Efficient Transformer for Single Image Super-Resolution

Zhisheng Lu, Hong Liu*, Juncheng Li, and Linlin Zhang

***Abstract*—Single image super-resolution task has witnessed the great strides with the development of deep learning. However, most existing studies focus on building a more complex neural network with a massive number of layers, bringing heavy computational cost and memory storage. Recently, as Transformer yields brilliant results in NLP tasks, more and more researchers start to explore the application of Transformer in computer vision tasks. But with the heavy computational cost and high GPU memory occupation of vision Transformer, the network can not be designed too deep. To address this problem, we propose a novel Efficient Super-Resolution Transformer (ESRT) for fast and accurate image super-resolution. ESRT is a hybrid Transformer where a CNN-based SR network is first designed in the front to extract deep features. Specifically, there are two backbones for formatting the ESRT: lightweight CNN backbone (LCB) and lightweight Transformer backbone (LTB). Among them, LCB is a lightweight SR network to extract deep SR features at a low computational cost by dynamically adjusting the size of the feature map. LTB is made up with an efficient Transformer (ET) with small GPU memory occupation, which benefited from the novel efficient multi-head attention (EMHA). In EMHA, a feature split module (FSM) is proposed to split the long sequence into sub-segments and then these sub-segments are applied by attention operation. This module can significantly decreases the GPU memory occupation. Extensive experiments show that our ESRT achieves competitive results. Compared with the original Transformer which occupies 16057M GPU memory, the proposed ET only occupies 4191M GPU memory with better performance.**

***Index Terms*—Image super-resolution, transformer, high-frequency information, lightweight network.**

I. Introduction

SINGLE image super-resolution (SISR) aims at recovering a high-resolution (HR) image from its degraded low-resolution (LR) counterpart. SISR is still an active area for offering the promise of overcoming resolution limitations in many applications, such as video transmission, smart camera, and so on. However, SISR is an ill-posed problem since there exist infinite HR images that can be downsampled to an identical LR image. To address this issue, numerous deep neural networks have been proposed. Although these methods have achieved outstanding performance, they cannot be easily utilized in real applications due to high computation cost and memory storage.

* Corresponding author.

Z. Lu, H. Liu and L. Zhang are with Key Laboratory of Machine Perception, Shenzhen Graduate School, Peking University, Beijing 100871, China. Email: zhisheng_lu, hongliu, catherinezll@pku.edu.cn.

J. Li is with the Department of Mathematics, The Chinese University of Hong Kong, Hong Kong. Email: cvjunchengli@gmail.com.

This work is supported by National Natural Science Foundation of China (No.62073004), Science and Technology Plan of Shenzhen (No. JCYJ20200109140410340, JCYJ20190808182209321)

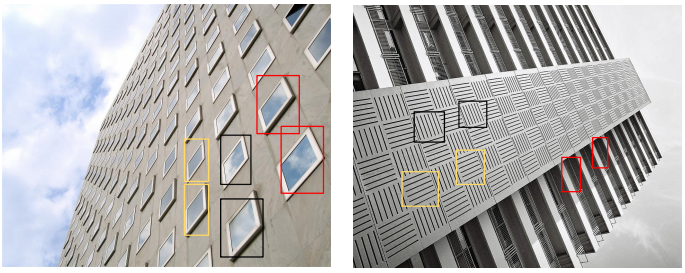

Fig. 1: Examples of similar patches in the images. These patches can help to recover details from each other.

The typical pattern to lighten the network is reducing the number of parameters. There are many ways to achieve this. The most effective and simple approach is to use the recurrent mechanism. For example, SRFBN [1] uses a feedback manner to share the weights of several middle layers and enhance the reconstruction ability of the model iteratively. RNN-based models [1], [2], [3], [4] decrease the number of parameters effectively compared to the standard CNN and obtain good performance. However, these models repeat the forward process several times, resulting in a long inference time and a large number of operations. Meanwhile, some works focus on neural architecture search (NAS) [5], [6] to design effective networks automatically. For instance, FALSR [5] proposes an elastic search tactic at both micro and macro level and acquire a great trade-off between restoration capacity and time complexity. But NAS-based methods usually have limited performances for the constraints of search space and strategy.

Nowadays, more and more lightweight SR works pay attention to the efficient architecture design, such as channel grouping [7], [8], multi-scale structure [9], [10] and information distillation [11], [12]. As we all know, the computation cost of the model is related to not only the complexity of the network but also the size of the feature map in the pipeline. Former works generally concentrate on constructing a more efficient network structure, but the reduced network capacity leads to poor performance. As Fig. 1 shows, the inner areas of the boxes with the same color are very similar to each other. Like the reference-based super-resolution task, these similar image patches can be used as reference images for each other, so that the texture details of the certain patch can be restored with reference patches. Transformer has a strong feature expression ability to model such a long-term dependency in the image, so we explore the feasibility of using Transformer in the lightweight SR task.

Vision-Transformer usually needs to occupy heavy GPU

memory, which greatly limits the development of Transformer in computer vision tasks. In this work, an Efficient SR Transformer (ESRT) architecture is proposed to enhance the ability to capture the long-distance context dependence for the SR network while significantly decrease the GPU memory cost. It is worth noting that, training a Transformer usually needs a very large dataset, but SR datasets are usually very small (DIV2K [13] only has 1000 images). Therefore, we propose a hybrid Transformer architecture for ESRT and uses a "CNN+Transformer" pattern to handle the small SR dataset.

Specifically, ESRT can be divided into two parts: lightweight CNN backbone (LCB) and lightweight Transformer backbone (LTB). For LCB, we consider more on reducing the shape of the feature map in the middle layers and maintain a deep network depth to ensure large network capacity. Firstly, inspired by the high-pass filter which can obtain the high-frequency information in the image, we design a high-frequency filtering module (HFM) which is differential and can capture the texture details of the image. With the aid of HFM, a novel high preserving block (HPB) is proposed to extract the potential features efficiently by size variation. Specifically, before size reduction, the high-frequency information is preserved by HFM and then is added back to the refined features to prevent the resolved image from visually unnaturally. By this operation, our HPB can not only efficiently extract SR features but save computational cost. For feature extraction, a novel adaptive residual feature block (ARFB) is proposed as the basic feature extraction unit with the ability to adaptively adjust the weight of the residual path and identity path. In residual path, ARFB uses **Reduction** and **Expansion** operation to reduce the number of parameters. In LTB, an efficient Transformer (ET) is proposed to be embedded into the behind of LCB. Meanwhile, a novel efficient multi-head attention (EMHA) is designed to significantly decrease the GPU memory cost In the ET. EMHA just considers the relationship between image blocks in a local region, as the pixel in SR image is commonly related to its neighbor pixels. Even though it is a local region, it is much wider than a regular convolution and has more useful context information. ET can efficiently explore the relationship between similar local blocks in the image, making the super-resolved region having more references.

Based on LCB and LTB, we build an Efficient SR-Transformer named ESRT. Compared with the recurrent mechanism, our method significantly reduces the inference time when processing the image. In comparison to the NAS-based approach, ESRT paves a new way to design a better efficient architecture manually. Meanwhile, our HPB and ARFB are general units that can be embedded into previous SR models to replace their feature extraction module, making them more lightweight while maintain high performance. Compared with other common vision-Transformer, our ESRT occupies less GPU memory and takes less time, while significantly improves the performance of SR networks at low resource consumption. Meanwhile, ESRT achieves state-of-the-art performance with few parameters and little computational cost compared with other lightweight SR models. The main contributions of this paper are summarized as follows:

- A novel hybrid Transformer that contains two backbones is proposed to effectively enhance the feature expression ability and the long-term dependence of similar patches in the captured image, so as to achieve better performance.
- A lightweight CNN backbone (LCB) is proposed to solve the small SR datasets for Transformer. LCB is used to obtain the basic feature extraction ability at a low computational cost. It is worth noting that, only use the LCB, the model can also achieve comparable performance with other lightweight methods.
- A lightweight Transformer backbone (LTB) is proposed to capture long-term dependency among similar patches in the image. Meanwhile, an efficient Transformer (ET) is designed in the LTB, which can significantly decrease the GPU memory cost and computational cost compared to the original Transformer architecture.

II. Related Work

A. Deep Learning based SR Models

SRCNN [14] is the first work that introduces deep CNN to solve the SR problem, where there are only three convolution layers. FSRCNN [15] proposes a post-upsampling mode to reduce the computational cost. VDSR [16] deepens the depth of the network by employing the skip connections for learning the residual information. DRRN [2] and MemNet [4] utilize the recurrent mechanism to refine the SR result iteratively. EDSR [17] optimizes the residual block by removing unnecessary modules and expands the model size, which wined the champion of the NTIRE2017 challenge. RCAN [18] proposes a very deep residual network with residual-in-residual architecture and channel attention mechanism. SAN [19] presents a second-order attention network to enhance the feature expression and feature correlation learning. EBRN [20] thinks that the lower-frequency and higher-frequency information in images have different levels of complexity and should be restored by models of different representational capacity. SRFBN [1] transmits the high-level information to low-level features through feedback connections in a top-down manner. CS-NL [21] proposes the concept of cross-scale feature correlation and designs the cross-scale non-local attention module. Although all these works achieve superior performance in the SR task, they typically spend highly on computation cost and memory cost, which is not suitable for practical applications.

B. Lightweight SR Models

Building a lightweight SR model has attracted extensive concern for saving computing resources. They can be mainly divided into two categories: the architecture manual-designed methods and the neural architecture search-based methods. For the first category, early models mainly focus on the recursive mechanism. DRCN [3] first applies the recursive operation into the SR model and DRRN [2] deepens the recursive neural network with weight-shared residual block. Moreover, CARN [7] adds the channel splitting technology on the basis of recursive operation. Later approaches pay more attention to designing an efficient feature extraction module. For example, MSRN [9] proposes a multi-scale block to adaptively detect

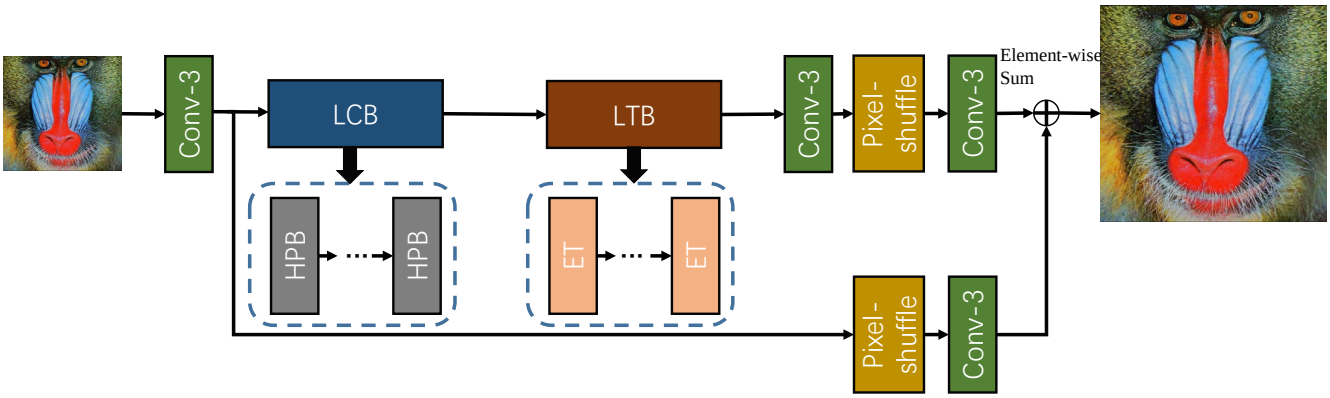

Fig. 2: The overall architecture of the proposed Efficient SR Transformer (ESRT).

the image features in different scales. IDN [12] compresses the model size by using the group convolution and combine short-term and long-term features. IMDN [11] improves the architecture of IDN and introduces the information multi-distillation blocks to extract the hierarchical features effectively. PISR [22] proposes a novel distillation framework, which consists of a teacher network and a student network, that allows boosting the performance of FSRCNN [15] which has extremely small parameters. LatticeNet [23] designs the lattice block that simulates the realization of Fast Fourier Transformation with the butterfly structure.

As for neural architecture search-based methods, MoreMNAS [6] presents a new multi-objective oriented algorithm by leveraging good virtues from both evolution algorithms and reinforced learning methods. Besides, FALSR [5] propose an elastic search tactic at both micro and macro level, based on a hybrid controller. Although the researches based on lightweight SR models has made good progress, there still remain many challenges to overcome.

C. Transformers in Vision

The breakthroughs from Transformer networks in the NLP area lead to great interest in the computer vision community. The key idea of Transformer is "selfattention" which can capture long-term information between sequence elements. By adapting Transformer in vision tasks, it has been successfully applied in image recognition [24], [25], object detection [26], [27], low-level image processing [28], [29], action recognition [30], [31] and so on. Among them, ViT [24] is the first work to replace the standard convolution with Transformer. To produce the sequence elements, ViT flattened the 2D image patches in a vector and fed them into the Transformer. DETR [26] is a Transformer for object detection task. DETR can model the prediction of a set of objects and model their relationships. DETR discards some complex hand-crafted operations (like NMS) and modules (like RPN [32]). Through this manner, there is no need to design some strong prior knowledge for this task. Although Transformers in vision has achieved great progress, they still need heavy GPU resources to train the whole model, which is not friendly to most researchers. Hence, building efficient vision-Transformer has become a research hotspot recently.

III. Proposed Method

In this section, we first describe the overall architecture of the proposed Efficient SR Transformer (ESRT). Then we present the LCB with novel high preserving block (HPB) and high-frequency filtering module (HFM). HPB first isolates the high-frequency information with the help of HFM. The size of the feature map is reduced to decrease the redundant features and save memory cost. In HPB, an adaptive residual feature block (ARFB) is also introduced as the basic feature extraction unit. Next, we present the LTB with an efficient Transformer (ET). Finally, we present the difference between our ESRT and other SR methods.

A. Overall Architecture

As shown in Fig. 2, our ESRT mainly consists of four parts: shallow feature extraction, lightweight CNN backbone (LCB), lightweight Transformer backbone (LTB), and image reconstruction. We define I_{SR} and I_{LR} as the input and output of ESRT, respectively. Therefore, we first extract the shallow feature F_0 from I_{LR} with one convolution layer:

$$F_0 = f_s(I_{LR}), \tag{1}$$

where f_s denotes the shallow feature extraction function. F_0 is then used for LCB with several HPBs, which can be formulated as:

$$F_n = \zeta^n(\zeta^{n-1}(...(\zeta^1(F_0)))), \tag{2}$$

where ζ^n denotes the mapping of n-th HPB and F_n represents the output of n-th HPB. All outputs of HPB are concatenated to be sent to LTB with several ETs to fuse all intermediate features in LCB:

$$F_d = \phi^n(\phi^{n-1}(...(\phi^1([F_1, F_2, ..., F_n])))), \tag{3}$$

where F_d is the output of LTB and ϕ stands for the function of ET. To get the I_{SR}, F_d, and F_0 are simultaneously fed into the reconstruction module:

$$I_{SR} = f(f_p(f(F_d))) + f_p(f(F_0)), \tag{4}$$

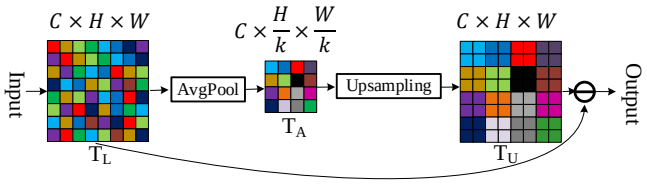

Fig. 3: The schematic diagram of the propose HFM.

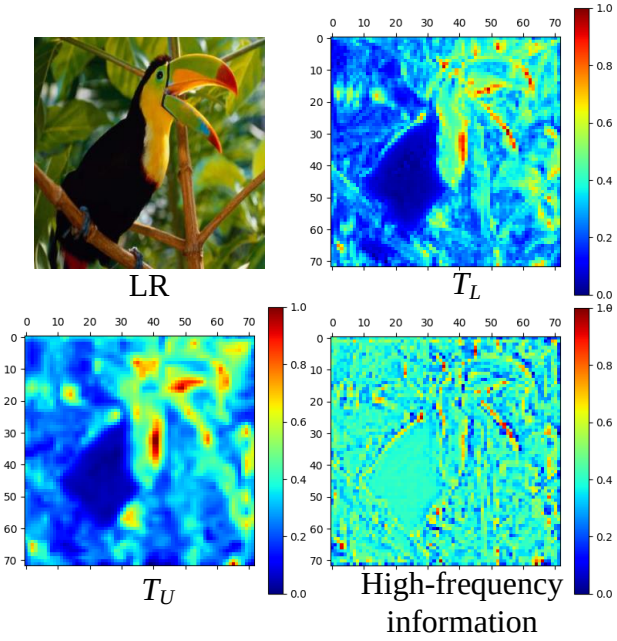

Fig. 4: Visual activation maps of T_L, T_U, and obtained high-frequency information. Best viewed in color.

where f and f_p stand for the convolution layer and PixelShuffle layer, respectively.

B. Lightweight CNN Backbone (LCB)

Lightweight CNN Backbone (LCB) is built like other SR models, which served as the front part of ESRT. The function of LCB is to extract the latent SR features in advance so that the model has the initial ability of super-resolution. According to the Fig. 2, we can observe that LCB is mainly composed of a series of high preserving blocks (HPBs). Here we introduce the LCB in detail.

1) High-frequency Filtering Module: Before introducing the HPB, we first present the high-frequency filtering module (HFM) which is embedded in HPB. Since the Fourier transform is difficult to embed in CNN, a differentiable HFM is proposed. The target of HFM is to estimate the high frequency information of the image from the LR space. Assuming the size of the input feature map T_L is $C \times H \times W$, an average pooling layer is first applied to T_L:

$$T_A = avgpool(T_L, k), \tag{5}$$

where k denotes the kernel size (same to the stride) of pooling layer. As shown in Fig. 3, the size of the intermediate feature map T_A is $C \times \frac{H}{k} \times \frac{W}{k}$. Each value in T_A can be viewed as the average intensity of each specified small area of T_L. After that,

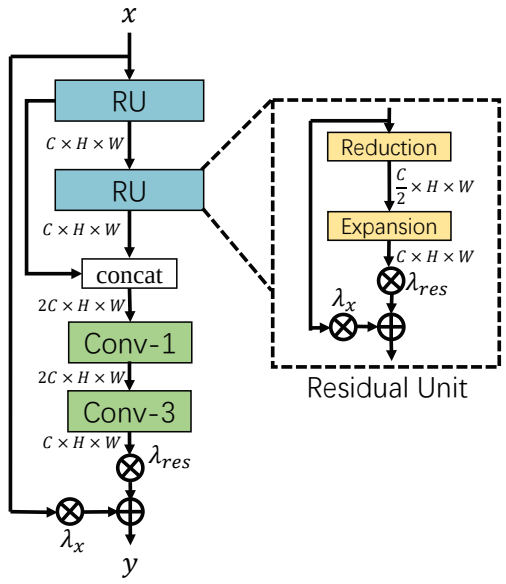

Fig. 5: The architecture of the proposed ARFB.

T_A is upsampled to get a new tensor T_U of size $C \times H \times W$. T_U is regarded as an expression of the average smoothness information compared with the original T_L. Finally, T_U is element-wise subtracted from T_L to obtain the high-frequency information.

The visual activation maps of T_L, T_U, and high-frequency information are also shown in Fig. 4. It can be observed that the T_U is more smooth than the T_L as it is the average information of the T_L. And the high-frequency information retains the details and edges of the feature map before downsampling. Hence it is essential to save these information.

2) Adaptive Residual Feature Block: Plenty of works have proven that the depth of the model is highly correlated to the performance. As explored in ResNet and VDSR, when the depth grows, the residual architecture can mitigate the gradient vanishing problem and augment the representation capacity of the model. In this paper, a novel Adaptive Residual Feature Block (ARFB) is proposed as the basic feature extraction block which is efficient and fast.

As Fig. 5 shows, ARFB contains two residual units (RUs) and two convolution layers. To save memory and number of parameters, RU is made up of two modules: Reduction and Expansion. For Reduction, the channels of the feature map are reduced by half and recovered in Expansion. Like EDSR [17], BN [33] is not used in RU. Meanwhile, a residual scaling with adaptive weights (RSA) is designed to dynamically adjust the importance of residual path and identity path. Compared with fixed residual scaling, RSA can improve the flow of gradients and automatically adjust the content of the residual feature maps for the input feature map. Assume that x_{ru} is the input of RU, the process of RU can be formulated as:

$$y_{ru} = \lambda_{res} \cdot f_{ex}(f_{re}(x_{ru})) + \lambda_x \cdot x, \tag{6}$$

where y_{ru} is the output of RU, f_{re} and f_{ex} represent the Reduction and Expansion, respectively. λ_{res} and λ_x are two adaptive weights for two paths.

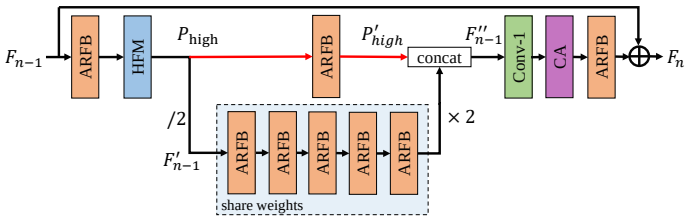

Fig. 6: The architecture of the proposed HPB.

For ARFB, the outputs of two RUs are concatenated followed by a 1×1 convolution layer, to fully utilize the hierarchical features. In the end, a 3×3 convolution layer is adopted to reduce the channels of the feature map and extract valid information from the fused features.

3) High Preserving Block: The main target of SISR is the amplification of resolution while retaining the texture details of the image. Hence, previous SR networks commonly keep the spatial resolution of the feature map unchanged in the pipeline. In this work, to lessen the computational cost of the network, a novel high preserving block (HPB) is proposed to reduce the shape of processing features. However, the reduction of the size of the feature map always leads to the loss of image details, which cause the visually unnatural SR images. To solve this problem, in HPB, we creatively preserve the high-frequency information with the aid of HFM while reducing the size of feature maps.

The architecture of HPB is shown in Fig. 6. Firstly, an ARFB is adopted to extract the input features F_{n-1} for HFM. HFM then calculates the high-frequency information (marked as P_{high}) of the features. After the P_{high} is obtained, we reduce the size of the feature map to reduce computational cost and feature redundancy. The downsampled feature map is denoted as $F^{'}_{n-1}$. For $F^{'}_{n-1}$, several ARFBs are utilized to explore the potential information for completing the SR image. It is worth noting that these ARFBs share weights to reduce parameters. Meanwhile, a single ARFB is used to process the P_{high} to align the feature space with $F^{'}_{n-1}$. After feature extraction, $F^{'}_{n-1}$ is upsampled to the original size by bilinear interpolation. Next, we fuse the $F^{'}_{n-1}$ with $P^{'}_{high}$ for preserving the initial details and obtain the feature $F^{''}_{n-1}$. This operation can be expressed as:

$$F^{''}_{n-1} = [f_a(P_{high}), \uparrow f_a^{\circlearrowleft 5}(\downarrow F^{'}_{n-1})], \tag{7}$$

where f_a denotes the operation of ARFB, $f_a^{\circlearrowleft 5}$ means that ARFB is called five times, $\uparrow$ denotes the upsampling, and $\downarrow$ denotes the downsampling.

For $F^{''}_{n-1}$, as it is concatenated by two features, a 1×1 convolution layer is used to reduce the channel number. Then, a channel attention module [34] is employed to highlight channels with high activated values. After that there is a ARFB to extract the final features. Finally, the global residual connection is proposed to add the original features F_{n-1} to F_n. The goal of this operation is to learn the residual information from the input and stabilize the training.

C. Lightweight Transformer Backbone (LTB)

Recent years, Transformer has made great progress in computer vision task as its strong selfattention mechanism. In SISR, similar image blocks within the image can be used as reference images to each other, so that the texture details of the currenct image block can be restored with reference to other image blocks, which is proper to use Transformer. However, previous variants of vision Transformer commonly need heavy GPU memory cost, which hinders the development of Transformer in the vision area. In this paper, we propose a lightweight Transformer backbone (LTB), which is composed of efficient Transformer (ET), to capture the long-term dependence of similar local regions in the image at a low computational cost.

1) Pre- and Post-processing for ET: Typically, the standard Transformer takes a 1-D sequence as input, learning the long-distance dependency of the sequence. However, for the vision task, the input is always a 2-D image. The common way to turn a 2-D image into a 1-D sequence is to sort the pixels in the image one by one. But this method will lose the unique local correlation of the image, leading to suboptimal performance. In ViT [24], the 1-D sequence is generated by non-overlapping block partitioning, which means there is no pixel overlap between each block. In this paper, this pre-processing way is eliminated as the experimental result is not good. Hence, a novel processing way is proposed to handle the feature map.

To handle the 2-D feature maps for LR images, we use unfolding technique to split the feature maps into patches. Each patch is considered as a "word". Specifically, the feature map $F_{ori} \in \mathbb{R}^{C \times H \times W}$ are unfolded (by $k \times k$ kernel) into a sequence of patches, i.e., $F_{p_i} \in \mathbb{R}^{k^2 \times C}$, $i = \{1, ..., N\}$, where $N = H \times W$ is the amount of patches. Here, the learnable position embeddings are eliminated for each patch because the "Unfold" operation automatically reflects the position information for each patch. It is redundant to add the location embedding. Those patches F_p are directly sent to the ET. The output of ET has the same shape as the input and we use the "Fold" operation to reconstruct the feature map.

2) The Architecture of ET: The main architecture of ET is shown in Fig. 8. For simplicity and efficiency, like ViT [24], ET only uses the encoder structure of the standard Transformer. In the ET encoder, there consists of an efficient multi-head attention (EMHA) and an MLP. Meanwhile, layer-normalization (Norm) [35] is employed before every block, and residual connection is also applied after each block. Assume that the input embeddings are E_i, the output embeddings E_o can be obtained by:

$$\begin{aligned} E_{m1} &= EMHA(Norm(E_i)) + E_i, \\ E_o &= MLP(Norm(E_{m1})) + E_{m1}. \end{aligned} \tag{8}$$

Efficient Multi-Head Attention (EMHA). In EMHA, there are several modifications to make the EMHA more efficient and occupy lower GPU memory cost compared with the original MHA [36]. Assume the shape of the input embedding E_i is $B \times C \times N$. Firstly, a Reduction layer is used to reduce the number of channels by half ($B \times C_1 \times N, C_1 = \frac{C}{2}$). After that, a linear layer is adopted to project the feature map into

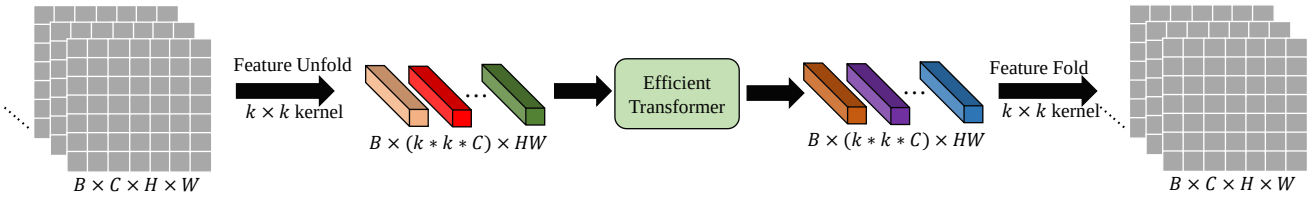

Fig. 7: The pre- and post-processing for the Efficient Transformer (ET).

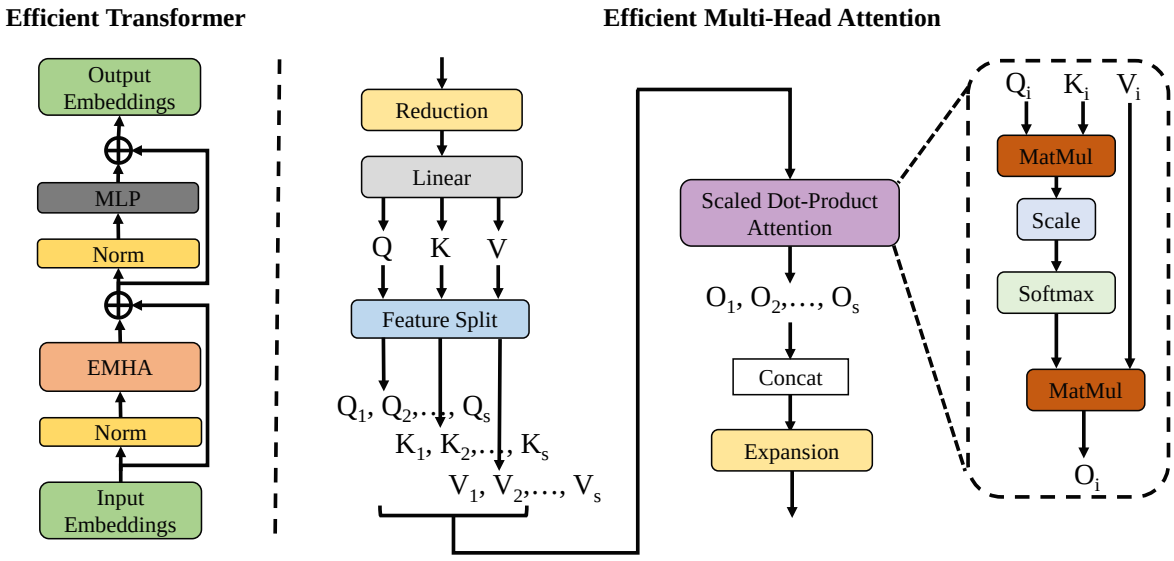

Fig. 8: The architecture of the propose Efficient Transformer (ET).

three elements: Q (query), K (keys), V (values). As employed in Transformer, we linearly project the Q, K, V m times to perform the multi-head attention. m is the number of heads. Next, the shape of three elements is reshaped and permuted to $B \times m \times N \times \frac{C_1}{m}$.

In original MHA, Q, K, V are directly used to calculate the self-attention with large scale matrix multiplication, which is a huge memory expense. Assume Q and K calculate the self-attention matrix with shape $B \times m \times N \times N$. Then this matrix computes the self-attention with V, the dimension in 3-th and 4-th are $N \times N$. For SISR, the images usually have high resolution, causing that N is very large and the calculation of self-attention matrix consumes a lot of GPU memory cost and computational cost.

In the SR task, the predicted pixels in super-resolved images commonly only depend on the local adjacent areas in LR. Hence, in ET, a Feature Split Module (FSM) is used to split Q, K, and V into s equal segments with splitting factor s. Therefore, the dimension in 3-th and 4-th of the last self-matrix is $\frac{N}{s} \times \frac{N}{s}$, which can significantly reduce the computational and GPU memory cost. We denote this segments as $Q_1, ..., Q_s$, $K_1, ..., K_s$, and $V_1, ..., V_s$. Each triplet of these segments is applied with Scaled Dot-Product Attention (SDPA), respectively. The structure of SDPA is also shown in Fig. 8, which just omits the $Mask$ operation. Afterwards, all the outputs $(O_1, O_2, ..., O_s)$ of SDPA are concatenated together to generate the whole output feature O. Finally, an Expansion layer is used to recover the number of channels.

D. Implements Details

In our proposed ESRT, we set 3×3 as the size of all convolution layer except that in the **Reduction** module, whose kernel size is 1×1. Each convolution layer has 32 channels except for the fusion layer which is twice. For image reconstruction part, following most previous methods, we use PixelShuffle [37] to upscale the last coarse features to fine features. The k in HFP is 2 which means that the feature map is down-scaled by half. In ESRT, the number of HPB is set to 3 and we set the initial value of learnable weight in ARFB to 1. Meanwhile, the number of ET in LTB is set to 1 to save the GPU memory. The splitting factor s in ET is set to 4, the k in pre- and post-process of ET is set to 3, and the head number m in EMHA is set to 8, respectively.

IV. Experiments

A. Datasets and Metrics

Our model is trained with the DIV2K [13] dataset, which is widely used in SISR task. DIV2K contains 800 RGB training images and 100 validation images with rich textures (2K resolution). For evaluation, we use five benchmark datasets to validate the effectiveness of our method, including Set5 [38], Set14 [39], B100 [40], Urban100 [41], and Manga109 [42]. Meanwhile, Peak signal-to-noise ratio (PSNR) and structure similarity index (SSIM) are used to evaluate the performance of the reconstructed SR images. Following previous works, we calculate the results on Y channel of YCbCr color space.

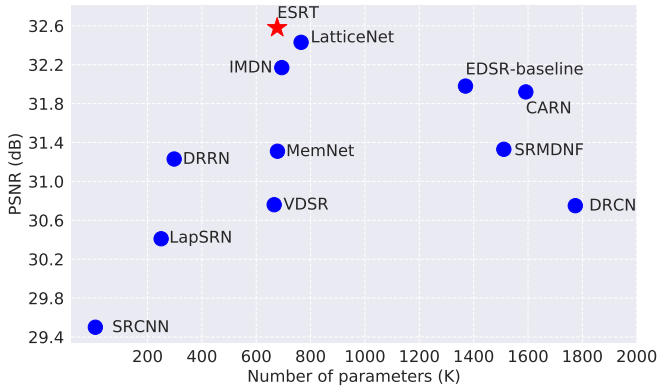

Fig. 9: Trade-off between the number of parameters and performance on Urban100 ($\times 2$).

Training Setting. Following previous works, we randomly crop 16 LR image patches with the size of 48×48 as inputs of the model for training in each epoch. Random horizontal flipping and 90 degree rotation are used for data augment. The initial learning rate is 2×10^{-4} and decreased half for every 200 epochs. The model is trained by Adam optimizer with momentum equal to 0.9. L1 loss is used as it can produce more sharp images compared with L2 loss. We implement our model on Pytorch platform. Training an ESRT roughly takes two days with one GTX1080Ti GPU for the whole training.

B. Comparisons With State-of-The-Art Methods

1) Objective Evaluation: In TABLE I, we compare our ESRT with 13 state-of-the-art lightweight SR models, including SRCNN [14], FSRCNN [15], VDSR [16], DRCN [3], LapSRN [43], DRRN [2], MemNet [4], EDSR-baseline [17], SRMDNF [44], CARN [7], FALSR [5], IMDN [11], and LatticeNet [23]. Most of them achieve the best results at the time with a well-designed structure in the lightweight SR task.

TABLE I shows the performance comparison for $\times 2$, $\times 3$, and $\times 4$ SR. Obviously, our ESRT achieves the best results under all scaling factors. These CNN-based models use a well-designed network to learn the mapping function between LR and HR in a lightweight manner. They use channel splitting or reducing the number of layers to lighten the model, but they all ignore the model depth which is also of great importance for SR. Our ESRT reduces the computation cost of each module while ensuring that the network is very deep. According to TABLE II we can see that our ESRT can achieve 163 layers while other methods' layers are extremely shallow compared with our method. This is benefited from our HPB and ARFB which can efficiently extract SR features while preserving the high-frequency information. It is worth noting that ESRT performs much better on the Urban100 dataset. The reason is that the images in this dataset usually have many similar patches in each image. Hence, profit from our LTB, ESRT can easily capture the long-term dependencies among these similar image patches and explore their relevance.

Also, the parameters comparison of these models is also provided in TABLE I. We can clearly observe that although EDSR-baseline (the champion of NTIRE2017 Super-Resolution Challenge) has a close performance to our ESRT, it has almost twice the parameters compared to ESRT. Moreover, LatticeNet has close parameters to ESRT, but ESRT performs better on Set14, B100, Urban100, and Manga109. We also visualize the trade-off analysis between the number of parameters and performance among these lightweight SR models in Fig. 9. All experiments fully demonstrated that our ESRT achieves a great trade-off between model size and performance.

2) Comparison on Computational Cost: In TABLE II, the GFlops of other SR methods and our ESRT is calculated as the input size is 1280×720 on $\times 4$ scale. It can be seen that IMDN has the least amount of computation as it uses the channel splitting in many convolution layers. The GFlops of our ESRT is 67.7G which is the second-least among these methods. With so little computation, our method can even achieve 168 layers, which extremely surpasses the other methods. Such many layers are the reason why ESRT performs well. This table also shows the running time of these methods. Our ESRT needs 0.01085 seconds for inference, which is longer than IMDN but still meets the actual needs. From the last two rows, we can see that the addition of ET only adds little running time (only 0.00189s). In summary, our ESRT is an efficient and accurate lightweight SR model.

3) Subjective Evaluation: In Fig. 10, we provide the visual comparison between ESRT and other lightweight SR models on $\times 2$, $\times 3$, and $\times 4$. Obviously, SR images reconstructed by our ESRT have more refined details, especially in the edges and lines. It is worth noting that in $\times 4$ scale, the gap between our method and other SR models is more apparent. This further validates the effectiveness of the propose ESRT.

C. Network Investigations

1) Study of High Preserving Block (HPB): HPB is an important component of ESRT, which not only can reduce the model size but maintain the high SR performance of the model. As mentioned in Sec. III-B3, HPB is mainly composed of HFM, ARFB, and CA. In TABLE III, we provide a series of ablation studies to explore their effectiveness.

A. **High Pathway**: In cases 1 and 2, we investigate the effectiveness of high pathway in HPB. Case 1 presents that the HPB does not use in the HFM to extract the high-frequency information and the red line in Fig. 6 is eliminated. This means that the high-frequency information cannot be preserved and added to the recovered features, which may cause the super-resolved image visually unnatural. According to the results, we can observe that the join of the high pathway adds almost 100K parameters and improve the 0.17dB PSNR. The above experiment illustrates the effectiveness and necessity of the high pathway.

B. **Channel Attention (CA)**: In case 3, we drop the CA in HPB to investigate its effectiveness. Compared with case 2, case 3 achieves the worse results with little parameter reduction. This is because the CA can obtain the correlation between channels and augment the representation ability of some important channels, making the network focusing on more useful information.

TABLE I: Average PSNR/SSIM comparison on Set5, Set14, BSD100, Urban100, and Manga109. Best and second best results are **highlighted** and underlined, respectively.

Method	Scale	Params	Set5	Set14	BSD100	Urban100	Manga109
			PSNR / SSIM	PSNR / SSIM	PSNR / SSIM	PSNR / SSIM	PSNR / SSIM
Bicubic		-	33.66 / 0.9299	30.24 / 0.8688	29.56 / 0.8431	26.88 / 0.8403	30.80 / 0.9339
SRCNN [14]		8K	36.66 / 0.9542	32.45 / 0.9067	31.36 / 0.8879	29.50 / 0.8946	35.60 / 0.9663
FSRCNN [15]		13K	37.00 / 0.9558	32.63 / 0.9088	31.53 / 0.8920	29.88 / 0.9020	36.67 / 0.9710
VDSR [16]		666K	37.53 / 0.9587	33.03 / 0.9124	31.90 / 0.8960	30.76 / 0.9140	37.22 / 0.9750
DRCN [3]		1,774K	37.63 / 0.9588	33.04 / 0.9118	31.85 / 0.8942	30.75 / 0.9133	37.55 / 0.9732
LapSRN [43]		251K	37.52 / 0.9591	32.99 / 0.9124	31.80 / 0.8952	30.41 / 0.9103	37.27 / 0.9740
DRRN [2]	×2	298K	37.74 / 0.9591	33.23 / 0.9136	32.05 / 0.8973	31.23 / 0.9188	37.88 / 0.9749
MemNet [4]		678K	37.78 / 0.9597	33.28 / 0.9142	32.08 / 0.8978	31.31 / 0.9195	37.72 / 0.9740
EDSR-baseline [17]		1,370K	37.99 /0.9604	33.57 /0.9175	32.16 / 0.8994	31.98 / 0.9272	38.54 / 0.9769
SRMDNF [44]		1,511K	37.79 / 0.9601	33.32 / 0.9159	32.05 / 0.8985	31.33 / 0.9204	38.07 / 0.9761
CARN [7]		1,592K	37.76 / 0.9590	33.52 / 0.9166	32.09 / 0.8978	31.92 / 0.9256	38.36 / 0.9765
IMDN [11]		694K	38.00 / 0.9605	33.63 / 0.9177	32.19 / 0.8996	32.17 / 0.9283	38.88 / 0.9774
LatticeNet [23]		756K	**38.15 / 0.9610**	**33.78 / 0.9193**	32.25 / **0.9005**	32.43 / 0.9302	— / —
ESRT(ours)		677K	38.03 / 0.9600	33.75 / 0.9184	**32.25** / 0.9001	**32.58 / 0.9318**	**39.12 / 0.9774**
Bicubic		-	30.39 / 0.8682	27.55 / 0.7742	27.21 / 0.7385	24.46 / 0.7349	26.95 / 0.8556
SRCNN [14]		8K	32.75 / 0.9090	29.30 / 0.8215	28.41 / 0.7863	26.24 / 0.7989	30.48 / 0.9117
FSRCNN [15]		13K	33.18 / 0.9140	29.37 / 0.8240	28.53 / 0.7910	26.43 / 0.8080	31.10 / 0.9210
VDSR [16]		666K	33.66 / 0.9213	29.77 / 0.8314	28.82 / 0.7976	27.14 / 0.8279	32.01 / 0.9340
DRCN [3]		1,774K	33.82 / 0.9226	29.76 / 0.8311	28.80 / 0.7963	27.15 / 0.8276	32.24 / 0.9343
LapSRN [43]		502K	33.81 / 0.9220	29.79 / 0.8325	28.82 / 0.7980	27.07 / 0.8275	32.21 / 0.9350
DRRN [2]	×3	298K	34.03 / 0.9244	29.96 / 0.8349	28.95 / 0.8004	27.53 / 0.8378	32.71 / 0.9379
MemNet [4]		678K	34.09 / 0.9248	30.00 / 0.8350	28.96 / 0.8001	27.56 / 0.8376	32.51 / 0.9369
EDSR-baseline [17]		1,555K	34.37 / 0.9270	30.28 / 0.8417	29.09 / 0.8052	28.15 / 0.8527	33.45 / 0.9439
SRMDNF [44]		1,528K	34.12 / 0.9254	30.04 / 0.8382	28.97 / 0.8025	27.57 / 0.8398	33.00 / 0.9403
CARN [7]		1,592K	34.29 / 0.9255	30.29 / 0.8407	29.06 / 0.8034	28.06 / 0.8493	33.50 / 0.9440
IMDN [11]		703K	34.36 / 0.9270	30.32 / 0.8417	29.09 / 0.8046	28.17 / 0.8519	33.61 / 0.9445
LatticeNet [23]		765K	**34.53 / 0.9281**	30.39 / 0.8424	29.15 / 0.8059	28.33 / 0.8538	—/ —
ESRT(ours)		770K	34.42 / 0.9268	**30.43 / 0.8433**	**29.15 / 0.8063**	**28.46 / 0.8574**	**33.95 / 0.9455**
Bicubic		-	28.42 / 0.8104	26.00 / 0.7027	25.96 / 0.6675	23.14 / 0.6577	24.89 / 0.7866
SRCNN [14]		8K	30.48 / 0.8628	27.50 / 0.7513	26.90 / 0.7101	24.52 / 0.7221	27.58 / 0.8555
FSRCNN [15]		13K	30.72 / 0.8660	27.61 / 0.7550	26.98 / 0.7150	24.62 / 0.7280	27.90 / 0.8610
VDSR [16]		666K	31.35 / 0.8838	28.01 / 0.7674	27.29 / 0.7251	25.18 / 0.7524	28.83 / 0.8870
DRCN [3]		1,774K	31.53 / 0.8854	28.02 / 0.7670	27.23 / 0.7233	25.14 / 0.7510	28.93 / 0.8854
LapSRN [43]		502K	31.54 / 0.8852	28.09 / 0.7700	27.32 / 0.7275	25.21 / 0.7562	29.09 / 0.8900
DRRN [2]	×4	298K	31.68 / 0.8888	28.21 / 0.7720	27.38 / 0.7284	25.44 / 0.7638	29.45 / 0.8946
MemNet [4]		678K	31.74 / 0.8893	28.26 / 0.7723	27.40 / 0.7281	25.50 / 0.7630	29.42 / 0.8942
EDSR-baseline [17]		1,518K	32.09 / 0.8938	28.58 / 0.7813	27.57 / 0.7357	26.04 / 0.7849	30.35 / 0.9067
SRMDNF [44]		1,552K	31.96 / 0.8925	28.35 / 0.7787	27.49 / 0.7337	25.68 / 0.7731	30.09 / 0.9024
CARN [7]		1,592K	32.13 / 0.8937	28.60 / 0.7806	27.58 / 0.7349	26.07 / 0.7837	30.47 / 0.9084
IMDN [11]		715K	32.21 / 0.8948	28.58 / 0.7811	27.56 / 0.7353	26.04 / 0.7838	30.45 / 0.9075
LatticeNet [23]		777K	**32.30 / 0.8962**	28.68 / 0.7830	27.62 / 0.7367	26.25 / 0.7873	—/ —
ESRT(ours)		751K	32.19 / 0.8947	**28.69 / 0.7833**	**27.69 / 0.7379**	**26.39 / 0.7962**	**30.75 / 0.9100**

C. **Adaptive Residual Feature Block (ARFB)**: In TABLE III, RF stands for the residual block proposed in EDSR [17], which is widely used in SISR. In Case 2 and Case 4, we compare the performance and the number of parameters between ARFB and RB (in EDSR [17]). We can see that if ESRT replaces ARFB with RB as the basic feature extraction unit, the PSNR just rises 0.01dB but the parameters go up to 972K. This means that our ARFB significantly reduces the number of parameters without losing too much performance.

All the above experiments fully demonstrated the effectiveness of HPB. Meanwhile, these experiments indicate the necessity and effectiveness of the introduction of these modules and mechanisms within HPB.

2) Study of Efficient Transformer (ET): To capture the long-term dependencies of similar local regions in the image, we add the ET behind the LCB. To illustrate the efficiency and effectiveness of ET, we provide the following experiments:

A. **TR v.s. w/o TR**: Firstly, we analyze the model with and without Transformer in TABLE IV. We can see that if ESRT drops the ET, the model performance descends obviously from 32.18dB to 31.96dB. From this, it can be inferred that the correlation of long-term image patches is beneficial for image super-resolution. We think the reason is that a natural scene picture has many similar pixel blocks and these blocks always can complete other missing information as a reference. Hence, the addition of Transformer provides this completion message.

B. **TR v.s. Original TR**: Secondly, we compare the model with the original Transformer in vision (we use the ViT [24] version) to our ESRT ('1 ET'). From TABLE IV we can see that for the original TR, it increases the 417M parameters while our ET just adds 197M parameters. This benefits from the **Reduction** module to reduce the number of channels. Also, for GPU memory, the original TR needs to occupy 16057M memory which even cannot run on some common NVIDIA GPUs like 1080Ti and 2080Ti. Contrastly, our ET just occupies 4191M GPU memory, which only 1/4 of the original ET. More surprising is that the performance of the model with the original Transformer is even worse than our ESRT. This is because the model with the original Transformer needs more data to train while the datasets are usually small

TABLE II: Network structure settings comparison between our ESRT and other lightweight SR models.

Method	Input	#Layers	Residual learning	Parameters	GFlops(x4)	Running time
SRCNN [14]	LR+bicubic	3	No	0.008M	52.7G	0.00111s
VDSR [16]	LR+bicu	20	Yes	0.67M	612.6G	0.00597s
LapSRN [43]	LR	27	Yes	0.25M	149.4G	0.00330s
DRRN [2]	LR+bicubic	52	No	0.30M	6796.9G	0.08387s
CARN [7]	LR	34	Yes	1.6M	90.9G	0.00278s
IMDN [7]	LR	34	Yes	0.7M	**40.9G**	0.00258s
ESRT w/o ET	**LR**	-	-	-	-	0.00896s
ESRT	**LR**	**163**	Yes	0.68M	67.7G	0.01085s

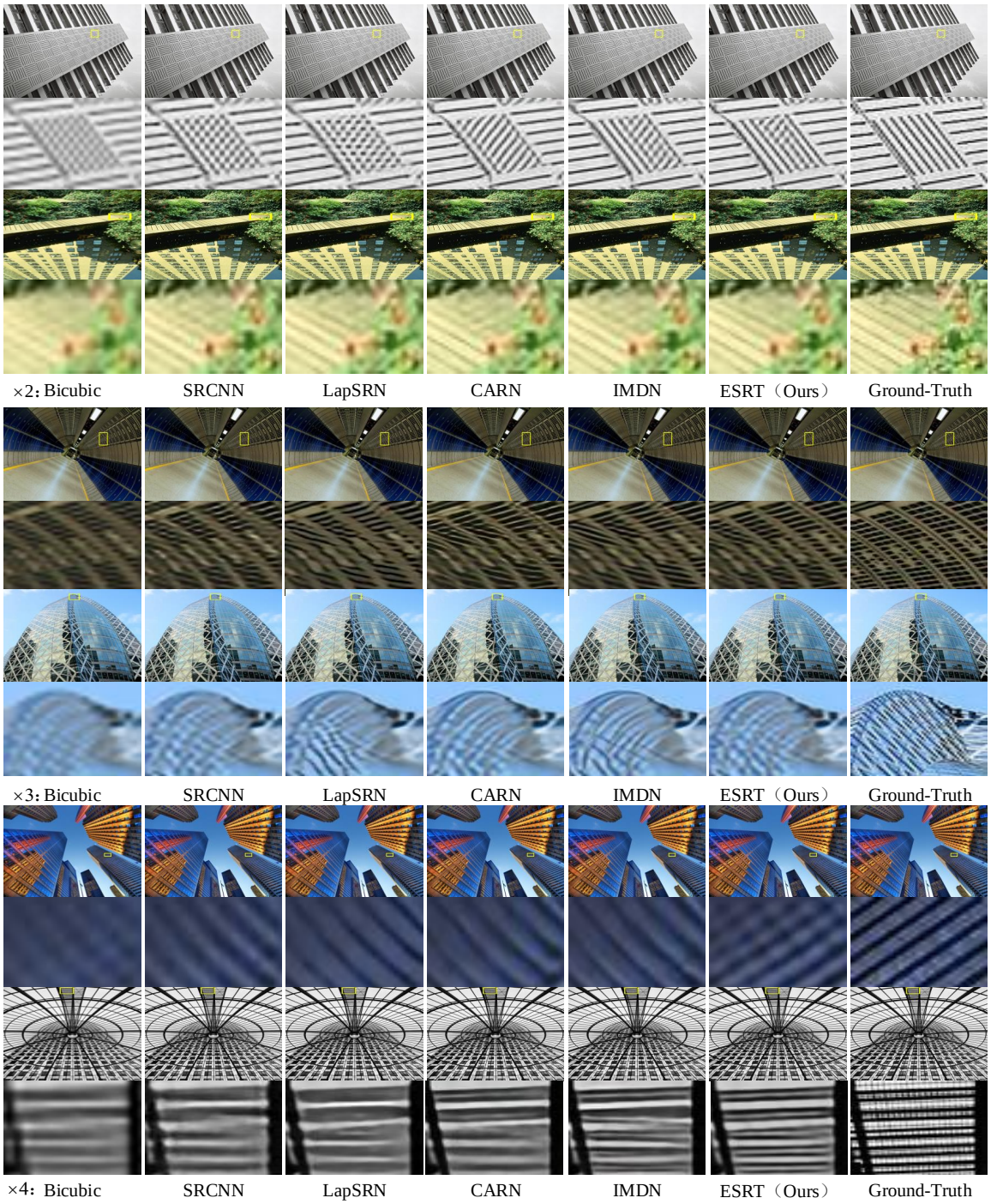

Fig. 10: Visual comparison with other SR methods. Our ESRT can reconstruct realistic SR images with sharper edges

TABLE III: Study of each component in HPB. The experiment is performed on Set5 ($\times 4$).

Case Index	1	2	3	4
High pathway		√	√	√
CA	√	√		√
ARFB	√	√	√	
RB				√
Parameters(K)×4	658	751	724	972
PSNR(dB) ×4	32.02	32.19	32.08	**32.20**

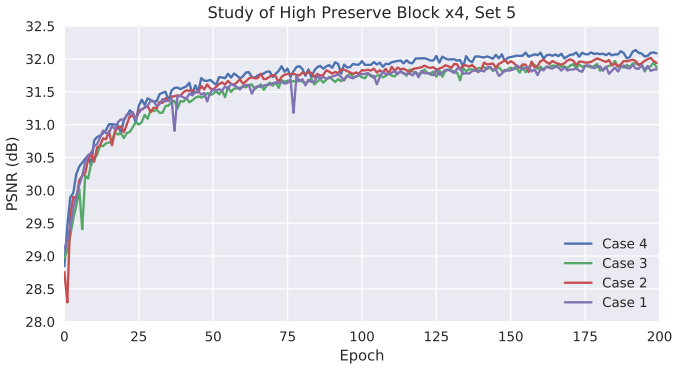

Fig. 11: Performance comparison of cases in TABLE III.

in the SISR task.

C. **The Number of ET**: In general, increasing the number of convolutional layers can increase the model performance. In view of this, we add the number of ET. From TABLE IV, we can see that if we add the number of ET to two, the PSNR is much higher than the model with only 1 ET, while the parameters add up to 949K. And if we add the number of ET to three, the PSNR is the same as the '2 ET' model. This indicates that the performance of the model cannot be improved simply by increasing the number of ET. In this work, to keep consistent with other lightweight models in the aspect of parameters, only one ET is used in ESRT.

D. **The Splitting Factor** s: In MHA, a Feature Split Module (FSM) is used to split the original Q, K, and V into s segments

TABLE IV: Study of Efficient Transformer. The experiment is performed on Set5 ($\times 4$).

Case	PSNR(dB)	Parameters(K)	GPU memory (M)
w/o TR	31.96	554	1931M
Original TR	32.14	971	16057M
1 ET	32.18	751	4191M
2 ET	32.25	949	6499M
3 ET	32.25	1147	8217M
s=2	32.15	751	6731M
s=4	**32.18**	751	4191M
s=6	32.04	751	3159M

TABLE V: Adding ET into RCAN.

Scale	Model	#Param	Set5	Set14	B100	Urban100
×2	RCAN	16M	**38.27**	34.12	**32.46**	33.54
	RCAN/2+ET	8.5M	38.25	**34.15**	32.42	**33.61**
×3	RCAN	16M	**34.74**	**30.65**	29.32	29.09
	RCAN/2+ET	8.7M	34.69	30.63	**29.35**	**29.16**
×4	RCAN	16M	**32.63**	28.87	**27.77**	26.82
	RCAN/2+ET	8.7M	32.60	**28.90**	27.76	**26.87**

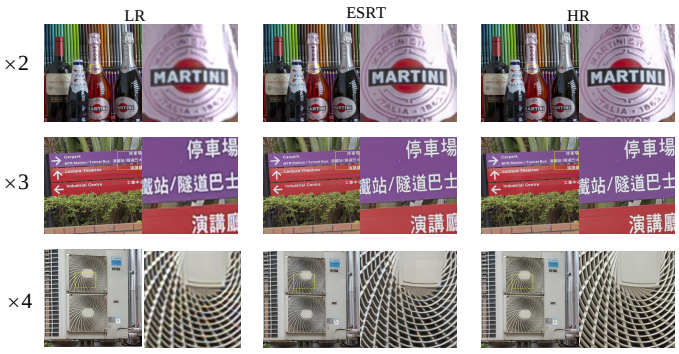

Fig. 12: Visual comparison on RealSR dataset.

to save the GPU memory. Commonly, the s is smaller, the split segments are shorter and the GPU memory occupation is less. In TABLE IV, we investigate the different value of s. As shown in TABLE IV, the model achieves the best performance when s is set to 4. Meanwhile, we can observe that GPU memory is not relevant to s linearly. The change of s does not affect the number of parameters.

3) The Universality of ET: In order to verify the effectiveness and universality of the proposed ET, we add the ET into RCAN [18]. It is worth noting that we only use the half structure of RCAN (group number=5, original=10) and add the ET before the reconstruction part. According TABLE V we can see that the performance of the model "RCAN/2+ET" is close to the original RCAN with fewer parameters. It is worth noting that the PSNR on Urban100 of "RCAN/2+ET" is always higher than the original RCAN. This is because our ET can effectively model the relationship between image blocks, so it can effectively improve the model performance.

4) Study of Pure Transformer: In general, the pure Transformer-based architecture is more efficient and scalable than previous CNN-based architecture in both model size and computational scale. And the hybrid Transformer can perform better than the pure Transformer model on a smaller model. To verify whether this property exists in the proposed ET, we test the performance of pure Transformer-based ESRT. Specifically, we modify ESRT to the pure Transformer by removing the LCB, to test the effectiveness of ET in the setting of pure Transformer. We define the modified Transformer as "Pure-ESRT". The result is shown in TABLE VI. It can be seen that if the number of ET in LTB is 1 in both Pure-ESRT and ESRT, the performance of Pure-ESRT will significantly decrease compared with ESRT. This means that LCB can effectively make up for the feature extraction ability of Transformer with little computation cost. Meanwhile, if the number of ET is small, the Pure-ESRT performs badly without the huge pretrained dataset.

Meanwhile, we also tested the PSNR of Pure-ESRT when adding the quantity of ET. The model "2ET" has a huge improvement compared to "1ET". Moreover, "4ET" has a close performance to the ESRT on five datasets. However, the parameters and GPU memory cost of "4ET" are higher than ESRT. When the number of ET is continued to increase to 6, Pure-ESRT outperforms ESRT on all five benchmarks,

TABLE VI: The performance of Pure-ESRT.

Model	Parameter	GPU occupy	Set5	Set14	BSD100	Urban100	Manga109
			PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR / SSIM
ESRT	751K	4191M	32.19/0.8947	28.69/0.7833	27.69/0.7379	26.39/0.7962	30.75/0.9100
1ET	357K	3967M	31.01/0.8751	27.85/0.7636	27.10/0.7203	25.00/0.7459	28.22/0.8726
2ET	564K	5685M	31.77/0.8878	28.39/0.7758	27.42/0.7312	25.73/0.7728	29.76/0.8978
3ET	771K	7409M	32.10/0.8926	28.59/0.7808	27.57/0.7360	26.13/0.7853	30.32/0.9057
4ET	978K	9121M	32.29/0.8948	28.71/0.7830	27.64/0.7384	26.42/0.7936	30.69/0.9109
6ET	1392K	12647M	32.36/0.8965	28.80/0.7850	27.70/0.7405	26.69/0.8016	30.97/0.9135
8ET	1806K	16163M	32.40/0.8751	28.84/0.7858	27.73/0.7412	26.83/0.8048	31.11/0.9146
SAN [19]	15700K	12912M	32.64/0.9003	28.92/0.7888	27.78/0.7436	26.79/0.8068	31.18/0.9169

TABLE VII: PSNR/SSIM comparison on the RealSR dataset.

Scale	Bicubic		SRCNN [14]		VDSR [16]		SRResNet [45]		IMDN [11]		ESRT	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
$\times 2$	32.61	0.907	33.40	0.916	33.64	0.917	33.69	0.919	33.85	0.923	**33.92**	**0.924**
$\times 3$	29.34	0.841	29.96	0.845	30.14	0.856	30.18	0.859	30.29	0.857	**30.38**	**0.857**
$\times 4$	27.99	0.806	28.44	0.801	28.63	0.821	28.67	0.824	28.68	0.815	**28.78**	**0.815**

especially on Set5 and Urban100. For "8ET", the performance of the model continues to increase, but the increased amplification is not obvious. From the table, we can see that the biggest PSNR improvement with each increase in the number of ET is the Urban100 benchmark. This illustrates that Transformer architecture is able to model similar areas within the image in the SR task. We can see that our "8ET" model has close performance compared with the state-of-the-art method SAN [19] and our model has fewer parameters (only one-ninth the size of SAN). Our model even performs better on Urban100 benchmark than SAN. The increase of ET can improve the performance of Pure-ESRT, which also reflects that the built Pure-ESRT can achieve comparable SR performance compared with a well-designed CNN model.

D. Real World Image Super-resolution

To further verify the validity of the model, we also compare our ESRT with some classic lightweight SR models (e.g., SRCNN [14], VDSR [16], SRResNet [45], and IMDN [11]) on the real image dataset (RealSR [46]). All these models are retrained on RealSR dataset for a fair comparison. It is worth noting that as the shape of LR and HR is the same, the PixelShuffle is removed in our model and only one convolution layer is applied to change the feature map into SR images. The patch size is 128×128 during training. Other settings are the same as the training for DIV2K. The results are shown in TABLE VII. According to the table, the bicubic interpolation obtains the least average accuracy in three scales, indicating its low suitability for handling the real scene. Contrastly, our ESRT achieves the best PSNR and SSIM in all three scales with a large margin. Particularly, compared to IMDN, the performance of ESRT gains 0.07dB, 0.09dB, and 0.10dB for scaling factors $\times 2$, $\times 3$, and $\times 4$, respectively. Simultaneouslywe provide the visual comparisons in Fig. 12. Obviously, our ESRT recovers line edges effectively, such as some Chinese words and English words. Also, ESRT can restore the texture details well, such as the grid lines in the air conditioner. All these experiments shows that our method can still obtain a good SR property in the real world.

V. DISCUSSIONS

Here, we will give a brief view of the benefits and limitations of the proposed methods.

Benefits of LCB. LCB solves the problem of Transformer's poor feature extraction ability on small datasets. It is a lightweight architecture that can efficiently extract deep SR features. Meanwhile, LCB can be easily embedded into any SR model to reduce parameters and calculation costs, and maintain good performance.

Benefits of ET. ET solves the problem of heavy GPU memory consumption and large parameters in other vision Transformer. Meanwhile, ET can model the dependence between long-term sub-image blocks in the LR, enhancing the structural information of every image region. It has been improved that model such a long-term dependency of similar local regions is helpful for SR task. Meanwhile, ET is a lightweight and universal module that can be embedded into any present SR model to further improve model performance.

Limitations of ESRT. As we know, CNN with few layers has a small receptive field while with many layers has a heavy computational cost. By contrast, Transformer can capture the global contextual information by attention mechanism to establish long-term dependence in a sequence, to extract more powerful features. However, the current ESRT mainly consists of a convolutional neural network and a Transformer. This means that the convolution operations are not completely eliminated in the model. In future work, we will explore a full Transformer SR model to further improve the model performance and reduce the computational cost.

VI. CONCLUSION

In this work, we propose a novel efficient SR Transformer (ESRT) for lightweight SISR. ESRT first utilizes a lightweight CNN backbone (LCB) to extract deep features and then uses a lightweight Transformer backbone (LTB) to model the long-term dependence between similar local regions in the image. In LCB, we propose a high preserving block (HPB)

to reduce the computational cost by down-sampling and up-sampling the feature map progressively and retain the high-frequency information to the restored feature map with the help of the designed high-frequency filtering module. In LTB, an efficient Transformer is designed to enhance the feature representation ability at a low computational cost and GPU memory occupation with the help of proposed efficient multi-head attention (EMHA). Extensive benchmark and real-world datasets demonstrate that our ESRT achieves the best trade-off between model performance and computation cost.

REFERENCES

[1] Z. Li, J. Yang, Z. Liu, X. Yang, G. Jeon, and W. Wu, "Feedback network for image super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3867–3876.

[2] Y. Tai, J. Yang, and X. Liu, "Image super-resolution via deep recursive residual network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3147–3155.

[3] J. Kim, J. K. Lee, and K. M. Lee, "Deeply-recursive convolutional network for image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1637–1645.

[4] Y. Tai, J. Yang, X. Liu, and C. Xu, "Memnet: A persistent memory network for image restoration," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 4539–4547.

[5] X. Chu, B. Zhang, H. Ma, R. Xu, and Q. Li, "Fast, accurate and lightweight super-resolution with neural architecture search," *arXiv preprint arXiv:1901.07261*, 2019.

[6] X. Chu, B. Zhang, and R. Xu, "Multi-objective reinforced evolution in mobile neural architecture search," in *European Conference on Computer Vision*. Springer, 2020, pp. 99–113.

[7] N. Ahn, B. Kang, and K. A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," in *Proc. ECCV*, 2018, pp. 252–268.

[8] X. Wang, Q. Wang, Y. Zhao, J. Yan, L. Fan, and L. Chen, "Lightweight single-image super-resolution network with attentive auxiliary feature learning," in *Proceedings of the Asian Conference on Computer Vision*, 2020.

[9] J. Li, F. Fang, K. Mei, and G. Zhang, "Multi-scale residual network for image super-resolution," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 517–532.

[10] J. Li, F. Fang, J. Li, K. Mei, and G. Zhang, "Mdcn: Multi-scale dense cross network for image super-resolution," *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.

[11] Z. Hui, X. Gao, Y. Yang, and X. Wang, "Lightweight image super-resolution with information multi-distillation network," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 2024–2032.

[12] Z. Hui, X. Wang, and X. Gao, "Fast and accurate single image super-resolution via information distillation network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 723–731.

[13] R. Timofte, S. Gu, J. Wu, and L. Van Gool, "Ntire 2018 challenge on single image super-resolution: Methods and results," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 852–863.

[14] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2015.

[15] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *European conference on computer vision*. Springer, 2016, pp. 391–407.

[16] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1646–1654.

[17] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 136–144.

[18] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 286–301.

[19] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang, "Second-order attention network for single image super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 065–11 074.

[20] Y. Qiu, R. Wang, D. Tao, and J. Cheng, "Embedded block residual network: A recursive restoration model for single-image super-resolution," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4180–4189.

[21] Y. Mei, Y. Fan, Y. Zhou, L. Huang, T. S. Huang, and H. Shi, "Image super-resolution with cross-scale non-local attention and exhaustive self-exemplars mining," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5690–5699.

[22] W. Lee, J. Lee, D. Kim, and B. Ham, "Learning with privileged information for efficient image super-resolution," in *European Conference on Computer Vision*. Springer, 2020, pp. 465–482.

[23] X. Luo, Y. Xie, Y. Zhang, Y. Qu, C. Li, and Y. Fu, "Latticenet: Towards lightweight image super-resolution with lattice block."

[24] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[25] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," *arXiv preprint arXiv:2012.12877*, 2020.

[26] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*. Springer, 2020, pp. 213–229.

[27] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," *arXiv preprint arXiv:2010.04159*, 2020.

[28] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao, "Pre-trained image processing transformer," *arXiv preprint arXiv:2012.00364*, 2020.

[29] F. Yang, H. Yang, J. Fu, H. Lu, and B. Guo, "Learning texture transformer network for image super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5791–5800.

[30] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Decoupled spatial-temporal attention network for skeleton-based action-gesture recognition," in *Proceedings of the Asian Conference on Computer Vision*, 2020.

[31] C. Plizzari, M. Cannici, and M. Matteucci, "Spatial temporal transformer network for skeleton-based action recognition," *arXiv preprint arXiv:2008.07404*, 2020.

[32] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *arXiv preprint arXiv:1506.01497*, 2015.

[33] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[34] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[35] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[37] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. 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, pp. 1874–1883.

[38] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," 2012.

[39] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE transactions on image processing*, vol. 19, no. 11, pp. 2861–2873, 2010.

[40] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 2. IEEE, 2001, pp. 416–423.

[41] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5197–5206.

[42] K. Aizawa, A. Fujimoto, A. Otsubo, T. Ogawa, Y. Matsui, K. Tsubota, and H. Ikuta, "Building a manga dataset "manga109" with annotations for multimedia applications," *IEEE MultiMedia*, vol. 27, no. 2, pp. 8–18, 2020.
[43] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Fast and accurate image super-resolution with deep laplacian pyramid networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 11, pp. 2599–2613, 2018.
[44] K. Zhang, W. Zuo, and L. Zhang, "Learning a single convolutional super-resolution network for multiple degradations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3262–3271.
[45] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4681–4690.
[46] J. Cai, H. Zeng, H. Yong, Z. Cao, and L. Zhang, "Toward real-world single image super-resolution: A new benchmark and a new model," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3086–3095.

Zhisheng Lu received the B.S. degree from School of Computer Science and Engineering, Nanjing University of Science and Technology (NJUST). He is currently pursuing his master's degree in Computer Science at Peking University (PKU), China. His research interest lies in computer vision and image processing.

Hong Liu received the Ph.D. degree in mechanical electronics and automation in 1996. He is currently a Full Professor in the School of Electronics Engineering and Computer Science, Peking University (PKU), Beijing, China. He has been selected as Chinese Innovation Leading Talent supported by "National High-level Talents Special Support Plan" since 2013. He is also the Director of Open Lab on Human Robot Interaction, PKU. He has published more than 150 papers. His research interests include computer vision and robotics, image processing, and pattern recognition. He received the Chinese National Aero-space Award, the Wu Wenjun Award on Artificial Intelligence, the Excellence Teaching Award, and the Candidates of Top Ten Outstanding Professors in PKU. He is the Vice President of Chinese Association for Artificial Intelligent (CAAI), and the Vice Chair of Intelligent Robotics Society of CAAI. He has served as keynote speakers, Co-Chairs, Session Chairs, or PC members of many important international conferences, such as IEEE/RSJ IROS, IEEE ROBIO, IEEE SMC, and IIHMSP, and serves as reviewers for many international journals such as Pattern Recognition, the IEEE Transactions on Signal Processing, and the IEEE Transactions on Pattern Analysis and Machine Intelligence.

Juncheng Li received the Ph.D. degree in computer science from East China Normal University, Shanghai, China, in 2021. He is currently a postdoctoral researcher with the Department of Mathematics, The Chinese University of Hong Kong. His main research interests include artificial intelligence and its applications to computer vision and image processing (e.g., image super-resolution, image denoising, and image dehazing).

Linlin Zhang received the B.S. degree from School of Information Engineering, Minzu University of China. She is currently pursuing his master's degree in Computer Science at Peking University (PKU), China. Her research interest lies in computer vision and human action recognition.