MDCN: Multi-scale Dense Cross Network for Image Super-Resolution

Juncheng Li, Faming Fang, Jiaqian Li, Kangfu Mei, and Guixu Zhang

***Abstract*—Convolutional neural networks have been proven to be of great benefit for single-image super-resolution (SISR). However, previous works do not make full use of multi-scale features and ignore the inter-scale correlation between different upsampling factors, resulting in sub-optimal performance. Instead of blindly increasing the depth of the network, we are committed to mining image features and learning the inter-scale correlation between different upsampling factors. To achieve this, we propose a Multi-scale Dense Cross Network (MDCN), which achieves great performance with fewer parameters and less execution time. MDCN consists of multi-scale dense cross blocks (MDCBs), hierarchical feature distillation block (HFDB), and dynamic reconstruction block (DRB). Among them, MDCB aims to detect multi-scale features and maximize the use of image features flow at different scales, HFDB focuses on adaptively recalibrate channel-wise feature responses to achieve feature distillation, and DRB attempts to reconstruct SR images with different upsampling factors in a single model. It is worth noting that all these modules can run independently. It means that these modules can be selectively plugged into any CNN model to improve model performance. Extensive experiments show that MDCN achieves competitive results in SISR, especially in the reconstruction task with multiple upsampling factors. The code is provided at https://github.com/MIVRC/MDCN-PyTorch.**

***Index Terms*—Single image super-resolution, multi-scale, feature distillation, dynamic reconstruction.**

I. Introduction

IMAGE super-resolution, especially single image super-resolution (SISR) is an extremely hot topic in the computer vision field, which aims to reconstruct a super-resolution (SR) image from its degraded low-resolution (LR) one. In order to generate high-quality SR images, plenty of SR methods have been proposed, including interpolation-based [1]–[4], anchored neighborhood regression based [5], [6], self-example learning based [7]–[9], and learning based [10], [11] methods.

Recently, convolutional neural networks (CNNs) have achieved great success in computer vision tasks, which also profoundly promote the development of SISR. Therefore, CNN-based SR methods [10]–[31] have become the mainstream today, which aim to learn the mapping between LR and HR images by constructing a well-designed network. Among them, Dong et al. [10] proposed the SRCNN, which was the first successful model adopting CNN to the SISR task. After that, Kim et al. [11] introduced residual learning into SISR and proposed the VDSR; Lim et al. [16] introduced residual block and multi-factor strategy to construct EDSR and MDSR models. Li et al. [32] proposed the MSRN by introducing multi-scale residual learning; Zhang et al. [33] introduced the attention mechanism into residual blocks to construct a very deep RCAN; He et al. [22] presented a cascaded network (CDN_MRF) with multi-receptive fields to increase the spatial resolution; Li et al. [26] proposed a deep adaptive information filtering network (FilterNet) for accurate and fast SR image reconstruction; Apart from these models, more CNN-based SR models can be found in [34] and [35].

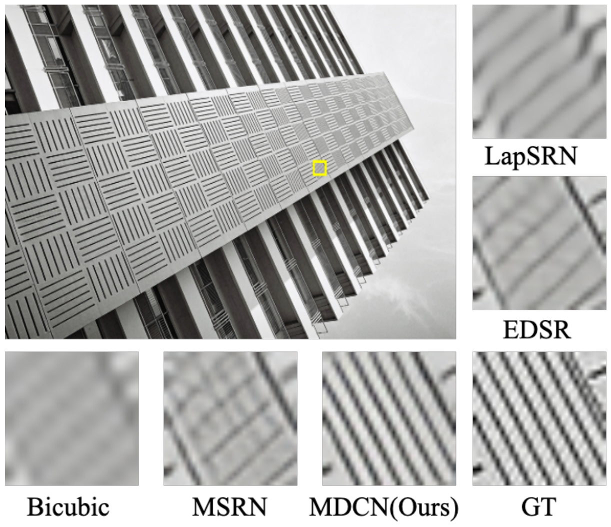


Fig. 1. Visual comparisons of $\times 3$ SR on a challenging image. The SR images reconstructed by other models have been severely blurred, but our MDCN can suppress artifacts and reconstruct clear texture details.

Although SR models mentioned above can achieve great results (Fig. 1), they are usually accompanied by complicated structures and huge computational overhead. Among all of them, MSRN [32] is the focus of this paper. MSRN [32] was proposed in 2018, which used multi-scale features to reconstruct SR images. To achieve this, the multi-scale residual block (MSRB) was proposed for feature extraction. Now, many works [36]–[45] have proved the effectiveness of multi-scale residual learning strategy and multi-scale features have been used in various fields, such as image denoising [45], remote sensing image super-resolution [41], [42], and medical image enhancement [43], [44]. These models directly migrate

This work was supported by the Key Project of the National Natural Science Foundation of China (NSFC) under Grant 61731009, the NSFC-RGC under Grant 61961160734, and the NSFC under Grant 61871185. Corresponding author: Guixu Zhang.

J. Li, F. Fang, J. Li, and G. Zhang are with the Shanghai Key Laboratory of Multidimensional Information Processing, East China Normal University, Shanghai 200062, China, and also with the School of Computer Science and Technology, East China Normal University, Shanghai 200062, China. E-mail: cvjunchengli@gmail.com, {fmfang, gxzhang}@cs.ecnu.edu.cn

K. Mei is with the Department of Mathematics, The Chinese University of Hong Kong (ShengZheng), ShengZheng, China.

MSRN to other tasks or by optimizing the model structure of MSRN to improve model performance. Although these models greatly improved the architecture and application of MSRN, none of them solved the core problem of MSRN. In this work, we aim to propose a more efficient and general model.

(A). Multi-scale Feature Extraction: Massive studies [46]–[49] have pointed out that image will exhibit different characteristics at different scales and making full use of these features can further improve model performance. Therefore, Li et al. [32] proposed a MSRB for multi-scale feature extraction, which integrated different convolutional kernels in a block to adaptively extract image features at different scales. However, it does not make full use of image features in the previous layers, so local features are difficult to transfer to other layers and the multi-scale feature flow will be suppressed. Therefore, exploring a more efficient multi-scale feature extraction block is essential for image reconstruction.

(B). Hierarchical Feature Distillation: As the depth of the network increases, image features will gradually disappear during the conduction process. Therefore, taking advantage of hierarchical features will greatly improve model performance. However, most models ignore this or simply concatenate all hierarchical features. These methods cannot eliminate redundant features, resulting in sub-optimal results and inefficient image reconstruction. Therefore, an effective method that can exploit hierarchical features and eliminate redundant features is crucial for SISR.

(C). Inter-scale Correlation Exploration: Most SR models introduce the deconvolutional layer or sub-pixel convolutional layer to achieve image magnification. However, due to their characteristics, these models need to train specific models for different upsampling factors. Although some models introduce simple and flexible reconstruction modules, they still fail to achieve the expectation that a model can be suitable for multiple upsampling factors. Therefore, a model that can adapt to multiple factors and learn the inter-scale correlation between different upsampling factors is needed.

To solve these issues, we propose a Multi-scale Dense Cross Network (MDCN). MDCN consists of multi-scale dense cross blocks (MDCBs), hierarchical feature distillation block (HFDB), and dynamic reconstruction block (DRB). Among them, MDCB is an efficient feature extraction module that can extract rich high-frequency details through the integrated dual-path dense network and multi-scale learning. HFDB introduces dimension transformation and channel attention mechanism to adaptively recalibrate channel-wise feature responses, thus redundant hierarchical features will be removed. DRB aims to maximize the reuse of model parameters and learn the inter-scale correlation between different upsampling factors by dynamic activating the corresponding upsampling module. In summary, our contributions are as follows:

(i) We propose a Multi-scale Dense Cross Network (MDCN) for SISR, which achieves competitive results with fewer parameters and less execution time.

(ii) We devise a Multi-scale Dense Cross Block (MDCB) for feature extraction, which is essentially a dual-path dense network that effectively detects local and multi-scale features.

(iii) We design a Hierarchical Feature Distillation Block (HFDB) to maximize the use of hierarchical features. It is the first CNN module specifically designed for hierarchical feature learning.

(iv). We introduce the Dynamic Reconstruction Block (DRB) to learn the inter-scale correlation between different upsampling factors, which enables MDCN to reconstruct SR images with multiple factors in a single model.

TABLE I
COMPARISONS OF DIFFERENT FEATURE EXTRACTION BLOCKS. √ AND × INDICATE WHETHER THE BLOCK USES THE CORRESPONDING MECHANISM.

Block	Residual Learning	Dense Connection	Multi-scale Learning
Residual Block [16]	√	×	×
Dense Block [18]	×	√	×
RDB [50]	√	√	×
MSRB [32]	√	×	√
MDCB (Ours)	√	√	√

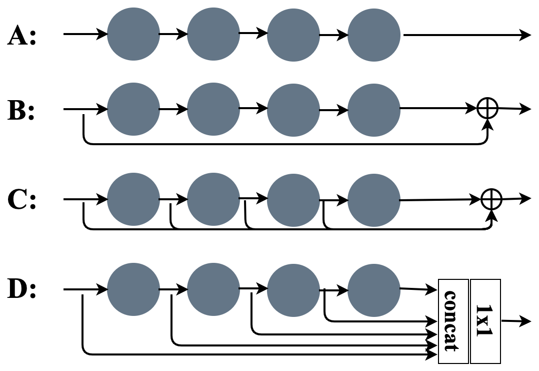


Fig. 2. Comparisons of different hierarchical feature utilization methods.

II. Related Work

SISR has attracted increasing attention in recent years since the quality of reconstructed SR images will seriously affect the accuracy of high-level computer vision tasks such as image classification [51], [52], image segmentation [53], [54], and object detection [55], [56]. In all relevant studies, feature extraction and multi-factor model have attracted our attention.

Feature Extraction: Recently, numerous feature extraction blocks have been proposed for local feature extraction. In TABLE I, we provide several classic feature extraction blocks, including Residual Block [57], Dense Block [58], Residual Dense Block (RDB [50]), and Multi-scale Residual Block (MSRB [32]). According to the table, we can clearly see the mechanism introduced in each block. Different from previous feature extraction blocks, MSRB introduces a new multi-scale learning strategy. Although this allows MSRB to obtain multi-scale features, it ignores the use of features in the previous layers. In this paper, by rethinking the influence of residual learning, dense connection, and multi-scale learning, we aim to explore a new feature extraction block that can skillfully combine these strategies without adding additional parameters.

On the other hand, as the depth of the network increases, image features will be gradually lost in the process of transmission. Therefore, making full use of hierarchical features is also important for image restoration. In Fig. 2, we show four

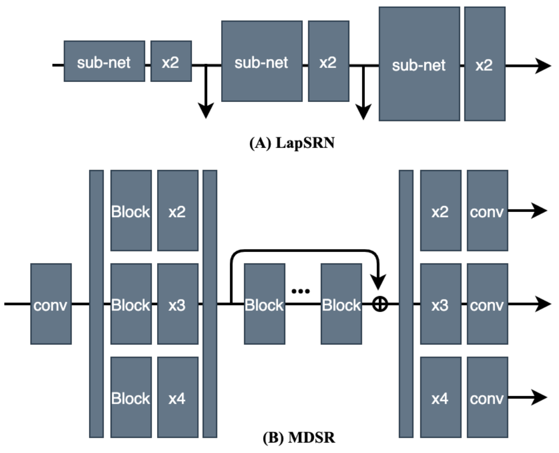


Fig. 3. The main structure of LapSRN [15] and MDSR [16].

different methods, including the series connection (A, without using hierarchical feature), residual connection (B, only use the input image features), dense residual connection (C, use all hierarchical features for residual learning), and hierarchical connection (D, concatenate all hierarchical features and apply a 1×1 layer for feature fusion). However, by investigating the performance of methods A-D, we find that hierarchical features can improve the model performance while redundant hierarchical features will make the model more difficult to train. Therefore, a new method that can effectively utilize hierarchical features and eliminate redundant features is crucial.

Multi-factor Model: A model that can be applied to multiple upsampling factors is the hot topic in SISR, which is the basic condition for inter-scale correlation learning. Nonetheless, due to the limitations of deconvolutional layer and sub-pixel convolutional layer, there is still no perfect solution for this problem. Recently, some models have been proposed for multiple upsampling factors, such as LapSRN [15] and MDSR [16]. LapSRN [15] (Fig. 3 (A)) is a Laplacian pyramid framework, which can progressively reconstruct high-resolution images. However, this method essentially decomposes the large upsampling factor into multiple small upsampling factors for image reconstruction, which does not consider the inter-scale correlation between different upsampling factors. MDSR [16] (Fig. 3 (B)) introduces the scale-specific processing modules at the head and tail of the model to handle different upsampling factors. This structure can be suitable for multiple upsampling factors, but the scale-specific processing modules at the head of the model are not conducive to the transmission of information and the interaction of inter-scale features. In order to maximize the reuse of model parameters and further exploit the inter-scale correlation between different upsampling factors, we aim to further optimize the strategy used in MDSR and propose a more efficient method.

Notice that some papers term the models that can be used for multiple upsampling factors as *multi-scale network*. In this paper, we name this type of model as *multi-factor model* to distinguish it from the multi-scale feature extraction block.

III. Multi-scale Dense Cross Network (MDCN)

In this paper, we propose a Multi-scale Dense Cross Network (MDCN) for SISR. As shown in Fig. 4, MDCN can be divided into two stages: feature extraction and dynamic reconstruction. In stage I, we use N multi-scale dense cross blocks (MDCBs) and a hierarchical feature distillation block (HFDB) for local feature extraction and hierarchical features distillation, respectively. In stage II, we introduce a dynamic reconstruction block (DRB) for SR image reconstruction. This block allows the model can be suitable for multiple upsampling factors, which makes MDCN more scalable.

Define I_{LR} and I_{SR} as the input and output of MDCN. L_{input}, L_{output}, and L_{mix} are the input of the first MDCB, the output of the last MDCB, and the input of DRB, respectively. Following previous works, we first use a 3×3 convolutional layer to upgrade the LR image to a high dimensional

$$L_{\text{input}} = F_{\text{input}}(I_{\text{LR}}), \tag{1}$$

where $F_{\text{input}}(\cdot)$ denotes the corresponding convolutional layer and L_{input} is the converted features. Meanwhile, L_{input} is also served as the input of MDCB for local feature extraction

$$L_{\text{output}} = F_{\text{MDCG}}(L_{\text{input}}), \tag{2}$$

where $F_{\text{MDCG}}(\cdot)$ represents the multi-scale dense cross group (MDCG), which consists of N MDCBs. To make full use of the hierarchical features, we also introduce the HFDB for hierarchical feature fusion and distillation

$$L_{\text{dis}} = F_{\text{HFDB}}([L^{1}_{\text{hie}}, L^{2}_{\text{hie}}, ..., L^{N-1}_{\text{hie}}]), \tag{3}$$

where $F_{\text{HFDB}}(\cdot)$ denotes the HFDB and L^{n}_{hie} $(n = 1, 2, ..., N-1)$ represents the output of the n-th MDCB. In addition, L_{dis} denotes the distilled hierarchical features, which can be used for image reconstruction.

After feature extraction, we combine the original image feature L_{input}, the extracted high-level features L_{output}, and the distilled hierarchical features L_{dis} for the final SR image reconstruction

$$L_{\text{mix}} = F_{\text{mix}}(L_{\text{input}} + L_{\text{dis}} + L_{\text{output}}), \tag{4}$$

where $F_{\text{mix}}(\cdot)$ is a 3×3 convolutional layer used for feature fusion. Finally, the merged features are delivered to the DRB for high-quality SR image reconstruction and a 3×3 convolutional layer is applied to convert it to RGB space

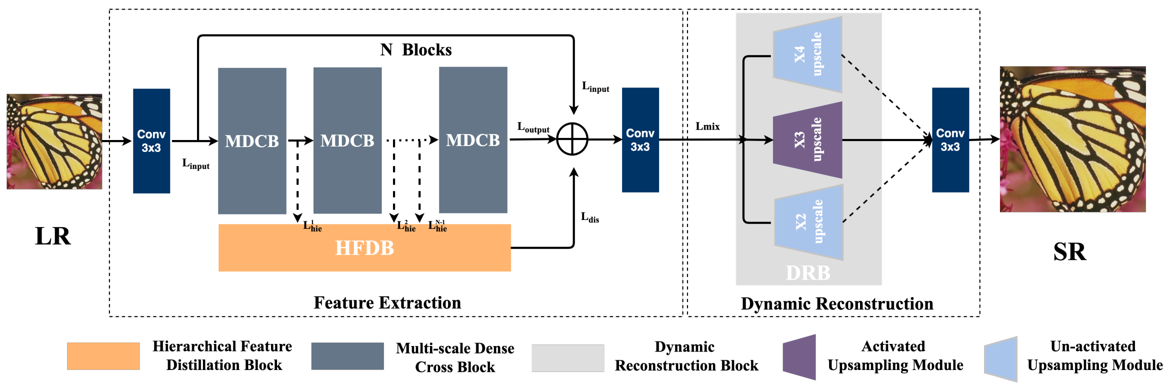
$$I_{\text{SR}} = F_{\text{output}}(F_{\text{DRB}}(L_{\text{mix}})), \tag{5}$$

where $F_{\text{DRB}}(\cdot)$ denotes the dynamic reconstruction block, $F_{\text{output}}(\cdot)$ represents the corresponding operation that converts image feature maps from high dimensional space to RGB space, and I_{SR} is the finally reconstructed SR image.

During training, MDCN is optimized with L1 loss function. Therefore, Given a training dataset $\left\{I^{i}_{\text{LR}}, I^{i}_{\text{HR}}\right\}^{M}_{i=1}$, we solve

$$\hat{\theta} = arg \min_{\theta} \frac{1}{M} \sum_{i=1}^{M} \left\| F_{\theta}(I^{i}_{\text{LR}}) - I^{i}_{\text{HR}} \right\|_{1}, \tag{6}$$

where θ denotes the parameter set of our model and $F(\cdot)$ denotes the MDCN. Each module of MDCN will be described in the following section.



N Blocks
Conv 3x3
MDCB
MDCB
MDCB
HFDB
LR
SR
Feature Extraction
Dynamic Reconstruction
DRB
X4 upscale
X3 upscale
X2 upscale
Lmix
Hierarchical Feature Distillation Block
Multi-scale Dense Cross Block
Dynamic Reconstruction Block
Activated Upsampling Module
Un-activated Upsampling Module

Fig. 4. The complete architecture of our proposed Multi-scale Dense Cross Network (MDCN), which consists of two stages: feature extraction and dynamic reconstruction. The dark grayish block, orange block, and gray block denote the MDCB, HFDB, and DRB, respectively.

A. Multi-scale Dense Cross Block (MDCB)

Features extraction is the most important step in image restoration. In order to make full use of image features from the LR image, we propose a new module named multi-scale dense cross block (MDCB). MDCB is the basic component of MDCN, which is inspired by MSRB [32] and aims to solve the problem that MSRN can not obtain features from the previous layers. Compared with MSRB [32], MDCB adopts a dual dense network as the backbone, which enables it to extract image features at different scales as well as take advantage of features from the previous layers.

Dual-path Network: As shown in Fig. 5, MDCB is essentially a dual-path network, which uses two dense networks as the backbone. In order to better explain the working mechanism of the module, we provide the decomposed structure of MDCB in Fig. 6. In Fig. 6, (A) is the simplified version of MDCB, which removes residual learning for better explanation, (B) is the decomposed structure of MDCB, and (C) is the equivalent structure after straightening (B). As shown in Fig. 6 (C), the model can be divided into two parts: DenseNet-Top and DenseNet-Bottom. Each part is actually a modified dense network [58], which reduces the depth of the network and introduces bottleneck layers for features fusion. Following the dense network, our Dense-Top or DenseNet-Bottom connects each layer to every other layers in a feed-forward fashion (red lines), which can detect rich features in the previous layers. The operations of Dense-Top are:

$$L_{22} = C^1_{3\times3}(L_{11}), \tag{7}$$

$$L_{33} = C^2_{3\times3}(C^2_{1\times1}([L_{12}, L_{22}])), \tag{8}$$

$$L_{\text{out}} = C^3_{1\times1}([L_{13}, L_{23}, L_{33}]), \tag{9}$$

where $C^p_{3\times3}$ denotes the p-depth 3×3 convolutional layer, $C^p_{1\times1}$ denotes the p-depth 1×1 convolutional layer, and $[\cdot,\cdot,\cdot]$ denotes the concatenation operation. In this part, we use 3×3 convolutional layers for feature extraction and use 1×1 convolutional layer for feature fusion. Inspired by MSRB [32], we also introduce multi-scale learning to extract image features at different scales. Therefore, we replace all 3×3 convolutional layers with 5×5 convolutional layers in the DenseNet-Bottom

$$H_{22} = C^1_{5\times5}(H_{11}), \tag{10}$$

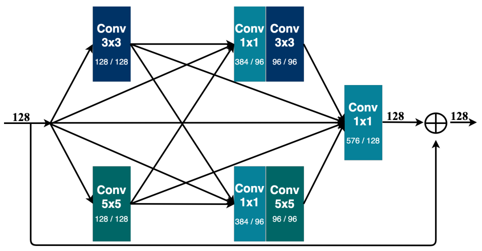


Fig. 5. The architecture of multi-scale dense cross block (MDCB).

$$H_{33} = C^2_{5\times5}(C^2_{1\times1}([H_{12}, H_{22}])), \tag{11}$$

$$H_{\text{out}} = C^3_{1\times1}([H_{13}, H_{23}, H_{33}]). \tag{12}$$

Feature Exchange & Fusion Mechanism: As mentioned above, MDCB uses two dense networks to extract image features at different scales. However, if these two subnets are independent of each other, image features at different scales will be difficult to transfer and fuse. Therefore, we associate the DenseNet-Top and DenseNet-Bottom by introducing two skip connections: M_{35} and M_{53}. As shown in Fig. 6, the blue lines are the introduced skip connection. M_{35} and M_{53} transmit the unique features extracted by themselves to the other sub-net. This facilitates the exchange of image features at different scales and improves the model performance. Meanwhile, we also introduce a bottleneck layer at the tail of the module to achieve feature fusion and dimensionality reduction, which combines the multi-scale features (L_{33} and H_{33}) extracted by DenseNet-Top and DenseNet-Bottom with the local features (L_{23}, H_{23}, and L_{13}/H_{13}) provided by the previous layers to obtain the high-frequency features. Therefore, the complete operations of multi-scale dense feature extraction can be defined as

$$L_{22} = C^1_{3\times3}(L_{11}), H_{22} = C^1_{5\times5}(H_{11}), \tag{13}$$

$$L_{33} = C^2_{3\times3}(C^2_{1\times1}([L_{12}, L_{22}, M_{53}])), \tag{14}$$

$$H_{33} = C^2_{5\times5}(C^2_{1\times1}([H_{12}, H_{22}, M_{35}])), \tag{15}$$

$$L_{\text{out}} = C^3_{1\times1}([L_{23}, L_{33}, H_{23}, H_{33}, L_{\text{in}}]). \tag{16}$$

Among them, $L_{\text{in}} = L_{11} = L_{12} = L_{13} = H_{11} = H_{12} = H_{13}$, $L_{22} = L_{23} = M_{35}$, and $H_{22} = H_{23} = M_{53}$, which represent

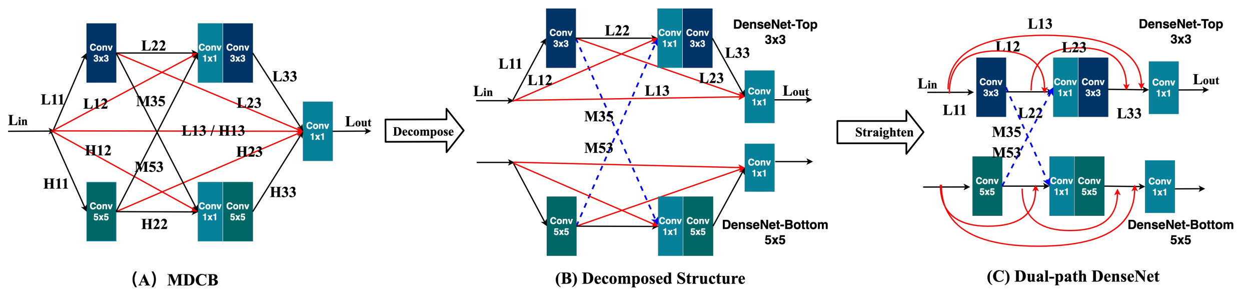


Fig. 6. The decomposed structure of MDCB, which remove the residual learning for better representation. (A) is the MDCB structure that removes residual learning strategy, (B) is the decomposed structure of MDCB, and (C) is the equivalent structure after straightening (B).

the same features while delivering to different convolutional layers. In addition, L_{in} and L_{out} denote the input and output of MDCB, respectively.

Local residual learning: Following previous works [32], [50], we also introduce local residual learning into our MDCB to further improve the information flow. Therefore, the output of n-th MDCB can be written as $L_n = L_{n-1} + L_{\text{out}}$. L_{n-1} denotes the output of the previous MDCB and serves as the input of this MDCB ($L_{\text{in}} = L_{n-1}$).

B. Hierarchical Feature Distillation Block (HFDB)

As mentioned in MSRN [32], hierarchical features can further improve model performance. However, the method used in MSRN is crude. This is because only use the 1×1 convolutional layer to compress hierarchical features can not extract effective features. Meanwhile, this method will produce massive redundant features, which is not conducive to model training. In order to remove redundant features and fully mine useful hierarchical features, we propose a hierarchical feature distillation block (HFDB, Fig. 7). HFDB focuses on adaptively recalibrate channel-wise feature response to achieve feature distillation, which is an efficient module that only needs low computational overhead. The core of HFDB is the introduced dimension transformation and channel attention mechanism.

Dimension Transformation Mechanism (DTM): Inspired by AutoEncoder, we introduce the dimension transformation mechanism to the module to achieve feature distillation. As shown in Fig. 7, HFDB introduces the bottleneck layer (1×1 layer) at the head and tail of the module, respectively. Firstly, all hierarchical features are concatenate like MSRN [32]. Then, these M feature maps are sent to the bottleneck layer to reduce the feature dimension. Then, the compressed features are sent to the channel attention module to adaptively recalibrate channel-wise feature response. Finally, the dimension of recalibrated features will be upgraded by the bottleneck layer at the tail of the module. It is worth noting that we set $\mathrm{M} > \mathrm{C} > 96$. Therefore, this module achieves a similar effect to the AutoEncoder, thus achieving feature distillation.

Channel Attention Mechanism (CAM): As mentioned above, we introduce CAM to the module to mine the most useful hierarchical features. CAM was firstly proposed in SENet [59], which could adaptively calibrate channel-wise feature response by explicitly modeling interdependency between channels. Inspired by this, RCAN [33] and

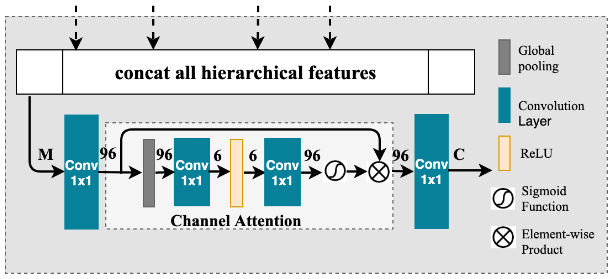


Fig. 7. The architecture of hierarchical feature distillation block (HFDB).

SrSENet [60] introduced CAM into all feature extraction blocks to obtain useful local features. However, this method will only bring a slight performance improvement while the increasement in execution time and memory consumption is huge [28]. Different from RCAN [33] and SrSENet [60] that pay attention to the local features extraction, we focus on using CAM to mine the relationship between hierarchical features and extract the most useful hierarchical features.

Taking $X = [x_1, ..., x_c, ..., x_C]$ as input, which has C feature maps with size of $H \times W$. The c-th element of z can be defined as

$$z_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} x_c(i,j), \tag{17}$$

where $x_c(i,j)$ is the value at position (i,j) of c-th feature x_c. Then, we apply a gating mechanism with the sigmoid function $s = f(C_1(\delta(C_2(z))))$, where $f(\cdot)$ and $\delta(\cdot)$ denote the sigmoid function and ReLU function, respectively. $C_1(\cdot)$ is a convolutional layer, which acts as channel-downscaling with reduction ratio r and is activated by ReLU, the low-dimensional feature is increased with ratio r by a channel-upscaling layer $C_2(\cdot)$. Finally, the channel statistics s is used to rescale the input: $\hat{x_c} = s_c \cdot x_c$, where s_c and x_c are the scaling factor and feature maps of the c-th channel.

Due to the ability of DTM and CAM, HFDB can not only eliminate redundant features but also extract useful hierarchical features. Meanwhile, MDCN only uses the channel attention mechanism once, which can greatly improve the efficiency of the model.

C. Dynamic Reconstruction Block (DRB)

In order to exploit the inter-scale correlation between different upsampling factors and maximize the use of model parameters, we introduce the dynamic reconstruction block (DRB). DRB is essentially a multi-element module that combines several different upsampling modules, which was first proposed in MDSR [16]. For single upsampling factor, DRB will create a corresponding scale-specific upsampling module. However, as for multi-factor image super-resolution task, DRB will generate a set of corresponding scale-specific upsampling modules (e.g., $\times2$, $\times3$, and $\times4$). In the upsampling module, we use the sub-pixel convolutional layer [14] to learn an array of upscaling filters to upscale feature maps into SR output.

As shown in Fig. 4, DRB is located at the tail of the model, which contains three sub-modules ($\times2$, $\times3$, and $\times4$ upsample modules). Meanwhile, these upsampling modules are arranged in parallel and only one upsampling module will be activated when used. In other words, the data stream will only flow into the module corresponding to the upsampling factor for image reconstruction. To better exploit the inter-scale correlation between different upsampling factors, we construct a dataset with different factors and introduce a mixed training strategy for training. This means that different size of images will be randomly sent to the model. Then it will automatically activate the corresponding upsampling modules according to the upsampling factors. Compared with specifically training a model for each upsampling factor, this method can learn the latent inter-scale information, maximize the reuse of model parameters, and reduce the training time. Thanks to this strategy, the trained model can be directly applied to multiple upsampling factors without re-training.

Different from MDSR [16], all parameters of our model are shared under different upsampling factors, except for the DRB. This can maximize the reuse of model parameters and benefit for the model to learn the inter-scale correlation between different upsampling factors, thus improve the model performance and robustness. In Sec. V-C, we will further verify the effectiveness of this module.

IV. Experiments

Following previous works, we only use DIV2K (1-800) [61] as our training dataset. Meanwhile, we choose Set5 [62], Set14 [63], B100 [64], Urban100 [9], and Manga109 [65] as our test datasets. All of them are the most widely used benchmark datasets, which can provide a fair comparison of the performance and generalization capability of each model.

A. Implementation Details

Model setting: In the final version of MDCN, we use 12 MDCBs ($N = 12$) for feature extraction, the input and output channel of each MDCB are set as 128 ($C = 128$). Detailed configuration of the input and output channels for each layer in each module has been provided in Fig. 5 and Fig. 7. We also introduce the self-ensemble mechanism [68] to further improve MDCN, which is denoted as **MDCN+**. Specifically, we first flip and rotate the input image to generate seven augmented inputs for each sample, thereby, we can obtain eight corresponding inputs. Then, we reconstruct the corresponding SR images using our MDCN. Finally, we apply the inverse transformation to those SR images and average them to generate the final SR image.

Training setting: During training, we use RGB image as input and augment the image with random horizontal flips and rotations. It is worth noting that our MDCN is a multi-factor model. Therefore, we use the mix training method to train the model. During training, we extract 16 LR patches with a randomly selected scale among $\times2$, $\times3$, and $\times4$, 1,000 iterations of back-propagation constitute an epoch. The learning rate is initialized to 10^{-4} and halved every 200 epochs. Different from previous works [32], [33] which set the size of LR images as 48$\times$48, we set the size of HR images as 48$\times$48. Thence, the size of the corresponding LR images of different upsampling factors is 24$\times$24, 16$\times$16 and 12$\times$12, respectively. Using small patch size for training will cause performance degradation, but greatly reduce the training time. However, our MDCN still achieves great results due to the fully use of the inter-scale correlation between different upsampling factors. We implement our model with the PyTorch framework and update it with Adam optimizer, all our experiments are performed on GTX TitanX.

B. Comparisons with state-of-the-art SR methods

We compare MDCN with more than 19 SR methods, including Bicubic, A+ [6], SelfExSR [9], SRCNN [10], ESPCN [14], FSRCNN [12], VDSR [11], LapSRN [15], DRCN [13], MRFN [27], SRMDNF [21], MSRN [32], EDSR [16], RDN [50], RCAN [33], FilterNet [26], DNCL [30], RAN [29], and SeaNet [31]. All SR images are evaluated with PSNR and SSIM [69] on the Y channel in YCbCr space.

Quantitative Comparison: In TABLE II, we show the quantitative comparisons with some advanced SR methods, all of them have achieved competitive results at the time. Among them, best results are highlighted and the second best results are underlined. Besides, the 'Average' denotes the average results of these 5 test datasets. Obviously, our MDCN achieves competitive results on all upsampling factors. Among them, RDN is slightly better than MDCN under small upsampling factors ($\times2$). However, our MDCN can achieve better results under large upsampling factors (e.g., $\times3$, $\times4$). This is because the introduced MDCB in MDCN can fully extract multi-scale features, which is conducive to large upsampling factor SR reconstruction. Considering that RCAN achieves the state-of-the-art results, we make a detailed comparison with it in TABLE III. We can observe that RCAN is slightly better than MDCN ($\times2$: 0.24dB, $\times3$: 0.15dB, and $\times4$: 0.09dB). However, it should be noticed that the execution time of our MDCN is 3 times faster than RCAN. This means that MDCN can achieve similar results as RCAN with less execution time. Meanwhile, it is worth noting that: (1). Except for MDCN, all reported SR methods are specially trained for different upsampling factor; (2). EDSR [16], RDN [50], and RCAN [33] use the pre-trained model ($\times2$) as the initialization model to train large upsampling factor model like $\times4$; (3). DRCN [13] introduces the recursive mechanism to further improve model

TABLE II
QUANTITATIVE COMPARISONS WITH STATE-OF-THE-ART SR METHODS. THE BEST RESULTS ARE **HIGHLIGHTED** AND THE SECOND BEST RESULTS ARE UNDERLINED. OBVIOUSLY, MDCN ACHIEVES COMPETITIVE RESULTS ON ALL BENCHMARK DATASETS.

Algorithm	Scale	Set5 [62] PSNR / SSIM	Set14 [63] PSNR / SSIM	BSDS100 [66] PSNR / SSIM	Urban100 [67] PSNR / SSIM	Manga109 [65] PSNR / SSIM	Average PSNR / SSIM
Bicubic	×2	33.66 / 0.9299	30.24 / 0.8688	29.56 / 0.8431	26.88 / 0.8403	30.80 / 0.9339	30.22 / 0.8832
A+ [6] (2014)	×2	36.60 / 0.9542	32.42 / 0.9059	31.24 / 0.8870	29.25 / 0.8955	35.37 / 0.9663	32.98 / 0.9218
SelfExSR [9] (2015)	×2	36.60 / 0.9537	32.46 / 0.9051	31.20 / 0.8863	29.55 / 0.8983	35.82 / 0.9671	33.13 / 0.9221
SRCNN [10] (2014)	×2	36.66 / 0.9542	32.45 / 0.9067	31.36 / 0.8879	29.50 / 0.8946	35.60 / 0.9663	33.11 / 0.9219
ESPCN [14] (2016)	×2	37.00 / 0.9559	32.75 / 0.9098	31.51 / 0.8939	29.87 / 0.9065	36.21 / 0.9694	33.47 / 0.9271
FSRCNN [12] (2016)	×2	37.06 / 0.9554	32.76 / 0.9078	31.53 / 0.8912	29.88 / 0.9024	36.67 / 0.9694	33.58 / 0.9252
VDSR [11] (2016)	×2	37.53 / 0.9590	33.05 / 0.9130	31.90 / 0.8960	30.77 / 0.9140	37.22 / 0.9750	34.09 / 0.9314
DRCN [13](2016)	×2	37.63 / 0.9584	33.06 / 0.9108	31.85 / 0.8947	30.76 / 0.9147	37.63 / 0.9723	34.19 / 0.9302
LapSRN [15] (2017)	×2	37.52 / 0.9591	33.08 / 0.9130	31.80 / 0.8950	30.41 / 0.9101	37.27 / 0.9740	34.02 / 0.9302
EDSR [16] (2017)	×2	38.11 / 0.9602	33.92 / 0.9195	32.32 / 0.9013	32.93 / 0.9351	39.10 / 0.9773	35.27 / 0.9387
SRMDNF [21] (2018)	×2	37.79 / 0.9601	33.32 / 0.9159	32.05 / 0.8985	31.33 / 0.9204	38.07 / 0.9761	34.51 / 0.9342
MSRN [32] (2018)	×2	38.07 / 0.9608	33.68 / 0.9184	32.22 / 0.9002	32.32 / 0.9304	38.64 / 0.9771	34.99 / 0.9374
RDN [50] (2018)	×2	38.24 / 0.9614	34.01 / 0.9212	32.34 / 0.9017	32.89 / 0.9353	39.18 / 0.9780	35.33 / 0.9395
RAN [29] (2019)	×2	37.58 / 0.9592	33.10 / 0.9133	31.92 / 0.8963	N / A	N / A	N / A
DNCL [30] (2019)	×2	37.65 / 0.9599	33.18 / 0.9141	31.97 / 0.8971	30.89 / 0.9158	N / A	N / A
FilterNet [26] (2019)	×2	37.86 / 0.9610	33.34 / 0.9150	32.09 / 0.8990	31.24 / 0.9200	N / A	N / A
MRFN [27] (2019)	×2	37.98 / 0.9611	33.41 / 0.9159	32.14 / 0.8997	31.45 / 0.9221	38.29 / 0.9759	34.65 / 0.9349
SeaNet [31] (2020)	×2	38.08 / 0.9609	33.75 / 0.9190	32.27 / 0.9008	32.50 / 0.9318	38.76 / 0.9774	35.07 / 0.9380
MDCN (Ours)	×2	38.19 / 0.9612	33.86 / 0.9202	32.32 / 0.9014	32.92 / 0.9355	39.09 / 0.9780	35.28 / 0.9393
MDCN+ (Ours)	×2	**38.25 / 0.9614**	**34.01 / 0.9208**	**32.36 / 0.9019**	**33.08 / 0.9367**	**39.27 / 0.9784**	**35.39 / 0.9399**
Bicubic	×3	30.39 / 0.8682	27.55 / 0.7742	27.21 / 0.7385	24.46 / 0.7449	26.95 / 0.8556	27.31 / 0.7963
A+ [6] (2014)	×3	32.63 / 0.9085	29.25 / 0.8194	28.31 / 0.7828	26.05 / 0.8019	29.93 / 0.9089	29.23 / 0.8443
SelfExSR [9] (2015)	×3	32.66 / 0.9089	29.34 / 0.8222	28.30 / 0.7839	26.45 / 0.8124	27.57 / 0.7997	28.86 / 0.8254
SRCNN [10] (2014)	×3	32.75 / 0.9090	29.30 / 0.8215	28.41 / 0.7863	26.24 / 0.7989	30.48 / 0.9117	29.44 / 0.8455
ESPCN [14] (2016)	×3	33.02 / 0.9135	29.49 / 0.8271	28.50 / 0.7937	26.41 / 0.8161	30.79 / 0.9181	29.64 / 0.8537
FSRCNN [12] (2016)	×3	33.20 / 0.9149	29.54 / 0.8277	28.55 / 0.7945	26.48 / 0.8175	30.98 / 0.9212	29.75 / 0.8552
VDSR [11] (2016)	×3	33.67 / 0.9210	29.78 / 0.8320	28.83 / 0.7990	27.14 / 0.8290	32.01 / 0.9340	30.29 / 0.8630
DRCN [13] (2016)	×3	33.85 / 0.9215	29.89 / 0.8317	28.81 / 0.7954	27.16 / 0.8311	32.31 / 0.9328	30.40 / 0.8625
LapSRN [15] (2017)	×3	33.82 / 0.9227	29.87 / 0.8320	28.82 / 0.7980	27.07 / 0.8280	32.21 / 0.9350	30.36 / 0.8631
EDSR [16] (2017)	×3	34.65 / 0.9280	30.52 / 0.8462	29.25 / 0.8093	28.80 / 0.8653	34.17 / 0.9476	31.48 / 0.8793
SRMDNF [21] (2018)	×3	34.12 / 0.9254	30.04 / 0.8382	28.97 / 0.8025	27.57 / 0.8398	33.00 / 0.9403	30.74 / 0.8692
MSRN [32] (2018)	×3	34.48 / 0.9276	30.40 / 0.8436	29.13 / 0.8061	28.31 / 0.8560	33.56 / 0.9451	31.18 / 0.8757
RDN [50] (2018)	×3	34.71 / 0.9296	30.57 / 0.8468	29.26 / 0.8093	28.80 / 0.8653	34.13 / 0.9484	31.49 / 0.8799
RAN [29] (2019)	×3	33.71 / 0.9223	29.84 / 0.8326	28.84 / 0.7981	N / A	N / A	N / A
DNCL [30] (2019)	×3	33.95 / 0.9232	29.93 / 0.8340	28.91 / 0.7995	27.27 / 0.8326	N / A	N / A
FilterNet [26] (2019)	×3	34.08 / 0.9250	30.03 / 0.8370	28.95 / 0.8030	27.55 / 0.8380	N / A	N / A
MRFN [27] (2019)	×3	34.21 / 0.9267	30.03 / 0.8363	28.99 / 0.8029	27.53 / 0.8389	32.82 / 0.9396	30.72 / 0.8689
SeaNet [31] (2020)	×3	34.55 / 0.9282	30.42 / 0.8444	29.17 / 0.8071	28.50 / 0.8594	33.73 / 0.9463	31.27 / 0.8771
MDCN (Ours)	×3	34.69 / 0.9294	30.54 / 0.8470	29.26 / 0.8095	28.83 / 0.8662	34.17 / 0.9485	31.50 / 0.8801
MDCN+ (Ours)	×3	**34.76 / 0.9299**	**30.63 / 0.8480**	**29.31 / 0.8103**	**29.00 / 0.8687**	**34.43 / 0.9497**	**31.63 / 0.8813**
Bicubic	×4	28.42 / 0.8104	26.00 / 0.7027	25.96 / 0.6675	23.14 / 0.6577	24.89 / 0.7866	25.68 / 0.7250
A+ [6] (2014)	×4	30.33 / 0.8565	27.44 / 0.7450	26.83 / 0.6999	24.34 / 0.7211	27.03 / 0.8439	27.19 / 0.7733
SelfExSR [9] (2015)	×4	30.34 / 0.8593	27.55 / 0.7511	26.84 / 0.7032	24.83 / 0.7403	27.83 / 0.8598	27.48 / 0.7827
SRCNN [10] (2014)	×4	30.48 / 0.8628	27.50 / 0.7513	26.90 / 0.7101	24.52 / 0.7221	27.58 / 0.8555	27.40 / 0.7804
ESPCN [14] (2016)	×4	30.66 / 0.8646	27.71 / 0.7562	26.98 / 0.7124	24.60 / 0.7360	27.70 / 0.8560	27.53 / 0.7850
FSRCNN [12] (2016)	×4	30.73 / 0.8601	27.71 / 0.7488	26.98 / 0.7029	24.62 / 0.7272	27.90 / 0.8517	27.59 / 0.7781
VDSR [11] (2016)	×4	31.35 / 0.8830	28.02 / 0.7680	27.29 / 0.7267	25.18 / 0.7540	28.83 / 0.8870	28.13 / 0.8037
DRCN [13] (2016)	×4	31.56 / 0.8810	28.15 / 0.7627	27.24 / 0.7150	25.15 / 0.7530	28.98 / 0.8816	28.22 / 0.7987
LapSRN [15] (2017)	×4	31.54 / 0.8850	28.19 / 0.7720	27.32 / 0.7270	25.21 / 0.7560	29.09 / 0.8900	28.27 / 0.8060
EDSR [16] (2017)	×4	32.46 / 0.8968	28.80 / 0.7876	27.71 / 0.7420	26.64 / 0.8033	31.02 / 0.9148	29.33 / 0.8289
SRMDNF [21] (2018)	×4	31.96 / 0.8925	28.35 / 0.7787	27.49 / 0.7337	25.68 / 0.7731	30.09 / 0.9024	28.71 / 0.8161
MSRN [32] (2018)	×4	32.25 / 0.8958	28.63 / 0.7833	27.61 / 0.7377	26.22 / 0.7905	30.57 / 0.9103	29.05 / 0.8235
RDN [50] (2018)	×4	32.47 / 0.8990	28.81 / 0.7871	27.72 / 0.7419	26.61 / 0.8028	31.00 / 0.9151	29.32 / 0.8292
RAN [29] (2019)	×4	31.43 / 0.8847	28.09 / 0.7691	27.31 / 0.7260	N / A	N / A	N / A
DNCL [30] (2019)	×4	31.66 / 0.8871	28.23 / 0.7717	27.39 / 0.7282	25.36 / 0.7606	N / A	N / A
FilterNet [26] (2019)	×4	31.74 / 0.8900	28.27 / 0.7730	27.39 / 0.7290	25.53 / 0.7680	N / A	N / A
MRFN [27] (2019)	×4	31.90 / 0.8916	28.31 / 0.7746	27.43 / 0.7309	25.46 / 0.7654	29.57 / 0.8962	28.53 / 0.8117
SeaNet [31] (2020)	×4	32.33 / 0.8970	28.72 / 0.7855	27.65 / 0.7388	26.32 / 0.7942	30.74 / 0.9129	29.13 / 0.8257
MDCN (Ours)	×4	32.48 / 0.8985	28.83 / 0.7879	27.74 / 0.7423	26.69 / 0.8049	31.10 / 0.9163	29.37 / 0.8300
MDCN+ (Ours)	×4	**32.61 / 0.9000**	**28.90 / 0.7893**	**27.79 / 0.7434**	**26.86 / 0.8083**	**31.40 / 0.9188**	**29.51 / 0.8320**

TABLE III
QUANTITATIVE COMPARISONS (PSNR/SSIM, PARAMETERS, AND EXECUTION TIME) WITH RCAN [33]. THE BEST RESULTS ARE **HIGHLIGHTED** AND THE FASTEST EXECUTION TIME ARE RED.

Algorithm	Scale	Parameters	Set5 [62] PSNR / SSIM / Time	Set14 [63] PSNR / SSIM / Time	BSDS100 [66] PSNR / SSIM / Time	Urban100 [67] PSNR / SSIM / Time	Manga109 [65] PSNR / SSIM / Time	Average PSNR / SSIM / Time
RCAN [33]	×2	15M	**38.27 / 0.9614** / 0.60s	**34.12 / 0.9216** / 1.11s	**32.41 / 0.9027** / 0.75s	**33.34 / 0.9384** / 3.78s	**39.44 / 0.9786** / 4.55s	**35.52 / 0.9405** / 2.16s
MDCN (Ours)	×2	15M	38.19 / 0.9612 / 0.22s	33.86 / 0.9202 / 0.38s	32.32 / 0.9014 / 0.29s	32.92 / 0.9355 / 1.19s	39.09 / 0.9780 / 1.45s	35.28 / 0.9393 / 0.71s
RCAN [33]	×3	15M	**34.74 / 0.9299** / 0.34s	**30.65 / 0.8482** / 0.55s	**29.32 / 0.8111** / 0.41s	**29.09 / 0.8702** / 1.89s	**34.44 / 0.9499** / 2.33s	**31.65 / 0.8818** / 1.10s
MDCN (Ours)	×3	15M	34.69 / 0.9294 / 0.15s	30.54 / 0.8470 / 0.24s	29.26 / 0.8095 / 0.16s	28.83 / 0.8662 / 0.71s	34.17 / 0.9485 / 0.87s	31.50 / 0.8801 / 0.43s
RCAN [33]	×4	15M	**32.63 / 0.9002** / 0.30s	**28.87 / 0.7889** / 0.40s	**27.77 / 0.7436** / 0.30s	**26.82 / 0.8087** / 1.21s	**31.22 / 0.9173** / 1.50s	**29.46 / 0.8317** / 0.74s
MDCN (Ours)	×4	15M	32.48 / 0.8985 / 0.12s	28.83 / 0.7879 / 0.18s	27.74 / 0.7423 / 0.12s	26.69 / 0.8049 / 0.52s	31.10 / 0.9163 / 0.62s	29.37 / 0.8300 / 0.31s

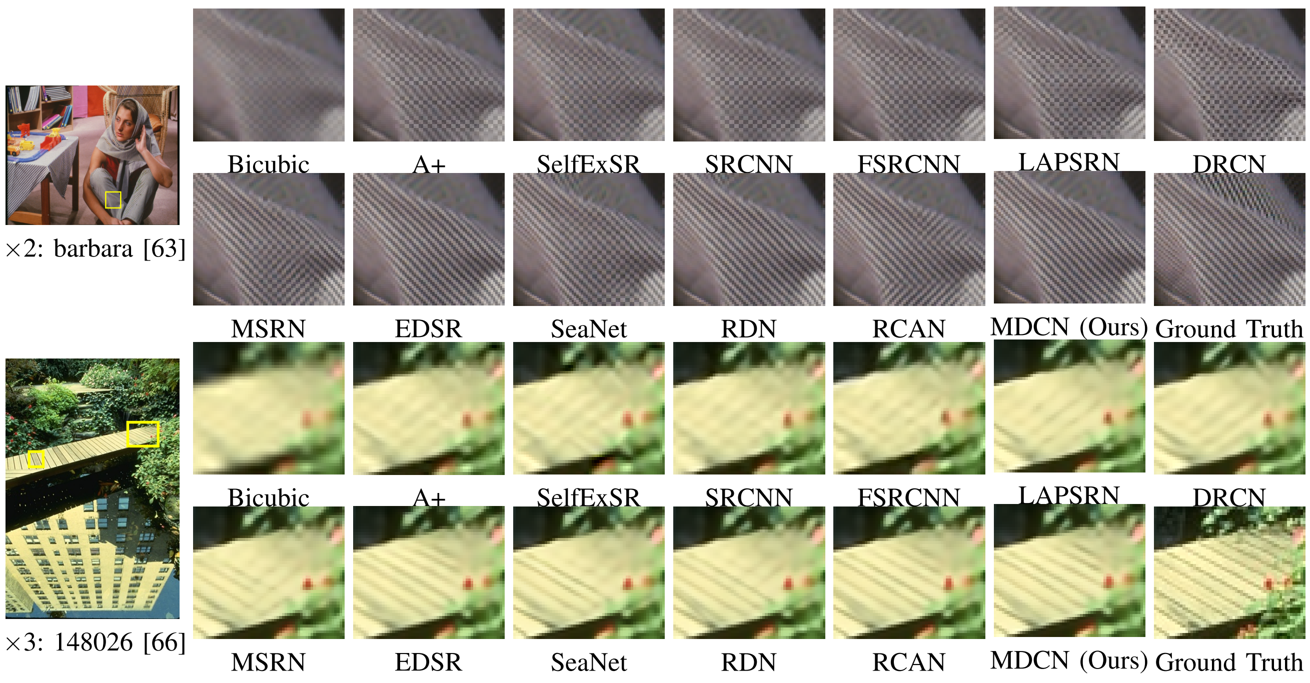


Fig. 8. Visual comparison with different SR methods on Set14 [63] and BSDS100 [66] under small upsampling factors. **Please zoom in to view details.**

performance. (4). Other models use large LR images as inputs for training. All these strategies can further boost the performance. However, in order to verify the effectiveness of MDCN, we do not use any training tricks in our experiment. Nevertheless, since MDCB, HFDB, and DRB can extract rich image features and learn the inter-scale correlation, our MDCN still achieves competitive results.

Visual Comparison: In Figs. 8 and 9, we show visual comparisons under small (×2, ×3) and large upsampling factor (×4), respectively. Among them, EDSR [16] was the champion model of the NTIRE2017 SR Challenge, RDN [50] and RCAN [33] are superior models that achieved SOTA results. According to the figure, we can clearly observe that: (i). Most compared SR methods (e.g., SRCNN, MSRN, and SeaNet) cannot recover clear and accurate image edges. Furthermore, under large upsample factor (e.g., ×4), the reconstructed SR images are blurred with severe artifacts and incorrect edges. In contrast, our MDCN can reconstruct more realistic SR images with clear and sharp edges; (ii). Compared with large size models (e.g., EDSR, RDN, and RCAN), our MDCN still shows competitive performance and better results in edges reconstruction. Overall, with the help of MDCB and HFDB, MDCN can reconstruct high-quality SR images.

V. Model Analysis

A. *Study of Multi-scale Dense Cross Block (MDCB)*

MDCB is the most important component of MDCN, which is designed for local and multi-scale feature extraction.

(1). As mentioned in Sec. III-A, MDCB is essentially a dual-path dense network, which introduces multi-scale learning, feature exchange & fusion mechanism, and residual learning. In TABLE IV, we provide a series of ablation studies to investigate their effectiveness.

TABLE IV
STUDY OF MDCN. 'RL' DENOTES 'RESIDUAL LEARNING', 'FEFM' DENOTES 'FEATURE EXCHANGE & FUSION MECHANISM'. ALL OF THESE MODELS WERE TRAINED UNDER THE SAME EXPERIMENTAL CONDITIONS AND TESTED ON DIV2K(896-900).

Case Index		1	2	3	4	5	6	7	8
MDCB	RL	×	×	×	×	√	√	√	√
	FEFM	×	×	×	√	√	√	√	√
	DenseNet-Top	√	×	√	√	√	√	√	√
	DenseNet-Bottom	×	√	√	√	√	√	√	√
HFDB		×	×	×	×	×	√	√	√
MDCB Number		3	3	3	3	3	3	6	6
Channel Number		128	128	128	128	128	128	128	256
PSNR(dB) ×3		32.85	32.90	33.02	33.08	33.10	33.17	<u>33.35</u>	**33.49**

Dual-path Dense Network: In Cases 1 and 2, we remove all introduced mechanisms, only leaving DenseNet-Top or DenseNet-Bottom, respectively. In Case 3, we use both DenseNet-Top and DenseNet-Bottom to build the dual-path network for image reconstruction. It is worth noting that Case 3 is a simple combination of these two dense networks and does not introduce skip connections (M_{35} and M_{53}) for feature exchange and fusion. According to the table, we can clearly see that Case 3 achieves better results, which demonstrates the dual-path network is effective.

Feature Exchange & Fusion Mechanism (FEFM): We introduce skip connections (M_{35} and M_{53}) in Case 4 to achieve feature exchange and fusion. Compared with Case 3, Case 4 achieves better results. This is because the introduced FEFM can transfer the different scales of image features to the other part, which greatly enriches the diversity of extracted features. Therefore, the model can achieve better results.

Residual Learning (RL): Plenty of previous works have proved the effectiveness of residual learning. Following these works, we also introduce residual learning into our MDCN to improve the information flow. In Cases 4 and 5, we provide

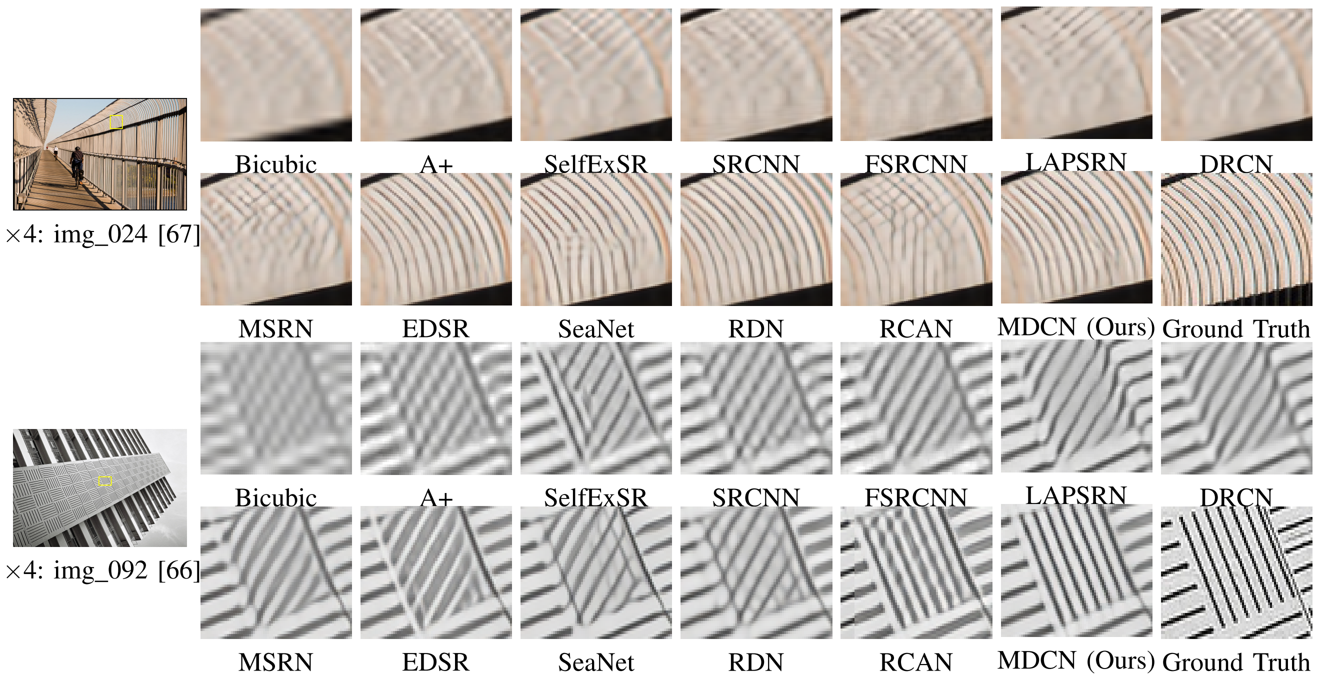


Fig. 9. Visual comparison with different SR methods on Urban100 [67] under large upsampling factor (×4). **Please zoom in to view details.**

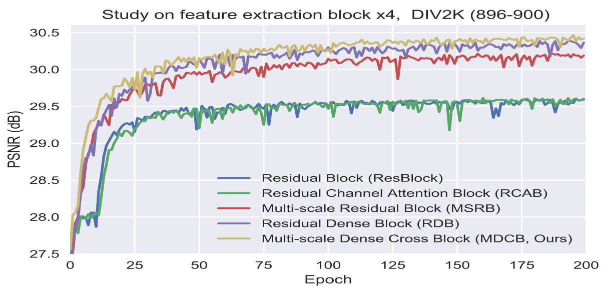


Fig. 10. Performance comparison between different feature extraction blocks.

the results of the model without and with residual learning, respectively. According to the results, we confirmed the effectiveness of residual learning. Although the improvement of PSNR is not obvious, this is because we only use 3 MDCBs here. Furthermore, residual learning can accelerate model convergence and solve the problem of gradient disappearance and explosion, which is beneficial for deep networks.

MDCB and Channel Number: MDCN is a modular network, so the size of the model can be easily adjusted by changing the number of MDCB (N) and channels (C). In Cases 6, 7, and 8, we provide the results of different numbers of MDCBs and channels. Obviously, as the number of MDCBs and channel increases, the model performance can be further improved. This means the reported results are not the best, and its performance still has room for improvement. However, in order to achieve a well balance between performance and model size, we set $N = 12$ and $C = 128$ in the final model.

(2).To further demonstrate the effectiveness of MDCB, we compare our MDCB with ResBlock [57], RDB [50], RCAB [33], and MSRB [32]. All of them are the most widely used feature extraction blocks, which have well-designed architecture and have been widely validated. In the experiment, we use MDCN as the basic backbone and replace the original MDCB with different feature extraction blocks (e.g., RDB and RCAB) to build new models. After that, all these models are retrained and tested under the same equipment, environment, and dataset. Meanwhile, all of these feature extraction blocks are adjusted to the same parameter level by adjusting the number of blocks or channels for a fair comparison. In Fig. 10, we show the performance curve of these models during training. According to this figure, we can draw the following conclusions: (i) simply stacking ResBlock or RCAB will not effectively boost model performance; (ii) compared with MSRB, the result of MDCB has been significantly improved; (iii) the performance of MDCB is comparable to RDB, even slightly better than it. This is because MDCB skillfully combines multi-scale residual learning with dense connections to make it have the characteristics of both MSRB and RDB, so as to obtain better results. All aforementioned experiments fully demonstrate the effectiveness of MDCB.

B. *Study of Hierarchical Feature Distillation Block (HFDB)*

In order to eliminate redundant features and extract the most useful hierarchical features, we propose HFDB. In TABLE IV, Cases 5 and 6 represent the results of the model without and with HFDB, respectively. Obviously, with the help of HFDB, the model performance can be further improved. On the other hand, we compare HFDB with four hierarchical feature utilization methods mentioned in Fig. 2. To ensure the fairness of comparison, we apply these methods to the same model and the performance curves of each method are presented in Fig. 11. Obviously, our HFDB achieves the best results (the blue line). It is worth noting that the introduced dimension transformation mechanism in HFDB can greatly

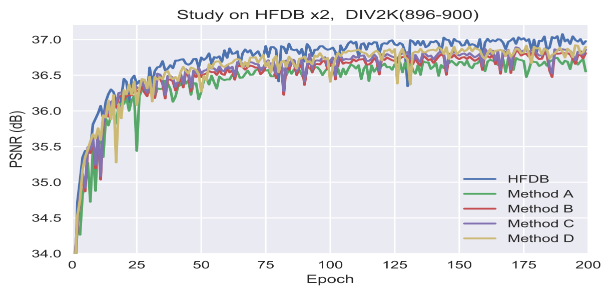


Fig. 11. Performance comparison between different hierarchical feature utilization methods.

TABLE V
QUANTITATIVE COMPARISONS WITH MSRN, MDCN-S, AND MDCN.

Algorithm	Scale	Set5 [62] PSNR / SSIM	Set14 [63] PSNR / SSIM	BSDS100 [66] PSNR / SSIM	Urban100 [67] PSNR / SSIM	Manga109 [65] PSNR / SSIM
MSRN [32]	$\times 4$	32.25 / 0.8958	28.63 / 0.7833	27.61 / 0.7377	26.22 / 0.7905	30.57 / 0.9103
MDCN-S	$\times 4$	32.39 / 0.8977	28.75 / 0.7863	27.66 / 0.7398	26.45 / 0.7985	30.87 / 0.9141
MDCN	$\times 4$	**32.48 / 0.8985**	**28.83 / 0.7879**	**27.74 / 0.7423**	**26.69 / 0.8049**	**31.10 / 0.9163**

reduce model parameters. Taking 12 MDCBs as an example, Method D needs $245,760$ parameters while our HFDB only needs $173,184$ parameters, $5/7$ of Method D. In addition, this gap will increase as the number of MDCB increases. In summary, compared with Method D (proposed in MSRN [32]), our HFDB can achieve better results with fewer parameters and less computational overhead.

C. Study of Dynamic Reconstruction Block (DRB)

To realize the reconstruction of SR images with different upsampling factors in a single model, we introduce DRB into MDCN. DRB can create a set of scale-specific upsampling modules according to actual needs. This can maximizes the reuse of model parameters and enables the model to learn the inter-scale correlation between different upsampling factors. It is worth noting that when there is only one upsampling module in DRB, it is the same as previous works that specifically trained for different upsampling factors. This special case is named as 'MDCN-S'. In order to illustrate the effectiveness of DRB, we provide the following experiments:

(1) In TABLE V, we provide PSNR results of MSRN, MDCN-S, and MDCN on 5 benchmark test datasets. According to the table, we can observe that: (a) whether MDCN-S or MDCN, the performance is better than MSRN; (b) compared with MDCN-S, the performance of MDCN has been significantly improved. This proves that the inter-scale correlation can further improve the model performance.

(2). As described in Section II, DRB was first proposed in MDSR [16]. Different from MDSR [16], we only apply DRB at the tail of the model. This means that all feature extraction modules in MDCN are weight sharing, which benefits for inter-scale learning. In TABLE VI, we show the quantitative comparisons of multi-factor models, including VDSR [11], MDSR [16], and our MDCN. For a fair comparison, all models are re-trained under the same training dataset and settings. Among them, ($\times$2), ($\times$3), and ($\times$4) stand for the specifically trained model for one single upsampling factor and ($\times$2,$\times$3,$\times$4) denote the multi-factor model with mix training

TABLE VI
PSNR COMPARISONS OF MULTI-FACTOR MODELS. BEST RESULTS UNDER DIFFERENT UPSAMPLING FACTORS AND DIFFERENT TRAINING STRATEGIES ARE **RED**. BEST RESULTS UNDER THE SAME UPSAMPLING FACTORS WITH DIFFERENT TRAINING STRATEGIES ARE UNDERLINE.

		($\times$2)		($\times$3)		($\times$4)		($\times$2,$\times$3,$\times$4)	
	Method	Urban100	Manga109	Urban100	Manga109	Urban100	Manga109	Urban	Manga109
x2	VDSR	30.77	37.22	*	*	*	*	30.80	37.33
	MDSR	32.30	38.73	*	*	*	*	32.74	38.95
	MDCN	**32.95**	**39.13**	*	*	*	*	**32.92**	**39.09**
x3	VDSR	*	*	27.14	32.01	*	*	27.25	32.20
	MDSR	*	*	28.19	33.46	*	*	28.68	34.02
	MDCN	*	*	**28.78**	**34.11**	*	*	**28.83**	**34.17**
x4	VDSR	*	*	*	*	25.18	28.83	25.28	28.92
	MDSR	*	*	*	*	25.98	30.28	26.56	30.98
	MDCN	*	*	*	*	**26.45**	**30.87**	**26.69**	**31.10**

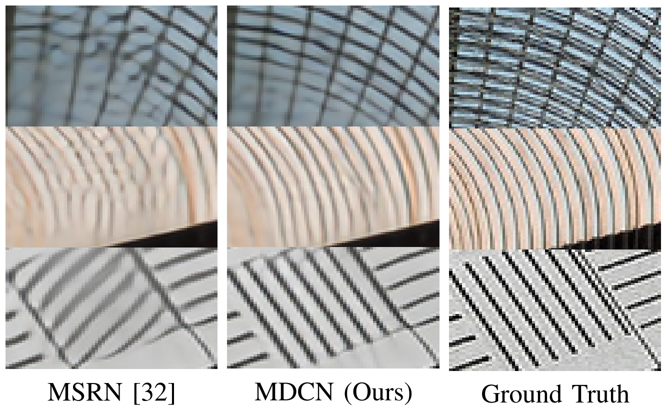


Fig. 12. Visual comparison between MSRN and MDCN ($\times$4).

method. According to the table, we can find that: (i). Our MDCN achieves the best results under all upsampling factors whether using the single factor or multi-factor mixed training method, ; (ii). Most results of multi-factor models (mixed training) are superior to the special training models (single factor training). This indicates that fully using the inter-scale correlation between different upsampling factors can improve the model performance; (iii). At the small upsampling factor ($\times$2), the results of single-factor MDCN are slightly better than the mixed trained MDCN. This is because SR images reconstructed with small upsampling factor is a relatively simple task, so that the introduced other scale information will interference it. Fortunately, this gap is acceptable. Therefore, we recommend to introduce DRB and mix training strategy to build a general model.

VI. RESEARCH AND ANALYSIS

A. Compare with MSRN

Different from MSRN, MDCN introduced MDCB, HFDB, and DRB to build a more efficient and general SR model. The effectiveness of these modules have been verified in Sec.V. Considering the similarity of these models, we provide more detailed comparisons to manifest the effectiveness of MDCN.

(1) In TABLE II, we provide the PSNR and SSIM results of MSRN and MDCN. Obviously, MDCN achieves better results. In addition, we provide some high-frequency detail comparisons between MSRN and MDCN in Fig. 12. We can clearly observe that the high-frequency details of the

TABLE VII
REPLACE THE MSRB IN THE MSRN WITH MDCB TO GET THE MDCN'. MDCN' ACHIEVES BETTER RESULTS WITH FEWER PARAMETERS.

Methods	MSRN			MDCN'(Ours)		
Scale	x2	x3	x4	x2	x3	x4
Parameters	5.92M	6.11M	6.07M	**4.34M** ↓	**4.52M** ↓	**4.48M** ↓
Set5	38.07/0.9608	34.48/0.9276	32.25/0.8958	**38.10/0.9608**	**34.52/0.9278**	**32.30/0.8965**
Set14	33.68/0.9184	30.40/0.8436	28.63/0.7833	**33.74/0.9186**	**30.45/0.8444**	**28.68/0.7844**
BSD100	32.22/0.9002	29.13/0.8061	27.61/0.7377	**32.23/0.9003**	**29.16/0.8067**	**27.63/0.7383**
Urban100	32.32/0.9304	28.31/0.8560	26.20/0.7905	**32.34/0.9304**	**28.39/0.8575**	**26.26/0.7918**
Manga109	38.64/0.9771	33.56/0.9451	30.57/0.9103	**38.73/0.9774**	**33.77/0.9462**	**30.68/0.9119**
Average	34.99/0.9374	31.18/0.8754	29.05/0.8235	**35.03/0.9375**	**31.26/0.8765**	**29.11/0.8246**

TABLE VIII
REPLACE THE MDCB IN THE MDCN WITH MSRB TO GET THE MSRN'. MDCN ACHIEVES BETTER RESULTS WITH FEWER PARAMETERS.

Methods	MSRN'(x2,x3,x4)			MDCN (x2,x3,x4, Ours)		
Scale	x2	x3	x4	x2	x3	x4
Parameters	16.77M			**15.62M** ↓		
Set5	38.07/0.9608	34.51/0.9279	32.32/0.8967	**38.19/0.9612**	**34.69/0.9294**	**32.48/0.8985**
Set14	33.78/0.9192	30.45/0.8445	28.71/0.7850	**33.86/0.9202**	**30.54/0.8470**	**28.83/0.7988**
BSD100	32.23/0.9001	29.17/0.8070	27.66/0.7391	**32.32/0.9014**	**29.26/0.8095**	**27.74/0.7423**
Urban100	32.43/0.9314	28.43/0.8584	26.35/0.7946	**32.92/0.9355**	**28.83/0.8662**	**26.69/0.8049**
Manga109	38.55/0.9779	33.72/0.9461	30.75/0.9120	**39.09/0.9780**	**34.17/0.9485**	**31.10/0.9163**
Average	35.01/0.9379	31.26/0.8768	29.16/0.8255	**35.28/0.9393**	**31.40/0.8801**	**29.37/0.8322**

image reconstructed by MSRN are severely damaged, and even wrong textures and edges are generated. Contrastly, our MDCN can reconstruct high-quality SR images with accurate and clear details. Meanwhile, different from MSRN that specially trained for various upsample factors, we only use one MDCN to perform SR reconstruction task for different factors.

(2) Considering that the parameter amounts of MSRN and MDCN are not equal, we design two experiments to further explore their performance. To achieve this, we build two models, named MDCN' and MSRN', respectively. MDCN' is a variant of MSRN, which uses MSRN as the backbone and replaces all MSRBs in the network with MDCB ($C = 64$). In contrast, MSRN' is the variant of MDCN, which uses MDCN as the backbone and replaces all MDCBs in the network with MSRB ($C = 128$). Quantitative comparisons are shown in TABLE VII and VIII, respectively. Compared with MSRN, MDCN' achieves better results on all benchmark test datasets with fewer parameters. This also verifies the feature extraction ability of MDCB is superior to MSRB. Similarly, MDCN achieves better results than MSRN' with fewer parameters. This fully proves the effectiveness of MDCN.

In summary, although MSRN is an efficient SR model, the performance can be further improved. Meanwhile, extensive experiments demonstrate that MDCN can achieve better results than MSRN with fewer parameters.

B. Compare with Other Multi-scale SR Models

As described in Sec. I, many improved versions of MSRN have been proposed such as MSDN [70], MSFFRN [38], MWRN [71], and MSRCAN [36]. They introduce dense connection, feature fusion, wide-activated, or channel attention mechanisms to improve the performance of MSRN. However, these models do not pay attention to the core defects of MSRN, so the introduced mechanism does not significantly boost model performance. To solve this problem, we proposed MDCN. As shown in TABLE IX, we provide the quantitative comparisons of these models. According to the table, we can clearly observe that the performance of MDCN has been greatly improved compared to others. This further demonstrates the effectiveness of MDCN.

TABLE IX
COMPARISONS WITH MULTI-SCALE SR MODELS ON URBAN100.

Method	MSRN	MSRCAN	MWRN	MSDN	MSFFRN	MDCN (Ours)
x2	32.32/0.9304	31.72/0.9242	32.46/0.9313	32.51/0.9342	32.60/0.9326	**32.92/0.9355**
x3	28.31/0.8560	27.72/0.8397	28.40/0.8569	28.63/0.8593	28.65/0.8619	**28.83/0.8662**
x4	26.20/0.7905	25.75/0.7733	26.29/0.7926	26.25/0.7931	26.47/0.7980	**26.69/0.8049**

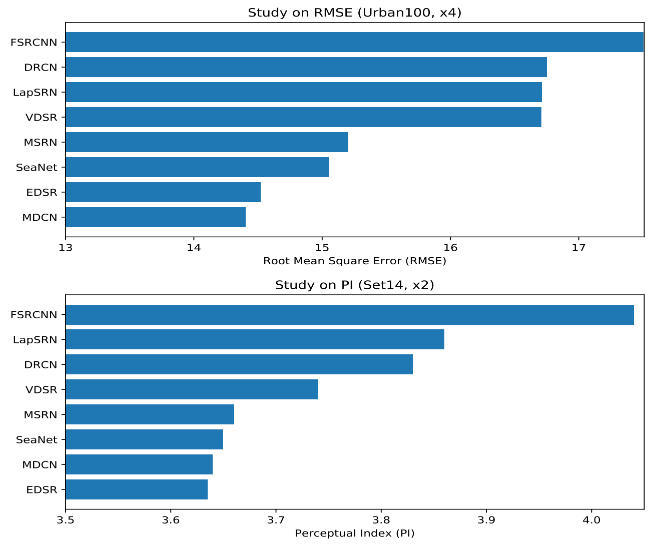


Fig. 13. RMSE and PI comparison with other SR methods. **A lower index indicates better image quality.**

C. Investigation of Image Naturalness

In Sec. IV, we evaluate the performance of MDCN from PSNR, SSIM, and visual effects. In order to further verify the distribution and perceptual quality of the reconstructed images, we introduce new indicators in this part, including Root Mean Square Error (RMSE) and Perceptual Index [72] (PI). Among them, RMSE is used to measure the standard deviation of the difference between SR and HR images, and PI is a new criterion that bridges the visual effect with computable index. Specifically, PI is a perceptual quality index which is judged by the Ma and $NIQE$ scores:

$$PI = \frac{1}{2}((10 - Ma) + NIQE), \tag{18}$$

where Ma is the non-reference measures of Ma's [73] score and NIQE [74] is a natural image quality evaluator. Meanwhile, a lower perceptual index (PI) represents a better perceptual quality.

In Fig. 13, we provide the results of RMSE and PI. According to the top figure, we can clearly see that the RMSE result of MDCN is much lower than other methods. This means that the SR images reconstructed by MDCN have a closer data distribution to real HR images, in other words, the reconstructed images have higher quality. For the below one, we can observe that the PI results of EDSR [16] and MDCN are smaller than others and the performance of them

TABLE X
IMAGE SEGMENTATION PERFORMANCE. DUE TO PAGE LIMIT, ONLY 5 CATEGORIES ARE PROVIDED AND 'MEAN IOU' DENOTES THE AVERAGE ACCURACY OF 19 CATEGORIES.

Evaluation	building	traffic light	sky	car	truck	Mean IoU
Bicubic	41.38	15.08	47.82	59.03	20.59	27.17
MDCN (Ours)	55.59	25.08	81.34	59.63	27.71	35.61
Original Image	63.15	26.26	82.16	65.08	29.97	37.19

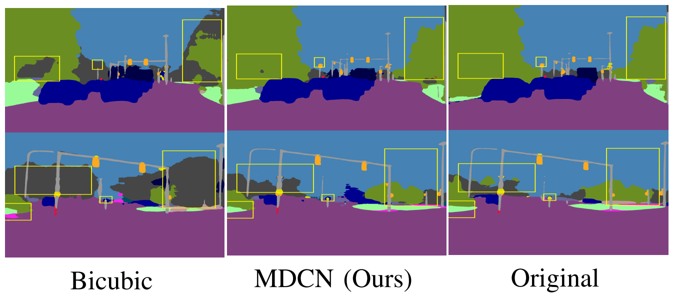


Fig. 14. Comparison of image segmentation results (x4).

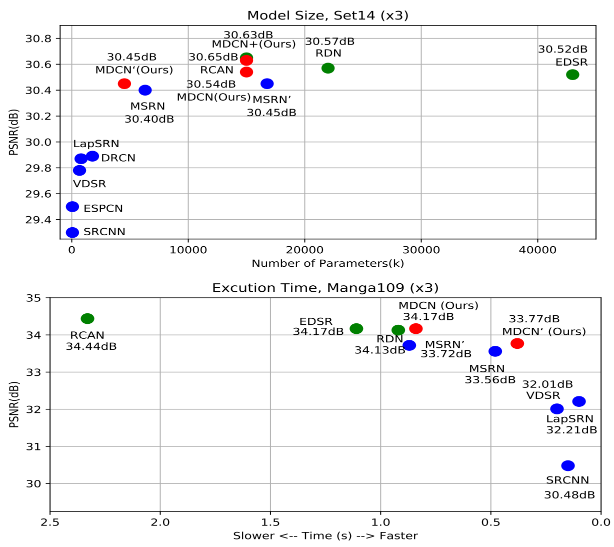


Fig. 15. Investigations of the model size and execution time.

are very close. This indicates that the images reconstructed by EDSR and MDCN have better perceptual quality and image naturalness. All these results fully manifest that our MDCN can achieve competitive performance.

D. Exploring on High-level Task

As we know, the quality of SR images will seriously affect the accuracy of high-level visual tasks such as image segmentation. In this part, we evaluate the performance of image segmentation to prove it. We use DRN-38 [75] as our segmentation model and Cityscapes [76] (foggy driving dataset) as the test dataset. Firstly, we downsample (Bicubic) the original image to obtain the LR input with the factor of $\times 4$. Then, we use Bicubic and our MDCN to reconstruct the corresponding SR image. Finally, we use DRN-38 [75] to segment the reconstructed SR image. In TABLE X, we provide the IoU results of these methods and we show some segmentation results in Fig. 14. In order to show the accuracy of the segmentation, we mark some areas with green rectangular boxes. We hope the segmentation result to be consistent with the result marked in "Original", which means the better quality of the reconstructed image. According to the figure, we can observe that there are a lot of gray areas in the "Bicubic", which represents the objects are marked incorrectly. In contrast, our MDCN obtains better segmentation results. This demonstrates that MDCN can reconstruct high-quality images and the quality of SR images will seriously affect the accuracy of high-level visual tasks. However, it cannot be ignored that there is still a large gap between the image reconstructed by MDCN and the original image. This means that SISR technology can be further improved. We will explore the performance of more SISR models on image segmentation or other high-level tasks in future works. Meanwhile, we aim to combine the feedback from high-level tasks to further boost model performance.

E. Model Size and Execution Time

Increasing the depth of the network is the easiest way to improve model performance. Therefore, some large size models have been proposed in recent years such as EDSR [16] and RDN [50]. However, it cannot be ignored that these models are also accompanied by numerous parameters. It means that these models require more storage space, computing resources, and execution time. In Fig. 15, we show the comparison of model parameters and execution time. Among them, red dots represent our MDCN', MDCN, and MDCN+, respectively. According to the figure, we can draw the following conclusions: (i) Compared with lightweight SR models (e.g., SRCNN, VDSR, and MSRN), the performance of MDCN is greatly improved; (ii) Compared with large models (e.g., EDSR and RDN), MDCN achieves close or better results with fewer parameters and less execution time; (iii) Compared with RCAN, we can find that the performance of RCAN is slightly better than MDCN. However, it should not be ignored that the execution time of RCAN is 3 times of MDCN. Furthermore, MDCN can be suitable for multiple upsampling factors (x2, x3, and x4) without any re-training. This means that MDCN can save lots of storage space, training time, and computational overhead. In summary, MDCN achieves a well balance between model performance, model size, and execution time.

VII. DISCUSSION

Although MDCN is an efficient and general SR model, it still has some limitations:

(1) In this paper, we focus on the reconstruction effect of MDCN on simulated degradation (e.g., Bicubic downsampling) images and verify its effectiveness from multiple indicators (e.g., PSNR, SSIM, RMSE, PI, and visual effect). It is worth noting that the task of real image super-resolution is more difficult due to the unknown and random degradation modes. However, this does not mean that our research is meaningless. Like previous works [10]–[21], [25]–[32], we

concentrate on the design of efficient and universal network structure. Recently, some methods [77]–[80] have been proposed for real image super-resolution. These models use real-world SR datasets for training, thus the model can achieve real image super-resolution. This manifests that MDCN can also be implemented to real image super-resolution by fine-tuning on the real SR dataset. We will explore the performance of MDCN on real images in future works.

(2) To make MDCN applicable to multiple upsampling factors without any re-training, we introduce the dynamic reconstruction block (DRB) to learn the inter-scale correlation between different upsampling factors. However, due to the limitations of sub-pixel convolutional layer, this method cannot directly handle non-integer factors. In order to remdy this problem, we suggest combining our MDCN with Bicubic to handle any upsampling factors, including non-integer factors. Specifically, we first utilize MDCN to perform integer magnification, and then use Bicubic to adjust the non-integer part. For example, for the factor of $\times 3.2$, use MDCN to process $\times 3$ first, then apply Bicubic to adjust it to the $\times 3.2$. Although the strategy is simple, it is very effective. At the same time, the performance of this strategy far exceeds the method of using Bicubic to directly enlarge the image. We also notice that some researchers [81], [82] claim that their models can deal with arbitrary upsampling factors. In the future work, we will further explore the reliability of these strategies and improve our MDCN to handle arbitrary upsampling factors.

VIII. Conclusions

In this paper, we propose a Multi-scale Dense Cross Network (MDCN) for SISR. MDCN is a robust SR model that can be applied to multiple upsampling factors without any re-training. Specifically, the proposed multi-scale dense cross blocks (MDCB), hierarchical feature block (HFDB), and dynamic reconstruction block (DRB) together form the MDCN. Among them, MDCB aims to fully detect and utilize local and multi-scale features, HFDB focuses on maximizing distillation and uses hierarchical features, and the introduced DRB enables the model to learn inter-scale correlations between different upsampling factors. Extensive evaluations demonstrate that MDCN can achieve competitive results with fewer parameters and less execution time, which achieves an excellent balance between model size and performance. Furthermore, MDCN can be easily expanded to other low-level computer vision tasks such as image denoising, image dehazing, and image enhancement. We will further verify the performance of MDCN on other image restoration tasks in future works.

References

[1] Lei Zhang and Xiaolin Wu. An edge-guided image interpolation algorithm via directional filtering and data fusion. *IEEE Transactions on Image Processing*, 15(8):2226–2238, 2006.

[2] Xianming Liu, Debin Zhao, Ruiqin Xiong, Siwei Ma, Wen Gao, and Huifang Sun. Image interpolation via regularized local linear regression. *IEEE Transactions on Image Processing*, 20(12):3455–3469, 2011.

[3] Weisheng Dong, Lei Zhang, Rastislav Lukac, and Guangming Shi. Sparse representation based image interpolation with nonlocal autoregressive modeling. *IEEE Transactions on Image Processing*, 22(4):1382–1394, 2013.

[4] Yunfeng Zhang, Qinglan Fan, Fangxun Bao, Yifang Liu, and Caiming Zhang. Single-image super-resolution based on rational fractal interpolation. *IEEE Transactions on Image Processing*, 27(8):3782–3797, 2018.

[5] Radu Timofte, Vincent De Smet, and Luc Van Gool. Anchored neighborhood regression for fast example-based super-resolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1920–1927, 2013.

[6] Radu Timofte, Vincent De Smet, and Luc Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *ACCV*, pages 111–126, 2014.

[7] Min-Chun Yang and Yu-Chiang Frank Wang. A self-learning approach to single image super-resolution. *IEEE Transactions on Multimedia*, 15(3):498–508, 2012.

[8] Zhiliang Zhu, Fangda Guo, Hai Yu, and Chen Chen. Fast single image super-resolution via self-example learning and sparse representation. *IEEE Transactions on Multimedia*, 16(8):2178–2190, 2014.

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

[10] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision*, pages 184–199, 2014.

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

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

[13] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1637–1645, 2016.

[14] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. 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*, 2016.

[15] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 624–632, 2017.

[16] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1132–1140, 2017.

[17] Ying Tai, Jian Yang, and Xiaoming Liu. Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, page 5, 2017.

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

[19] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1664–1673, 2018.

[20] Wei Han, Shiyu Chang, Ding Liu, Mo Yu, Michael Witbrock, and Thomas S Huang. Image super-resolution via dual-state recurrent networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1654–1663, 2018.

[21] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Learning a single convolutional super-resolution network for multiple degradations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[22] Zewei He, Siliang Tang, Jiangxin Yang, Yanlong Cao, Michael Ying Yang, and Yanpeng Cao. Cascaded deep networks with multiple receptive fields for infrared image super-resolution. *IEEE Transactions on Circuits and Systems for Video Technology*, 29:2310–2322, 2019.

[23] Huapeng Wu, Zhengxia Zou, Jie Gui, Wen-Jun Zeng, Jieping Ye, Jun Zhang, Hongyi Liu, and Zhihui Wei. Multi-grained attention networks for single image super-resolution. *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.

[24] Yifan Zuo, Qiang Wu, Yuming Fang, Ping An, Liqin Huang, and Zhifeng Chen. Multi-scale frequency reconstruction for guided depth map super-

resolution via deep residual network. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(2):297–306, 2019.

[25] Yanting Hu, Jie Li, Yuanfei Huang, and Xinbo Gao. Channel-wise and spatial feature modulation network for single image super-resolution. *IEEE Transactions on Circuits and Systems for Video Technology*, 2018.

[26] Feng Li, Huihui Bai, and Yao Zhao. Filternet: Adaptive information filtering network for accurate and fast image super-resolution. 2019.

[27] Zewei He, Yanpeng Cao, Lei Du, Baobei Xu, Jiangxin Yang, Yanlong Cao, Siliang Tang, and Yueting Zhuang. Mrfn: Multi-receptive-field network for fast and accurate single image super-resolution. *IEEE Transactions on Multimedia*, 2019.

[28] Juncheng Li, Yiting Yuan, Kangfu Mei, and Faming Fang. Lightweight and accurate recursive fractal network for image super-resolution. In *ICCVW*, 2019.

[29] Yifan Wang, Lijun Wang, Hongyu Wang, and Peihua Li. Resolution-aware network for image super-resolution. *IEEE Transactions on Circuits and Systems for Video Technology*, 29:1259–1269, 2019.

[30] Chao Xie, Weili Zeng, and Xiaobo Lu. Fast single-image super-resolution via deep network with component learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 29:3473–3486, 2019.

[31] Faming Fang, Juncheng Li, and Tieyong Zeng. Soft-edge assisted network for single image super-resolution. *IEEE Transactions on Image Processing*, 29:4656–4668, 2020.

[32] Juncheng Li, Faming Fang, Kangfu Mei, and Guixu Zhang. Multi-scale residual network for image super-resolution. In *European Conference on Computer Vision*, 2018.

[33] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *European Conference on Computer Vision*, 2018.

[34] Wenming Yang, Xuechen Zhang, Yapeng Tian, Wei Wang, Jing-Hao Xue, and Qingmin Liao. Deep learning for single image super-resolution: A brief review. *IEEE Transactions on Multimedia*, 2019.

[35] Zhihao Wang, Jian Chen, and Steven C. H. Hoi. Deep learning for image super-resolution: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[36] Feilong Cao and Huan Liu. Single image super-resolution via multi-scale residual channel attention network. *Neurocomputing*, 358:424–436, 2019.

[37] Yu Sang, Jinguang Sun, Simiao Wang, Yanfei Peng, Xinjun Zhang, and Zhiyang Yang. Multi-scale information distillation network for image super resolution in nsct domain. In *International Conference on Neural Information Processing*, pages 50–59, 2019.

[38] Jinghui Qin, Yongjie Huang, and Wushao Wen. Multi-scale feature fusion residual network for single image super-resolution. *Neurocomputing*, 379:334–342, 2020.

[39] Zhuangzi Li, Shanshan Li, Naiguang Zhang, Lei Wang, and Ziyu Xue. Multi-scale invertible network for image super-resolution. In *Proceedings of the ACM Multimedia Asia*, pages 1–6. 2019.

[40] Jae Woong Soh and Nam Ik Cho. Lightweight single image super-resolution with multi-scale spatial attention networks. *IEEE Access*, 8:35383–35391, 2020.

[41] Xinying Wang, Yingdan Wu, Yang Ming, and Hui Lv. Remote sensing imagery super resolution based on adaptive multi-scale feature fusion network. *Sensors*, 20(4):1142, 2020.

[42] Shu Zhang, Qiangqiang Yuan, Jie Li, Jing Sun, and Xuguo Zhang. Scene-adaptive remote sensing image super-resolution using a multiscale attention network. *IEEE Transactions on Geoscience and Remote Sensing*, 2020.

[43] Wei-Liang Zhang, Qin-Yan Zhang, Ji-Jiang Yang, and Qing Wang. Multi-scale network with the deeper and wider residual block for mri motion artifact correction. In *IEEE Annual Computer Software and Applications Conference*, volume 2, pages 405–410, 2019.

[44] Yongjun Qi, Junhua Gu, Weixun Li, Zepei Tian, Yajuan Zhang, and Juanping Geng. Pulmonary nodule image super-resolution using multi-scale deep residual channel attention network with joint optimization. *The Journal of Supercomputing*, 76(2):1005–1019, 2020.

[45] Yu Sang, Jinguang Sun, Simiao Wang, Xiangfu Meng, and Heng Qi. Contourlet transform based seismic signal denoising via multi-scale information distillation network. In *Pacific Rim International Conference on Artificial Intelligence*, pages 660–672, 2019.

[46] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[47] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[48] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1251–1258, 2017.

[49] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5835–5843, 2017.

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

[51] Mosin Russell, Ju Jia Zou, Gu Fang, and Weidong Cai. Feature-based image patch classification for moving shadow detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 29:2652–2666, 2019.

[52] Na Ra Han, Jigang Wu, Xiaozhao Fang, Wai Keung Wong, Yong Xu, Jian Yang, and Xuelong Li. Double relaxed regression for image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 30:307–319, 2020.

[53] Huiqun Wang, Di Huang, Kui Jia, and Yunhong Wang. Hierarchical image segmentation ensemble for objectness in rgb-d images. *IEEE Transactions on Circuits and Systems for Video Technology*, 29:93–103, 2019.

[54] Yiren Zhou, Thanh-Toan Do, Haitian Zheng, Ngai-Man Cheung, and Lu Fang. Computation and memory efficient image segmentation. *IEEE Transactions on Circuits and Systems for Video Technology*, 28:46–61, 2018.

[55] Wei Li, Hongliang Li, Qingbo Wu, Fanman Meng, Linfeng Xu, and King Ngi Ngan. Headnet: An end-to-end adaptive relational network for head detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 30:482–494, 2020.

[56] Lars Wilko Sommer, Tobias Schuchert, and Jürgen Beyerer. Comprehensive analysis of deep learning-based vehicle detection in aerial images. *IEEE Transactions on Circuits and Systems for Video Technology*, 29:2733–2747, 2019.

[57] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[58] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 3, 2017.

[59] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7132–7141, 2018.

[60] Kangfu Mei, Aiwen Jiang, Juncheng Li, Jihua Ye, and Mingwen Wang. An effective single-image super-resolution model using squeeze-and-excitation networks. In *International Conference on Neural Information Processing*, pages 542–553, 2018.

[61] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern RecognitionW*, pages 1110–1121, 2017.

[62] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In *The British Machine Vision Conference*, 2012.

[63] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *International Conference on Curves and Surfaces*, pages 711–730, 2010.

[64] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 416–423, 2001.

[65] Yusuke Matsui, Kota Ito, Yuji Aramaki, Azuma Fujimoto, Toru Ogawa, Toshihiko Yamasaki, and Kiyoharu Aizawa. Sketch-based manga retrieval using manga109 dataset. *Multimedia Tools and Applications*, 76(20):21811–21838, 2017.

[66] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Trans-*

actions on Pattern Analysis and Machine Intelligence, 33(5):898–916, 2011.

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

[68] Radu Timofte, Rasmus Rothe, and Luc Van Gool. Seven ways to improve example-based single image super resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1865–1873, 2016.

[69] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[70] Chia-Yang Chang and Shao-Yi Chien. Multi-scale dense network for single-image super-resolution. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1742–1746, 2019.

[71] Kan Chang, Minghong Li, Pak Lun Kevin Ding, and Baoxin Li. Accurate single image super-resolution using multi-path wide-activated residual network. *Signal Processing*, page 107567, 2020.

[72] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6228–6237, 2018.

[73] Chao Ma, Chih-Yuan Yang, Xiaokang Yang, and Ming-Hsuan Yang. Learning a no-reference quality metric for single-image super-resolution. *Computer Vision and Image Understanding*, 158:1–16, 2017.

[74] Anish Mittal, Rajiv Soundararajan, and Alan C Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal Processing Letters*, 20(3):209–212, 2012.

[75] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[76] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016.

[77] Jianrui Cai, Shuhang Gu, Radu Timofte, Lei Zhang, and et al. Ntire 2019 challenge on real image super-resolution: Methods and results. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[78] Du Chen, Zewei He, Anshun Sun, Jiangxin Yang, Yanlong Cao, Yanpeng Cao, Siliang Tang, and Michael Ying Yang. Orientation-aware deep neural network for real image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[79] Shangqi Gao and Xiahai Zhuang. Multi-scale deep neural networks for real image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[80] Thomas Köhler, Michel Bätz, Farzad Naderi, André Kaup, Andreas Maier, and Christian Riess. Toward bridging the simulated-to-real gap: Benchmarking super-resolution on real data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[81] Xuecai Hu, Haoyuan Mu, Xiangyu Zhang, Zilei Wang, Tieniu Tan, and Jian Sun. Meta-sr: A magnification-arbitrary network for super-resolution. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1575–1584, 2019.

[82] Longguang Wang, Yingqian Wang, Zaiping Lin, Jungang Yang, Wei An, and Yulan Guo. Learning for scale-arbitrary super-resolution from scale-specific networks. 2020.

Juncheng Li received the B.Sc. degree in Computer Science and Technology from Jiangxi Normal University, Nanchang, China, in 2016. He is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, East China Normal University, Shanghai, China. His main research interests includes image restoration, computer vision, deep learning, and medical image processing.

Faming Fang received the Ph.D. degree in Computer Science from East China Normal University, Shanghai, China, in 2013. He is currently a Professor with the School of Computer Science and Technology, East China Normal University. His main research area is image processing using the variational methods, PDEs and deep learning.

Jiaqian Li received the B.Sc. degree in Computer Science and Technology from Zhejiang Normal University, Jinhua, China, in 2018. She is pursuing the M.Phil degree with the School of Computer Science and Technology, East China Normal University, Shanghai, China. Her research interests includes low-light image enhancement and image processing.

Kangfu Mei received the B.Sc degree in Computer Science and Technology from Jiangxi Normal University, Nanchang, China, in 2019. He is currently pursuing the M.Phil degree with the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, China. His main research interests includes image generation, image processing, and computational photography.

Guixu Zhang received the Ph.D. degree from the Institute of Modern Physics, Chinese Academy of Sciences, Lanzhou, China, in 1998. He is currently a Professor with the School of Computer Science and Technology, East China Normal University, Shanghai, China. His research interests includes hyperspectral remote sensing, image processing, and artificial intelligencests are image processing, machine learning and scientific computing.