SRUNet: Stacked Reversed U-shape Network for Lightweight Single Image Super-Resolution

ZHIIYI FENG1,2, FEIFEI LEE1,2,*, (Member, IEEE), and QIU CHEN3,*, (Member, IEEE)
1Shanghai Engineering Research Center of Assistive Devices, University of Shanghai for Science and Technology, Shanghai 200093, China
2Rehabilitation Engineering and Technology Institute, University of Shanghai for Science and Technology, Shanghai 200093, China
3Graduate School of Engineering, Kogakuin University, Tokyo 163-8677, Japan
*
Corresponding author: Feifei Lee (feifeilee@ieee.org) and Qiu Chen (q.chen@ieee.org)

This research is partially supported by The Programme for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning, and also partially supported by JSPS KAKENHI Grant number 22K12079.

ABSTRACT In recent years, for relieving the heavy computational cost, lightweight models have been successfully applied to single image super-resolution (SISR) task. However, most lightweight models adopt low-resolution images as input and apply transposed convolution or subpixel convolution only in the tail of the model to reconstruct the super-resolution image which neglect to fully utilize the multi-scale features. To resolve this problem, we propose a stacked reversed U-shape network (SRUNet) to further extract and utilize the multi-scale features at different resolutions. In detail, SRUNet consists of shallow feature extraction (SFE), stacked reversed U-shape module (SRUM), multi-scale backward fusion module (MSBFM) and feature refinement module (FRM). Instead of upsampling the feature map at the tail of the model, we perform the upsampling and downsampling operation progressively and iteratively by the stacked reversed U-shape module to extract richer multi-scale features. Furthermore, for archiving better use of multi-scale features, a scale-wise dense connection with residual channel-wise attention and multi-scale backward fusion is added to the network. The fused super-resolved features are refined by the feature refinement module and reconstructed to the image. Extensive experiments demonstrate that our model can achieve competitive performance compared with the state-of-the-art methods. When scaling factor is 4, SRUNet achieves the highest SSIM performance in all benchmarks and it takes 29.2ms to process per image in Urban100 dataset.

INDEX TERMS Attention mechanism, dense connection, lightweight, multi-scale mechanism, single image super-resolution.

I. INTRODUCTION

Single image super-resolution (SISR) is a low-level computer vision task which refers to recover the high-resolution (HR) image from its low-resolution (LR) counterpart. SISR is a typical ill-posed problem since there are many potential HR images correspond to a single LR image. Even so, SISR technique has employed in various domains, including surveillance imaging [1], medical imaging [2] and portable devices photography [3].

Due to the rapid development of deep learning methods, recent years, deep learning methods are widely used to perform SISR. Dong et al. [4] are pioneers to introduce a convolutional neural network (CNN) with three layers for SISR which outperforms the traditional methods (such as the bicubic interpolation). Subsequent models [5], [6], [7], [8] further enhance the performance of SISR but they are accompanying heavier computation cost. For relieving this problem, lightweight models [9], [10], [11], [12], [13], [14], [15], [16], [17] are proposed and achieve reliable performance with less parameters. Most existing lightweight models are formed on the post-upsampling architecture, which only applies the upsampling operation at the tail of the model. Thus, these models can only extract and utilize the features on LR scale. Even though models like MADNet [15] propose the building blocks which utilize the multi-scale features, the post-upsampling structure still constrain the ability of models to leverage multi-scale features at different intermediate scales. Besides, some SISR models construct on progressively upsampling architecture to tackle large scaling factors and utilize the multi-scale feature. For example, LapSRN [18] first employs the Laplacian pyramid to construct the progressively upsampling structure to reconstruct the SR image. SUSR [19] and DRN [8] employ the U-shape structure based on U-Net [20] to extract multi-scale features and utilize the feature on SR scale to reconstruct the SR image. The difference is that the structure of DRN is in a single U-shape, but SUSR construct the U-shape blocks as the building blocks and stack these blocks to extract multi-scale features. However, DRN use the bicubic interpolated LR image as input which destroys the end-to-end...
end learnable manner of the network. The U-shape block in SUSR first upscale the input feature map with the scaling factor of task and process the feature on U-shape manner. After that, a compression block behind the U-shape block downscale the feature map which on SR scale back to LR scale. As reported in [11], when the scaling factor is large, these layers will suffer from the sudden size change of features maps which may leads to sudden shock to the network resulted in training unstable. Furthermore, SUSR reconstruct the SR image only using the outputs of U-shape blocks which may neglect to fully leverage the multi-scale features.

In this paper, we propose a stacked reversed U-shape network (SRUNet) composed of the reversed U-shape blocks (RUBs) to fully utilize the multi-scale features and make our network end-to-end learnable. Our SRUNet consists of four parts, which are shallow feature extraction (SFE), stacked reversed U-shape module (SRUM), multi-scale backward fusion module (MSBFM) and feature refinement module (FRM), respectively. Different from the U-shape block in SUSR [19], both the input and output of our proposed RUB are on LR scale. Instead of directly upscaling the feature map to the SR scale with the first layer, upscale and downscale are progressively taken place in the RUB to relieve the effect of large scaling factor. Benefit from this structure, our network can directly take the LR as input and don’t need to interpolate the LR image to the SR scale first as in [8]. As far as we know, iterative up-and-down sampling architecture is not yet employed in lightweight SISR models. Thus, it may be the first attempt to employ iterative up-and-down sampling architecture in lightweight SISR model. For enhancing the ability of utilizing the multi-scale features, based on the dense connection proposed in [21], scale-wise dense connection is added. Furthermore, residual channel-wise attention proposed in [19] follow by a $1 \times 1$ convolution is added to fuse the scale-wise dense features before passing to RUB. In order to fully leverage the multi-scale features extracted by the RUBs, we employ a multi-scale backward fusion module based on the backward fusion mechanism proposed in [16] to fuse them. The fused feature is refined by the RFB proposed in [14] and reconstruct to SR image. The proposed model is illustrated in Fig. 1. In brief, our contributions are in five-folds:

- We propose a novel lightweight model with stacked reversed U-shape blocks for single image super-resolution task, namely SRUNet.
- The proposed RUBs progressively upscale and downscale the feature maps for multi-scale feature extraction and relieving the effect of large scaling factor.
- For further leveraging the multi-scale features, scale-wise dense connection and residual channel-wise attention are added.
- A MSBFM is introduced to fuse the multi-scale features, so as to reconstruct the SR image with fully utilizing the multi-scale features.
- Extensive experimental results demonstrate our model is competitive compared with the state-of-
the-art methods. When scaling factor is 4, it takes 29.2ms for SRUNet to process per image in Urban100 dataset.

The remainder of this paper is organized as follows. In Section II, we provide the relevant works on proposed method. In Section III, we introduce the architecture of our model in detail. In Section IV, we show the implement details and discuss the parameter setting then provide the ablation study as well as comparison with the state-of-the-art methods. In Section V, we conclude the proposed method.

II. RELATED WORKS

A. DEEP CNN-BASED SINGLE IMAGE SUPER-RESOLUTION

From the perspective of the model framework, deep CNN-based SISR models can be divided into four categories: pre-upscaling, post up-sampling, progressively upsampling and iterative up-and-down sampling [22]. In pre-upscaling framework, the image is upsampled to desired scale before input to the network. Generally speaking, bicubic interpolation would be adopted as the upscaling method. This framework let model design easy since a single model can perform SR with arbitrary scaling factors. Thus, it is first adopted by Dong et al. [4], the first CNN-based SISR model, to construct a three layers CNN for SISR. Based on it, Kim et al. [5] add a global skip connection in the network to ease the training. Since the pre-upscaling method use the interpolated LR image as input, feature maps in network have same size with the SR image which resulted in large memory space cost. For relieving this problem, post-upscaling structure which place the upsampling layer at the tail of the network is employed in subsequent models. FSRCNN [23] directly adopts LR image as input and upscale the feature maps to desired size with a transposed convolutional layer. Different from FSRCNN, Shi et al. [24] introduce a sub-pixel convolutional layer in ESPCN for upsampling which is widely used by recent models [7], [8]. When scaling factor become large, models with aforementioned architectures will suffer from performance degradation since they only process features on a single scale. To resolve this problem, Lai et al. [18] construct a model by progressively upsampling the feature maps and reconstruct the SR image by Laplacian pyramid. By upsampling the feature maps progressively, the large scaling factor is divided into multiplication of several smaller scaling factors which not only ease the training process but also enhance the SR image quality. Different from these methods, Haris et al. [25] propose the DBPN which iteratively upsample and downsample the feature map so as to learn better representations of various mapping between LR and HR.

In pursuit of higher image quality, benefited from the residual learning [26], models are getting deeper [6], [7], [27]. But accompanying with the increment of depth, computation complexity and space consumption increased. For achieving the balance of performance and consumption, recent years, lightweight models are attached more attention. DRCN [9] and DRRN [10] construct the model in a recursive scheme that make the parameters reusable. Ahn et al. [17] propose the CARN which composed of cascading blocks so as to leverage the multi-level features. Hui et al. [12] propose the IDN using the group convolution and combines the extracted features by long and short skip connections to learn better representation. Hui et al. [13] further propose the IMDN composed of information multi-distillation blocks which combines multiple receptive fields to learn hierarchical features. Tian et al. [14] propose the CFSRCNN which add a feature refinement module to refine the features on SR scale and form a coarse-to-fine scheme. Lan et al. [15] propose the MADNet which combines the dilation convolution and simple convolution to extract the multi-scale features for image reconstruction. Recently, Luo et al. [16] propose the lattice block which implicitly includes multiple combinations of the residual blocks in order to enhance the representation ability of the network.

B. U-SHAPE NETWORK

The first U-shape network, U-Net, is proposed by Ronneberger et al. [20] for semantic segmentation. U-Net first abstract the compact features in bottom-up scheme and employ an upper-down branch to map the compact features back to origin scale. More than that, U-Net also employs several skip connections to combine the low-level features with the high-level features. These structures make U-Net performs well on semantic segmentation task. Due to its great performance, several models based on it to perform SR are proposed. Mao et al. [28] propose a U-shape network which downscale and upscale the feature maps and symmetrically connect layers in these two parts achieved good performance on image restoration. Guo et al. [8] construct a U-shape network to progressively fetch the feature maps on different scale in order to reconstruct the intermediate images. More than that, they further add a downscale model which progressively downscale the SR image to the LR resolution to form a dual regression learning scheme. Different with these, Zhu et al. [19] construct the U-shape blocks and stack them together which iteratively upscale and downscale the feature maps so as to extract the multi-scale features. Inspired by these, combining the intrinsic of SR task, we propose the reversed U-shape block which first progressively upscale the feature maps to SR scale and then downscale back to the LR scale.

C. ATTENTION MECHANISM

Attention mechanism can urge the model focus on the important features so as to improve the efficiency of the network. Recent years, attention mechanism is widely used in computer vision tasks [29], [30] including image super-resolution. Based on [29], Zhang et al. [7] employ the channel attention mechanism considering the relationship between channels to highlight the important channels. Liu et al. [31] propose a non-local attention mechanism that considers the relationship between a point and its neighborhood to highlight the important positions. Dai et al. [32] propose a
second order attention mechanism, which takes into account the second-order statistic of features for more discriminative representation. Zhu et al. [19] propose a residual channel-wise attention mechanism by adding a skip connection based on channel attention in [7] to highlight important features while maintaining its good properties. Luo et al. [16] utilize the attention mechanism to calculate the combination weight to decide the equivalent structure of the lattice blocks, thereby enhancing the representation ability of the network. Inspired by these, we employ the residual channel-wise attention in our proposed reversed U-shape block to promote the efficiency.

D. MULTI-SCALE REPRESENTATION

Generally speaking, objects in images have different scales, thus need multi-scale representation and the key to multi-scale representation is the combination of different receptive field sizes. Szegedy et al. [33] propose the Inception blocks which combines the feature maps produced by convolution with different kernel sizes. Since multi-scale feature is essential to SISR, recent years multi-scale representation is widely used in SR models. Li et al. [34] propose the multi-scale residual block composed of two types of convolutional layers which kernel size is different and extract the multi-scale feature by learning the different combination types of extracted features with a butterfly structure. Shang et al. [35] introduce the receptive filed block which based on human visual intrinsic. The receptive field block is composed of different combinations of dilatation convolutions and simple convolutions which equivalent to different receptive fields. The multi-scale feature is produced by employing a channel-wise concatenation on the features. Lan et al. [15] also employ a scheme similar to receptive block, but employed more combinations of the receptive fields.

These multi-scale representation methods [33], [34], [35], [15] work well, but they only extract multi-scale features based on a single resolution of the feature map which neglect to leverage hierarchical features. Thus, we propose the reversed U-shape block to leverage the multi-scale features on different scale of the feature maps and employ the scale-wise dense connection to further leveraging the multi-scale features. To some extent, multi-level features also can be seen as multi-scale features. The multi-level features are extracted by different depths of the network while the increasing of depth, in general, accompanying with the enlargement of the receptive field. Hence, the dense connection can combine both multi-scale and multi-level features.

III. PROPOSED METHOD

A. NETWORK STRUCTURE

In this part, we will describe our proposed SRUM in detail. As shown in Fig. 1, our model consists of four parts: shallow feature extraction (SFE), stacked reversed U-shape Module (SRUM), multi-scale backward fusion module (MSBFM) and feature refinement module (FRM). We directly adopt the LR image $I_{LR}$ as input and similar to recent works, use a single 3×3 convolutional layer without activation function for SFE which is formulated as

$$F_{SF} = H_{SFE}(I_{LR})$$  \hspace{1cm} (1)

where the $H_{SFE}$ denotes the shallow feature extraction function and $F_{SF}$ denotes the extracted shallow feature. Then, the shallow feature is sent into the SRUM to fetch the multi-scale features which is formulated as

$$F_{SRUM} = H_{SRUM}(F_{SF})$$  \hspace{1cm} (2)

where $H_{SRUM}$ denotes the SRUM and $F_{SRUM}$ denotes the features produced by SRUM. It is worth nothing that $F_{SRUM}$ is a group of features containing all outputted features of RUBs in SRUM. For fully utilizing these multi-scale features, MSBFM is employed and formulated as

$$F_{SR} = H_{MSBFM}(F_{SRF}, F_{SRUM})$$  \hspace{1cm} (3)

where $H_{MSBFM}$ denotes the MSBFM and $F_{SR}$ is the coarse super-resolved features. Finally, a feature refinement module is appended at the end of the network to refine the coarse super-resolved features and reconstruct the final SR image. The FRM can be formulated as

$$I_{SR} = H_{FRM}(F_{SR}) = H_{SRUN}(I_{LR})$$  \hspace{1cm} (4)

where $H_{FRM}$ denotes the FRM, $I_{SR}$ denotes the output SR image and $H_{SRUM}$ denotes the SRUM.

We adopt the mean absolute error (MAE) loss function for optimization. Given a set of image patches for training: $\{I_{LR}^{i}, I_{HR}^{i}\}^N_{i=1}$, where $N$ denotes the number of patches, the MAE loss can be formulated as

$$L = \frac{1}{N} \sum_{i=1}^{N} \|H_{SRUM}(I_{LR}^{i}) - I_{HR}^{i}\|_1$$  \hspace{1cm} (5)

B. REVERSED U-SHAPE BLOCK

Next, we describe our proposed RUB in detail. In order to divide a large scaling factor into the multiplications of several smaller scaling factors, we symmetrically employ several upscale blocks and downscale blocks, so as to progressively upsample the feature maps to the desired scale and downscale to the origin scale. Simply speaking, an upscale block would upsample the input feature map to two times of width and height and a downscale block would downscale this feature map back to its origin scale. For dealing with the $\times 3$ scaling task, the scaling factor of a single upscale block is $3$. To be specific, if the scaling factor of a task is $s$, there will be $\lceil\log_2 s \rceil$ upscale blocks and downscale blocks, respectively. With these blocks, the RUB can be divided into $\lceil\log_2 s \rceil + 1$ phases which include feature maps with different scales, respectively. The structure of upscale block and downscale block is as shown in Fig. 1. Following [8], we employ a sub-pixel convolutional layer followed by a 3×3 convolution as upscale block and employ a strided convolution followed by LeakyReLU and a 3×3 convolution to form the downscale block. But different from it, we employ the ReLU in upscale block for enhancing the representation ability. Skip connections are also added to constitute the
residual learning manner. The shallow feature maps are transmitted to the deep feature maps with same size and fused with element-wise add.

Furthermore, we add a scale-wise dense connection between RUBs and employ residual channel-wise attention feature fusion blocks (RCAFB) in RUB for feature fusion. The RCAFB is composed by three parts: concatenation, residual channel-wise attention (RCA) and 1×1 convolution. The RCAFB receives the dense features from prior RUBs in same phase and the feature maps from prior phase (if has). The RCAFB can be formulated as

$$F^p_{RFAB} = \begin{cases} H^p_{RFAB}\left(F^p_{RB1}, F^p_{RB2}, \ldots, F^p_{RBi-1}\right), & p = 0 \\ H^p_{RFAB}\left(F^p_{RB1}, F^p_{RB2}, \ldots, F^p_{RBi-1}, F^p_{RBi}\right), & p \neq 0 \end{cases}$$  (6)

where $F^p_{RB}$ denotes the outputted feature map of the $i$-th RUB belong to phase $p$, $F^p_{RBi}$ denotes the feature maps produced by the upscale block belonging to phase $p-1$ in the $i$-th RUB, $H^p_{RFAB}$ denotes the RCAFB belonging to phase $p$ in the $i$-th RUB and $F^p_{RFAB}$ is the feature produced by it.

After a channel-wise attention, RCA is used to highlight the important features and then fuse them with a 1×1 convolution (as shown in Fig. 2). RCA is proposed in SUSR [19] for highlighting the important features while maintaining the good properties of original features. The structure is illustrated in Fig. 2. The input which has $C$ channels will first map to a vector with $C$ channels using global average pooling (GAP). Then the first 1×1 convolution followed with ReLU maps the vector to $C/r$ channels, where $r$ refers to the decreasing rate. Generally speaking, decreasing rate is a number no less than 1. By adjusting $r$, one can control the number of intermediate feature maps. (We will further discuss $r$ in section IV.C.) The second 1×1 convolution is used to map the vector back to $C$ channels and use sigmoid to fetch the mask $s$. These processes can be formulated as

$$s = \sigma(W_2\left(\delta(W_1\left(f_{GAP}(F)\right))\right))$$  (7)

where $F_{in}$ denotes the input feature, $f_{GAP}$ denotes the GAP function, $W_1$ and $W_2$ denotes the 1×1 convolution, $\sigma$ and $\delta$ denotes the sigmoid and ReLU, respectively. Then, the output can be formulated as

$$F_{att} = s \cdot F_{in} + F_{in} = (1 + s) \cdot F_{in}$$  (8)

where $F_{att}$ denotes the feature maps of output. The values of elements in mask are range from $0, 1$ since the sigmoid function. After adding the skip connection, it is equivalently mapping the value domain of mask to $[0, 1, 2]$ so as to highlight the important features in input while maintain its origin good property. In [19], multiple residual channel-wise attention blocks are used to processing the feature maps produced by the building blocks in trunk, respectively. These produced feature maps are further fused with concatenation followed by 1×1 convolution for SR image reconstruction. Different from this, we employ the RCA to form a fusion block for receiving the dense features. Specifically, for fully observing the feature maps, we place the RCA between the concatenation and 1×1 convolution.

By stacking multiple RUBs to form the trunk of our network, features maps are iteratively up-and-down sampling so as to learn various representation.

C. MULTI-SCALE BACKWARD FUSION MODULE

In order to fully leverage the multi-scale features extracted by the RUBs, we construct a multi-scale backward fusion module (MSBFM) for progressively fuse and upscale the features from RUBs and produce the coarse features on SR scale (as illustrated in Fig. 1). MSBFM composed by several backward fusion blocks (BFB) (Fig. 3) which fuse the scale-wise features and upscale blocks that upscale the feature maps to next scale. The backward fusion structure is first proposed by Luo et al. [16] for enhancing the ability of feature leveraging while consuming less parameters. We append a 1×1 convolution at the back of it to constitute our BFB since the backward fusion structure directly outputs the final concatenated features. The BFB can be formulated as

$$F^p_{BF} = \begin{cases} Conv(F^p_{RB}), & i = n \\ Conv(Concat(F^p_{RBi}, F^p_{BFi+1})), & i \neq n \end{cases}$$  (9)

where $Conv$ denotes the 1×1 convolution, $Concat$ denotes the channel-wise concatenation, $F^p_{BFi}$ denotes the $i$-th feature produced in backward fusion block which belongs to phase $p$. For fusing all the scale-wise features, we first employ a BFB to fuse the features on LR scale, then add the shallow feature extracted by the SFE to the fused feature using element-wise addition. After that, an upscale block is employed to upscale the feature to next scale and add to the fused feature produced by the next BFB which fusing the features with the second
scale. We will repeat these processes for several times until the feature maps are mapped to the desired size. We only employ a subpixel layer in the last upscale block but other upscale blocks have the same structure with which in RUBs.

D. FEATURE REFINEMENT MODULE

Feature refinement module (FRM) is first proposed by Tian et al. [14] as one of the methods for overcoming the sudden shock incurred by the upsampling layer. We employ the FRM to refine the fused features on desired scale produced by MSBFM in order to reconstruct images with higher quality. The FRM here is consisted of four convolutional layers followed by ReLU and a convolutional layer for image reconstruction (illustrated in Fig. 1).

IV. EXPERIMENTAL RESULTS

A. DATASETS

We train our network on the training set of DIV2K [36] which contains 800 HR images (2K resolution). We employ bicubic interpolation to produce the LR images. During training, we randomly sample a $48 \times 48$ patch in the LR image and on the correspond position in HR image with a specific scaling factor. For data argumentation, we randomly perform 90° rotation, vertical and horizontal flipping on the patch. We employ four widely used benchmark datasets: Set5 [37], Set14 [38], BSD100 [39] and Urban100 [40] to evaluate the effect of proposed method. For comparison, we transforming the HR image and the SR image to the YCbCr color space and calculate the peak signal-to-noise ratio (PSNR) and structure similarity (SSIM) [41] on the Y channel between them.

B. IMPLEMENTATION DETAILS

We set up the parameters for all experiments reference to recent methods [8], [13], [19]. During the study of model setting and ablation analysis, we use the Adam optimizer [42] with $\beta_1=0.9$, $\beta_2=0.999$ and $\epsilon=10^{-8}$ to optimize the models. After performing experiments with different optimizers for our models, we choose RMSprop optimizer [43] with $\alpha=0.999$ to optimize our final model. The batch size is set to 16. The initial learning rate is set to $2 \times 10^{-4}$ and reduced by half each $2 \times 10^4$ iterations for total $6 \times 10^4$ iterations. Since our model for $\times 2$ and $\times 3$ scaling tasks only employ a single upscale block and a single downscale block in the RUB, we add two convolutional layers in the upscale block and the downscale block respectively in order to ensure the RUB has the same depth in our three final models. Our models are implemented under PyTorch framework and is trained on a single NVIDIA Tesla V100 GPU.

C. STUDY OF NETWORK SETTING

To investigate the efficiency of the network, we train networks with different setting of the filter number, the RUB number and the decreasing rate on the $\times 4$ scaling task. We denote the
TABLE 1. Quantitative evaluation on PSNR and SSIM of models with different setting on B, F and r. We provide the best result of the models on Set5, Set14, BSD100 and Urban100 on scaling factor ×4. The red color indicates the best result.

| Model  | Param(K) | Set5 PSNR/SSIM | Set14 PSNR/SSIM | BSD100 PSNR/SSIM | Urban100 PSNR/SSIM |
|--------|----------|---------------|----------------|------------------|-------------------|
| B3F16  | 136      | 31.80 / 0.8984 | 28.29 / 0.7738 | 27.37 / 0.7286   | 25.45 / 0.7643    |
| B4F16  | 173      | 31.90 / 0.8907 | 28.36 / 0.7759 | 27.41 / 0.7302   | 25.57 / 0.7686    |
| B5F16  | 213      | 31.92 / 0.8912 | 28.40 / 0.7769 | 27.43 / 0.7310   | 25.69 / 0.7716    |
| B5F32  | 843      | 32.13 / 0.8946 | 28.55 / 0.7810 | 27.53 / 0.7350   | 26.03 / 0.7842    |
| B5F48  | 1,893    | 32.33 / 0.8965 | 28.61 / 0.7828 | 27.61 / 0.7370   | 26.22 / 0.7902    |
| B5F32r8 | 861     | 32.17 / 0.8945 | 28.58 / 0.7809 | 27.55 / 0.7351   | 26.09 / 0.7847    |
| B5F32r4 | 897     | 32.19 / 0.8945 | 28.58 / 0.7812 | 27.55 / 0.7351   | 26.11 / 0.7855    |
| B5F32r1 | 1,110   | 32.17 / 0.8943 | 28.57 / 0.7810 | 27.56 / 0.7348   | 26.10 / 0.7846    |

TABLE 2. Quantitative evaluation on PSNR and SSIM of models trained with different optimizers. We provide the best result of the models on Set5, Set14, BSD100 and Urban100 on scaling factor ×4. The red color indicates the best result.

| Model  | Optimizer           | Set5 PSNR/SSIM | Set14 PSNR/SSIM | BSD100 PSNR/SSIM | Urban100 PSNR/SSIM |
|--------|---------------------|----------------|----------------|------------------|-------------------|
| B5F32  | SGD                 | 30.25 / 0.8596 | 27.32 / 0.7490 | 26.76 / 0.7076   | 24.21 / 0.7129    |
|        | SGD w/ momentum [44]| 31.02 / 0.8768 | 27.89 / 0.7632 | 27.08 / 0.7181   | 24.82 / 0.7383    |
|        | RMSprop [43]        | 32.12 / 0.8944 | 28.55 / 0.7812 | 27.53 / 0.7351   | 26.03 / 0.7838    |
|        | Adam [42]           | 32.13 / 0.8946 | 28.55 / 0.7810 | 27.53 / 0.7350   | 26.03 / 0.7842    |
|        | Adamax [42]         | 31.97 / 0.8919 | 28.41 / 0.7771 | 27.45 / 0.7316   | 25.71 / 0.7734    |

TABLE 3. Ablation study of the effect of the reversed U-shape block (RUB), feature fusion before residual channel-wise attention (RCA) and multi-scale backward fusion module (MSBFM). "\( \checkmark \)" denotes we employ this method in model while "\( \times \)" denotes we don't employ this method. We provide the best result of the models on BSD100 and Urban100 on scaling factor ×4. The red color indicates the best result.

| Model  | RUB | RCAFB | MSBFM | #Params(K) | BSD100 PSNR/SSIM | Urban100 PSNR/SSIM |
|--------|-----|-------|-------|-----------|------------------|-------------------|
| baseline| \( \times \) | \( \times \) | \( \times \) | 1,282      | 27.50 / 0.7339   | 25.88 / 0.7797    |
| 1st    | \( \checkmark \) | \( \times \) | \( \times \) | 816        | 27.54 / 0.7348   | 26.03 / 0.7838    |
| 2nd    | \( \times \) | \( \checkmark \) | \( \times \) | 1,292      | 27.50 / 0.7342   | 25.97 / 0.7821    |
| 3rd    | \( \times \) | \( \times \) | \( \checkmark \) | 1,287      | 27.50 / 0.7338   | 25.88 / 0.7800    |
| 4th    | \( \checkmark \) | \( \checkmark \) | \( \times \) | 824        | 27.55 / 0.7342   | 26.05 / 0.7826    |
| 5th    | \( \checkmark \) | \( \times \) | \( \checkmark \) | 826        | 27.55 / 0.7349   | 26.05 / 0.7837    |
| 6th    | \( \times \) | \( \checkmark \) | \( \checkmark \) | 1,297      | 27.52 / 0.7339   | 26.00 / 0.7832    |
| 7th    | \( \checkmark \) | \( \checkmark \) | \( \times \) | 784        | 27.52 / 0.7343   | 25.95 / 0.7816    |
| B5F32  | \( \checkmark \) | \( \checkmark \) | \( \times \) | 843        | 27.53 / 0.7350   | 26.03 / 0.7842    |

number of filters in convolutional layer as F, denote the number of RUBs as B and denote the decreasing rate as r. We first set r to 16 and train five models for investigating the effect of different B and F. We first constrain F as 16 and set B to 3, 4 and 5, then constrain B as 5 and set F to 32 and 48. These five models can be denoted as B3F16, B4F16, B5F16, B5F32 and B5F48, respectively. After B and F are determined as 5 and 32, we modify the r to 8, 4 and 1 and these three models are denoted as B5F32r8, B5F32r4, B5F32r1. The evaluated results on PSNR and SSIM can be found in Table 1. We also investigate the suitable optimizer based on B5F32 and the experiment results are as shown in Table 2.

1) STUDY OF RUBS NUMBER
Increasing the RUBs number in network can make network deeper. We compare the performance of B3F16, B4F16 and B5F16 for studying the effect of the RUBs number. One can find out that with the increasing of B, the performance of model increases. Changing B from 4 to 5 gain less enhancement on both PSNR and SSIM than changing B from 3 to 4 while accompanying with heavier parameter cost.

2) STUDY OF FILTERS NUMBER
More filters in convolutional layer will enhance the width of the network but lead to higher parameter cost. We compare the performance of B5F16, B5F32 and B5F48 to study the effect of filter number. With the increasing of F, the performance of model increases on great scale but accompanying with larger cost on parameters amount.

3) STUDY OF DECREASING RATE
The decreasing rate r in residual channel-wise attention controls the amount of intermediate feature maps. Thus, it is a key factor in the representation ability of the attention mechanism. We compare the performance of B5F32, B5F32r8,
TABLE 4. Comparison between proposed method and 8 state-of-the-art methods on ×2, ×3, ×4 scaling tasks. The average PSNR and average SSIM on each benchmark datasets are provided. The red and blue colors indicate the best and the second-best results, respectively.

| Method        | Scale | #Params | Set5 PSNR/SSIM | Set14 PSNR/SSIM | BSD100 PSNR/SSIM | Urban100 PSNR/SSIM |
|---------------|-------|---------|----------------|-----------------|------------------|-------------------|
| Bicubic       | -     | -       | 33.66 / 0.9299 | 30.24 / 0.8688  | 29.56 / 0.8431   | 26.88 / 0.8403    |
| SRCNN [45]    | 57K   | 32.75   | 32.45 / 0.9067 | 31.36 / 0.8879  | 30.41 / 0.9101   | 29.50 / 0.8946    |
| LapSRN [18]   | 813K  | 33.08   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| IDN [12]      | 553K  | 33.20   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| CARN [17]     | 1,592K| 33.51   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| SUSR [19]     | 1,200K| 33.61   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| CFSRCNN [14]  | 1,200K| 33.61   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| MADNet [15]   | 978K  | 33.61   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| SRUNet (Ours) | 978K  | 33.61   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| SRUNet+ (Ours)| 978K  | 33.61   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| Bicubic       | -     | -       | 30.39 / 0.8682  | 27.55 / 0.7742  | 27.21 / 0.7385   | 24.46 / 0.7394    |
| SRCNN [45]    | 57K   | 32.75   | 32.45 / 0.9067 | 31.36 / 0.8879  | 30.41 / 0.9101   | 29.50 / 0.8946    |
| IDN [12]      | 553K  | 33.08   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| CARN [17]     | 1,592K| 33.51   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| SUSR [19]     | 1,200K| 33.61   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| CFSRCNN [14]  | 1,200K| 33.61   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| MADNet [15]   | 978K  | 33.61   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| SRUNet (Ours) | 978K  | 33.61   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| SRUNet+ (Ours)| 978K  | 33.61   | 33.30 / 0.9148 | 32.08 / 0.8985  | 31.27 / 0.9196   | 31.92 / 0.9256    |
| Bicubic       | -     | -       | 28.42 / 0.8104  | 26.00 / 0.7027  | 27.21 / 0.7385   | 24.46 / 0.7394    |
| SRCNN [45]    | 57K   | 30.48   | 27.50 / 0.7513  | 26.90 / 0.7010  | 24.52 / 0.7221   | 23.14 / 0.6577    |
| LapSRN [18]   | 813K  | 31.54   | 28.19 / 0.7720  | 27.32 / 0.7270  | 25.21 / 0.7560   | 24.52 / 0.7221    |
| IDN [12]      | 553K  | 31.54   | 28.19 / 0.7720  | 27.32 / 0.7270  | 25.21 / 0.7560   | 24.52 / 0.7221    |
| CARN [17]     | 1,592K| 32.13   | 28.60 / 0.7806  | 27.58 / 0.7349  | 26.07 / 0.7837   | 25.21 / 0.7560    |
| SUSR [19]     | 1,200K| 32.05   | 28.57 / 0.7817  | 27.55 / 0.7351  | 26.01 / 0.7828   | 25.21 / 0.7560    |
| CFSRCNN [14]  | 1,200K| 32.05   | 28.57 / 0.7817  | 27.55 / 0.7351  | 26.01 / 0.7828   | 25.21 / 0.7560    |
| MADNet [15]   | 978K  | 32.26   | 28.57 / 0.7817  | 27.55 / 0.7351  | 26.01 / 0.7828   | 25.21 / 0.7560    |
| SRUNet (Ours) | 978K  | 32.26   | 28.57 / 0.7817  | 27.55 / 0.7351  | 26.01 / 0.7828   | 25.21 / 0.7560    |
| SRUNet+ (Ours)| 978K  | 32.26   | 28.57 / 0.7817  | 27.55 / 0.7351  | 26.01 / 0.7828   | 25.21 / 0.7560    |

B5F32r4 and B5F32r1 to investigate the effect of the decreasing rate. With the decreasing of $r$, the number of parameters increases but the performance does not simply improve. Compared with B5F32, modifying $r$ can improve the performance on PSNR, but achieve similar performance on SSIM in most benchmarks. B5F32r4 achieves the best performance on PSNR in this part.

4) STUDY OF OPTIMIZERS

A suitable optimizer can provide better guidance to the model during the training phase. In this part, we optimize the B5F32 model using five optimizers: SGD, SGD with momentum [44], RMSprop [43], Adam [42] and Adamax[42]. The experimental results are provided in Table 2. We also provide the loss value during training in Fig 5. The model optimized with Adam achieves the best PSNR performance on all datasets. The model optimized with RMSprop also achieves relatively similar performance and achieves the best SSIM performance on Set14 and BSD100 dataset. In Fig. 5, even though the loss with RMSprop (green line) decreasing slower than Adam at the preliminary stage of training, but it is in larger decreasing trend than Adam after 400K iterations.

![Figure 5](image-url)
Considering the performance and the number of parameters, we set B to 5 and F to 32 for each final model. Since we mainly focus on SSIM performance, we set $r$ to 4 when scaling factor is 4 for our final model. Since model with RMSprop seems can be further optimized, we employ RMSprop for all of our final model. The comparison between state-of-the-art methods and ours is provided in section IV.E.

### D. ABLATION ANALYSIS

For demonstrating the effect of reversed U-shape block (RUB), residual channel-wise attention feature fusion block (RCAFB) and multi-scale backward fusion module (MSBFM). The ablation study is operated on scaling factor ×4 and set the base feature number as 32, set block number as 5. In the baseline model, RUB is replaced by the combination of a single upscale block and a single downscale block, but convolutional layers are added in those blocks to keep the layer number as same as RUB. Scale-wise dense connection is also employed to the baseline model. Instead of fusing these features with the RCAFB (as our model), these features are first fused by channel-wise concatenation and 1×1 convolution and then employ the residual channel-wise attention mechanism. Furthermore, a backward fusion module is applied to fuse the features on the top scale. Shallow feature extraction and feature refinement module is also used in baseline module. The PSNR and SSIM results are provided in Table 3. It should notice that, experiments in this part is still using Adam optimizer [42].

1) REVERSED U-SHAPE BLOCK

For demonstrating the effect of the RUB, we put RUBs in the baseline model instead of the upscale and downscale blocks. As shown in row “1st”, after employing RUBs, the model can not only save 456K parameters, but also increase the SSIM by 0.0009 and 0.0041 on BSD100 and Urban100, respectively. This proves that our RUB can promote the performance efficiently. Actually, the stacked RUBs can be seen as another type of stacked U-shape blocks when turn one’s eyes to the connection of two RUBs and U-shape structure is powerful as discussed in section II.B. We also explore the effect of the scale-wise dense connection by employing the RUB without scale-wise dense connection on the baseline model. As shown in row “7th”, the model with RUB without scale-wise dense connection also works, but has lower performance on benchmarks compared with row “1st”. This result also proves that models with scale-wise dense connection can work.

2) MULTI-SCALE BACKWARD FUSION MODULE

We replace the backward fusion module with MSBFM in the baseline model to evaluate the effect of MSBFM. Suffer from the large scaling factor in the baseline model, employing only MSBFM in baseline model does not work (as shown in row “3rd”). But as row “5th” shows, after employing MSBFM and RUB together, the model achieves the best PSNR result
among these models. But the improvement is not as obvious as using RUB.

3) RESIDUAL CHANNEL-WISE ATTENTION FEATURE FUSION BLOCK

After employing RCAF based on the baseline model, we obtain the performance in row “2nd”. The PSNR and SSIM performance on Urban100 is increased by 0.09dB and 0.0024, respectively. This result implies that the model with RCAF can recover more accurate structural information. As shown in row “4th”, employing RCAF together with RUB also achieves the best PSNR performance among these models. Row “6th” shows that using RCAF with MSBFM can improve the performance compared with using only one of them.

Since we pay more attention on recovering accurate structure in the image, we apply both of RUB, RCAF and MSBFM in our final model and it achieves the best performance in SSIM on both BSD100 and Urban100.

E. COMPARISON WITH STATE-OF-THE-ART METHODS

We perform a comprehensive comparison on four official benchmark datasets with 10 state-of-the-art methods: Bicubic, SRCNN [45], LapSRN [18], IDN [12], CARN [17], SUSR [19], CFRCNN [14], MADNet [15], IMDN [13] and DRN [8]. Similar to [6], we also introduce the self-ensemble strategy to enhance the performance of SRUNet and denote the model with self-ensemble as SRUNet+. Self-ensemble strategy based on the self-similarity of the image. With the combinations of using horizontal flip, vertical flip and 90° rotation or not, single LR image can be augmented to 8 LR images and each image is fed into the network, respectively. After inference, reverse operation is performed on each image and obtain the final result by perform element-wise add followed by averaging on these images. We use the public released results of these methods for comparison which are shown in Table 4. It should notice that the SSIM results acquired in the original papers of SUSR and DRN contain only three significant digits, and the number of parameters is recalculated based on the implementation details described in the original papers. We list the results in PSNR and SSIM of DRN which also employs the U-shape structure for reference. However, DRN is not a lightweight model, with about ten times the number of parameters than other models, and it mainly benefits from its dual regression scheme. Therefore, we do not include DRN into the ranking. Since our model for 2 and ×3 scaling tasks only employ one upscale block and one downscale block in RUB, and the structure of our upscale block and downscale block is relatively simple, even if we enhance the depth of RUB, our method performs worse than IMDN on ×2 and ×3 scaling tasks. In contrast, SUSR which performs two upscale blocks and downscale blocks in a single U-shape block, outperforms our model relatively on the ×2 scaling task, especially on the Urban100 dataset. But SUSR suffers from the large scaling factor on the ×4 scaling task. SRUNet achieves the best SSIM performance on the ×4 scaling task, especially on the Urban100 dataset, which means our method is better at recovering the structural information. With the strong help of self-ensemble strategy, SRUNet+ achieves the best PSNR and SSIM performance on all benchmark when scaling factor is 4.

For demonstrating the effect of our method visually, we further provide the comparisons of SR images in the BSD and Urban100 datasets on the ×4 scaling task. From image “302008” in BSD100 in Fig. 4, we can observe that our method can recover the stripes on the collar more accurately. Besides, “img005” in Urban100 shows that our method can recover more proper shape and color of the windows and the walls. Furthermore, both images in Fig. 6 demonstrate that our method do better in recovering the vertical and horizontal lines than other methods. These experimental results show that our method can achieve competitive performance against the state-of-the-art methods.

V. CONCLUSION

In this paper, we propose a stacked reversed U-shape network (SRUNet) composed of the stacked reversed U-shape blocks (RUBs), which iteratively progressive up-and-down sampling the feature maps thus can extract various multi-scale features. In order to fully leverage the multi-scale features, we further add the scale-wise dense connection between RUBs and leverage the residual channel-wise attention to fuse the dense features. Furthermore, we employ the multi-scale backward fusion module for feature fusion and reconstruct the SR image by the feature refinement module. It may be the first attempt to employ iterative up-and-down sampling architecture in lightweight SISR model. It takes 24.2ms, 24.6ms and 29.6ms for SRUNet to process per image in Urban100 dataset when scaling factor is 2, 3 and 4, respectively. Experimental results on official benchmarks demonstrate that the performance of our method is competitive with the state-of-the-art methods.

REFERENCES

[1] J. Kim, G. Li, I. Yun, C. Jung, and J. Kim, “Edge and identity preserving network for face super-resolution,” Neurocomputing, vol. 446, pp. 11-22, Jul. 2021.
[2] J. Wang, Y. Chen, Y. Wu, J. Shi and J. Gee, “Enhanced generative adversarial network for 3D brain MRI super-resolution,” in Proc. IEEE/CVF Winter Conf. on Appl. of Comput. Vis. (WACV), May 2020, pp. 3616-3625.
[3] A. Lugmayr et al., “NTIRE 2020 challenge on real-world image super-resolution: methods and results,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshop (CVPRW), Jun. 2020, pp. 2058-2076.
[4] C. He, C. C. Loy, K. He, and X. Tang, “Learning a deep convolutional network for image super-resolution,” in Proc. Eur. Conf. Comput. Vis. (ECCV), Sep. 2014, pp. 184-199.
[5] J. Kim, J. K. Lee and K. M. Lee, “Accurate image super-resolution using very deep convolutional networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 1646-1654.
[6] B. Lim, S. Son, H. Kim, S. Nah and K. M. Lee, “Enhanced deep residual networks for single image super-resolution,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshop (CVPRW), Jul. 2017, pp. 1132-1140.
[7] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, “Image super-resolution using very deep residual channel attention networks,” in Proc. Eur. Conf. Comput. Vis. (ECCV), Sep. 2018, pp. 294-310.
VOLUME XX, 2017

[8] Y. Guo, J. Chen, J. Wang, Q. Chen, J. Cao, Z. Deng, Y. Xu, M. Tan, “Closed-loop matters: Dual regression networks for single image super-resolution,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 5406-5415.

[9] J. Kim, J. K. Lee and K. M. Lee, “Deeply-recurvational convolutional network for image super-resolution,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 1677-1645.

[10] Y. Tai, J. Yang and X. Liu, “Image super-resolution via deep recursive residual network,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 2790-2798.

[11] N. Ahn, B. Kang and K. Sohn, “Image super-resolution via progressive cascading residual network,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshop (CVPRW), Jun. 2018, pp. 904-912.

[12] Z. Hui, X. Wang, and X. Gao, “Fast and accurate single image super-resolution via information distillation network,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2018, pp. 723-731.

[13] Z. Hui, X. Gao, Y. Yang, X. Wang, “Lightweight image super-resolution with information multi-distillation network,” in Proc. 27th ACM Int. Conf. Multimedia (MM), Oct. 2019, pp. 2024-2032.

[14] C. Tian, Y. Xu, W. Zuo, B. Zhang, L. Fei and C.-W. Lin, “Coarse-to-fine CNN for image super-resolution,” IEEE Trans. Multimedia, vol. 23, pp. 1489-1502, Jun. 2021.

[15] R. Lan, L. Sun, Z. Liu, H. Lu, C. Pang and X. Luo, “MADNet: A fast and lightweight network for single-image super-resolution,” IEEE Trans. Cybern., vol. 51, no. 3, pp. 1443-1453, Mar. 2021.

[16] X. Luo, Y. Xie, Y. Zhang, Y. Qu, C. Li, Y. Fu, “LatticeNet: Towards lightweight image super-resolution with lattice block,” in Proc. IEEE Conf. Comput. Vis. (ECCV), Nov. 2020, pp. 272-289.

[17] N. Ahn, B. Kang, and K.-A. Sohn, “Fast, accurate, and lightweight super-resolution with cascading residual network,” in Proc. Eur. Conf. Comput. Vis. (ECCV), Sep. 2018, pp. 256-272.

[18] W. Lai, J. Huang, N. Ahuja and M. Yang, “Deep Laplacian pyramid networks for fast and accurate super-resolution,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 5835-5843.

[19] L. Zhu, S. Zhan, and H. Zhang, “Stacked U-shape networks with channel-wise attention for image super-resolution,” Neurocomputing, vol. 345, pp. 58-66, Jun. 2019.

[20] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional networks for biomedical image segmentation,” in Proc. Med Image Comput Comput Assist Interv. (MICCAI), Nov. 2015, pp. 234-241.

[21] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, “Densely connected convolutional networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 2261-2269.

[22] Z. Wang, J. Chen and S. C. H. Hoi, “Deep learning for image super-resolution: A survey,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 10, pp. 3365-3387, Oct. 2021.

[23] C. Dong, C. C. Loy, and X. Tang, “Accelerating the super-resolution convolutional neural network,” in Proc. Eur. Conf. Comput. Vis. (ECCV), Sep. 2016, pp. 391-407.

[24] W. Shi et al., “Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 1874-1883.

[25] M. Haris, G. Shakhnarovich and N. Ukitu, “Deep back-projection networks for super-resolution,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2018, pp. 1664-1673.

[26] K. He, X. Zhang, S. Ren and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770-778.

[27] C. Ledig, L. Theis, F. Huszáry, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, W. Shi, “Photo-realistic single image super-resolution using a generative adversarial network,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 105-114.

[28] X. Mao, C. Shen, Y.-B. Yang, “Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections,” in Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS), Dec. 2016, pp. 2810-2818.

[29] J. Hu, L. Shen and G. Sun, “Squeeze-and-excitation networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2018, pp. 7132-7141.

[30] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, “CBAM: Convolutional block attention module,” in Proc. Eur. Conf. Comput. Vis. (ECCV), Sep. 2018, pp. 3-19.

[31] D. Liu, B. Wen, Y. Fan, C. C. Loy, and T. S. Huang, “Non-local recurrent network for image restoration,” in Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS), Dec. 2018, pp. 1680-1689.

[32] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang, “Second-order attention network for single image super-resolution,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 11057-11066.

[33] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, “Going deeper with convolutions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 1-9.

[34] J. Li, F. Fang, K. Mei, and G. Zhang, “Multi-scale residual network for image super-resolution,” in Proc. Eur. Conf. Comput. Vis. (ECCV), Sep. 2018, pp. 527-542.

[35] T. Shang, Q. Dai, S. Zhu, T. Yang and Y. Guo, “Perceptual extreme super resolution network with receptive field block,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshop, Jun. 2020, pp. 1778-1787.

[36] E. Agustsson, R. Timofte, “NTIRE 2017 challenge on single image super-resolution: Dataset and study,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshop (CVPRW), Jul. 2017, pp. 1122-1131.

[37] M. Bevilacqua, A. Ronny, C. Guillelmet, and M. L. Alberi-Morel, “Low-complexity single-image super-resolution based on nonnegative neighbor embedding,” in Proc. Brit. Mach. Vis. Conf. (BMVC), Sep. 2012, pp. 1-10.

[38] R. Zeyde, M. Elad, M. Protter, “On single image scale-up using sparse representations,” in Proc. Int. Conf. Curves Surf., Jun. 2010, pp. 711-730.

[39] P. Arbelaez, M. Maire, C. Fowlkes, J. Malik, “Contour detection and hierarchical image segmentation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 5, pp. 898-916, May 2011.

[40] J.-B. Huang, A. Singh, N. Ahuja, “Single image super-resolution from transformed self-exemplars,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 5197-5206.

[41] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” IEEE Trans. Image Process., vol. 13, no. 4, pp. 600-612, Apr. 2004.

[42] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. Int. Conf. Learn. Represent. (ICLR), May 2015, pp. 1-13.

[43] T. Tieleman and G. Hinton, “Lecture 6.5-RMSprop: Divide the gradient by a running average of its recent magnitude,” COURSERA: Neural Networks for Machine Learning 4, 2012.

[44] I. Sutskever et al. “On the importance of initialization and momentum in deep learning” in Proc. Int. Conf. Mach. Learn. (ICML), Jun. 2013, pp. 1139-1147.

[45] C. Dong, C. C. Loy, K. He, and X. Tang, “Image super-resolution using deep convolutional networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 38, no. 2, pp. 295-307, Jun. 2016.