Coarse-to-Fine CNN for Image Super-resolution

Chunwei Tian, Yong Xu*, Senior Member, IEEE, Wangmeng Zuo, Senior Member, IEEE, Bob Zhang, Senior Member, IEEE, Lunke Fei, Member, IEEE, and Chia-Wen Lin, Fellow, IEEE

Abstract—Deep convolutional neural networks (CNNs) have been popularly adopted in image super-resolution (SR). However, deep CNNs for SR often suffer from the instability of training, resulting in poor image SR performance. Gathering complementary contextual information can effectively overcome the problem. Along this line, we propose a coarse-to-fine SR CNN (CFSRCNN) to recover a high-resolution (HR) image from its low-resolution version. The proposed CFSRCNN consists of a stack of feature extraction blocks (FEBs), an enhancement block (EB), a construction block (CB) and, a feature refinement block (FRB) to learn a robust SR model. Specifically, the stack of FEBs learns the long- and short-path features, and then fuses the learned features by expending the effect of the shallower layers to the deeper layers to improve the representing power of learned features. A compression unit is then used in each FEB to distill important information of features so as to reduce the number of parameters. Subsequently, the EB utilizes residual learning to integrate the extracted features to prevent from losing edge information due to repeated distillation operations. After that, the CB applies the global and local LR features to obtain coarse features, followed by the FRB to refine the features to reconstruct a high-resolution image. Extensive experiments demonstrate the high efficiency and good performance of our CFSRCNN model on benchmark datasets compared with state-of-the-art SR models. The code of CFSRCNN is accessible on https://github.com/hellloxiaotian/CFSRCNN.

Index Terms—Image super-resolution, convolutional neural network, cascaded structure, feature fusion, feature refinement

I. INTRODUCTION

SINGLE image super-resolution (SISR) aims to reconstruct a high-resolution (HR) image from a given low-resolution (LR) image, which has been widely applied in many fields, such as visual analysis [1], medical images [2], and person identification [3, 4]. Since SISR is an ill-posed inverse problem, prior information is important to guarantee the quality of reconstructed SR image. For example, a set of patterns through Bayesian knowledge was learned for SISR in [5]. The method proposed in [6] utilizes sparse representation to predict the SR counterparts of LR patches. The random forest-based method proposed in [7] can directly map SR patches from LR patches to overcome the difficulty of training. In addition, non-local self-similarity (NLSS) [8], regression [9], dictionary learning [10], and gradient methods [11] were shown to be effective for SISR.

Although these SISR methods can achieve impressive performances, most of them still suffer from two drawbacks. First, they usually rely on complex optimization methods to recover HR images, which are time-consuming. Second, they need manually-tuned parameters to obtain a good performance of SISR, making them inflexible.

Deep learning techniques have found wide applications in low-level vision, such as image denoising [12], rain removal [13, 14], deblurring [15] and image SR [16]. Taking image SR as an example, the super-resolution CNN (SRCNN) proposed in [17] utilizes three convolutional layers to predict the HR image in a pixel-mapping manner, which, however, leads to slow convergence and large training cost. To break the bottleneck of the SRCNN, a very deep SR (VDSR) network [18] uses residual learning and small filter sizes to accelerate the speed of training while achieving good visual quality. Moreover, reducing the number of parameters is effective to overcome the difficulty in training a SR model. For example, the deeply-recursive convolutional network (DRCN) [19] and the deep recursive residual network (DRRN) [20] utilize recursive learning and residual learning techniques to improve training efficiency. Besides, using skip connections to fuse global and local features, such as the 30-layer convolutional residual encoder-decoder network (RED30) [21], is shown effective in enhancing the expressive ability of the SISR model. Moreover, the very deep persistent memory network (MemNet) in [22] applies recursive and gate units to mine useful features for some low-level tasks. However, very deep networks are not easy to train. Additionally, some of these methods perform bicubic interpolation to upscale a LR image to the same size as the HR image on SISR, which resulted in low efficiency for training [23]. Some existing methods only extract LR features for the SR task and magnify the obtained LR features in the final layer, which ignores the effect of HR features on SISR and can lead to the instability of training.

In this paper, we propose a coarse-to-fine super-resolution CNN (CFSRCNN) for SISR. It consists of a stack of feature extraction blocks (FEBs), an enhancement block (EB), a construction block (CB), and a feature refinement block
The combination of the stacked FEBs, EB and CB can more effectively make use of hierarchical LR features extracted from a LR image with much fewer parameters to enhance LR features and infer better initial HR features. Specifically, the stack of FEBs learns the long- and short-path features, and then fuses these features by expending the effect of the shallower layers to the deeper layers to improve the representing power of the learned features. A compression unit is then used in each FEB to distill important information of features so as to reduce the number of parameters. Subsequently, the EB utilizes residual learning to integrate the learned features to prevent from losing edge information due to repeated distillation operations. After that, the CB applies global and local LR features to obtain initial HR features, followed by the FRB to refine the HR features to reconstruct the final SR image. The proposed CFSRCNN has the following contributions.

1. We propose a cascaded network that combines LR and HR features to prevent possible training instability and performance degradation caused by upsampling operations.

2. We propose a novel feature fusion scheme based on heterogeneous convolutions to well resolve the long-term dependency problem and prevent information loss so as to significantly improve the efficiency of SISR without sacrificing the visual quality of reconstructed SR images.

3. The proposed network achieves both good performance and high computational efficiency for SISR.

The remainder of this paper is organized as follows. Section II provides related work. Section III presents the proposed method. Section IV shows extensive experimental results. Section V reports conclusion.

II. RELATED WORK

A. Deep CNNs based on cascaded structures for SISR

In image SR, using LR information, especially with a large upscaling factor to recover a HR image is very challenging. To address this problem, deep CNNs based on cascaded structures have been proposed to minimize the error between prediction results and their ground-truths. They can be divided into two categories in general. The first category applies bicubic interpolation to upscale a given LR image to the same size as the HR image, and then uses the upscaled image to predict a SR image. Specifically, a simple, effective, robust, and fast (SERF) method [24] cascades several linear least squares functions to extract effective features and then compresses the model in a coarse-to-fine manner. The second category gradually uses upsampling in different stages to predict the SR image. Specifically, the deep network cascade (DNC) in [25] magnifies a LR image layer by layer and utilizes NLSS in each sub-network to extract HR texture features. The cascaded multi-scale cross (CMSC) network in [26] cascades different sub-networks to obtain SR features. Then, it uses a reconstruction stage to fuse the obtained features in a weighted way to reconstruct a SR image. To reduce computation, the cascading residual network (CARN) in [27] cascades residual networks with small filter sizes to train a fast, accurate, and lightweight model. All the methods demonstrated the effectiveness of cascading operations in mitigating the discrepancy between a predicted SR image and its ground-truth.

B. Deep CNNs based on modules for SISR

Due to their flexible end-to-end architectures, CNNs have been widely adopted in many fields, i.e., image processing [29, 30], video surveillance [31] and speech processing [32], and text recognition [33]. To facilitate more features, CNNs based on modules are developed for image SR. Specifically, these methods can be divided into two categories in general: high accuracy (also referred to as performance) and efficiency.

For improving the accuracy of SR, fusing multiple features has been found useful in enhancing the expressive ability of a SR model. For example, the multi-scale dense network (MSDN) in [34] employs a multi-scale dense block to fuse intermediate features of different layers for SISR. Further, different views of an image are used as the inputs of a SR network to improve accuracy. The residual channel attention network (RCAN) in [35] utilizes residual channel attention blocks to mine and integrate different features from the channels of a LR image. The deep networks in [36, 37] fuse color and depth information of a given LR image to enhance the expressive ability of learned features. Besides, to make better use of hierarchical features, the residual dense network (RDN) in [38] combines local and global features via residual dense blocks to recover HR image details. Using residual learning techniques to learn hierarchical features is shown beneficial to build a depth map for SISR [39]. To obtain more detailed information, the SR CliqueNet (SRCliqueNet) in [40] utilizes a clique block and an upsampling module to extract textural details useful for SISR. The recursively dilated residual network in [16] increases the impact of local spatial information by using a recursion module, followed by a refinement module to learn more accurate LR features for recovering HR image details.

For improving efficiency, reducing the number of parameters is a common way. The information distillation network (IDN) in [41] applies a feature extraction block, a stacked information distillation block, and a reconstruction block to recover HR details. The stacked information distillation block exploits part convolutional filters of size 1 × 1 to compress the model for improving the speed of training. To reduce complexity and computational cost, the block state-based recursive network (BSRN) in [42], comprising an initial feature extractor, a recursive residual block and an upsampling part, increases the resolution of input images at the last stage to reduce the model complexity. That was also extended to a deeper or wider network, the convolutional anchored regression network (CARN) in [43] employs regression blocks to convert a LR input image to an other domain, according to the regression and similarity so as to achieve a better trade-off between speed and accuracy in contrast to other SISR methods. The Laplacian pyramid distillation network (LapIDN) in [44] first applies a Laplacian pyramid module to upscale extracted features gradually, followed by a distillation network to achieve high SR performance while compressing the network complexity. Moreover, there are other effective SISR methods, such as the lightweight feature fusion network (LFFN) in [45] that...
aggregates different hierarchical features in an adaptively convex weighed manner to control the number of parameters through a spindle block and a softmax feature fusion module, and the adaptive weighted SR network (AWSRN) in [46] that utilizes adaptive weighted residual unit and local residual fusion units, and an adaptive weighted multi-scale module to reduce parameters, according to the contribution of obtained features at different scales.

To achieve both high performance and efficiency in SISR, we propose a novel cascaded structure consisting of modular CNN blocks to learn accurate features.

III. PROPOSED METHOD

As shown in Fig. 1, our proposed CFSRCNN is composed of a stack of Feature Extraction Blocks (FEBs), an Enhancement Block (EB), a Construction Block (CB) and a Feature Refinement Block (FRB). The combination of the stacked FEBs, EB and CB can make use of hierarchical LR features extracted from the LR image with fewer parameters to enhance obtained LR features and derive coarse SR features. Specifically, combining an FEU and a CU into an FEB obtains long- and short-path features. Also, fusing the obtained features via the two closest FEUs can enlarge the effects of shallow layers on deep layers to improve the representing power of the SR model. The CU can distill more useful information and reduce the number of parameters. The EB fuses the features of all FEUs to offer complementary features for the stacked FEBs and prevent from the loss of edge information caused by the repeated distillation operations. Gathering several extra stacked FEUs into the EB removes over-enhanced pixel points from the previous stage of the EB. After that, the CB utilizes the global and local LR features to obtain coarse SR features. Finally, the FRB utilizes HR features to more effectively learn HR features and reconstruct a HR image. We introduce these techniques in the later sections.

A. Network architecture

The proposed 46-layer CFSRCNN is composed of four parts, a stacked FEBs, an EB, a CB and an FRB. Let \( I_{LR} \) and \( I_{SR} \) denote the input LR image and its corresponding SR output image of CFSRCNN, respectively. We divide the four blocks into two kinds, according to the obtained feature types (i.e., LR and HR features): the combination of the stack of FEBs, EB and CB, and FRB. For the combination of the three functional blocks, a 40-layer network is used to extract LR features from a given LR image and derive the coarse SR features, as formulated below:

\[
O_{CB} = f_{CB}(f_{EB}(f_{FEBs}(I_{LR}))),
\]

where \( f_{FEBs}, f_{EB} \) and \( f_{CB} \) denote the functions of the stack of FEBs, EB and CB, respectively. \( O_{CB} \) is the output of the the combination of the stack of FEBs, EB and CB. Specifically, \( O_{CB} \) is used as the input of a 6-layer FRB, which utilizes HR features to reduce the discrepancy between the predicted SR and target HR images, as formulated below:

\[
O_{FRB} = f_{FRB}(O_{CB}) = f_{CFSRCNN}(I_{LR}),
\]

where \( f_{FRB} \) and \( O_{FRB} \) denote the function and output of FRB, respectively. \( f_{CFSRCNN} \) is the function of the CFSRCNN. Also, \( O_{FRB} = I_{SR} \). Finally, CFSRCNN is optimized by the loss function that will be explained in Section III.B.

B. Loss function

We use a set of training pairs \( \{I_{LR}^j, I_{HR}^j\}_{j=1}^N \) to train the model, where \( N \) is the size of training set, and \( I_{LR}^j \) and \( I_{HR}^j \) denote the \( j \)-th LR and HR training images, respectively. We choose mean square error (MSE) [47] as the loss function to minimize the difference between the predicted SR and target HR images as follows:

\[
l(\theta) = \frac{1}{2N} \sum_{j=1}^{N} \left\| f_{CFSRCNN}(I_{LR}^j) - I_{HR}^j \right\|^2_2,
\]

where \( \theta \) denotes the parameter set of the trained model.

C. The combination of stacked FEBs, EB, and CB

The 40-layer block consists of 33-layer stacked FEBs, 6-layer EB and 1-layer CB. Specifically, FEBs aim to improve the efficiency and performance of SR by combining a FEU with a CU, where CU is used to distill more useful information. EB offers complementary features by fusing hierarchical LR features to address the loss of edge information from the CUs. Besides, EB also utilizes several additional stacked FEUs to learn finer LR features to mitigate over-enhanced pixel points caused by the previous stage of EB. Subsequently, CB can apply the global and local LR features to obtain coarse SR features. The detailed information is shown in later subsections.

1) Stacked FEBs: As illustrated in Fig. 2, the 33-layer stacked FEBs fuses obtained features from the pair of FEU and CU to enhance the performance and efficiency of the SR model. FEB of the stacked FEBs is composed of a pair of FEU and CU that respectively perform \( 3 \times 3 \) and \( 1 \times 1 \) heterogeneous convolutions, except for the last FEB that only has a FEU without CU. Each FEB (except the last one) first concatenates the output of the FEU from the previous FEB (aka the long-path features) and that of its own FEU (aka the short-path features) as the input of its own CU to enhance the representing power for SR. Subsequently, CU distills useful information from the enhanced LR features above, which can reduce the number of parameters and improve the training efficiency for a SR model. In practice, each FEU and CU comprise Conv+ReLU: a convolution filter followed by a rectified linear unit [48]. As mentioned above, the filter sizes of the FEU and CU are \( 3 \times 3 \) and \( 1 \times 1 \), respectively. The ReLU is used to non-linearly transform the fused features, which is then used as the input of the following CU. Further, the sizes of the first FEU and CU are \( 3 \times 3 \times 3 \times 64 \) and \( 4 \times 1 \times 1 \times 64 \), respectively, where \( 3 \) and \( 64 \) are the channel numbers of the input and output of the first FEU, respectively, and \( 64 \) is the channel number of the input and output of the first CU, respectively. The sizes of FEU and CU of the other FEBs are \( 64 \times 3 \times 3 \times 64 \) and \( 128 \times 1 \times 1 \times 64 \), respectively, where \( 64 \) represents the channel numbers of input and output except...
Fig. 1. Network architecture of CFSRCNN.

Fig. 2. Architecture of the CFSRCNN.
for the first FEU, and 128 and 64 are the channel numbers of input and output except the first CU, respectively. Due to the 1 × 1 convolution, CUs can better distill useful information as well as reduce the number of parameters [41], as will be shown in Section IV.D.

We further clarify the formulation of the stacked FEBs in details. Let \( O_{\text{CU}}^i \) and \( O_{\text{CU}}^{i+1} \) denote the output of the FEU and CU of the \( i \)-th FEB, respectively. According to the previous descriptions, the function of the \( i \)-th FEB can be expressed as

\[
O_{\text{CU}}^{i+1} = \text{CU}(\text{Cat}(\text{FEU}(O_{\text{CU}}^i), O_{\text{CU}}^{i-1})),
\]

where \( i = 2, 3, \ldots, 16 \) and \( \text{Cat} \) represents a concatenation operation. Specifically, the first FEB can be represented as \( O_{\text{CU}}^1 = \text{CU}(\text{FEU}(I_{\text{LR}})) \). Because the last FEB (the 17th FEB) only has a FEU, it can be formulated as

\[
O_{\text{CU}}^{17} = \text{Cat}(\text{FEU}(O_{\text{CU}}^{16}), O_{\text{CU}}^{16}),
\]

where \( O_{\text{sFEBS}} = O_{\text{CU}}^{17} \) and \( O_{\text{sFEBS}} \) is the output of the stacked FEBs.

2) Enhancement block: It is known that as the depth of the network increases, the extracted features are more accurate. However, this will also lead to the information loss of shallow layers [49]. Also, although the CUs can reduce the number of parameters and improve the efficiency for a SR model, repeated distillation operations lead to the loss of edge information extracted from the shallow layers. Taking into account these two factors, we propose a two-step feature enhancement block (EB) as demonstrated in Fig. 2. The first step of the EB, involving one Conv+ReLU with a filter size of \( 128 \times 3 \times 3 \times 64 \), gathers hierarchical features extracted by the FEUs of all FEBs through residual learning to offer features complementary to the stacked FEBs. The second step, involving five Conv+ReLU with the same filter size of \( 64 \times 3 \times 3 \times 64 \), fine-tunes the LR features to mitigate possible over-enhancement caused by the first step. The first-step operation of EB can be formulated as follows:

\[
O_{\text{EB}}^1 = \sum_{i=1}^{17} (\text{FEU}(O_{\text{sFEBS}}) + O_{\text{FEU}}^i),
\]

where \( O_{\text{EB}}^1 \) denotes the first-step output of EB. The EB’s second-step can be expressed as:

\[
O_{\text{EB}}^2 = \text{FEU}(\text{FEU}(\text{FEU}(\text{FEU}(O_{\text{EB}}))))),
\]

where \( O_{\text{EB}}^2 \) represents the final output of EB.

3) Construction block: It is known that a given LR image uses bicubic interpolation to obtain the same size as the HR image as the input of a SR network for training the model, which can result in high computational cost and memory consumption [23]. To address this problem, the up-sampling technique was proposed [50]. For example, a fast SR convolutional neural network (FSRCNN) [23] utilized deconvolution as the last layer to upscale the extracted LR features for SISR. FSRCNN directly utilized a LR image as input to extract LR features, then applied the deconvolution technique in the last layer to reconstruct the HR image. Although this approach can improve the efficiency of training, its performance in SISR is not satisfactory. In this paper, we use two steps in the construction block (CB) to overcome this problem. CB first performs a sub-pixel convolution with a filter size of \( 64 \times 3 \times 3 \times 64 \) to obtain global- and local-features. Subsequently, CB fuses the obtained features by residual learning to derive coarse HR features. The operation of CB can be formulated as follows:

\[
O_{\text{CB}} = S(O_{\text{FEU}}^1) + S(O_{\text{EB}}^2),
\]

where \( S \) and + stand for the functions of the sub-pixel convolution and residual learning, respectively. In addition, the residual learning operation is denoted as \( \oplus \) in Figs. 1 and 2.

D. Feature refinement block

Feature refinement block (FRB) is used to reduce the discrepancy between the predicted SR and ground-truth HR images. It involves five cascaded FEUs followed by a convolutional filter, where each FEU consists of Conv+ReLU with the same filter size of \( 64 \times 3 \times 3 \times 64 \), where 64 is the channel numbers of the input and output. This operation of cascaded FEUs is shown in Eq. (9).

\[
O_{\text{FRB}1} = \text{FEU}(\text{FEU}(\text{FEU}(\text{FEU}(\text{FEU}(O_{\text{CB}})))))
\]

where \( O_{\text{FRB}1} \) is the output of five cascaded FEUs of FRB, which is then filtered by the final convolution with a filter size of \( 64 \times 3 \times 3 \times 3 \) as follows:

\[
I_{\text{SR}} = C(O_{\text{FRB}1}),
\]

where \( C \) denotes the final convolution function.

IV. EXPERIMENTAL RESULTS

A. Training dataset

Following the state-of-the-art SR methods in [27, 51, 52], we utilize the public DIV2K dataset [53] to train our model. The DIV2K dataset contains 800 training images, 100 validation images, and 100 test images at three different scales: \( \times 2 \), \( \times 3 \), and \( \times 4 \). We merge the training and validation datasets of the DIV2K to expand the training dataset. Further, to improve the efficiency of model training, we divide each LR image into patches of size \( 77 \times 77 \). Besides, we use random horizontal flipping and \( 90^\circ \) rotation operations [27] to augment the training patches.

B. Testing datasets

We utilize five benchmark datasets, including Set5 [54], Set14 [6], BSD100 [55], Urban100 [56] and 720p, which are constructed at three different scales \( \times 2 \), \( \times 3 \), and \( \times 4 \), to evaluate the performance of the trained SR model. Set5 and Set14 respectively contain 5 and 14 images from different scenes, and BSD100 (a.k.a. B100) and Urban100 (a.k.a. U100) both consist of 100 images. The 720p dataset is composed of three typical images from clean images in the PolyU dataset [57], which are cropped to 1280 × 720.

Note, existing methods, such as DnCNN [58] and RED30 utilize YCbCr channel (also named Y channel) to conduct experiments. Thus, we covert the RGB image predicted by CFSRCNN into Y channel to evaluate the performance for SISR.
C. Implementation details

In the training process, the initial parameters are batch size of 64, beta_1 of 0.9, beta_2 of 0.999 and epsilon of 1e-8. All steps of the training are 6e+4 and halved every 4e+5 steps. Also, the initial weights and biases are the same as [27].

The propose CFSRCNN is implemented on Pytorch 0.41 and Python 2.7 for training and inference, respectively. Besides, all experiments are conducted on Ubuntu 16.04 on a PC equipped with an Intel Core i7-7800 CPU, 16G RAM, and two GPUs of Nvidia GeForce GTX 1080Ti with Nvidia CUDA 9.0, and CuDNN 7.5.

D. Network analysis

Extracting suitable features has been shown useful to accelerate the training process and improve performance in many image applications [59, 60]. To this end, feature extraction for a SR network usually involves HR features, LR features, and the combination of HR and LR features. For HR features, using bicubic interpolation to upscale a LR image as the input for training a SR model was popular [61]. This method, however, may lose some LR features, thereby resulting in performance degradation in SISR. Also, it leads to high computational cost as well. To address this issue, directly using LR features to train a SR model has been proposed. For example, FSRCNN [23] accelerates the SR task by utilizing only LR features prior to upsampling LR features in the last layer to obtain the HR image. However, as reported in [62], the upsampling operation may lead to a sudden shock to the model, which makes the training procedure unstable. To address the problem, an additional refinement process was found useful in [62] which refines the HR features obtained by the upsampling process by combining HR and LR information. Motivated by this, we also cascade a feature refinement block (FEB) to recover HR image details. Moreover, when the network depth goes deeper, the shallow-layer features would have weaker effect on deep-layer features. To address this problem, fusing hierarchical information has been proposed. Notably, the RDN in [38] fuses hierarchical non-linear features extracted from all convolutional layers via a residual dense block (see Fig. 3(a)) to enhance the memory ability of shallow layers. Besides, it employs global residual learning to learn global features complementary to local features obtained from the residual dense blocks, thereby achieving performance improvement in SISR. Moreover, the channel-wise and spatial feature modulation (CSFM) method proposed in [63] applies channel-wise and spatial attentions as block to extract hierarchical features and fuse them for enhancing the expressive ability of the SR model, as illustrated in Fig. 3(b). These methods achieve great performances for SISR. Similarly, our proposed CFSRCNN makes full use of hierarchical features to enhance LR features in the cascaded network for SISR. Nevertheless, as can be easily seen by comparing Figs. 1–2 and Fig. 3, CFSRCNN is different from the RDN and CSFM in the following aspects:

1) We concatenate the features of two neighboring FEBs, instead of using solely the current layer (used in RDB and CSFM), as the input of all the following layers to propagate the effect of shallower-layer features to deeper layers. Besides, we use a pair of heterogeneous ($3 \times 3$ and $1 \times 1$) convolutions with two layers, rather than stacking multiple ($3 \times 3$) convolutions, and fuse them into a block (FEB) to reduce the network depth and complexity (i.e., the numbers of parameters and ops). The above two changes together largely reduce the number of parameters to only 5.5% of RDB and 9.3% of CSFM, and the run-time, without severely sacrificing the visual quality of reconstructed SR images, as shown in the next subsection.

2) The Enhancement Block (EB) is not simply a concatenation of multiple FEUs. Instead, it adopts residual learning, rather than concatenating FEUs, to integrate the hierarchical LR features obtained from FEUs for enhancing the robustness of obtained LR features, which are complementary to the sFEBs. To prevent possible over-enhancement caused by previous operations, inspired by VDSR [18], we stack several convolutional layers to smooth out sharp features.

3) By gathering the local and global features via the residual learning and sub-pixel convolution to obtain coarse HR features, rather than using solely local features, CB can effectively address the long-term dependency problem.

4) Different from most existing learning-based SR methods which utilize LR features to train their models, we additionally make use of HR features to boost the SR performance via FRB that learns more accurate HR features by stacking multiple convolutional layers to reduce the discrepancy between the predicted SR image and its ground-truth. This also can enhance the stability of the training process.

CFSRCNN is composed of a number of stacked feature extraction blocks (sFEBs), an enhancement block (EB), a construction block (CB), and a feature refinement block (FRB). Such a combination efficiently makes full use of extracted hierarchical features to enhance the LR features, that are then used to infer initial SR features. The feature refinement block further employs HR features to learn more robust SR features to reduce the discrepancy between the predicted SR and its ground-truth. Thus, the combination of the sFEBs, EB, CB, and FRB can enhance the stability of SR model training. These key modules constituting CFSRCNN are elaborated below.

1) Stacked Feature Extraction Blocks (sFEBs): The sFEBs well collaborate with the EB and CB to extract more robust LR features for training a SR model, where each FEB includes an FEU and a CU. The design of FEB breaks two rules: less parameters and better performance for a SR model. For the first aspect, we use partial heterogeneous convolutions with $P = 2$ [64] to reduce computation and improve efficiency of training the SR network, where $P$ denotes part. These heterogeneous convolutions consist of 16 standard convolutions with size $3 \times 3$ and 16 small convolutions with size $1 \times 1$. The convolutions of size $3 \times 3$ and $1 \times 1$ are integrated into FEU and CU, respectively. Since the convolution of size $1 \times 1$ can remove redundant information and distill important features [41], the proposed CU can effectively reduce the number of parameters and improve the efficiency of training CFSRCNN.

The higher diversity the network architecture is, the better performance the SR model achieves as revealed in [38]. We therefore append a standard convolutional layer of $3 \times 3$ to heterogeneous convolutions. Given a LR input image, we
consolidate the sub-pixel layer to learn SR features after heterogeneous convolutions by using a standard convolution. To reconstruct the final SR image, we employ a standard convolution with size $3 \times 3$ as the final layer. The depth of heterogeneous convolutional network (HCN) is set to be 35. For fair comparison, we compare HCN with a standard convolutional network (SCN) of the same depth as that of HCN. Table I shows that HCN consumes significantly fewer computational cost and memory space than SCN for $\times 2$ upscaling. Besides, HCN is also more computationally efficient than SCN in run-time complexity as shown in Table II. These results verify that our method effectively reduces the number of parameters and complexity. Additionally, Table III show that HCN achieves the same Peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) performance as that of SCN on U100 for $\times 2$ upscaling.

Note, with the increase of network depth, shallow-layer features would make weaker effect on deep-layer ones, making a deep network suffer from SR performance degradation [49]. To address this problem, gathering hierarchical features can offer complementary contextual information from shallower layers to deeper layers [38]. Motivated by this, we propose a two-step enhancement mechanism in the sFEBs to enhance the expressive ability of the SR model. The first step utilizes the FEUs and CUs from two contiguous FEBs to extract long- and short-path features, respectively. The second step exploits a concatenation operation to fuse the extracted long- and short-path features to address the long-term dependency problem. Further, the effectiveness of the two-step enhancement mechanism for SR is illustrated in Table III, where the sFEBs with long and short-path achieve higher PSNR and SSIM than that of HCN without long and short-path on B100 and U100, showing the effectiveness of the combination of long- and short-path.

2) Enhancement Block (EB): Although a convolution of size $1 \times 1$ can distill useful features [41], repeated distillation operations may lead to information loss of the original images. To address this issue, a two-step enhancement mechanism has been adopted in the sFEBs. However, the two-step enhancement mechanism can fuse long- and short-path features by concatenating their half feature points to enhance the generalization ability of a SR model. The features extracted from the deep layers only inherit partial features from shallow layers, which cannot completely address the above-mentioned problem. To better handle this problem, we propose a two-phase EB. The first phase of EB (named EB1) applies the residual learning technique to integrate hierarchical features of all FEUs to offer complementary features for the sFEBs. The second phase uses several additional stacked FEUs to refine the learned LR features, which can mitigate over-enhanced pixels from EB1. The two phases not only provide extra information for SISR, but also enhance the diversity of the network architecture. These are useful to recover a latent HR image as verified in Table III. That is, the ‘combination of stacked FEUs and EB1’ outperforms sFEBs in PSRN and SSIM on B100 and U100, showing the effectiveness of EB1 for SISR. ‘The combination of stacked FEUs and EB’ achieves higher PSNR and SSIM performances than ‘The combination of stacked FEUs and EB1’, implies that the second phase of EB improves SR performance.

3) Construction Block (CB): Using bicubic interpolation to upscale a given LR image as the input of a SR model leads to high computational cost and memory consumption [23]. Taking this into consideration, we use the sub-pixel technique to magnify the obtained LR features as SR features, which can improve the training efficiency by reducing the computational cost and memory consumption. However, local LR features benefit from multiple convolutions may ignore some useful information in the original LR images, making the extracted SR features anemic. It is known that global features are complementary with local features to promote the expressive ability for SISR [38]. Inspired by that, we utilize the residual learning technique to fuse global and local
features for enhancing the robustness of the extracted SR features as follows. First, the outputs of the FEU from the first FEB and the EB are treated as global and local SR features, respectively. The global and local features are then upsampled by the sub-pixel technique as the global and local SR features, respectively. Second, residual learning is employed to fuse the obtained global and local SR features to derive the coarse SR features. This phase makes full use of global and local information from LR and SR features to improve the expressive ability of a SR model, where is tested by Table III. That is, ‘The combination of sFEBs, EB and CB’ has better results of PSNR and SSIM on B100 and U100 than that of ‘The combination of sFEBs and EB’ in SISR.

4) Feature Refinement Block (FRB): The combination of the sFEBs, EB and CB mainly extracts LR features to learn coarse SR features, which, however, cannot fully characterize the latent HR image. Also, the training of a SISR model is unstable. To tackle these problems, a six-layer FRB is proposed to refine SR features. That is, FRB can use HR features to refine SR features so as to reduce the discrepancy between the predicted SR and its ground-truth, which is complementary with the combination of the sFEBs, EB and CB. As a consequence, combining the sFEBs, EB and CB can enhance the stability of training the SR model. This fact is verified by comparing ‘CFSRCNN’ with ‘The combination of sFEBs, EB and CB’ about PSNR and SSIM on U100 and B100 in Table III. Also, we employ FRB in the LR space (called FRNet) to extract features with approximately the same number of parameters to validate the effectiveness of FRB as shown in Table III. However, solely using FRB in the LR space behaves like a VGG network with a very deep depth that usually leads to gradient vanishing/explosion, thereby significantly degrading performance. Since such validation method does not provide meaningful comparison, we do not include it in the comparison. Finally, we use a convolution of size $64 \times 3 \times 3 \times 3$ as the final layer to reconstruct a SR image.

E. Comparisons with state-of-the-arts

We conduct quantitative and qualitative analyses to evaluate the performance of CFSRCNN for SISR. Specifically, we evaluate quantitatively the average PSNR and SSIM performances and the run-time and model complexities of various SR methods on five benchmark datasets: Set5, Set14, B100, U100 and 720p. The compared methods include Bicubic, A+ [9], RFL [7], self-exemplars SR method (SelfEx) [56], cascade of sparse coding based network (CSCN) [67], RED30 [21], DnCNN [58], trainable nonlinear reaction diffusion (TNRD) [68], fast dilated SR method (FDSR) [69], SRCNN [17], FSRCNN [23], response context sub-network (RCN) [61], VDSR [18], DRCN [19], context-wise network fusion (CNF) [51], Laplacian SR network (LapSRN) [70], DRRN [20], balanced two-stage residual networks (BTSRN) [71], MemNet [22], CARN-M [27], CARN [27], end-to-end deep and shallow network (EEDS)+ [72], two-stage convolutional network (TSCN) [73], deep recurrent fusion network (DRFN) [74], RDN [38], CSFM [63], and super-resolution feedback network (SRFBN) [75]. We also demonstrate a few reconstructed SR images for subjective visual comparisons.

The average PSNR and SSIM performances of various SR methods on the Set5, Set14, B100, U100 and 720p datasets are demonstrated in Tables IV–VIII, respectively. As shown in Table IV, our CFSRCNN with scaling factors of $\times 3$ and $\times 4$ outperforms the state-of-the-art SR methods, such as DRFN, TSCN, EEDS+ and CARN-M on Set5, and achieves a comparable performance with that of CARN-M for $\times 2$ upscaling. Specifically, compared with CARN-M, CFSRCNN achieves a notable gain of 0.14dB in PSNR for $\times 4$ upscaling. Moreover, as shown in Tables V–VIII, CFSRCNN achieves excellent performances for all the three scaling factors: $\times 2$, $\times 3$ and $\times 4$. For example, Table V shows that, CFSRCNN outperforms MemNet by 0.23dB, 0.27dB, and 0.31dB in PSNR on Set14 for $\times 2$, $\times 3$, and $\times 4$ upscaling, respectively. Similarly, Table VI also shows that CFSRCNN outperforms several popular SR methods, such as CARN-M, TSCN and DRFN. As illustrated in Table VII, CFSRCNN performs stably well.

Figs. 4–6 compare the subjective visual quality of CFSRCNN with that of four SR methods, including Bicubic, SRCNN, SelfEx, and CARN-M for $\times 2$ upscaling on Set14, $\times 3$ upscaling on B100 and $\times 4$ upscaling on U100, respectively. To facilitate subjective comparison visually, we enlarge selected regions in the SR images, showing that the images super-resolved by CFSRCNN are clearer than those super-resolved by the other methods for $\times 2$, $\times 3$, and $\times 4$ upscaling.
We then compare the run-time complexity of CSFCRN with that of six methods, including VDSR, DRRN, MemNet, CARN-M, and CARN-M, on HR images of sizes 256×256, 512×512 and 1024×1024 for ×2 upsampling. Table IX shows that CSFCRN achieves the fastest processing speed. Besides run-time, we also evaluate the number of parameters and flops [76] in Table X that reflect the SR model complexity (i.e., computational cost and memory consumption) for SR methods for ×2, ×3, and ×4 upsampling on U100.

RDN, SRFBN, and CARN-M, on HR images of sizes 256×256, 512×512 and 1024×1024 for ×2 upsampling. Table IX shows that CSFCRN achieves the fastest processing speed. Besides run-time, we also evaluate the number of parameters and flops [76] in Table X that reflect the SR model complexity (i.e., computational cost and memory consumption) for SR methods for ×2, ×3, and ×4 upsampling on U100.
images of size $154 \times 154$. Table X shows that CFSRCNN consumes the third fewest number of flops while faithfully reconstructing high-quality SR images. Due to its shallower architecture and fewer concatenation operations, CFSRCNN does not outperform some deeper SR networks, such as RDN, CSFM and SRFBM. However, CFSRCNN offers an excellent trade-off among visual quality, computational efficiency, and model complexity.

V. Conclusion

In this paper, we proposed a coarse-to-fine super-resolution CNN (CFSRCNN) for single-image super-resolution. CFSRCNN combines low-resolution and high-resolution features by cascading several types of modular blocks to prevent possible training instability and performance degradation caused by upsampling operations. We have also proposed a novel feature fusion scheme based on heterogeneous convolutions to address the long-term dependency problem as well as prevent information loss so as to significantly improve the computational efficiency of super-resolution without sacrificing the visual quality of reconstructed images. Comprehensive evaluations on four benchmark datasets demonstrate that CFSRCNN offers an excellent trade-off among visual quality, computational efficiency, and model complexity.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Model</th>
<th>$\times 2$</th>
<th>$\times 3$</th>
<th>$\times 4$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR/SSIM</td>
<td>PSNR/SSIM</td>
<td>PSNR/SSIM</td>
<td></td>
</tr>
<tr>
<td>720p</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CARN-M [27]</td>
<td>43.62/0.9791</td>
<td>39.87/0.9602</td>
<td>37.61/0.9389</td>
<td></td>
</tr>
<tr>
<td>CARN [27]</td>
<td>44.57/0.9809</td>
<td>40.66/0.9633</td>
<td>38.03/0.9429</td>
<td></td>
</tr>
<tr>
<td>CFSRCNN (Ours)</td>
<td>44.77/0.9811</td>
<td>40.93/0.9656</td>
<td>38.34/0.9482</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Single Image Super-Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
</tr>
<tr>
<td>VDSR [18]</td>
</tr>
<tr>
<td>DRCN [19]</td>
</tr>
<tr>
<td>MemNet [22]</td>
</tr>
<tr>
<td>RDN [38]</td>
</tr>
<tr>
<td>SRFBN [75]</td>
</tr>
<tr>
<td>CARN-M [27]</td>
</tr>
<tr>
<td>CFSRCNN (Ours)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Methods</th>
<th>Parameters</th>
<th>Flops</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDSR [18]</td>
<td>665K</td>
<td>15.82G</td>
</tr>
<tr>
<td>DRCN [58]</td>
<td>556K</td>
<td>13.20G</td>
</tr>
<tr>
<td>MemNet [22]</td>
<td>677K</td>
<td>16.06G</td>
</tr>
<tr>
<td>CARN-M [27]</td>
<td>412K</td>
<td>2.50G</td>
</tr>
<tr>
<td>CARN [27]</td>
<td>1392K</td>
<td>10.13G</td>
</tr>
<tr>
<td>CSFM [63]</td>
<td>12341K</td>
<td>76.32G</td>
</tr>
<tr>
<td>RDN [38]</td>
<td>21.937K</td>
<td>130.75G</td>
</tr>
<tr>
<td>SRFBN [75]</td>
<td>3.631K</td>
<td>22.24G</td>
</tr>
<tr>
<td>CFSRCNN (Ours)</td>
<td>1.200K</td>
<td>11.08G</td>
</tr>
</tbody>
</table>

Fig. 6. Subjective visual quality comparison of various SR methods for $\times 4$ upsampling on U100: (a) HR image (PSNR/SSIM), (b) Bicubic (22.10/0.7862), (c) SRCNN (26.08/0.8547), (d) SelfEx (28.02/0.9026), (e) CARN-M (31.80/0.9324) and (f) CFSRCNN (33.21/0.9377).

### References


This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2020.2999182, IEEE Transactions on Multimedia


Fig. 5. Subjective visual quality comparison of various SR methods for \( \times 3 \) upscaling on B100: (a) HR image (PSNR/SSIM), (b) Bicubic (25.52/0.7731), (c) SRCNN (26.58/0.8217), (d) SelfEx (27.32/0.8424), (e) CARN-M (27.90/0.8626) and (f) CFSRCNN (28.56/0.8732).


Wangmeng Zuo (Senior Member, IEEE) received the Ph.D. degree in computer application technology from the Harbin Institute of Technology, Harbin, China, in 2007. He is currently a Professor in the School of Computer Science and Technology, Harbin Institute of Technology. His current research interests include image enhancement and restoration, object detection, visual tracking, and image classification. He has published over 70 papers in top-tier academic journals and conferences. He has served as a Tutorial Organizer in ECCV 2016, an Associate Editor of the IET Biometrics and Journal of Electronic Imaging, and the Guest Editor of Neurocomputing, Pattern Recognition, IEEE Transactions on Circuits and Systems for Video Technology, and IEEE Transactions on Neural Networks and Learning Systems.

Bob Zhang (Senior Member, IEEE) received the B.A. degree in computer science from York University, Toronto, ON, Canada, in 2006, the M.A.Sc. degree in information systems security from Concordia University, Montreal, QC, Canada, in 2007, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2011. He is currently an Associate Professor with the Department of Computer and Information Science, University of Macau, Macau, China. His current research interests include medical biometrics, pattern recognition, and image processing.

Chia-Wen Lin (Fellow, IEEE) received the Ph.D. degree in electrical engineering from National Tsing Hua University (NTHU), Hsinchu, Taiwan, in 2000. He is currently Professor with the Department of Electrical Engineering and the Institute of Communications Engineering, NTHU. He is also Deputy Director of the AI Research Center of NTHU. His research interests include computer vision and image and video processing. He has served as a Distinguished Lecturer for IEEE Circuits and Systems Society from 2018 to 2019, a Steering Committee member for the IEEE TRANSACTIONS ON MULTIMEDIA from 2014 to 2015, and the Chair of the Multimedia Systems and Applications Technical Committee of the IEEE Circuits and Systems Society from 2013 to 2015. His articles received the Best Paper Award of IEEE VCIP 2015, Top 10% Paper Awards of IEEE MMSP 2013, and the Young Investigator Award of VCIP 2005. He has been serving as President of the Chinese Image Processing and Pattern Recognition Association, Taiwan. He served as a General Co-Chair of IEEE VCIP 2018, a Technical Program Co-Chair of IEEE ICME 2010, and a Technical Program Co-Chair of IEEE ICIP 2019. He has served as an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, the IEEE TRANSACTIONS ON MULTIMEDIA, IEEE MULTIMEDIA, and the Journal of Visual Communication and Image Representation.