x_{s,k}, y_{s,k}\}_{k=1}^M, s \in S$ for a set of scale factors $S$, the network is used to minimize the following Mean Absolute Error (MAE) loss for every scale factor $s \in S$ alternately:

$$L(\Theta_s) = \frac{1}{M} \sum_{k=1}^M \|y_{s,k}^{(k)} - \hat{y}_s^{(k)}\|_1,$$

where $\Theta_s$ is the parameters of EMSRDPN for scale factor $s$, which includes shared parameters in FEB, DPBs, HFIB for all the different scale factors and private parameters in RB corresponding to scale factor $s$. Note that we have only one HR image and different LR images for different scale factors during training using standard training datasets, but we can infer multiple SR images from one LR image during testing.

In the following subsections, we will describe FEB, dual path block (DPB), HFIB, and RB in detail.

3.2 Feature Extraction Block

The FEB uses two $3 \times 3$ convolution layers to extract low level features from the LR image $x$. To be consistent with DPBs, we also take its output as two parts, the residual part and the dense part,

$$B_{-1} = W_{\text{F}}^1(x),$$

$$f_{\text{FEB}}(x) = B_0 = W_{\text{F}}^2(B_{-1}),$$

$$[R_{1}^{0}, D_{1}^{0}] = B_0,$$
where $W^1_F$ and $W^2_F$ denote the weights of the first and second convolution layer in FEB (we omit bias for simplicity in this section and following sections). $B_{-1}$ denotes the output of first convolution layer in FEB, $B_0$ denotes the output of second convolution layer in FEB, $R_0^d$ and $D_0^d$ denote the residual part and the dense part of output $B_0$ after splitting along feature dimension, which will be the input to the first DPB.

### 3.3 Dual path Block

As shown in Figure 3, each DPB is composed of several cascading DPUs and one transition unit (TU). Let us assume our network has $D$ DPBs and each DPB has $C$ DPUs. The input feature maps of $d$th DPB is the output feature maps of $(d-1)$th DPB or the output feature maps of FEB when $d = 1$. Let us denote $R_c^d$ and $D_c^d$ as the residual part and the dense part of output feature maps of $c$th DPU in $d$th DPB or the residual part and the dense part of output feature maps of TU in $(d-1)$th DPB when $c = 0, d \neq 1$ or the residual part and the dense part of output feature maps of FEB when $c = 0, d = 1$.

#### 3.3.1 Dual path Unit

DPU is shown in Figure 4. Each DPU takes its input feature maps as two parts, the residual part and the dense part. The input of first DPU in $d$th DPB is the output of transition unit in $(d-1)$th DPB or the output of FEB when $d = 1$, which is denoted as $R_0^d$ and $D_0^d$. The input of $c$th DPU except the first DPU in $d$th DPB is the output of $(c-1)$th DPU in $d$th DPB.

Each DPU first concatenates the residual part and the dense part of its input feature maps, then does non-linear transform to the concatenated input feature maps, because we adopt the “pre-activation” design of residual network [17], at last it uses a bottleneck design, which includes a $1 \times 1$ convolution layer followed by a $3 \times 3$ convolution layer to learn a residual function and new dense features while keeping the computation and memory budget low,

$$C_c^d = W_c^d \left( \sigma \left( T_c^d \left( \sigma \left( [R_{c-1}^d, D_{c-1}^d] \right) \right) \right) \right),$$

(9)
Efficient Single-image Super-resolution Using Dual path Connections

where $T^c_d$ and $W^c_d$ denote the weights of the $1 \times 1$ convolution layer and $3 \times 3$ convolution layer of $c$th DPU in $d$th DPB, $\sigma$ denotes nonlinear function, $C^c_d$ denotes the output of the $3 \times 3$ convolution layer in this DPU. Next it splits the output into two parts, a residual function to be added to the residual part of input and new dense features to be concatenated to the dense part of input,

$$[F^c_d, G^c_d] = C^c_d,$$  \hspace{1cm} (10)

where $F^c_d$ denotes the residual function to be added to the residual part of input feature maps and $G^c_d$ denotes newly produced dense features to be concatenated to the dense part of input feature maps. Finally, it add the residual function to the residual part of input feature maps to form “residual path,”

$$R^c_d = R^{c-1}_d + F^c_d,$$  \hspace{1cm} (11)

and concatenate the newly learned dense features to the dense part of the input feature maps to form “dense path,”

$$D^c_d = [D^{c-1}_d, G^c_d].$$  \hspace{1cm} (12)

These two paths of feature maps are used as input to the next DPU in current DPB or as input to the TU in current DPB when this DPU is the last DPU in current DPB. Batch normalization layer is removed according to Lim et al. [34] and non-linearity transform is changed to ReLU to improve computation and memory efficiency in this work compared to SRDPN.

### 3.3.2 Transition Unit

Because the dense path of each DPU in every DPB will concatenate dense part of its output to dense part of its input feature maps, which is all the dense parts of outputs of previous DPU and dense part of input of the first DPU in current DPB. Assuming the input feature maps to first DPU has $G_r$ residual path features and $G_d$ dense path features, and the grow rate of dense path of each DPU is $G$, after the $C$th DPU of $d$th DPB, we have $G_r$ residual path features and $(G_d + C \times G)$ dense path features, which are the input of $(d + 1)$ DPB. To build deeper EMSRDPN and make computation complexity and memory consumption under control, we add a transition unit in the tail of each DPB to transform all the $(G_r + G_d + C \times G)$ output features of current DPB to $(G_r + G_d)$ features to be the input to the first DPU of next DPB. Each transition unit also takes its input feature maps as two parts, the residual part and the dense part, which are output by the last DPU in current DPB. The structure of TU is shown in Figure 5, the function of TU in $d$th DPB is formulated as follows:

$$f_{DPB,d}(B_{d-1}) = B_d = W^T_d([R^c_{d-1}, D^c_{d-1}]),$$  \hspace{1cm} (13)

$$[R^0_{d+1}, D^0_{d+1}] = B_d.$$  \hspace{1cm} (14)
where $W_d^T$ denotes the weights of $1 \times 1$ convolutional layer in TU, $B_d$ denotes the output of convolution layer in TU, $R_{d+1}^0$ and $D_{d+1}^0$ denote the residual part and dense part of output feature maps of TU after splitting along feature dimension, which are the input of first DPU in next DPB. The output of $d$th DPB is the output of the TU in $d$th DPB.

### 3.4 Hierarchical Feature Integration Block

In hierarchical feature integration block, we first concatenate outputs of all the DPBs to use rich hierarchical features learned by multiple stacked DPBs,

$$ F = [B_1, B_2, \ldots, B_D], $$

then use a $1 \times 1$ convolution layer to transform the feature space to low dimensionality to reduce computation cost and memory consumption, and finally use a $3 \times 3$ convolution layer to learn a global residual function to be added to the output of the first convolution layer in FEB of network (denoted as $B_{-1}$) to improve forward information and backward gradient flows further following the strategy of Zhang et al. [62],

$$ H = f_{HFIB}(B_{-1}, F) = W_H^2(W_H^1(F)) + B_{-1}, $$

where $W_H^1$ and $W_H^2$ denote the weights of $1 \times 1$ convolution layer and $3 \times 3$ convolution layer in HFIB, respectively, $H$ denotes the output of HFIB.

### 3.5 Reconstruction Block

To use training data of multiple scale factors for each other to utilize feature correlation to boost performance and improve parameter efficiency and inference time of the proposed model further, we use multiple parallel RBs for different scale factors, respectively, and shared FEB, DPBs, and HFIB for different scale factors in EMSRDPN, which can produce HR images of different scale factors in one pass. The number of and corresponding scale factors of RBs can be configured flexibly according to availability of training data and requirements of applications. In the RB for a specific scale factor $s$, we use the output of HFIB shared by different scale factors as the input to an efficient sub-pixel convolution layer proposed by Shi et al. [42] followed by a $3 \times 3$ convolution layer to reconstruct the HR image for scale factor $s$,

$$ \hat{y}_s = f_{RB_s}(H) = W_{Rs}^2(W_{Rs}^1(H)), $$

where $W_{Rs}^1$ and $W_{Rs}^2$ denote the weights of efficient sub-pixel convolution layer and $3 \times 3$ convolution layer in RB for scale factor $s$, respectively.

### 4 EXPERIMENTS

In this section, we first discuss training and testing data sets and metrics for performance evaluation. Next, we describe implementation details of the proposed EMSRDPN. Then, we study effects to reconstruction performance of different number of scale factors for training. Furthermore, we compare the performance of EMSRDPN with the state-of-the-art CNN-based SISR methods. Finally, we investigate the model complexity and inference time of EMSRDPN.

### 4.1 Datasets and Evaluation Metrics

Following the setting in previous methods [16, 63], we use DIV2K [1] and Flickr2K [33] as the training sets, five standard benchmark data sets, Set5 [4], Set14 [55], BSD100 [37], Urban100 [18], and Manga109 [14], are used for performance evaluation. Because some methods only use DIV2K [1] as the training set, we also train a model denoted as EMSRDPN-D using only DIV2K [1] for
reference and comparison. DIV2K and Flickr2K datasets provide corresponding HR images and LR images, which are downsampled based on bicubic interpolation. We do two kinds of data augmentation during training: (1) Rotate the images of 90, 180, 270 degrees; (2) Flip the images vertically. We use peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) as evaluation metrics. We train the network using all three RGB channels and test the network from Y channel of YCbCr color space of images following the strategy of most state-of-the-art methods.

4.2 Implementation Details

We construct a network with \((D = 16)\) DPBs and each DPB has \((C = 4)\) DPUs. The width of residual path of network is set to \((G_r = 64)\), the width of basic dense path of network is set to \((G_d = 64)\), which is the width of dense part of TU and FEB, and the grow rate for the dense path of network is set to \((G = 64)\), which is the width of dense part of DPU. The width of output feature maps of convolutional layers in FEB and RBs is set to \((G_r + G_d)\), except that the output of the last \(3 \times 3\) convolutional layers in RBs is multiple reconstructed HR color images corresponding to different scale factors. All layers use \(3 \times 3\) convolutional kernels except transition layers in DPUs, TUs and HFIB, which use \(1 \times 1\) convolutional kernels to fuse features and reduce computation and memory cost. Compared to SRDPN, rectifier linear unit \([51]\) is used as non-linear function of our network and batch normalization \([20]\) is removed before non-linearity. Mini-batch size is set to 16 due to memory limit. We use \(48 \times 48\) LR image patches and corresponding HR image patches for \(\times 2, \times 3, \times 4,\) and \(\times 8\) scale factors to train our models. At each iteration of training EMSRDPN, we randomly sample a scale factor uniformly to sample mini-batch training pairs of this scale factor to update shared parameters in FEB, DPBs, HFIB and private parameters in RB for the sampled scale factor. For a sampled specific scale factor, we first uniformly sample an image pair from all pairs of corresponding HR and LR images, then uniformly sample a training patch pair from the sampled image pair. Adam optimizer \([25]\) is used to learn the network weights, learning rate is initialized to \(1 \times 10^{-4}\). We train our network for about \(1 \times 10^6\) iterations on average for each scale factor in set of all training scale factors, the learning rate is step-decayed by 2 after about number of training scale factors times \(2 \times 10^5\) of iterations for multiple scale training. We implement our method using PyTorch framework, train our models using NVIDIA Tesla V100 GPUs and test our models using NVIDIA TITAN Xp GPUs.

4.3 Ablation Study

4.3.1 Effects of Multiple scale Learning. In this subsection, we study the effects to performance of multiple scale training and the benefits to efficiency of multiple scale inference of our EMSRDPN model. We design three groups of experiments to study the effects to performance indices of number of training scale factors. First, we train four models each with only one RB tail for a different specific scale factor in set of scale factors \(\times 2, \times 3, \times 4,\) and \(\times 8\), denoted as EMSRDPN-2, EMSRDPN-3, EMSRDPN-4, and EMSRDPN-8. Second, we start with a single scale model of scale factor \(\times 2\), add one more scale factor each time according to the order of \(\times 3, \times 4, \times 8\) to train model of two scale factors, three scale factors and four scale factors, denoted as EMSRDPN-2, EMSRDPN-23, EMSRDPN-234, and EMSRDPN-2348. Third, we start with a single scale model of scale factor \(\times 8\), add one more scale factor each time according to the order of \(\times 4, \times 3, \times 2\) to train models of two scale factors, three scale factors and four scale factors, denoted as EMSRDPN-8, EMSRDPN-48, EMSRDPN-348, and EMSRDPN-2348. The same models in different groups are trained only once.

The performance comparison are shown in Table 1 and Figure 6. We have two observations. First, \(\times 3, \times 4, \times 8\) all benefit from multiple scale training, the performance is increasing with the number of training scale factors. Second, \(\times 2\) benefits from multiple scale training up to three training scale factors, including EMSRDPN-23, EMSRDPN-234 models, but the performance indices
Table 1. Average PSNR/SSIM Metrics for Scale Factors ×2, ×3, ×4, ×8 of EMSRDPN Using Different Number of Training Scale Factors

| Method       | Scale | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM | PSNR | SSIM |
|--------------|-------|------|------|------|------|------|------|------|------|------|------|
| EMSRDPN-2    | ×2    | 38.24 | 0.9658 | 33.96 | 0.9211 | 32.39 | 0.9115 | 32.95 | 0.9407 | 39.38 | 0.9801 |
| EMSRDPN-23   | ×2    | 38.28 | 0.9660 | 34.33 | 0.9186 | 32.42 | 0.9122 | 33.34 | 0.9439 | 39.53 | 0.9832 |
| EMSRDPN-234  | ×2    | 38.22 | 0.9659 | 34.28 | 0.9190 | 32.80 | 0.9210 | 33.25 | 0.9436 | 39.66 | 0.9835 |
| EMSRDPN-2348 | ×3    | 34.67 | 0.9374 | 30.68 | 0.8614 | 29.26 | 0.8276 | 28.95 | 0.8762 | 34.29 | 0.9526 |
| EMSRDPN-23   | ×3    | 34.69 | 0.9376 | 30.61 | 0.8624 | 29.30 | 0.8276 | 29.01 | 0.8790 | 34.52 | 0.9536 |
| EMSRDPN-234  | ×3    | 34.77 | 0.9381 | 30.68 | 0.8634 | 29.33 | 0.8284 | 29.16 | 0.8816 | 34.62 | 0.9541 |
| EMSRDPN-2348 | ×3    | 34.80 | 0.9380 | 30.71 | 0.8637 | 29.31 | 0.8282 | 29.17 | 0.8818 | 34.41 | 0.9541 |
| EMSRDPN-348  | ×3    | 34.80 | 0.9381 | 30.76 | 0.8650 | 29.32 | 0.8283 | 29.12 | 0.8810 | 34.62 | 0.9541 |
| EMSRDPN-4    | ×4    | 32.56 | 0.9090 | 28.91 | 0.8559 | 27.78 | 0.7681 | 26.88 | 0.8220 | 31.36 | 0.9121 |
| EMSRDPN-48   | ×4    | 32.62 | 0.9095 | 28.96 | 0.8567 | 27.80 | 0.7683 | 26.99 | 0.8246 | 31.45 | 0.9132 |
| EMSRDPN-348  | ×4    | 32.62 | 0.9095 | 28.93 | 0.8569 | 27.72 | 0.7687 | 26.99 | 0.8253 | 31.55 | 0.9135 |
| EMSRDPN-2348 | ×4    | 32.94 | 0.9095 | 29.05 | 0.8665 | 27.72 | 0.7682 | 26.96 | 0.8240 | 31.51 | 0.9137 |

Best is highlighted, and second best is underlined.

Fig. 6. Illustration of effects to reconstruction performance of number of training scale factors. (a) Performance trend of models in group 2 for ×2. (b) Performance trend of models in group 2 for ×3. (c) Performance trend of models in group 3 for ×4. (d) Performance trend of models in group 3 for ×8.

drop a little for the model of four training scale factors, i.e., EMSRDPN-2348. We argue that it is because patterns of LR images of large scale factor of ×8 are much different from patterns of LR images of small scale factor of ×2, which deteriorates the feature correlation of these two scale factors. According to these observations and analysis, we choose EMSRDPN-2348 as our final model.
Table 2. Parameters and Inference Overhead for a 256 × 256 Image of EMSRDPN Using Different Number of Training Scale Factors

| Method     | Params | Total Memory (MB) | Total Flops (T) |
|------------|--------|-------------------|-----------------|
| EMSRDPN-2  | 13,311,875 | 10627 | 0.87 |
| EMSRDPN-3  | 13,969,795 | 10951 | 0.92 |
| EMSRDPN-4  | 13,822,211 | 11660 | 1.03 |
| EMSRDPN-8  | 14,412,547 | 15792 | 1.66 |
| EMSRDPN-2348 | 17,522,188 | 17926 | 1.98 |

SSI: Single Scale Inference, MSI: Multiple Scale Inference.

Table 3. Average PSNR/SSIM Metrics for Scale Factor ×4 of EMSRDPN-Res-4, EMSRDPN-Dense-4, and EMSRDPN-4

| Method                | Scale | Set5 PSNR | Set5 SSIM | Set14 PSNR | Set14 SSIM | B100 PSNR | B100 SSIM | Urban100 PSNR | Urban100 SSIM | Manga109 PSNR | Manga109 SSIM |
|-----------------------|-------|-----------|-----------|-------------|-------------|-----------|-----------|---------------|---------------|---------------|---------------|
| EMSRDPN-Res-4         | ×4    | 32.52     | 0.9082    | 28.89       | 0.8050      | 27.74     | 0.7045    | 26.71         | 0.8179        | 31.17         | 0.9213        |
| EMSRDPN-Dense-4       | ×4    | 32.52     | 0.9084    | 28.87       | 0.8049      | 27.74     | 0.7045    | 26.76         | 0.8186        | 31.28         | 0.9215        |
| EMSRDPN-4             | ×4    | 32.50     | 0.9083    | 28.88       | 0.8048      | 27.74     | 0.7045    | 26.78         | 0.8188        | 31.28         | 0.9215        |

Best is highlighted, and second best is underlined.

Table 4. Network Configuration and Inference Overhead for a 256 × 256 Image of EMSRDPN-Res-4, EMSRDPN-Dense-4, and EMSRDPN-4

| Method       | G_r   | G_d   | G   | G_r + G_d | D   | C      | Total Memory (MB) | Total Flops (T) |
|--------------|-------|-------|-----|-----------|-----|--------|--------------------|-----------------|
| EMSRDPN-Res-4 | 125   | 5     | 130 | 130       | 16  | 0      | 13,915,803         | 11660 |
| EMSRDPN-Dense-4 | 0    | 122   | 122 | 122       | 16  | 4      | 13,718,671         | 12578 |
| EMSRDPN-4     | 64    | 64    | 128 | 128       | 16  | 4      | 13,822,211         | 11660 |

SSI: Single Scale Inference, MSI: Multiple Scale Inference.

abbreviated as EMSRDPN to compare to state-of-the-art SISR methods to achieve a good tradeoff between reconstruction performance and parameter and inference efficiency.

We also investigate the benefits to efficiency of multiple scale inference of our EMSRDPN model. As shown in Table 2, the multiple scale inference capability of our EMSRDPN model can amortize the parameters, memory consumption and computation flops between multiple scale factors. Our EMSRDPN model only add marginal parameters compared to a single scale model such as EMSRDPN-8, and only add marginal memory consumption and marginal computation flops to do inference of four scale factors compared to do inference of a single scale factor such as ×8 using EMSRDPN-8. The overhead of multiple scale inference of our EMSRDPN model is only fourth of overhead of multiple scale inference using single scale models.

4.3.2 Effects of Dual path Connections. In this subsection, we study the effects of dual path connections proposed in our EMSRDPN model. Because residual connections and dense connections are special cases of the proposed dual path connections, we can simply set G_d and G to 0 to form a model with a residual net structure or set G_r to 0 to form a model with a dense net structure. In this way, we can study the effects of dual path structure compared to pure residual net structure and dense net structure by limiting the models of three structures to have similar number of parameters and similar number of network units thereof network depth. In this ablation study, we only train models of different structures for ×4 scale factor. Table 4 shows the network configuration of models of three structures, we use EMSRDPN-Res-4 to denote the model with only a residual net structure and EMSRDPN-Dense-4 to denote the model with only a dense net structure. Table 3 shows the performance comparisons for models of three structures. Table 4 also includes the total memory and total flops of models with different structures during inference to compare memory and computation overheads.

As shown in Table 3, EMSRDPN-4 has comparable even better performance compared to EMSRDPN-Res-4 and EMSRDPN-Dense-4. As shown in Table 4, the memory consumption of
Table 5. Average PSNR/SSIM Metrics for Scale Factors ×2, ×3, ×4, ×8 of Compared Methods

| Method      | Scale | Bicubic | SRCNN [9] | FSRCNN [10] | VDSR [21] | LapSVM [28] | MemNet [46] | EMSRDPN-D | EMSRDPN | EMSRDPN++ |
|-------------|-------|---------|-----------|-------------|-----------|-------------|-------------|-----------|----------|-----------|
|              | ×2    | 24.87   | 30.24     | 33.95       | 33.85     | 33.78       | 33.52       | 33.73     | 33.75   | 33.85     |
|              | ×3    | 24.87   | 30.24     | 33.95       | 33.85     | 33.78       | 33.52       | 33.73     | 33.75   | 33.85     |
|              | ×4    | 24.87   | 30.24     | 33.95       | 33.85     | 33.78       | 33.52       | 33.73     | 33.75   | 33.85     |
|              | ×8    | 24.87   | 30.24     | 33.95       | 33.85     | 33.78       | 33.52       | 33.73     | 33.75   | 33.85     |

Best is highlighted, and second best is underlined.

EMSRDPN-4 is less than EMSRDPN-Dense-4 and the computation flops of EMSRDPN-4 are less than EMSRDPN-Res-4. These observations demonstrate dual path structure can achieve a good trade-off between performance, memory consumption and computation overhead.

4.4 Comparisons with State-of-the-art Methods

In this subsection, we compare our method with SRCNN [9], FSRCNN [10], VDSR [23], LapSRN [26], MemNet [46], EDSR [34], SRMDNF [58], D-DBPN [16], RDN [62], RCAN [60], MSRN [29], ACM Trans. Multimedia Comput. Commun. Appl., Vol. 19, No. 5, Article 115. Pub. date: February 2023.
Efficient Single-image Super-resolution Using Dual path Connections

Fig. 7. Visual comparison for ×4 on B100 and Manga109 datasets.

NLRN [35], SAN [7], HAN [39], and NLSN [38] to validate the effectiveness of EMSRDPN. Following the literature, self-ensemble strategy [34] is also adopted to improve EMSRDPN and self-ensembled EMSRDPN is denoted as EMSRDPN+. Table 5 shows the PSNR and the SSIM metrics on five benchmark datasets. Our method achieves comparable even better performance over state-of-the-art methods on almost all combinations of benchmark datasets and scale factors.

ACM Trans. Multimedia Comput. Commun. Appl., Vol. 19, No. 3, Article 115. Pub. date: February 2023.
Fig. 8. Visual comparison for ×8 on Urban100 and Manga109 datasets.
Figures 7 and 8 show visual comparisons. As shown in image 78004 from B100 and images img028 and img092 from Urban100, our method reconstructs more textures, less blurring and ring effects of images than other state-of-the-art methods. As shown in images OLLunch and MisutenaidaDaisy from Manga109, our method reconstructs clearer edges and less artifacts of images than other state-of-the-art methods. Visual comparisons show our method reconstructs more high-frequency details such as edges and textures of images and has less artifacts.

4.5 Model Complexity and Inference Time

In this subsection, we compare number of parameters and inference time of our EMSRDPN model with state-of-the-art methods. For single scale inference such as $\times 4$ as shown in Figure 9(a), our method has more parameters or inference time than LapSRN [26], MDSR [34], MSRN [29], and DBPN [16], but it has much better reconstruction performance, meanwhile, our method has comparable even much less parameters and inference time than other state-of-the-art methods except that it has better reconstruction performance than these state-of-the-art methods. The efficiency of our method is much obvious especially for multiple scale inference such as $\times 2$, $\times 3$, and $\times 4$ as shown in Figure 9(b), our method has comparable parameters and inference time to MDSR [34] and MSRN [29] and much less parameters and inference time than other state-of-the-art methods, but it has much better reconstruction performance than MDSR [34] and MSRN [29] and better reconstruction performance than other state-of-the-art methods. It is worth noting that the additional multiple scale inference time of $\times 2$, $\times 3$, and $\times 4$ is only marginal compared to the single scale inference time of $\times 4$, because the most computation can be shared between different scale factors during inference, which is consistent with results in Table 2 and Section 4.3.1. The statistics of number of parameters and inference time in this subsection and reconstruction performance in previous subsection show that our method can achieve a good trade-off between number of parameters, inference time and reconstruction performance.

5 CONCLUSION AND FUTURE WORK

In this work, we propose an efficient single-image super-resolution network using dual path connections with multiple scale learning named EMSRDPN to boost reconstruction performance, save
parameters, and improve computation efficiency for SISR. First, we introduce efficient dual path connections into a very deep convolutional neural network, which have the benefits of both residual connections to reuse common features and dense connections to explore new features to learn a good representation for SISR. Second, based on the improved network architecture to do SISR in LR space to save computation and memory cost, we propose a multiple scale training and inference architecture of network by sharing most network parameters between different scale factors to make use of training data of multiple scale factors for each other to utilize feature correlation during training and amortize most parameters and computation between multiple scale factors during inference, which has benefits of both achieving better performance and improving parameter efficiency and inference time of network. Experiments show that our new model achieves better performance, and improved visual effects can be seen in the results compared to state-of-the-art methods. Meanwhile, our method has comparable or even better parameter efficiency and inference time compared to state-of-the-art methods. We believe the proposed multiple scale training and inference strategy is generally applicable to various network designs to boost performance and efficiency of SISR, and dual path connections could also be useful to multimedia tasks other than SISR.

REFERENCES

[1] Eirikur Agustsson and Radu Timofte. 2017. NTIRE 2017 challenge on single-image super-resolution: Dataset and study. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW’17). 1122–1131.

[2] Jan P. Allebach and Ping Wah Wong. 1996. Edge-directed interpolation. In Proceedings of the International Conference on Image Processing. IEEE Computer Society, 707–710. https://doi.org/10.1109/ICIP.1996.560768

[3] Saeed Anwar and Nick Barnes. 2022. Densely residual Laplacian super-resolution. IEEE Trans. Pattern Anal. Mach. Intell. 44 (2022), 1192–1204.

[4] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie-Line Alberi Morel. 2012. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In Proceedings of the British Machine Vision Conference. BMVA Press, 135.1–135.10. https://doi.org/10.5224/C.26.135

[5] Hong Chang, Dit-Yan Yeung, and Yimin Xiong. 2004. Super-resolution through neighbor embedding. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’04), Vol. 1. IEEE, I–I.

[6] Yunpeng Chen, Jianan Li, Huaxin Xiao, Xiaojie Jin, Shuicheng Yan, and Jiashi Feng. 2017. Dual path networks. In Advances in Neural Information Processing Systems. 4467–4475.

[7] Tao Dai, Jianrui Cai, Yongbing Zhang, Shu-Tao Xia, and Lei Zhang. 2019. Second-order attention network for single-image super-resolution. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR’19). Computer Vision Foundation/IEEE, 11065–11074. https://doi.org/10.1109/CVPR.2019.01132

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchial image database. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 248–255.

[9] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. 2016. Image super-resolution using deep convolutional networks. IEEE Trans. Pattern Anal. Mach. Intell. 38, 2 (2016), 295–307.

[10] Chao Dong, Chen Change Loy, and Xiaoou Tang. 2016. Accelerating the super-resolution convolutional neural network. In Proceedings of the European Conference on Computer Vision. Springer, 391–407.

[11] Weisheng Dong, Lei Zhang, Guangming Shi, and Xiaolin Wu. 2011. Image deblurring and super-resolution by adaptive sparse domain selection and adaptive regularization. IEEE Trans. Image Process. 20, 7 (2011), 1838–1857. https://doi.org/10.1109/TIP.2011.2108306

[12] William T. Freeman, Thouis R. Jones, and Egon C. Pasztor. 2002. Example-based super-resolution. IEEE Comput. Graph. Appl. 22, 2 (2002), 56–65.

[13] William T. Freeman and Egon C. Pasztor. 1999. Learning low-level vision. In Proceedings of the International Conference on Computer Vision. IEEE Computer Society, 1182–1189. https://doi.org/10.1109/ICCV.1999.790414

[14] Azuma Fujimoto, Toru Ogawa, Kazuyoshi Yamamoto, Yusuke Matsui, Toshihiko Yamasaki, and Kiyoharu Aizawa. 2016. Manga109 dataset and creation of metadata. In Proceedings of the 1st International Workshop on colmics ANaLysis, Processing and Understanding (MANPU@ICPR’16). Jean-Marc Ogier, Kiyoharu Aizawa, Koichi Kise, Jean-Christophe Burie, Toshihiko Yamasaki, and Motoi Iwata (Eds.). ACM, 2:1–2:5. https://doi.org/10.1145/3011549.3011551

[15] Daniel Glasner, Shai Bagon, and Michal Irani. 2009. Super-resolution from a single image. In Proceedings of the IEEE 12th International Conference on Computer Vision. IEEE, 349–356.
[16] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Urita. 2018. Deep back-projection networks for super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 1664–1673.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity mappings in deep residual networks. In Proceedings of the European Conference on Computer Vision. Springer, 630–645.

[18] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. 2015. Single-image super-resolution from transformed self-exemplars. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 5197–5206.

[19] Zheng Hui, Xiumei Wang, and Xinbo Gao. 2018. Fast and accurate single-image super-resolution via information distillation network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 723–731.

[20] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32nd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 37), Francis Bach and David Blei (Eds.). PMLR, Lille, France, 448–456. Retrieved from http://proceedings.mlr.press/v37/ioffe15.html.

[21] Michal Irani and Shmuel Peleg. 1991. Improving resolution by image registration. CVGIP: Graph. Model Image Process. 53 (1991), 231–239.

[22] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual losses for real-time style transfer and super-resolution. In Proceedings of the European Conference on Computer Vision. Springer, 694–711.

[23] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. 2016. Accurate image super-resolution using very deep convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 1646–1654.

[24] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. 2015. Fast and accurate super-resolution with dense connection distillation network. In Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS’18). Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristian Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). PMLR, Lille, France, 694–711. Retrieved from http://arxiv.org/abs/1412.6980.

[25] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR’15), Yoshua Bengio and Yann LeCun (Eds.). Retrieved from https://proceedings.mlr.press/v51/kingsma16.html.

[26] Yanchun Li, Jianglian Cao, Zhetao Li, Sangyoon Oh, and Nobuyoshi Komuro. 2021. Lightweight single-image super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 624–632.

[27] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. 2017. Deep Laplacian pyramid networks for fast and accurate super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 617–626.

[28] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang et al. 2017. Photo-realistic single-image super-resolution using a generative adversarial network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 4681–4690.

[29] Juncheng Li, Faming Fang, Kangfu Mei, and Guixu Zhang. 2018. Multi-scale residual network for image super-resolution. In Proceedings of the European Conference on Computer Vision (ECCV’18). 517–532.

[30] Xin Li and Michael T. Orchard. 2001. New edge-directed interpolation. IEEE Trans. Image Process. 10, 10 (2001), 1521–1527. https://doi.org/10.1109/83.951537.

[31] Yanchun Li, Jiangan Cao, Zhetao Li, Sangyoon Oh, and Nobuyoshi Komuro. 2021. Lightweight single-image super-resolution with dense connection distillation network. ACM Trans. Multimedia Comput., Commun. Appl. 17, 1s (2021), 1–17.

[32] Lukas Liebel and Marco Körner. 2016. Single-image super-resolution for multispectral remote sensing data using convolutional neural networks. ISPRS-Int. Arch. Photogram., Remote Sens. Spatial Inf. Sci. 41 (2016), 883–890.

[33] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. 2017. Enhanced deep residual networks for single-image super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR’17).

[34] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. 2017. Enhanced deep residual networks for single-image super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 136–144.

[35] Ding Liu, Bihan Wen, Yuchen Fan, Chen Change Loy, and Thomas S. Huang. 2018. Non-local recurrent network for image restoration. In Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NeurIPS’18). Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 1680–1689. Retrieved from https://proceedings.neurips.cc/paper/2018/hash/ fc49306d97602c0ed1be1dfb0835ead-Abstract.html.

[36] Xiaojiang Mao, Chunhua Shen, and Yu-Bin Yang. 2016. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In Advances in Neural Information Processing Systems. 2802–2810.

[37] David R. Martin, Charless C. Fowlkes, Doron Tal, and Jitendra Malik. 2001. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In Proceedings
of the 8th International Conference on Computer Vision (ICCV’01). IEEE Computer Society, 416–425. https://doi.org/10.1109/ICCV.2001.976555

[38] Yiquan Mei, Yuchen Fan, and Yuqian Zhou. 2021. Image super-resolution with non-local sparse attention. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 3517–3526.

[39] Ben Niu, Weilei Wen, Wenqi Ren, Xiangde Zhang, Lianping Yang, Shuzhen Wang, Kaihao Zhang, Xiaochun Cao, and Haifeng Shen. 2020. Single-image super-resolution via a holistic attention network. In Proceedings of the European Conference on Computer Vision. Springer, 191–207.

[40] Yajun Qiu, Ruxin Wang, Dapeng Tao, and Jun Cheng. 2019. Embedded block residual network: A recursive restoration model for single-image super-resolution. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV’19). IEEE, 4179–4188. https://doi.org/10.1109/ICCV.2019.00428

[41] Msm Sajjadi, B. Scholkopf, and M. Hirsch. 2017. EnhanceNet: Single-image super-resolution through automated texture synthesis. In Proceedings of the IEEE International Conference on Computer Vision.

[42] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. 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. 1874–1883.

[43] Wenzhe Shi, Jose Caballero, Christian Ledig, Xiahai Zhuang, Wenjia Bai, Kanwal K. Bhatia, Antonio M. Simoes Monteiro de Marvao, Tim Dawes, Declan P. O’Regan, and Daniel Rueckert. 2013. Cardiac image super-resolution with global correspondence using multi-atlas patchmatch. In Proceedings of the 16th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI’13) (Lecture Notes in Computer Science, Vol. 8151). Springer, 191–207.

[44] Jian Sun, Zongben Xu, and Heung-Yeung Shum. 2008. Image super-resolution using gradient profile prior. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 1–8.

[45] Ying Tai, Jian Yang, and Xiaoming Liu. 2017. Image super-resolution via deep recursive residual network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 3147–3155.

[46] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. 2017. Memnet: A persistent memory network for image restoration. In Proceedings of the IEEE International Conference on Computer Vision. 4539–4547.

[47] Yu-Wing Tai, Shuaicheng Liu, Michael S. Brown, and Stephen Lin. 2010. Super resolution using edge prior and single-image detail synthesis. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE, 2400–2407.

[48] Radu Timofte, Vincent De Smet, and Luc Van Gool. 2014. A+: Adjusted anchored neighborhood regression for fast super-resolution. In Proceedings of the Asian Conference on Computer Vision. Springer, 111–126.

[49] Tong Tong, Gen Li, Xiejie Liu, and Qinquan Gao. 2017. Image super-resolution using dense skip connections. In Proceedings of the IEEE Conference on Computer Vision. 4799–4807.

[50] Zhaowen Wang, Ding Liu, Jianchao Yang, Wei Han, and Thomas Huang. 2015. Deep networks for image super-resolution with sparse prior. In Proceedings of the IEEE International Conference on Computer Vision. 370–378.

[51] Lie Xu, Chiu-sing Choy, and Yi-Wen Li. 2016. Deep sparse rectifier neural networks for speech denoising. In Proceedings of the IEEE International Workshop on Acoustic Signal Enhancement (IWAENC’16). IEEE, 1–5. https://doi.org/10.1109/IWAENC.2016.7602891

[52] Bin-Cheng Yang. 2019. Super resolution using dual path connections. In Proceedings of the 27th ACM International Conference on Multimedia (MM’19). Association for Computing Machinery, New York, NY, 1552–1560. https://doi.org/10.1145/3343031.3350878

[53] Chih-Yuan Yang and Ming-Hsuan Yang. 2013. Fast direct super-resolution by simple functions. In Proceedings of the IEEE International Conference on Computer Vision. 561–568.

[54] Jianchao Yang, John Wright, Thomas S. Huang, and Yi Ma. 2010. Image super-resolution via sparse representation. IEEE Trans. Image Process. 19, 11 (2010), 2861–2873.

[55] Roman Zeyde, Michael Elad, and Matan Protter. 2010. On single-image scale-up using sparse-representations. In Proceedings of the International Conference on Curves and Surfaces. Springer, 711–730.

[56] Dongyang Zhang, Jie Shao, and Heng Tao Shen. 2020. Kernel attention network for single-image super-resolution. ACM Trans. Multimedia Comput., Commun., Appl. 16, 3 (2020), 1–15.

[57] Kai Zhang, Wangmeng Zuo, and Lei Zhang. 2018. Learning a single convolutional super-resolution network for multiple degradations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR’18). Computer Vision Foundation/IIEEE Computer Society, 3262–3271. https://doi.org/10.1109/CVPR.2018.00344

[58] Lei Zhang and Xiaolin Wu. 2006. An edge-guided image interpolation algorithm via directional filtering and data fusion. IEEE Trans. Image Process. 15 (2006), 2226–2238.

ACM Trans. Multimedia Comput. Commun. Appl., Vol. 19, No. 3, Article 115. Pub. date: February 2023.
[60] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. 2018. Image super-resolution using very deep residual channel attention networks. In Proceedings of the European Conference on Computer Vision (ECCV'18). 286–301.

[61] Yulun Zhang, Kunpeng Li, Kai Li, Bineng Zhong, and Yun Fu. 2019. Residual non-local attention networks for image restoration. Retrieved from https://arXiv:1903.10082.

[62] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. 2018. Residual dense network for image super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2472–2481.

[63] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. 2020. Residual dense network for image restoration. IEEE Trans. Pattern Anal. Mach. Intell. 43, 7 (2020), 2480–2495.

[64] Wilman W. W. Zou and Pong Chi Yuen. 2010. Very low resolution face recognition problem. In Proceedings of the 4th IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS'10). 1–6.

Received 26 December 2021; revised 13 June 2022; accepted 25 October 2022