Lightweight Image Super-Resolution with Scale-wise Network

Xiaole Zhao*
zxlation@foxmail.com
Xinkun Wu
wulisinerti@gmail.com

School of Computing and
Artificial Intelligence,
SWJTU, Chengdu, Sichuan, China

Abstract

Recent advancements in Single Image Super Resolution (SISR) have been achieved by utilizing deep neural networks with a high number of layers. Incorporating multi-scale information is crucial in designing advanced super-resolution networks. In this paper, we present a novel feature upscaling method for SISR tasks which enhances the multi-scale information of images through multi-mode interaction. The multi-scale block design introduces more abundant image information, and the designed feature extraction module works in tandem with the multi-scale module to restore details in low-resolution images, thus enhancing image clarity. The static features and dynamic information in different scales of the image are fused and interacted through the local channel attention module to adjust the importance of different modes in recognizing different actions, thereby improving the final performance of the model. Numerous experiments confirm that our model achieves higher accuracy on Set5, Set14, B100, and Urban100 datasets compared to other state-of-the-art methods, while requiring relatively low computational and memory resources.

1 Introduction

The task of Super-Resolution (SR) involves reconstructing a low-resolution (LR) image into a high-resolution (HR) image. However, obtaining HR images from LR is an ill-posed problem, and therefore, the model needs to learn the raw data distribution to produce the most likely solution.

Convolutional neural networks (CNNs) have emerged as a dominant technique for solving SISR problems due to their increased depth and width, which results in improved performance. Despite their excellence, most deep networks suffer from certain limitations. Firstly, deep super resolution networks rely on a large number of model parameters, which lead to increased computing requirements and memory consumption during network training. Secondly, as the depth of the network increases, a significant amount of characteristic information is lost during the continuous nonlinear operations that produce the output.

Apart from the previously mentioned issues, most conventional upsampling modules utilize single-layer PixelShuffle [3] or Bicubic, which leads to a loss of feature information

*Corresponding author.

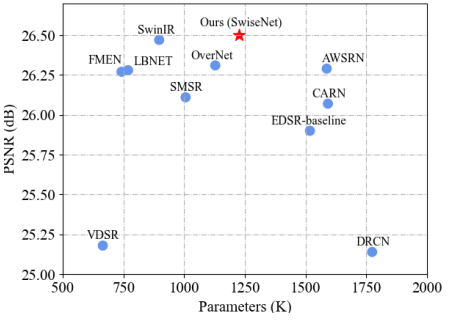

Figure 1: Comparison of model performance and complexity on Urban100 [14] with SR×4.

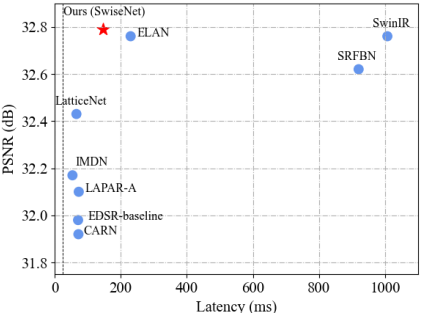

Figure 2: Comparison of model performance and latency on Urban100 [14] with SR×2.

from other scales. However, the missing information is also crucial for reconstructing high-quality image networks.

To address these issues, we propose a novel lightweight SISR method, dubbed Scale-wise Network (SwiseNet), which utilizes low-level feature information and explores the representational capabilities of deep networks. SwiseNet consists of two main components: lightweight feature extractor and Scale-wise Upsample Module (SUM) for image upsampling. SUM is a novel induction bias that generates accurate SR images by internally constructing multi-scale intermediate representations of output features. The feature extractor follows a new recursive framework of skip and dense connections to reduce low-level feature degradation. To further simplify the baseline, we reveal that nonlinear activation functions, such as the Rectifiers Linear Unit (RELU) [23] and Gaussian Error Linear Unit (GELU) [12], etc. are unnecessary: they can be replaced by simple multiplication or removed. SwiseNet is superior to the state-of-the-art lightweight SR models as shown in Figure 1 and Figure 2, we compare SwiseNet against various benchmark algorithms in terms of network parameters, latency and reconstruction PSNR, using the Urban100 dataset with a scale of $\times 4$ and $\times 2$. Our key contributions are summarized as follows:

- A lightweight recursive feature extractor that improves performance even in the most advanced models, including those with an order of magnitude more parameters.

- A Scale-wise Upsample module (SUM) to retain multi-scale information that helps restore HR images accurately. The module improves reconstruction accuracy effectively according to the number of parameters. Additionally, we demonstrate how the module can be integrated into an existing state-of-the-art model to improve its original performance.

- SUM favors lightweight model design. Based on this, we construct a lightweight SwiseNet, using the multi-scale information fusion strategy to extract multi-scale context information.

- Replacing the upsampling operation with our SUM produces better SR results for the baseline model. Compared to state-of-the-art methods, SwiseNet achieved excellent performance on several benchmark datasets while maintaining relatively low model scale and computational complexity.

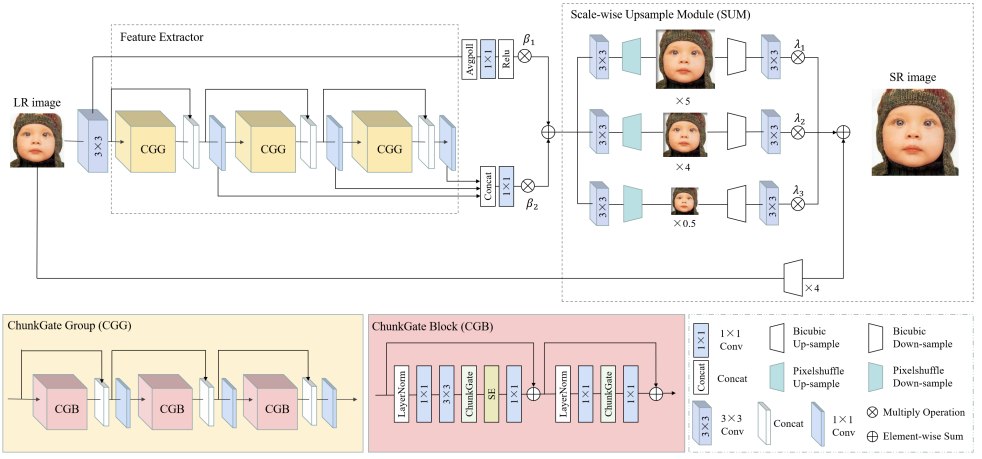

Figure 3: The overall structure of our proposed SwiseNet model. As the maximum scaling factor in this example is set to N = 4, the required scaling-factors are $\times 4$, $\times 5$ and $\times 0.5$.

2 Realated Works

2.1 Image Super-Resolution with Deep Learning

SRCNN is recognized as the first deep learning-based method for solving the SISR task [8], which has led to the development of numerous deep SISR models that achieve remarkable performance. Dong et al. [8] utilizes a three-layer convolutional neural network for SISR, and subsequent models have improved upon it in various ways. For instance, Anwar S and Barnes N. [2] employs cascading residual on the residual structure to allow the flow of low-frequency information to focus on learning high and mid-level features. Liang et al. [19] leverages the Swin Transformer for SR quest. Chen et al. [6] proposes a hybrid attention planning and pre-training strategies. Despite their impressive results, these deep SISR models are often limited by high memory storage and computational complexity.

2.2 Lightweight Image Super-Resolution Models

The significance of lightweight models in saving computing resources has prompted discussions on how to reduce the number of parameters and operations required for deep learn-based SR to improve its effectiveness in mobile applications. To this end, researchers have proposed various models with improved efficiency and reduced complexity. For instance, Kim, J et al. [15] implements a recursive neural network that reuses parameters multiple times to reduce the number of parameters in SISR. Ahn et al. [2] uses a recursive cascade mechanism and combines group convolution with 1×1 convolution. Similarly, Li et al. [18] replaces standard convolution with depth-separable convolution. Zhou et al. [39] significantly reduces the number of model parameters by introducing pixel normalization and increasing the receptive field of attention branches. Despite the significant progress made in the research of lightweight SR models, they are still in the early stages and require further discussions.

2.3 Upsampling Strategies

Upsampling is a critical step in SISR that involves generating high-resolution (HR) images based on high-level features extracted from a low-dimensional space. Interpolation is a widely used upsampling method in SR networks, such as [8, 16, 26]. In this method, the low-resolution (LR) image is first adjusted to the target size and then used as input for the CNN model to reconstruct the HR image. However, this increases the computation load due to the large input image size. Therefore, some networks such as [9] and [27] use the LR image directly as input and add a transposed convolution layer to perform the final upsampling reconstruction. This approach significantly reduces unnecessary computing overhead. Furthermore, Shi et al. [25] introduced Pixelshuffle as a method to overcome the checkerboard effect problem that arises in transposed convolution. Pixelshuffle [3] has been widely used in recent SR models, including [21, 32, 37]. However, traditional upsampling methods only use a single scale information for upsampling, which may result in the loss of a significant amount of detail and lead to fuzzy outputs.

3 Proposed Method

This section begins with an introduction of the overall network architecture, which is illustrated in Figure 3. The grid structure consists of two main components: feature extraction and Scale-wise Upsampling. We then present the loss function used to optimize the model. Finally, we compare our method with existing approaches to highlight the differences.

3.1 Network Architecture

Given a set of HR images and their reduced version I^{HR} and I^{LR}, the goal of SISR is to find a function $\mathcal{F}$ that maps the LR image to its original HR image. This problem is untenable because there are multiple possible HR images for one LR image. However, it is possible to learn the most likely reconstruction $\mathcal{L}$ by parameterizing $\mathcal{F}$ over a set of parameters θ and finding the most likely θ given the conditions:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum \mathcal{L}\left(\mathcal{F}(I^{LR}, \theta), I^{HR}\right) \tag{1}$$

We chose $\mathcal{L}$ to be the L_1 distance, since we empirically obtained superior PSNR results compared to L_2. In this work $\mathcal{F}$ is composed of two parts: (i) a feature extractor $\mathcal{H}$:

$$\mathbf{h} = \mathcal{H}\left(I^{LR}, \theta_h\right) \tag{2}$$

with parameters θ_h, and (ii) the Scale-wise Upsample Module $\mathcal{S}$:

$$\hat{I}^{HR} = \mathcal{S}(\mathbf{h}, \theta_s) \tag{3}$$

with θ_s the parameters used in this operation, and $\hat{I}^{HR}$ the reconstructed image. They are described in detail below.

3.2 Feature Extractor

The feature extractor is responsible for computing the LR patch that contains valid information for recovering the HR image. Our proposed approach utilizes a recursive structure called ChunkGate Blocks (CGBs), which is based on ChunkGate (CG) and incorporates channel

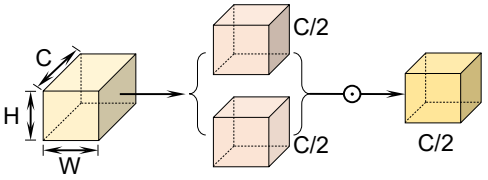

Figure 4: Diagram of ChunkGate unit (CG). The intermediate feature is evenly divided into 2 sub-features along the channel direction, which are fused together by simple element-wise product $\odot$. The operation halves the number of channels for the feature.

attention along with several convolutional layers. Figure 3 provides a schematic diagram of this structure.

Normalization. Normalization techniques have become popular in advanced computer vision tasks. Two types of normalization techniques are commonly used in deep learning neural networks: layer normalization and batch normalization [30]. In SISR tasks, layer normalization is often performs better than batch normalization for several reasons. Small batches can be unstable with batch normalization, while layer normalization is better suited for small batches. Additionally, layer normalization preserves spatial information better than batch normalization, which may damage spatial information in images. Layer normalization also reduces the model's dependence on initialization, while batch normalization may increase dependence on initial weights. Our findings suggest that layer normalization may be critical to feature extractors. Therefore, we added layer normalization to the ChunkGate Block, improving the training process and allowing for higher learning rates, resulting in significant performance improvements.

Nonlinear Activation Function. While GeLU [12] has gained popularity in computer vision, we aim to investigate whether its performance can be enhanced while retaining the same number of parameters, or if it can be simplified without compromising its performance. To answer these questions, we examine some of the latest state-of-the-art (SOTA) methods, such as [13, 20, 28, 33]. Our analysis reveals that all of these methods use gated linear units (GLU) [7]. Building on these observations, we propose a simplified version of GLU, called ChunkGate, which directly divides the feature map into two parts along the channel dimension and multiplies them, as illustrated in Figure 4 and more detailed analyse are given in supplementary material.

ChunkBlock and ChunkGroup. The feature extractors are organized into groups named ChunkGate Groups (CGGs). Each ChunkGate-Block (CGB) takes the input concatenated with the output of all preceding CGBs in the group and merges it with a 1×1 convolution. This process is repeated for all CGBs in the CGG, thus aggregating all local information through the 1x1 convolution layer step by step.

Besides, we introduce a 1×1 convolutional layer to adaptively merge the output information, as using the features of these connections directly would greatly increase the computational complexity. The output of these hierarchical features can be expressed as:

$$\mathbf{f}_D = \mathrm{Conv}_{1\times1}([\mathbf{f}_0,\ldots,\mathbf{f}_{D-1}]) \tag{4}$$

where $[\mathbf{f}_0,\ldots,\mathbf{f}_{D-1}]$ refers to the concatenation of feature maps produced by CGGs.

To preserve all the important information before the reconstruction step, Anwar et al. [9] used a structure of Residual In Residual (RIR) and Dense Residual Laplacian Module (DRLM).However, feature concatenation is widely-used in network design for enhancing the representation capacity of deep models, especially in low-level vision tasks. In general,

our SwiseNet differentiates with Anwar's work mainly in two aspects: (i) for the backbone of the networks, Anwar et al. used a structure of residual in residual (RIR) with the output of cascading blocks coming from residual connections. However, the output of feature extraction in our model is the channel concatenation of 1×1 convolutions after each CGG; (ii) for building blocks, the DRLM in Anwar's model adopted feature concatenation in a densely connected manner, while feature concatenation in our CGG is not densely connected.

We access the original information by adding a remote skip connection and facilitates gradient propagation from the output of the feature extractor back to the first 3×3 convolutional layer. Additionally, we have included a global average pooling layer and a 1×1 convolutional layer to fully capture the channel dependencies of the aggregated information. The final output just before the reconstruction step can be expressed as:

$$\mathbf{h} = \beta_1 \mathbf{f}_D + \beta_2 \sigma\left(\mathrm{Conv}_{1\times1}\left(\mathrm{GAP}\left(\mathrm{Conv}_{3\times3}\left(I^{LR}\right)\right)\right)\right) \tag{5}$$

where σ denotes the ReLU activation, GAP denotes global average pooling, and β_1 and β_2 are learned parameters.

3.3 Scale-wise Upsample Module

In this study, we propose a novel inductive bias for SISR that produces higher quality images with fewer artifacts. The traditional upsampling method employs a single scale information for upsampling, which may result in information loss and ambiguity. However, our proposed method, called multi-scale fusion, utilizes feature information from multiple scales simultaneously for up-sampling. This approach has the advantage of integrating information from different scales to provide a more comprehensive understanding of the image features, leading to improved performance and network robustness.

We propose that utilizing multiple scales can provide additional information for the same pixel, effectively acting as integrations and improving the SR performance by fusing these information. We hypothesize that the maximum scale factor N during network training can impact this process. To implement this idea, we first generate an intermediate representation of the final image using a multi-scale structure called H^{SR}, which comprises three different multiples (N_1, N_2 and N_3) of multi-scale coefficients. We then extract the feature vector $\mathbf{h}$ from the low-resolution image I^{LR} and map it to different scale channels using a 3×3 convolution layer. Finally, we apply the strided sub-pixel convolution proposed in [3] to obtain the HR image:

$$\begin{cases} H^{SR_1} = \mathrm{PixelShuffle}(\mathrm{Conv}_{3\times3}(\mathbf{h}), N_1) \\ H^{SR_2} = \mathrm{PixelShuffle}(\mathrm{Conv}_{3\times3}(\mathbf{h}), N_2) \\ H^{SR_3} = \mathrm{PixelShuffle}(\mathrm{Conv}_{3\times3}(\mathbf{h}), N_3) \end{cases} \tag{6}$$

In order to produce the ultimate outcome of the multi-scale module, we employ a long-range skip connection from the initial I^{LR} image. To achieve the final HR image, we adjust the multi-scale images and combine them into the upsampling of the original LR image.

$$\begin{cases} I^{HR_1} = \mathrm{Bicubic}\left(H^{SR_1}, N\right) \\ I^{HR_2} = \mathrm{Bicubic}\left(H^{SR_2}, N\right) \\ I^{HR_3} = \mathrm{Bicubic}\left(H^{SR_3}, N\right) \end{cases} \tag{7}$$

$$\hat{I}^{HR} = \sum_{i=1}^{3} \lambda_i \cdot \mathrm{Conv}_{3\times3}\left(I^{HR_i}\right) + \mathrm{Bicubic}_{\uparrow}\left(I^{LR}\right) \tag{8}$$

Table 1: Average PSNR/SSIM for 2×, 3×, 4× SR. The best results are highlighted in red color and the second best is in blue.

Method	Scale	#Params[K]	#FLOPs[G]	Set5	Set14	BSD100	Urban100
				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
SRCNN[8]		57	52.7	36.66 / 0.9542	32.45 / 0.9067	31.36 / 0.8879	29.50 / 0.8946
FSRCNN[9]		12	6.0	37.05 / 0.9560	32.66 / 0.9090	931.53 / 0.8920	29.88 / 0.9020
VDSR[16]		665	612.6	37.53 / 0.9590	33.05 / 0.9130	31.90 / 0.8960	30.77 / 0.9140
LAPAR-A[23]		1774	9788.7	37.63 / 0.9588	33.04 / 0.9118	31.85 / 0.8942	30.75 / 0.9133
EDSR-baseline[20]	×2	1370	316.2	37.91 / 0.9602	33.53 / 0.9172	32.15 / 0.8995	31.99 / 0.9270
CARN[1]		1592	222.8	37.76 / 0.9590	33.52 / 0.9166	32.09 / 0.8978	31.92 / 0.9256
SMSR[25]		985	224.1	38.00 / 0.9601	33.64 / 0.9179	32.17 / 0.8990	32.19 / 0.9284
LBNET[11]		731	153.2	38.05 / 0.9607	33.65 / 0.9177	32.16 / 0.8994	32.30 / 0.9291
FMEN[17]		748	172.0	38.10 / 0.9609	33.75 / 0.9192	32.26 / 0.9007	32.41 / 0.9311
SwinIR[15]		878	195.6	38.14 / 0.9610	33.86 / 0.9206	32.31 / 0.9012	32.76 / 0.9338
SwiseNet(w/o SUM)		1035	212.7	38.10 / 0.9605	33.85 / 0.9207	32.22 / 0.9008	32.64 / 0.9326
SwiseNet		1077	236.5	38.18 / 0.9610	33.94 / 0.9217	32.28 / 0.9012	32.79 / 0.9339
Bicubic		-	-	33.39 / 0.8682	27.55 / 0.7742	27.21 / 0.7385	24.46 / 0.7349
SRCNN[8]		57	52.7	32.75 / 0.9090	29.30 / 0.8215	28.41 / 0.7863	26.24 / 0.7989
FSRCNN[9]		13	5.0	33.18 / 0.9140	29.37 / 0.8240	28.53 / 0.7910	26.43 / 0.8080
VDSR[16]		666	612.6	33.66 / 0.9213	29.77 / 0.8314	28.82 / 0.7976	27.14 / 0.8279
DRCN[18]		1774	9788.7	33.82 / 0.9226	29.76 / 0.8311	28.80 / 0.7963	27.15 / 0.8276
EDSR-baseline[20]	×3	1554	160.4	34.28 / 0.9263	30.24 / 0.8405	29.06 / 0.8044	28.00 / 0.8493
CARN[1]		1592	118.8	34.29 / 0.9255	30.29 / 0.8407	29.06 / 0.8034	28.06 / 0.8493
SMSR[25]		993	100.5	34.40 / 0.9270	30.33 / 0.8412	29.10 / 0.8050	28.25 / 0.8536
LBNET[11]		736	68.4	34.47 / 0.9277	30.38 / 0.8417	29.13 / 0.8061	28.42 / 0.8559
FMEN[17]		757	77.2	34.45 / 0.9275	30.40 / 0.8435	29.17 / 0.8063	28.33 / 0.8562
SwinIR[15]		886	87.2	34.60 / 0.9289	30.54 / 0.8463	29.19 / 0.8082	28.66 / 0.8624
SwiseNet(w/o SUM)		1044	112.8	34.43 / 0.9268	30.44 / 0.8442	29.11 / 0.8066	28.45 / 0.8590
SwiseNet(Ours)		1068	128.0	34.61 / 0.9289	30.52 / 0.8453	29.20 / 0.8079	28.63 / 0.8615
Bicubic		-	-	28.42 / 0.8104	26.00 / 0.7027	25.96 / 0.6675	23.14 / 0.6577
SRCNN[8]		57	52.7	30.48 / 0.8628	27.50 / 0.7513	26.90 / 0.7101	24.52 / 0.7221
FSRCNN[9]		12	4.6	30.72 / 0.8660	27.61 / 0.7550	26.98 / 0.7150	24.62 / 0.7280
VDSR[16]		665	612.6	31.35 / 0.8830	28.02 / 0.7680	27.29 / 0.7260	25.18 / 0.7540
DRCN[18]		1774	9288.7	31.53 / 0.8854	28.02 / 0.7670	27.23 / 0.7233	25.14 / 0.7510
EDSR-baseline[20]	×4	1518	114.2	31.98 / 0.8927	28.55 / 0.7805	27.54 / 0.7348	25.90 / 0.7809
CARN[1]		1592	90.9	32.13 / 0.8937	28.60 / 0.7806	27.58 / 0.7349	26.07 / 0.7837
SMSR[25]		1006	57.2	32.12 / 0.8932	28.55 / 0.7808	27.55 / 0.7351	26.11 / 0.7868
LBNET[11]		742	38.9	32.29 / 0.8960	28.68 / 0.7832	27.62 / 0.7382	26.27 / 0.7906
FMEN[17]		769	44.2	32.24 / 0.8955	28.70 / 0.7839	27.63 / 0.7379	26.28 / 0.7908
SwinIR[15]		897	49.6	32.42 / 0.8976	28.77 / 0.7858	27.68 / 0.7406	26.47 / 0.7980
SwiseNet(w/o SUM)		1056	75.3	32.31 / 0.8955	28.65 / 0.7829	27.57 / 0.7363	26.35 / 0.7928
SwiseNet(Ours)		1227	88.6	32.43 / 0.8979	28.72 / 0.7844	27.68 / 0.7396	26.48 / 0.7969

Where, λ_1, λ_2 and λ_3 are the weights corresponding to I^{HR_1}, I^{HR_2} and I^{HR_3}.

The whole network can be regarded as a tool for improving or modifying the bicubic upsampling of LR input. To enhance the SR performance, multi-scale weighted summation can be used. This approach enables the integration of information from different scales, improving the model's ability to understand image features comprehensively. As a result, the model becomes more accurate and adaptable to various image scenes, producing images that closely resemble the actual HR image.

4 Experiments

Datasets. The model is trained with a high-quality dataset DIV2K [1], which is widely used for SISR task. It includes 800 training images and 100 validation images with rich textures. In addition, we used several baseline datasets for testing, including Set5 [5], Set14 [34], B100 [22], and Urban100 [14]. To evaluate the super resolution results, we used two commonly used indicators: PSNR (Peak Signal to Noise Ratio) and SSIM (Structural Similarity) on the Y channel of the YCbCr color space.

Degradation model. To demonstrate the effectiveness of the proposed method, we em-

Table 2: Quantitative comparison of different SR models with and without SUM on Set5 dataset for different scaling factors.

DataSet	Scale	CARN[1]	CARN-SUM	Margin	EDSR[23]	EDSR-SUM	Margin	RCAN[43]	RCAN-SUM	Margin	RDN[43]	RDN-SUM	Margin
Set5	×2	37.76	**37.85**	+0.09	38.20	**38.26**	+0.06	38.27	**38.38**	+0.11	38.24	**38.34**	+0.10
	×3	34.29	**34.36**	+0.07	34.76	**34.83**	+0.07	34.74	**34.81**	+0.07	34.71	**34.79**	+0.08
	×4	32.13	**32.18**	+0.05	32.62	**32.65**	+0.03	32.63	**32.68**	+0.05	32.47	**32.56**	+0.09
Set14	×2	33.52	**33.59**	+0.07	34.02	**34.07**	+0.05	34.12	**34.16**	+0.04	34.01	**34.08**	+0.07
	×3	30.29	**30.36**	+0.07	30.66	**30.73**	+0.07	30.65	**30.69**	+0.04	30.57	**30.66**	+0.09
	×4	28.60	**28.69**	+0.09	28.94	**29.00**	+0.06	28.87	**28.96**	+0.09	28.81	**28.89**	+0.08
Urban100	×2	31.92	**32.01**	+0.09	33.10	**33.13**	+0.03	33.34	**33.41**	+0.07	32.89	**32.95**	+0.06
	×3	28.06	**28.11**	+0.05	29.02	**29.07**	+0.05	29.09	**29.15**	+0.06	28.80	**28.87**	+0.07
	×4	26.07	**26.12**	+0.05	26.86	**26.93**	+0.07	26.82	**26.95**	+0.13	26.61	**26.69**	+0.08

Table 3: The effectiveness of Layer Normalization (LN), ChunkGate (CG), and Squeeze and Excitation (SE) was verified with SR×2 on Set5 and Set14 datasets.

Config	LN	GeLU→CG	SE	Set5		Set14	
				PSNR	SSIM	PSNR	SSIM
CGB	×	×	×	37.99	0.9596	33.69	0.9190
	✓	×	×	38.04	0.9598	33.82	0.9198
	✓	✓	×	38.10	0.9602	33.89	0.9208
	✓	✓	✓	**38.18**	**0.9611**	**33.94**	**0.9217**

Table 4: Comparisons between several typical lightweight models in terms of performance and efficiency indicators. The results are collected on Urban100 with SR×2, with PyTorch 1.11.0, CUDA 11.3 and NVIDIA Tesla V100 (32G). The latency and throughput are averaged over the entire Urban100.

SR Models	Parameters	Latency (ms)	Throughput (req/s)	PSNR (dB)
CARN	1592K	72	880	31.92
EDSR-baseline	1370K	71	905	31.98
IMDN	694K	54	1172	32.17
SwinIR-light	878K	1007	63	32.76
SwiseNet (ours)	1077K	147	440	32.79

ployed three degradation models to simulate LR images, as previously described in [35, 36, 38]. The first model, BI, involved generating LR images by bicubic-downsampling the ground truth HR image (×2, ×3, ×4). Details and more visual results on the other two models, DB and DN, can be found in the supplementary material.

Implementation details. We call the original model SwiseNet and further introduce SwiseNet without SUM. For training, we used 64 × 64 RGB input patches sampled from the LR image, with the LR patches randomly flipped and rotated by 90 degrees. In all experiments, we set the number of CGGs and CGBs to 3. We employed the ADAM optimizer [17] and applied weight normalization to all convolutional layers [31]. The mini-batch size was set to 64 and the learning rate was initialized to 10^{-3}. The learning rate was halved with each iteration of backpropagation. We implemented our network using the PyTorch framework [24] and trained it on the Tesla V100.

4.1 Comparison with State-of-the-art Methods

We compared our proposed SwiseNet with ten lightweight state-of-the-art SISR methods [2, 8, 9, 10, 11, 15, 16, 19, 21, 29]. We also trained SwiseNet without Scale-wise Upsample Module by using a typical sample on pixelshuffle (SwiseNet w/o SUM). To ensure a fair comparison, we trained the model separately for each scaling factor, including ×2, ×3, and

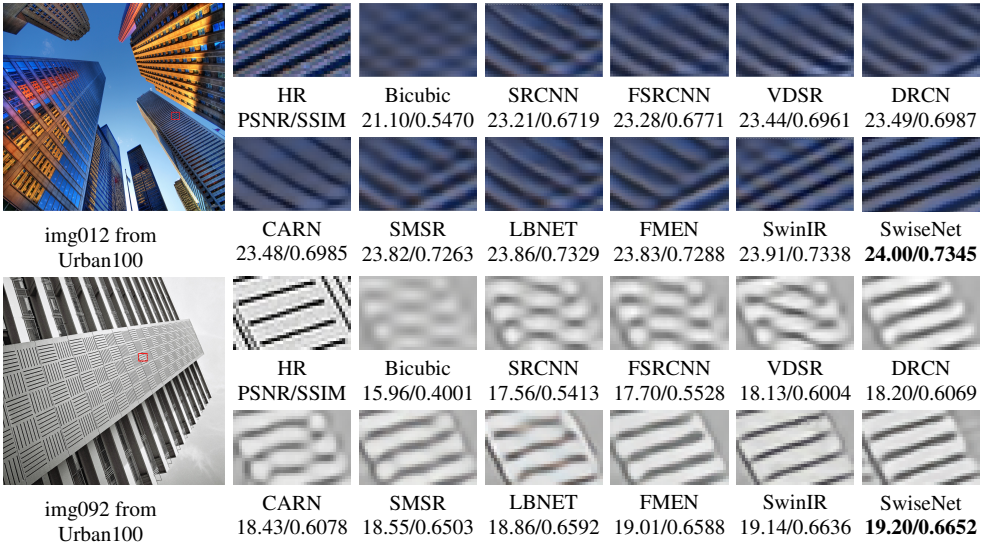

Figure 5: Visual comparisons of the state-of-the-art lightweight methods and our SwiseNet Urban100 for 4× SR. Zoom in for best view.

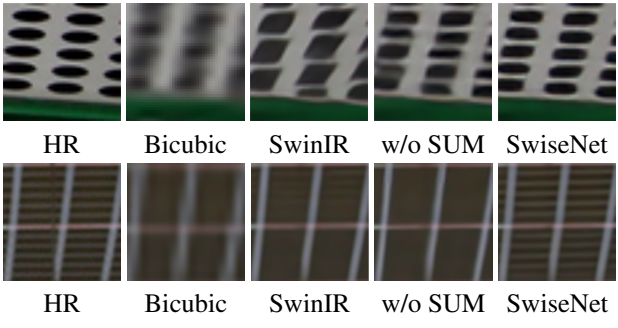

Figure 6: Qualitative comparison of SwiseNet (w/o SUM) for ×4 upscaling. The test images are img004 (top) and img044 (bottom) from Urban100, respectively.

×4. Our model was tested using PSNR and SSIM against various benchmarks.

In Table 1, we present the quantitative evaluation results, including the number of parameters and the number of multiplication and additions (Multi-Adds), for a more informative comparison (under the name of the method). Multi-Adds were calculated using 1280×720 SR images at all scales. It is important to note that we only compare models with similar numbers of parameters in this table. Our SwiseNet outperforms previous approaches on numerous baseline datasets. Additionally, our SwiseNet without SUM has similar or better results. The results show that both SUM and the proposed feature extractor independently improve PSNR compared to other SR methods. Finally, the proposed feature extractor is combined with SUM to further improve performance.

Table 4 illustrates the comparisons between several typical lightweight models in terms of performance on Urban100 (SR×2) and efficiency indicators, including SwinIR. It can be seen that (i) although SwinIR achieves excellent SR performance with relatively few parameters, it is not user-friendly for practical deployment since its inference is too slow and throughput is much smaller than CNN-based models; (ii) our SwiseNet shows better SR re-

sults than SwinIR with less inference time and larger throughput, as stated above. Moreover, our SwiseNet shows a significant performance margin compared to other CNN-based models (e.g., IMDN + 0.62dB, EDSR-baseline + 0.81dB and CARN + 0.87dB). Therefore, the proposed model significantly promotes the compromise between SR performance and model efficiency.

Furthermore, we present qualitative results in Figure 5 and Figure 6, which demonstrate that our solutions produce high-quality image structures.

4.2 Ablation Studies

To analyze the performance behavior of the proposed method, we examined the effects of the Scale-wise Upsample Module in other models. Additionally, we investigated the effects of the ChunkGate unit, Layer Normalization, and attention mechanism within the CGBs.

SUM across the architecture. The aim of this section is to demonstrate the effectiveness of SUM across various architectures. To achieve this, we utilized state-of-the-art networks, such as CARN [2], EDSR [21], RDN [38], and RCAN [37], as benchmarks. We replaced their conventional upsampling modules with our SUM, denoted as CARN-SUM, EDSR-SUM, RDN-SUM, and RCAN-SUM in Table 2, respectively. We trained these models on all scaling factors while keeping their original training settings unchanged.

ChunkGate Block. The ChunkGate block, which is defined in Section 3.2 and illustrated in Figure 3, was found to cause unstable training of SwiseNet in its default configuration. However, this issue was ultimately resolved by introducing layer normalization (LN), as illustrated in Section 3.2. This resulted in a more stable training process. Additionally, the effectiveness of ChunkGate was demonstrated, as shown in Table 3.

Several ablations are presented in the supplementary material, including a study on the effects of skip connections (SCs) in ChunkGate blocks and groups (CGBs, CGGs), the influence of different interpolation methods on reconstruction, evaluation results obtained by applying BD and DN degradation models and comparisons with six existing SR methods, and an analysis of the generalization ability of our architecture across different scales.

5 Conclution

In this paper, we propose a scale-wise upsample block, a lightweight structure that favors efficient model design and outperforms state-of-the-art algorithms with fewer parameters and lower computational requirements. Our main contributions are twofold: (i) a lightweight feature extractor that improves information propagation and maintains image details, where the attention mechanism is used to learn the combination coefficients of the ChunkGate Block, resulting in improved super-resolution performance. (ii) A Scale-wise Upsample Module that generates accurate SR images by combining different scale factors. This block can be embedded in any super-resolution network as an upsample block. We also created SwiseNet, a lightweight super-resolution model based on the Scale-wise Upsample Module, and used a skip connection strategy to integrate contextual information from different receptive fields. Experimental results on several benchmark datasets demonstrate that our method can achieve superior performance with moderate parameters on several benchmark datasets.

Acknowledgement: This work is supported in part by the National Natural Science Foundation of China (Nos. 62102330, 62306249) and in part by the Natural Science Foundation of Sichuan Province (Nos. 2022NSFSC0947, 2022NSFSC0945).

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 126–135, 2017.

[2] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In *Proceedings of the European conference on computer vision (ECCV)*, pages 252–268, 2018.

[3] Andrew Aitken, Christian Ledig, Lucas Theis, Jose Caballero, Zehan Wang, and Wenzhe Shi. Checkerboard artifact free sub-pixel convolution: A note on sub-pixel convolution, resize convolution and convolution resize. *arXiv preprint arXiv:1707.02937*, 2017.

[4] Saeed Anwar and Nick Barnes. Densely residual laplacian super-resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3):1192–1204, 2020.

[5] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. 2012.

[6] X Chen, X Wang, J Zhou, and C Dong. Activating more pixels in image super-resolution transformer. arxiv 2022. *arXiv preprint arXiv:2205.04437*.

[7] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *International conference on machine learning*, pages 933–941. PMLR, 2017.

[8] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part IV 13*, pages 184–199. Springer, 2014.

[9] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 391–407. Springer, 2016.

[10] Zongcai Du, Ding Liu, Jie Liu, Jie Tang, Gangshan Wu, and Lean Fu. Fast and memory-efficient network towards efficient image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 853–862, 2022.

[11] G Gao, Z Wang, J Li, W Li, Y Yu, and T Zeng. Lightweight bimodal network for single-image super-resolution via symmetric cnn and recursive transformer. arxiv 2022. *arXiv preprint arXiv:2204.13286*.

[12] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[13] Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc Le. Transformer quality in linear time. In *International Conference on Machine Learning*, pages 9099–9117. PMLR, 2022.

[14] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5197–5206, 2015.

[15] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1637–1645, 2016.

[16] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016.

[17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[18] Zheyuan Li, Yingqi Liu, Xiangyu Chen, Haoming Cai, Jinjin Gu, Yu Qiao, and Chao Dong. Blueprint separable residual network for efficient image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 833–843, 2022.

[19] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1833–1844, 2021.

[20] Jingyun Liang, Jiezhang Cao, Yuchen Fan, Kai Zhang, Rakesh Ranjan, Yawei Li, Radu Timofte, and Luc Van Gool. Vrt: A video restoration transformer. *arXiv preprint arXiv:2201.12288*, 2022.

[21] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[22] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423. IEEE, 2001.

[23] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[24] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[25] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.

[26] Ying Tai, Jian Yang, and Xiaoming Liu. Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3147–3155, 2017.

[27] Tong Tong, Gen Li, Xiejie Liu, and Qinquan Gao. Image super-resolution using dense skip connections. In *Proceedings of the IEEE international conference on computer vision*, pages 4799–4807, 2017.

[28] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxim: Multi-axis mlp for image processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5769–5780, 2022.

[29] Longguang Wang, Xiaoyu Dong, Yingqian Wang, Xinyi Ying, Zaiping Lin, Wei An, and Yulan Guo. Exploring sparsity in image super-resolution for efficient inference. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4917–4926, 2021.

[30] Junjie Yan, Ruosi Wan, Xiangyu Zhang, Wei Zhang, Yichen Wei, and Jian Sun. Towards stabilizing batch statistics in backward propagation of batch normalization. *arXiv preprint arXiv:2001.06838*, 2020.

[31] Jiahui Yu, Yuchen Fan, Jianchao Yang, Ning Xu, Zhaowen Wang, Xinchao Wang, and Thomas Huang. Wide activation for efficient and accurate image super-resolution. *arXiv preprint arXiv:1808.08718*, 2018.

[32] Jiahui Yu, Yuchen Fan, Jianchao Yang, Ning Xu, Zhaowen Wang, Xinchao Wang, and Thomas Huang. Wide activation for efficient and accurate image super-resolution. *arXiv preprint arXiv:1808.08718*, 2018.

[33] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Restormer: Efficient transformer for high-resolution image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5728–5739, 2022.

[34] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *Curves and Surfaces: 7th International Conference, Avignon, France, June 24-30, 2010, Revised Selected Papers 7*, pages 711–730. Springer, 2012.

[35] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3929–3938, 2017.

[36] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Learning a single convolutional super-resolution network for multiple degradations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3262–3271, 2018.

[37] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 286–301, 2018.

[38] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2472–2481, 2018.

[39] Lin Zhou, Haoming Cai, Jinjin Gu, Zheyuan Li, Yingqi Liu, Xiangyu Chen, Yu Qiao, and Chao Dong. Efficient image super-resolution using vast-receptive-field attention. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, pages 256–272. Springer, 2023.