TranMamba: A Lightweight Hybrid Transformer-Mamba Network for Super-Resolution

Long Zhang[1*] and Yi Wan[1†]

[1*]School of Information Science and Engineering, Lanzhou University, 222 S. Tianshui Rd, LanZhou, 730000, GanSu, China.

*Corresponding author(s). E-mail(s): lzhang2019@lzu.edu.cn;
Contributing authors: wanyi@lzu.edu.cn;
†These authors contributed equally to this work.

Abstract

Transformers excel in modeling long-range dependencies for computer vision, but the quadratic complexity of self-attention complicates lightweight model design. The Mamba model, with linear complexity, offers similar capabilities but underperforms compared to Transformers. Inspired by these insights, we propose TranMamba, a lightweight hybrid Transformer-Mamba network that enhances both performance and efficiency in Single Image Super-Resolution (SISR). Specifically, we reduce the computational cost associated with self-attention by alternating between Transformer and Mamba modules. To balance the extraction of both local and global information, we designed Transformer Aggregation Block (TAB) and Mamba Aggregation Block (MAB) to strengthen feature representation. Additionally, we developed a Reparameterized Spatial-Gate Feed-Forward Network (RepSGFN) to further improve the model's feature extraction capabilities. Extensive experiments demonstrate that TranMamba achieves SOTA performance among models of comparable size.

Keywords: single image super-resolution, transformer, mamba, hybrid transformer-mamba network.

1 Introduction

Single-image super-resolution (SISR) focuses on refining and enhancing low-resolution (LR) images to reconstruct high-resolution (HR) images with richer details. Benefiting from the powerful nonlinear mapping capabilities and adaptive optimization abilities of convolutional neural networks (CNNs), CNN-based SR methods are adept at learning complex image data relationships and demonstrate strong robustness [1–5]. This strength has led to the growth of CNN-based SR methods, with researchers continuously proposing novel network architectures and optimization strategies to further improve image reconstruction quality and efficiency [6–14]. While these methods have significantly improved image quality, the fixed receptive field of convolutional kernels limits their ability to capture long-range dependencies. To compensate, more layers are often stacked, which in turn increases computational costs in deep architectures.

To better capture global information within images, researchers have shifted their focus to Transformer models [15–20], which are inherently designed to model long-range dependencies through self-attention mechanisms. To ensure that

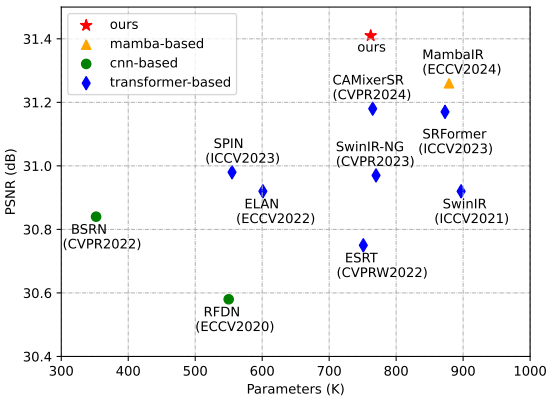


Fig. 1: PSNR vs. Params comparison on the Manga109 ×4 dataset. The proposed TranMamba achieves higher PSNR with fewer parameters compared to other SOTA methods.

both local and global features are effectively captured, recent approaches have explored hybrid models that integrate Transformers with CNN architectures. This combination leverages CNNs' strengths in extracting local features while utilizing Transformers' capability to model long-range dependencies, resulting in a more comprehensive feature representation [21–23]. Despite their significant improvements in image restoration quality, the quadratic complexity of self-attention continues to be a major bottleneck in the design of lightweight models. The Mamba model [24–26], emerging as a compelling alternative to Transformers, effectively models long-range dependencies with linear complexity, which addresses the high computational costs associated with self-attention mechanisms. However, current research shows that its performance still falls short of Transformers [27, 28]. Thus, further exploration and optimization of the Mamba architecture are needed.

Building on these insights, this paper proposes TranMamba, a lightweight hybrid Transformer-Mamba architecture for SISR. TranMamba effectively balances model efficiency with the ability to capture both local and global information for image reconstruction, which is primarily attributed to three key design innovations. First, to balance global information extraction with computational efficiency, TranMamba employs an alternating strategy of Transformer and Mamba modules, which proves effective in natural language processing (NLP) tasks while simultaneously reducing the computational complexity associated with self-attention and maintaining performance [27]. Second, inspired by the adaptive interaction module [23], we designed the Transformer Aggregation Block (TAB) and Mamba Aggregation Block (MAB) to enhance the local feature extraction capabilities of TranMamba. Specifically, while utilizing the self-attention module and state space module to capture long-range dependencies, we introduced a reparameterized convolution block (RepConv) to focus on extracting local features [29]. Finally, we designed the Reparameterized Spatial-Gate Feed-Forward Network (RepSGFN), based on SGFN, to enhance the model's feature representation capability. With these combined innovations, the proposed TranMamba achieves SOTA performance, as shown in Figure 1. The contributions of this paper are summarized as follows:

- We propose TranMamba, a lightweight hybrid Transformer-Mamba architecture for SISR. TranMamba reduces the computational complexity associated with self-attention by alternating between Transformer and Mamba modules without compromising performance.
- We design a Hybrid Transformer-Mamba Block (HTMB), consisting of the Transformer Aggregation Block (TAB) and the Mamba Aggregation Block (MAB), to strengthen TranMamba's local feature extraction capabilities. Additionally, we develop RepSGFN by introducing RepConv, enhancing the model's feature representation capacity.
- We conduct extensive experiments that demonstrate TranMamba's SOTA performance in SISR tasks while maintaining an efficient model architecture.

2 Related Work

2.1 Lightweight Super-Resolution Models

With the advancement of deep learning, CNN-based SR methods have progressively become mainstream. SRCNN [1], as the first model to apply CNNs to SISR, demonstrated that even

a simple three-layer convolutional network could surpass traditional methods in image generation quality, thereby laying the foundation for subsequent applications of CNNs in SR. To achieve higher image quality, numerous studies have focused on deepening network architectures. However, excessively deep networks often result in poor inference speed and high memory consumption [6, 8, 14]. To reduce model complexity, FSRCNN [2] optimized the network architecture based on SRCNN, minimizing computational cost while maintaining image reconstruction quality. ESPCN [3] reduces the computational burden of processing high-resolution images by placing a sub-pixel convolution layer at the network's end for upscaling image features. This approach significantly enhances both the accuracy and efficiency of SR tasks, making it a favored architecture for later lightweight models. Influenced by the ResNet [30] framework, many super-resolution models utilizing residual or dense residual connections have achieved significant enhancements in image quality [10, 31–33]. Furthermore, the attention mechanism's strong ability to capture essential features has led to the integration of attention modules in numerous studies, further boosting image restoration performance [9, 11–13, 34, 35].

As a result of the fixed size of convolutional kernels, traditional CNNs often struggle to achieve a global receptive field. In contrast, the remarkable performance of Transformer models in NLP has inspired new approaches in computer vision [15–17]. This success is largely attributed to the self-attention mechanism in Transformers, which overcomes the limitations of local receptive fields, enabling the effective capture of global information and long-range dependencies. In image restoration tasks, such as super-resolution and denoising, Transformers have also demonstrated their strengths. The self-attention mechanism allows Transformers to overcome the limitations of traditional methods, effectively capturing global information, resulting in more precise reconstruction of high-resolution images in these tasks [20, 21, 23, 36–39]. SwinIR [18] overcomes the limitations of fixed windows by incorporating a shifted window mechanism [17], effectively leveraging the advantages of the Transformer. This approach significantly enhances performance in image restoration tasks without substantially increasing computational costs. Lu *et al.* [22] designed a lightweight hybrid model architecture, ESRT, by combining the advantages of a lightweight CNN and Transformer network. This approach significantly reduces computational costs and GPU memory usage. DAT [23] enhances feature representation by using an adaptive interaction module to fuse local and global features, and alternating between spatial and channel attention modules. This approach achieves state-of-the-art performance in super-resolution tasks. NGSwin [19] integrates N-Gram [40] context into the Swin Transformer, enhancing the model's ability to capture relationships between neighboring windows. This improvement strengthens SwinIR's capability to capture global information and effectively reconstruct image details. CAMixerSR [21] introduces a content-aware model that combines the strengths of convolutional processing for simple contexts with deformable window attention for handling sparse textures, enabling dynamic feature extraction. This approach achieves outstanding performance in super-resolution tasks, significantly enhancing the model's capability to improve image resolution.

2.2 State Space Models

The State Space Model (SSM) is a mathematical model used to describe dynamic systems, and it is widely applied in fields such as control theory and signal processing. Although SSMs have strong capabilities in memory and capturing long-range dependencies in time series data, their application in deep learning is constrained by certain theoretical and computational complexities [41–44]. Linear State-Space Layer (LSSL) provide a unified framework that combines the strengths of Recurrent Neural Networks (RNNs), CNNs, and Continuous-Time Models (CTMs), enabling the potential application of SSMs in deep learning by treating these models as special cases of LSSL [41, 45]. The Structured State Space Sequence Model (S4) reparameterizes the structured state matrix by decomposing it into low-rank and normal components. This decomposition reduces the computational and memory demands of LSSL and resolves its numerical instability,

making the application of SSMs in deep learning practically feasible [45]. To achieve efficient parallel computation with SSMs, S5 [46] introduces a multi-input, multi-output SSM and leverages efficient parallel scan operations, resulting in exceptional performance on long-sequence modeling tasks. To address the inefficiency of Transformers in handling long-sequence tasks, Gu *et al.* [24, 25] designed the Mamba model based on S4 and other SSMs. Mamba achieves efficient computation and addresses long-range dependency issues by incorporating a selection mechanism and hardware-aware parallel algorithms. To adapt the Mamba model for visual tasks, recent research has focused on visual state space models, achieving significant progress [26, 28, 47, 48]. Although Mamba handles long-range dependencies effectively, its performance still lags behind that of Transformer models. However, its superior computational efficiency has made the exploration of hybrid models combining Transformer and Mamba a prominent area of research [27, 28].

3 The Method

In this section, we discuss the detailed technical aspects of the proposed TranMamba. First, we describe the overall architecture of the network in Section 3.1. Next, we introduce the HTMB in Section 3.2. Then, we discuss the integration of the Vision State Space Module (VSSM) in Section 3.3. Finally, we present the structure of the RepSGFN in Section 3.4.

3.1 Network Architecture

The TranMamba illustrated in Fig. 2(a) aims to learn the end-to-end mapping function $\mathcal{F}(\cdot)$ that upscales $I_{LR} \in \mathbb{R}^{H\times W\times 3}$ images into $\hat{I}_{HR} \in \mathbb{R}^{H\times W\times 3}$ images:

$$\hat{I}_{HR} = \mathcal{F}(I_{LR}; \theta), \tag{1}$$

where θ denotes the learnable parameters. The overall structure of the proposed TranMamba is composed of three parts: the expanding layer, the feature extraction layer, and the image reconstruction layer.

Expanding layer

The expanding layer is primarily used to enhance the model's representational capacity by increasing the channel dimensions of the input image, allowing it to capture more diverse feature information and providing richer input data for subsequent feature extraction:

$$F_s = Conv_3\left(I_{LR}\right), \tag{2}$$

where, $Conv_3\left(\cdot\right)$ denotes a 3×3 convolutional kernel, while F_s represents the shallow features extracted through the expanding layer.

Feature extraction layer

To extract more complex and contextually meaningful features from the input shallow features F_s, they are passed through the feature extraction layer to generate deep features F_d:

$$\begin{aligned} F_d &= \mathcal{F}_{ext}\left(F_s\right) + F_s, \\ \mathcal{F}_{ext}\left(\cdot\right) &= ConvB\left(LN\left(\mathcal{F}^i_{HTMB}\left(\cdot\right)\right)\right), \quad i = 1, \cdots, n, \\ ConvB\left(\cdot\right) &= Conv_3\left(\sigma_r\left(Conv_1\left(\sigma_r\left(Conv_3\left(\cdot\right)\right)\right)\right)\right), \end{aligned} \tag{3}$$

As demonstrated in the above equation, the feature extraction layer is composed of a feature extraction function $\mathcal{F}_{ext}\left(\cdot\right)$ and a residual connection. Where, $\mathcal{F}^i_{HTMB}\left(\cdot\right)$ denotes the i-th HTMB, $LN\left(\cdot\right)$ represents LayerNorm [49], $\sigma_r\left(\cdot\right)$ represents the ReLU activation function [50], and $ConvB$ represents a convolution block containing three convolution layers and two activation functions.

Reconstruction layer

Finally, we scale up the feature maps F_d in the image reconstruction layer to restore high-resolution images:

$$\begin{aligned} \hat{I}_{HR} &= \mathcal{F}_{rec}\left(F_d\right), \\ \mathcal{F}_{rec}\left(\cdot\right) &= UP_s\left(Conv_3\left(\cdot\right)\right), \end{aligned} \tag{4}$$

where $UP_s\left(\cdot\right)$ denotes the sub-pixel convolution operation [3], with s being the upscale factor. In the training process, we adopt the L_1 loss [51] as the cost function. The optimization objective can

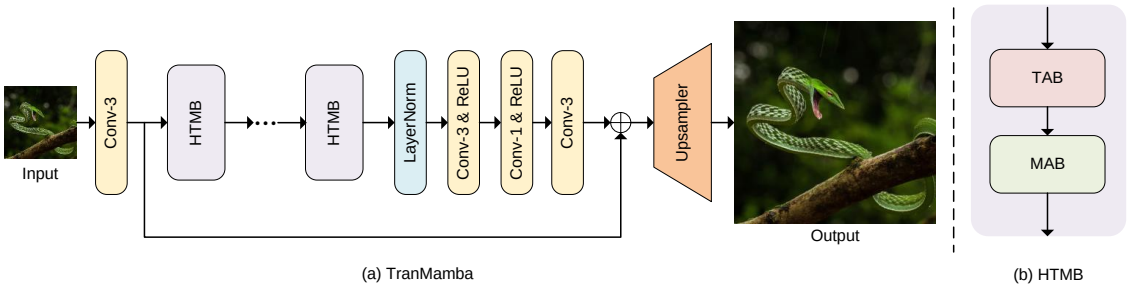


Fig. 2: The architecture of Hybrid Transformer-Mamba Network (TranMamba).

be formulated as:

$$\arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \left| \mathcal{F}\left(I_i^{LR}; \theta\right) - I_i^{HR} \right|, \tag{5}$$

where N is the batch size, I_i^{LR} and I_i^{HR} are the low-resolution and high-resolution images respectively.

3.2 Hybrid Transformer-Mamba Block

Although the performance of Mamba [24] is slightly lower than that of the Transformer module [27], it maintains exceptionally high efficiency and effectively handles long-range dependencies. Given the quadratic computational complexity of self-attention in Transformers, which impacts model efficiency, we designed an efficient MAB based on the VSSM [26, 48]. This block is used alternately with the TAB, forming the HTMB. This design allows us to benefit from Mamba's efficiency and long-distance relationship handling, while also harnessing the Transformer's powerful feature extraction capabilities. As shown in Fig. 2(b), the structure of the HTMB can be formally defined as follows:

$$\mathcal{F}_{HTMB}\left(\cdot\right) = \mathcal{F}_{MAB}\left(\mathcal{F}_{TAB}\left(\cdot\right)\right), \tag{6}$$

where $\mathcal{F}_{TAB}\left(\cdot\right)$ represents the Transformer Aggregation Block and $\mathcal{F}_{MAB}\left(\cdot\right)$ represents the Mamba Aggregation Block. As shown in Fig. 3(a) and Fig. 3(b), respectively, both TAB and MAB employ the MetaFormer architecture [52]. For convenience, we unify these two types of blocks under a common notation, $\mathcal{F}_{AB}\left(\cdot\right)$, which can be formulated as:

$$\begin{aligned} F_h &= \mathcal{F}_{TM}\left(LN\left(F_{in}\right)\right) + F_{in}, \\ \mathcal{F}_{AB}\left(F_{in}\right) &= RepSGFN\left(LN\left(F_h\right)\right) + F_h, \end{aligned} \tag{7}$$

where F_{in} and F_h respectively represent the input features and hidden layer features. $\mathcal{F}_{TM}$ represents the token mixer [52, 53], and RepSGFN stands for the reparameterized spatial-gated feed-forward network. For detailed information on RepSGFN, see Subsection 3.4.

To effectively capture both long-range dependencies and local features, inspired by the DAT [23], we employ a dual-branch architecture in our token mixer $\mathcal{F}_{TM}$. This architecture consists of a global branch $\mathcal{F}_g\left(\cdot\right)$ and a local branch $\mathcal{F}_l\left(\cdot\right)$, which work together to extract comprehensive features. The structure of $\mathcal{F}_{TM}$ can be formally defined as follows:

$$\mathcal{F}_{TM}\left(\cdot\right) = \mathcal{F}_g\left(\cdot\right) \odot CA\left(\mathcal{F}_l\right) + \mathcal{F}_l\left(\cdot\right) \odot SA\left(\mathcal{F}_g\right), \tag{8}$$

where $SA\left(\cdot\right)$ and $CA\left(\cdot\right)$ represent the spatial attention module and the channel attention module [23], respectively. $\mathcal{F}_l$ is the reparameterized convolution block [29, 54, 55], as shown in Fig. 3(d). The global branch $\mathcal{F}_g$ has two different forms:

$$\begin{aligned} \mathcal{F}_g\left(\cdot\right) &= \mathcal{F}_{Swin}\left(\cdot\right) \quad \text{in TAB}, \\ \mathcal{F}_g\left(\cdot\right) &= \mathcal{F}_{VSSM}\left(\cdot\right) \quad \text{in MAB}, \end{aligned} \tag{9}$$

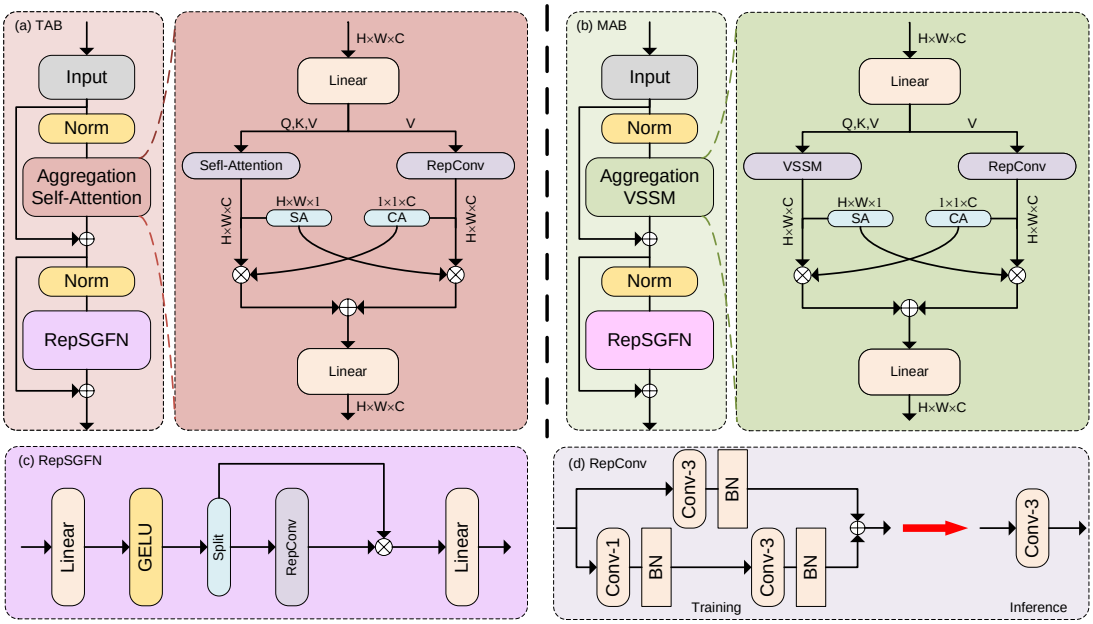


Fig. 3: (a) The architecture of Transformer Aggregation Block (TAB). (b) The architecture of Mamba Aggregation Block (MAB). (c) The architecture of Reparameterized Spatial-Gate Feed-Forward Network (RepSGFN). (d) Illustration of Reparameterized Convolution Block (RepConv).

where $\mathcal{F}_{VSSM}$ represents the vision state space module, the details of which can be found in Subsection 3.3. $\mathcal{F}_{Swin}$ represents the self-attention module in the Swin Transformer [17]. The self-attention is calculated as follows:

$$\begin{aligned} Q, K, V &= P\left(F_{in}\right), \\ \mathcal{F}_{Swin}\left(F_{in}\right) &= \mathcal{S}\left(\frac{Q \otimes K^{T}}{\sqrt{d}} + B\right) \odot M \otimes V, \end{aligned} \tag{10}$$

where Q, K, and V are the query, key, and value matrices, d is the feature dimension, M is the mask matrix, B is the bias matrix, $P\left(\cdot\right)$ is the linear projection layer, and $s\left(\cdot\right)$ denotes the softmax layer, $\otimes$ represents the matrix multiplication, and $\odot$ represents the Hadamard product (element-wise multiplication).

3.3 Vision State Space Module

Theories

A SSM is a mathematical framework used to represent the dynamic behavior of a system through state variables. It typically consists of two main equations: the state equation and the observation equation. These equations can be mathematically expressed as follows:

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t, \\ \mathbf{y}_t &= \mathbf{C}\mathbf{x}_t + \mathbf{D}\mathbf{u}_t, \end{aligned} \tag{11}$$

In sequence processing tasks, the direct transmission matrix $\mathbf{D}$ is often not explicitly included and are instead omitted (in the Mamba model, $\mathbf{D}\mathbf{u}_t$ is typically handled as a skip connection), the simplified form of the SSM is given by:

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t, \\ \mathbf{y}_t &= \mathbf{C}\mathbf{x}_t, \end{aligned} \tag{12}$$

To apply SSMs to the processing and analysis of sampled data, discretization methods such as bilinear transformation or zero-order hold (ZOH) are commonly used. Here, we present the system representation after discretization using the ZOH

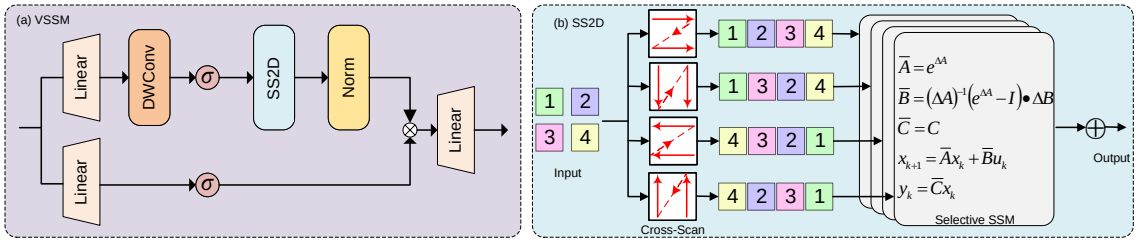


Fig. 4: (a) The architecture of Vision State Space Module (VSSM). (b) Illustration of 2D-Selective-Scan (SS2D).

method:

$$\begin{aligned} \mathbf{x}_{k+1} &= \bar{\mathbf{A}}\mathbf{x}_k + \bar{\mathbf{B}}\mathbf{u}_k, \\ \mathbf{y}_k &= \bar{\mathbf{C}}\mathbf{x}_k, \end{aligned} \tag{13}$$

where the discretized matrices $\bar{\mathbf{A}}$ and $\bar{\mathbf{B}}$ are computed as follows:

$$\bar{\mathbf{A}} = e^{\Delta\mathbf{A}}, \tag{14}$$

$$\bar{\mathbf{B}} = (\Delta\mathbf{A})^{-1}\left(e^{\Delta\mathbf{A}} - \mathbf{I}\right) \cdot \Delta\mathbf{B}, \tag{15}$$

$$\bar{\mathbf{C}} = \mathbf{C}, \tag{16}$$

where, k denotes the length of the sequence, Δ represents the sampling interval.

Based on the recursive relation of the state space model, the output $\mathbf{y}_k$ can be expressed in a convolutional form. Using convolution, the output is given by:

$$\begin{aligned} \mathbf{y}_k &= \sum_{i=0}^{k-1} \bar{\mathbf{C}}\bar{\mathbf{A}}^{k-i-1}\bar{\mathbf{B}}\mathbf{u}_i, \\ \mathbf{y} &= \bar{\mathbf{K}} * \mathbf{u}, \end{aligned} \tag{17}$$

where the symbol $*$ indicates the convolution operation, the SSM convolution kernel $\bar{\mathbf{K}}$ is expressed as:

$$\bar{\mathbf{K}} = (\bar{\mathbf{C}}\bar{\mathbf{B}}, \bar{\mathbf{C}}\bar{\mathbf{A}}\bar{\mathbf{B}}, \ldots, \bar{\mathbf{C}}\bar{\mathbf{A}}^{k-1}\bar{\mathbf{B}}), \tag{18}$$

Vision State Space Module

VSSM inherits the network structure characteristics of Mamba, serving as a multi-layer perceptron (MLP) block [56] with a gating mechanism [57–59]. As illustrated in Fig. 4(a), the architecture of the $\mathcal{F}_{VSSM}$ comprises two branches, denoted as $\mathcal{F}_1(\cdot)$ and $\mathcal{F}_2(\cdot)$, detailed as follows:

$$\begin{aligned} \mathcal{F}_{VSSM}(\cdot) &= P\left(\mathcal{F}_1(\cdot) \odot \mathcal{F}_2(\cdot)\right), \\ \mathcal{F}_1(\cdot) &= LN\left(SSM\left(\sigma_s\left(DWConv\left(Proj\left(\cdot\right)\right)\right)\right)\right), \\ \mathcal{F}_2(\cdot) &= \sigma_s\left(P\left(\cdot\right)\right), \end{aligned} \tag{19}$$

where, $DWConv(\cdot)$ represents a depthwise convolution layer [60] with a 3×3 kernel; $SSM(\cdot)$ represents the 2D-Selective-Scan module (SS2D), for details, see the next subsection; $\sigma_s(\cdot)$ represents the SiLU (Swish) activation function [59, 61].

Since SSMs are originally designed for NLP tasks, many works [26, 28, 47, 48] have adapted the input features F_{in} for visual tasks by flattening the 2D feature maps into 1D sequences before feeding them into VSSM. However, this flattening process diminishes the model's ability to capture spatial structure information. To address this limitation, VMamba [48] employs a cross-scan strategy in the proposed 2D-Selective-Scan (SS2D) to enhance Mamba's performance on visual tasks. We adopted this approach in our model design.

2D-Selective-Scan

As previously discussed, the flattening process disrupts the spatial relationships by transforming adjacent pixels into non-adjacent ones, thereby compromising the spatial structure of the image data [26, 48]. To address the potential loss of spatial structural information caused by converting 2D features into 1D sequences, SS2D [48] employs a cross-scanning strategy to preserve as much spatial information as possible. Specifically, as illustrated in Figure 4(b), SS2D arranges pixels (or

patches) in both forward and reverse orders along four directions: up, down, left, and right, resulting in four sequences with different orders. Each sequence is then processed according to the selective state space model (s6) [24], and the outputs are merged to obtain the final result.

3.4 Reparameterized Spatial-Gate Feed-Forward Network

The Feed-Forward Network (FFN) [15], a crucial component of the Transformer model, enhances the model's representation ability by addressing the limited nonlinear fitting capacity of self-attention or other token mixer approaches [62]. The Spatial-Gate Feed-Forward Network (SGFN) [23] further improves this by introducing a gating mechanism that compresses channel redundancy while selectively focusing on different spatial positions, functioning similarly to an attention mechanism [24]. To enhance spatial information extraction and improve the model's ability to capture local features, we incorporate a reparameterized convolution block into SGFN. This addition provides a more flexible network structure, thereby increasing the model's capacity to process spatial information effectively and better emphasize important local features. The detailed process is as follows:

$$\begin{aligned} F_h &= \sigma_g\left(P\left(F_{in}\right)\right), \\ F_{h1}, F_{h2} &= Split\left(F_h\right), \\ RepSGFN\left(F_{in}\right) &= P\left(F_{h1} \odot \mathcal{F}_{RepConv}\left(F_{h2}\right)\right), \end{aligned} \tag{20}$$

where $\sigma_g\left(\cdot\right)$ represents the GeLU activation function; $Split\left(\cdot\right)$ denotes the operation that divides the hidden feature $F_h \in \mathbb{R}^{H\times W\times C}$ into two equally-sized components, $F_{h1} \in \mathbb{R}^{H\times W\times \frac{C}{2}}$ and $F_{h2} \in \mathbb{R}^{H\times W\times \frac{C}{2}}$; $\mathcal{F}_{RepConv}\left(\cdot\right)$ signifies the reparameterized convolution operation applied to F_{h2}.

As shown in Figure 3(d), the reparameterized convolution block we designed consists of two branches. During inference, these parameters are merged into a single convolution operation, thereby reducing computational cost while enhancing the model's representational capacity. The merging process involves three transformations: batch normalization fusion, sequential convolutions fusion, and multi-branch fusion.

Batch normalization fusion embeds batch normalization parameters into the convolutional layer. This can be formulated as:

$$\begin{aligned} BN\left(Conv\left(F_{in}\right)\right) &= \frac{\gamma\mathcal{W}\left(F_{in}\right)}{\sigma} + \left(\frac{\gamma\left(b-\mu\right)}{\sigma} + \beta\right) \\ Conv\left(F_{in}\right) &= \mathcal{W}\left(F_{in}\right) + b \\ BN\left(F_{in}\right) &= \frac{\gamma\left(F_{in}-\mu\right)}{\sigma} + \beta \end{aligned} \tag{21}$$

where, $Conv(\cdot)$ is the convolutional layer, and $BN(\cdot)$ stands for the batch normalization layer. The terms $\mathcal{W}$ and b refer to the convolutional kernel's weights and biases, while μ, σ, γ, and β denote the mean, standard deviation, scaling factor, and shift factor of the batch normalization layer, respectively.

Sequential convolutions fusion involves merging a 1×1 convolution and a $k\times k$ convolution into a single $k\times k$ convolution. The dimensions of the 1×1 convolutional kernel are $K_1 \in \mathbb{R}^{C_h\times C_{in}\times 1\times 1}$, and the dimensions of the $k\times k$ convolutional kernel are $K_k \in \mathbb{R}^{C_{out}\times C_h\times k\times k}$. The input features are denoted as $F_{in} \in \mathbb{R}^{B\times C_{in}\times H\times W}$, and the output features as $F_{out} \in \mathbb{R}^{B\times C_{out}\times H\times W}$. The merging formula is:

$$\begin{aligned} F_{out} &= K_f * F_{in}, \\ K_f &= K_k * T\left(K_1\right), \end{aligned}$$

where $K_f \in \mathbb{R}^{C_{out}\times C_{in}\times k\times k}$ denotes the fused convolutional kernel, and T denotes the transpose operation.

The multi-branch fusion module integrates multiple branches by leveraging convolution additivity for convolutions of the same size. This is expressed as:

$$F * \mathcal{K}_{k1} + F * \mathcal{K}_{k2} = F * \left(\mathcal{K}_{k1} + \mathcal{K}_{k2}\right) \tag{22}$$

where $\mathcal{K}_{k1}$ and $\mathcal{K}_{k2}$ are two convolutional kernels of size $k\times k$.

4 Experiments

4.1 Setup

Datasets and Metrics

Aligned with other supervised SISR methods, we train our model using the DF2K [66, 67] dataset in our experimental setup. This dataset generates

Table 1: Quantitative comparison results of the state-of-the-art methods on public benchmark datasets, with the first and second best results highlighted in **Red** and **Blue** respectively. '−' indicates that the item is not included in the original paper.

Scale	Model	Params[K]	Set5 PSNR↑ / SSIM↑	Set14 PSNR↑ / SSIM↑	BSD100 PSNR↑ / SSIM↑	Urban100 PSNR↑ / SSIM↑	Manga109 PSNR↑ / SSIM↑
×2	VDSR[6]	666	37.53 / 0.9587	33.03 / 0.9124	31.90 / 0.8960	30.76 / 0.9140	37.22 / 0.9750
	CARN[10]	1592	37.76 / 0.9590	33.52 / 0.9166	32.09 / 0.8978	31.92 / 0.9256	38.36 / 0.9765
	IDN[63]	553	37.83 / 0.9600	33.30 / 0.9148	32.08 / 0.8985	31.27 / 0.9196	– / –
	IMDN[9]	694	38.00 / 0.9605	33.63 / 0.9177	32.19 / 0.8996	32.17 / 0.9283	38.88 / 0.9774
	RFDN[11]	534	38.05 / 0.9606	33.68 / 0.9184	32.16 / 0.8994	32.12 / 0.9278	38.88 / 0.9773
	RLFN[12]	527	38.07 / 0.9607	33.72 / 0.9187	32.22 / 0.9000	32.34 / 0.9299	– / –
	BSRN[13]	332	38.10 / 0.9610	33.74 / 0.9193	32.24 / 0.9006	32.34 / 0.9303	39.14 / 0.9782
	SWinIR-light[18]	878	38.14 / 0.9611	33.86 / 0.9206	32.31 / 0.9012	32.76 / 0.9340	39.12 / 0.9783
	ESRT[22]	677	38.03 / 0.9600	33.75 / 0.9184	32.25 / 0.9001	32.58 / 0.9318	39.12 / 0.9774
	ELAN-light[64]	582	38.17 / 0.9611	33.94 / 0.9207	32.30 / 0.9012	32.76 /0.9340	39.11 / 0.9782
	DAT-light[23]	–	– / –	– / –	– / –	– / –	– / –
	SWinIR-NG[19]	–	– / –	– / –	– / –	– / –	– / –
	SRFormer-light[20]	853	38.23 / 0.9613	33.94 / 0.9209	32.36 / 0.9019	32.91 / 0.9353	39.28 / 0.9785
	CAMixerSR[21]	–	– / –	– / –	– / –	– / –	– / –
	MambaIR[26]	859	38.16 / 0.9610	34.00 / 0.9212	32.34 / 0.9017	32.92 / 0.9356	39.31 / 0.9779
	TranMamba(Ours)	743	38.26 / 0.9619	33.99 / 0.9211	32.37 / 0.9020	33.07 / 0.9362	39.45 / 0.9789
×4	VDSR[6]	666	31.35 / 0.8838	28.01 / 0.7674	27.29 / 0.7251	25.18 / 0.7524	28.83 / 0.8870
	CARN[10]	1592	32.13 / 0.8937	28.60 / 0.7806	27.58 / 0.7349	26.07 / 0.7837	30.47 / 0.9084
	IDN[63]	553	31.82 / 0.8903	28.25 / 0.7730	27.41 / 0.7297	25.41 / 0.7632	– / –
	IMDN[9]	715	32.21 / 0.8948	28.58 / 0.7811	27.56 / 0.7353	26.04 / 0.7838	30.45 / 0.9075
	RFDN[11]	550	32.24 / 0.8952	28.61 / 0.7819	27.57 / 0.7360	26.11 / 0.7858	30.58 / 0.9089
	RLFN[12]	543	32.24 / 0.8952	28.62 / 0.7813	27.60 / 0.7364	26.17 / 0.7877	– / –
	BSRN[13]	352	32.10 / 0.8966	28.73 / 0.7847	27.65 / 0.7387	26.27 / 0.7908	30.84 / 0.9123
	SWinIR-light[18]	897	32.44 / 0.8976	28.77 / 0.7858	27.69 / 0.7406	26.47 / 0.7980	30.92 / 0.9151
	ESRT[22]	751	32.19 / 0.8947	28.69 / 0.7833	27.69 / 0.7379	26.39 / 0.7962	30.75 / 0.9100
	ELAN-light[64]	601	32.43 / 0.8975	28.78 / 0.7858	27.69 / 0.7406	26.54 / 0.7982	30.92 / 0.9150
	DAT-light[23]	573	32.57 / 0.8991	28.87 / 0.7879	27.74 / 0.7428	26.64 / 0.8033	31.37 / 0.9178
	SWinIR-NG[19]	770	32.44 / 0.8978	28.80 / 0.7863	27.70 / 0.7407	26.47 / 0.7977	30.97 / 0.9147
	SRFormer-light[20]	873	32.51 / 0.8988	28.82 / 0.7872	27.73 / 0.7422	26.67 / 0.8032	31.17 / 0.9165
	CAMixerSR[21]	765	32.51 / 0.8988	28.82 / 0.7870	27.72 / 0.7416	26.63 / 0.8012	31.18 / 0.9166
	MambaIR[26]	879	32.51 / 0.8993	28.85 / 0.7876	27.75 / 0.7423	26.75 / 0.8051	31.26 / 0.9175
	TranMamba(Ours)	762	32.54 / 0.9002	28.89 / 0.7889	27.79 / 0.7441	26.77 / 0.8066	31.43 / 0.9185

low-resolution images through bicubic downscaling and contains a total of 3,550 images, with 3,450 used for training and 100 reserved for validation. We evaluate our model's performance using peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) [68], comparing it to other models across five standard benchmark datasets: Set5[69], Set14 [70], BSD100 [71], Urban100 [72], and Manga109 [73].

Implementation details

The proposed TranMamba consists of 7 HTMB modules, with both input and output feature dimensions set to 64. All convolutions in the RepConv module are depthwise convolutions, and the dimension expansion factor for the linear projection layers in both VSSM and RepSGFN is 2. We randomly crop images to 192×192 for input into the model, using a mini-batch size of 8. Data augmentation techniques, such as horizontal flipping and random rotations at 90°, 180°, and 270°, are applied to enhance model performance. The model is trained using the Adam optimizer with an initial learning rate of 1×10^{-3}, β parameters set to $(0.9, 0.99)$, and a total of 2×10^6 iterations. The learning rate is dynamically adjusted using the Cosine Annealing scheduler, with periods of 2,000,000 and an $\eta_{\min}$ of 1×10^{-7}. The training process is implemented in the PyTorch framework on an NVIDIA GeForce RTX 3060 GPU.

4.2 Benchmark Results

As presented in Table 1, we conducted a comparative analysis of several state-of-the-art lightweight super-resolution (SR) models, including CAMixerSR, MambaIR, and others. This analysis covers both model parameters and performance metrics for 2× and 4× SR tasks, based on test results from five benchmark datasets. The results reveal that the TranMamba model demonstrates notable

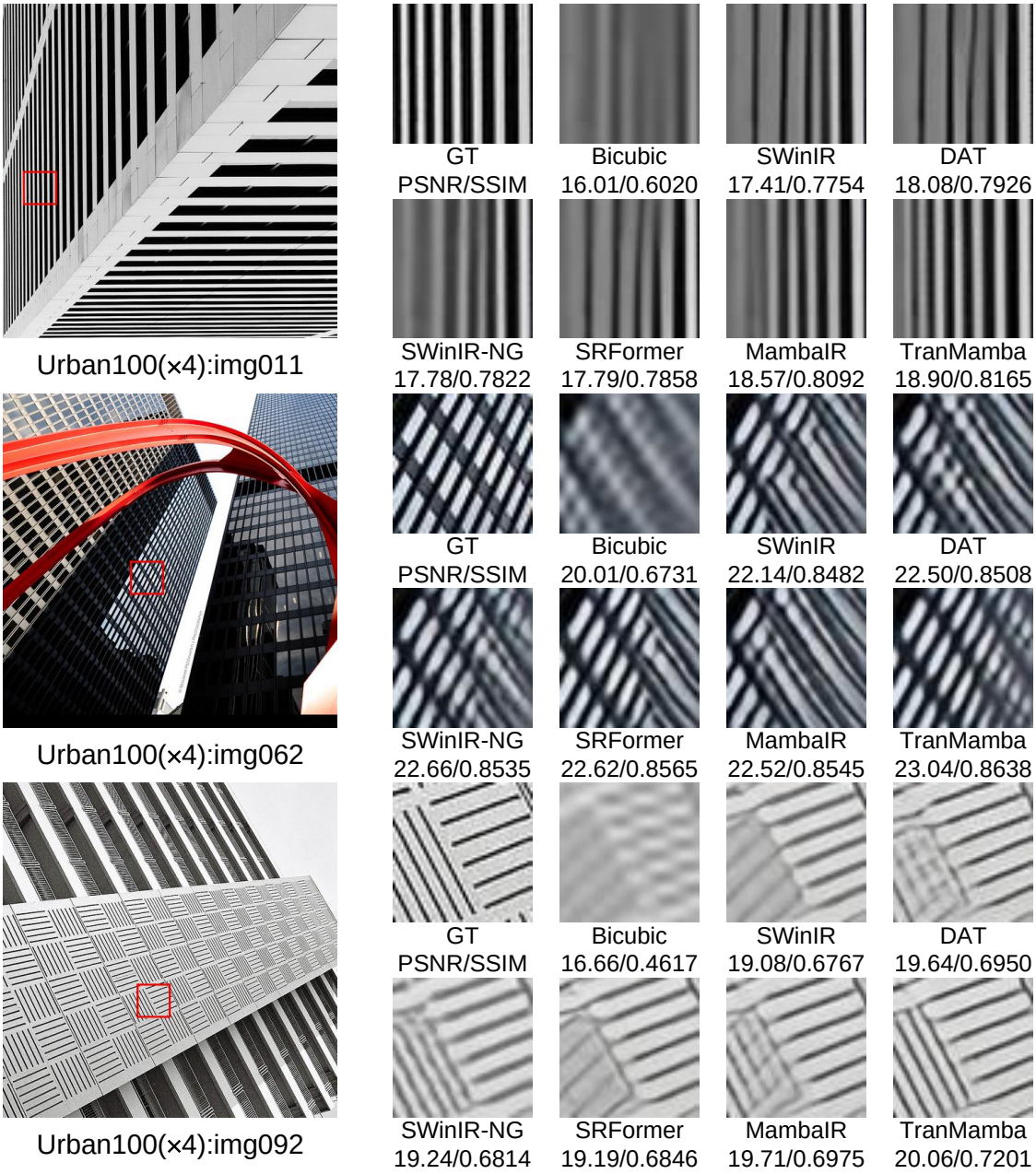
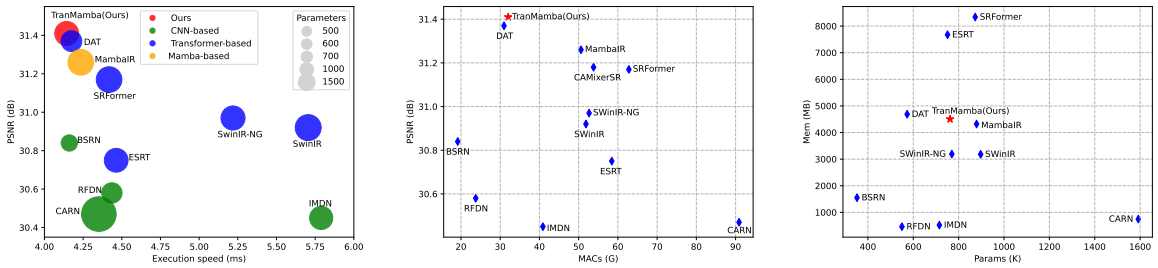


Fig. 5: Qualitative comparison with state-of-the-art SR(×4) methods on Urban100.

advantages in terms of model efficiency and multi-scale SR capabilities compared to other models. For instance, TranMamba outperforms MambaIR in SR performance while retaining a smaller model size. Although TranMamba shows clear performance benefits over the DAT model, it does not offer a significant advantage in model size, indicating that the network structure still has room for further optimization. Figure 5 illustrates TranMamba's visual SR results on the Urban100 dataset, emphasizing its strengths in restoring image details and textures. Overall, TranMamba's



(a) (b) (c)

Fig. 6: Comparative analysis of model performance and complexity. (a) Average execution time comparison across different models. (b) Comparison of MACs among various models. (c) Comparison of memory consumption among different models. The running times, MACs, and memory consumption are calculated based on the Manga109 dataset (GT image size: 816×1164).

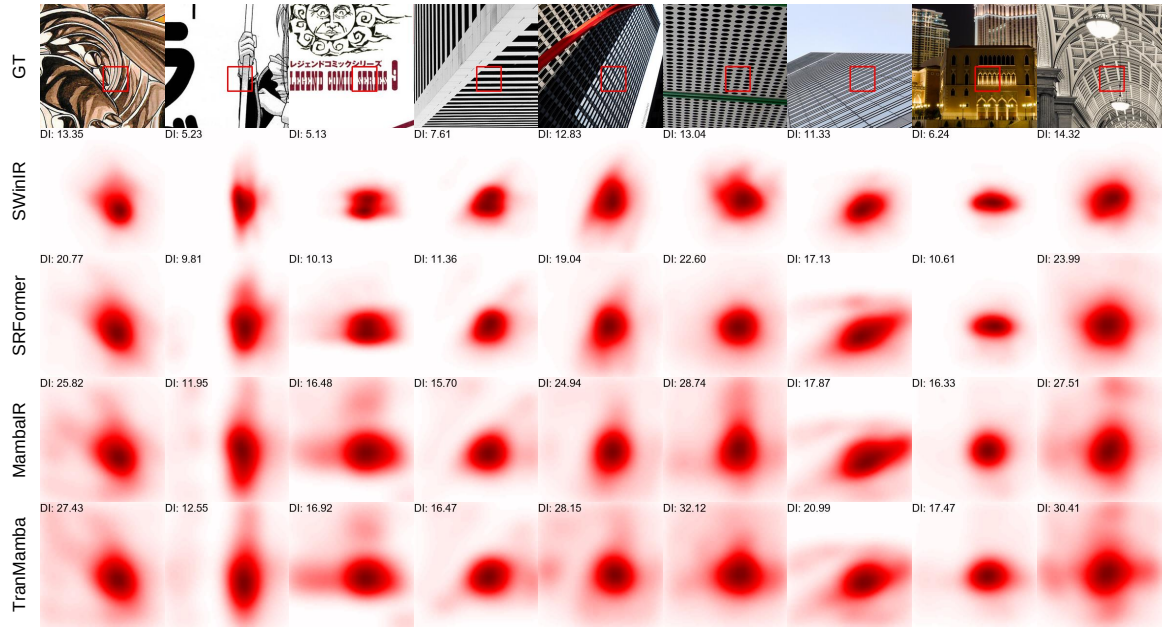


Fig. 7: The heat maps show pixel contributions to the red box in the GT images for different SR networks. DI stands for Diffusion Index, with a higher DI indicating more contributing pixels [65].

exceptional performance in PSNR and parameter metrics underscores its effectiveness in enhancing image quality and highlights its competitive edge among lightweight SR solutions.

Comparison on Model Complexity

We assessed the complexity of various models by comparing their parameters, running time, and multiply-accumulate operations (MACs). To ensure a fair evaluation, we included only models that offer pre-trained versions and can be reliably executed. For the execution time comparison, we ran each model 10 times on the Manga109 dataset and calculated the average. Figure 6a shows the

Table 2: Ablation study on the impact of hybrid Transformer-Mamba architecture on model performance, GPU memory during training, and parameter count. All models were trained from scratch with identical hyperparameters.

Attention	Mamba	Mem[M]	Params[K]	DIV2K100 PSNR/SSIM
	✓	5609	1069	30.6038/0.8412
✓		6043	459	30.6137/0.8419
✓	✓	5805	763	30.6798/0.8441

comparison of running times across different models, with the size of the circles indicating the number of parameters (for a detailed comparison of parameters, see Fig. 1). The results demonstrates that the proposed TranMamba model achieves a high PSNR while maintaining superior operational efficiency. Compared to other Transformer-based and Mamba-based models, TranMamba has fewer parameters, highlighting its advantages in lightweight SR tasks. Furthermore, as illustrated in Fig. 6b, TranMamba's computational complexity is comparable to that of CNN-based SR models but surpasses Transformer-based models in SR quality, achieving SOTA performance. This highlights our approach's effectiveness in balancing computational efficiency with superior performance. As shown in Fig. 6c, TranMamba demonstrates enhanced performance in certain aspects but also exhibits slightly higher memory consumption compared to other models. This suggests that further optimization could help reduce its memory requirements.

Attribution analysis

We conducted an attribution analysis using Local Attribution Maps (LAM)[65] on TranMamba, SWinIR, SRFormer, and MambaIR. As shown in Fig. 7, TranMamba's heatmaps reveal that its red regions encompass a larger area compared to the other models. This observation is supported by TranMamba's higher diffusion index (DI), indicating a greater number of contributing pixels. Together, the extensive heatmap coverage and higher DI highlight that TranMamba effectively leverages a wider range of pixel information, resulting in superior SR performance.

Table 3: Comparative analysis of reparameterized convolution block on model performance.

RepConv	Conv	BSD100 PSNR/SSIM	Urban100 PSNR/SSIM	Manga109 PSNR/SSIM
	✓	27.78/0.7435	26.73/0.8054	31.39/0.9179
✓		27.79/0.7441	26.77/0.8066	31.43/0.9185

Table 4: Comparative analysis of structural reparameterization on model computational complexity. 'Acts' denotes the total number of elements in all 'Conv2d' outputs, while 'Mem' represents memory consumption.

RepConv	Mem[M]	Params[K]	FLOPs[G]	Manga109 PSNR/SSIM
w/o Rep	7809.22	807.6	84.23	31.43 / 0.9185
w Rep	4505.23	762.4	79.83	31.43 / 0.9185

4.3 Ablation Study

Effectiveness of hybrid transformer-mamba architecture

Table. 2 illustrates the impact of different model architectures on performance, parameter count, and GPU memory usage during 4× SR tasks. The pure Mamba architecture delivers inferior performance in PSNR and parameter count (primarily in the VSSM mapping layers), though it exhibits relatively low memory usage, reflecting an advantage in execution efficiency. In contrast, the attention-only model achieves better performance and has a lower parameter count, but suffers from significantly higher memory usage. This increase is due to the quadratic computational complexity and memory demands of self-attention. Lastly, the hybrid Transformer-Mamba architecture improves PSNR further while keeping memory usage and parameter count intermediate between the other two models. This suggests that the hybrid approach effectively balances enhanced super-resolution performance with relatively low memory usage and parameter count.

Effectiveness of reparameterized convolution block

In TranMamba, the RepConv block is incorporated into both the Aggregation Block and the

RepSGFN. This block employs a multi-branch structure to enhance the model's ability to capture spatial features during training, and it consolidates all parameters into a single convolutional layer during inference to ensure high computational efficiency. As shown in Fig. 3, a comparison between the RepConv block and standard convolutional layers indicates that the RepConv block improves performance without adding extra computational load. Figure 4 illustrates the comparison of model complexity before and after reparameterization, demonstrating that the reparameterization technique significantly reduces computational complexity while maintaining image restoration quality.

5 Conclusion

This study introduces TranMamba, a lightweight hybrid Transformer-Mamba architecture for SISR. By alternating between Transformer and Mamba modules, TranMamba effectively reduces the computational complexity associated with self-attention mechanisms. Additionally, we designed TAB and MAB to balance the extraction of global and local information, and further enhanced the model's feature extraction capabilities with the RepSGFN. Extensive experiments demonstrate that TranMamba achieves SOTA performance in SISR tasks while maintaining a lightweight structure. Although TranMamba mitigates some of the complexity associated with Transformer modules through its hybrid architecture, this design may still face complexity trade-offs in certain scenarios. Future research could explore further simplification of the model structure.

Data availability. The code and data used in this study are publicly available on GitHub at the following repository: https://github.com/zl11250422/Tran-Mamba. The repository contains all scripts and datasets necessary to replicate the findings of this study.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Dong, C., Loy, C.C., He, K., Tang, X.: Image super-resolution using deep convolutional networks. IEEE Transactions on Pattern Analysis and Machine Intelligence **38**(2), 295–307 (2016) https://doi.org/10.1109/TPAMI.2015.2439281

[2] Dong, C., Loy, C.C., Tang, X.: Accelerating the super-resolution convolutional neural network. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016, pp. 391–407. Springer, Cham (2016)

[3] Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1874–1883 (2016). https://doi.org/10.1109/CVPR.2016.207

[4] Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z.: Photo-realistic single image super-resolution using a generative adversarial network. IEEE Computer Society (2016)

[5] Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Qiao, Y., Loy, C.C.: Esrgan: Enhanced super-resolution generative adversarial networks. In: The European Conference on Computer Vision Workshops (ECCVW) (2018)

[6] Kim, J., Lee, J.K., Lee, K.M.: Accurate image super-resolution using very deep convolutional networks. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1646–1654 (2016). https://doi.org/10.1109/CVPR.2016.182

[7] Tai, Y., Yang, J., Liu, X., Xu, C.: Memnet: A persistent memory network for image restoration. In: Proceedings of International Conference on Computer Vision (2017)

[8] Lim, B., Son, S., Kim, H., Nah, S., Lee, K.M.: Enhanced deep residual networks for

single image super-resolution. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (2017)

[9] Hui, Z., Gao, X., Yang, Y., Wang, X.: Lightweight image super-resolution with information multi-distillation network. In: Proceedings of the 27th ACM International Conference on Multimedia (ACM MM), pp. 2024–2032 (2019)

[10] Ahn, N., Kang, B., Sohn, K.-a.: Fast, accurate, and, lightweight super-resolution with cascading residual network. ArXiv **abs/1803.08664** (2018)

[11] Liu, J., Tang, J., Wu, G.: Residual feature distillation network for lightweight image super-resolution. In: Bartoli, A., Fusiello, A. (eds.) Computer Vision – ECCV 2020 Workshops, pp. 41–55. Springer, Cham (2020)

[12] Kong, F., Li, M., Liu, S., Liu, D., He, J., Bai, Y., Chen, F., Fu, L.: Residual local feature network for efficient super-resolution. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 765–775 (2022). https://doi.org/10.1109/CVPRW56347.2022.00092

[13] Li, Z., Liu, Y., Chen, X., Cai, H., Gu, J., Qiao, Y., Dong, C.: Blueprint separable residual network for efficient image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pp. 833–843 (2022)

[14] Kim, J., Lee, J.K., Lee, K.M.: Deeply-recursive convolutional network for image super-resolution. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1637–1645 (2016). https://doi.org/10.1109/CVPR.2016.181

[15] Vaswani, A., Shazeer, N.M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Neural Information Processing Systems (2017). https://api.semanticscholar.org/CorpusID:13756489

[16] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. ArXiv **abs/2010.11929** (2020)

[17] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2021)

[18] Liang, J., Cao, J., Sun, G., Zhang, K., Van Gool, L., Timofte, R.: Swinir: Image restoration using swin transformer. In: 2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), pp. 1833–1844 (2021). https://doi.org/10.1109/ICCVW54120.2021.00210

[19] Choi, H., Lee, J.-S., Yang, J.: N-gram in swin transformers for efficient lightweight image super-resolution. 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2071–2081 (2022)

[20] Zhou, Y., Li, Z., Guo, C.-L., Bai, S., Cheng, M.-M., Hou, Q.: Srformer: Permuted self-attention for single image super-resolution. In: 2023 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 12734–12745 (2023). https://doi.org/10.1109/ICCV51070.2023.01174

[21] Wang, Y., Liu, Y., Zhao, S., Li, J., Zhang, L.: CAMixerSR: Only Details Need More "Attention" (2024)

[22] Lu, Z., Li, J., Liu, H., Huang, C., Zhang, L., Zeng, T.: Transformer for single image super-resolution. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 456–465 (2022). https://doi.org/10.1109/CVPRW56347.2022.00061

[23] Chen, Z., Zhang, Y., Gu, J., Kong, L., Yang, X., Yu, F.: Dual aggregation transformer for image super-resolution. In: ICCV (2023)

[24] Gu, A., Dao, T.: Mamba: Linear-time sequence modeling with selective state spaces. arXiv preprint arXiv:2312.00752 (2023)

[25] Dao, T., Gu, A.: Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In: International Conference on Machine Learning (ICML) (2024)

[26] Guo, H., Li, J., Dai, T., Ouyang, Z., Ren, X., Xia, S.-T.: Mambair: A simple baseline for image restoration with state-space model. In: ECCV (2024)

[27] Lieber, O., Lenz, B., Bata, H., Cohen, G., Osin, J., Dalmedigos, I., Safahi, E., Meirom, S.H., Belinkov, Y., Shalev-Shwartz, S., Abend, O., Alon, R., Asida, T., Bergman, A., Glozman, R., Gokhman, M., Manevich, A., Ratner, N., Rozen, N., Shwartz, E., Zusman, M., Shoham, Y.: Jamba: A hybrid transformer-mamba language model. ArXiv **abs/2403.19887** (2024)

[28] Hatamizadeh, A., Kautz, J.: Mambavision: A hybrid mamba-transformer vision backbone. arXiv preprint arXiv:2407.08083 (2024)

[29] Ding, X., Zhang, X., Han, J., Ding, G.: Diverse branch block: Building a convolution as an inception-like unit. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10886–10895 (2021)

[30] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90

[31] Zhang, Y., Tian, Y., Kong, Y., Zhong, B., Fu, Y.: Residual dense network for image super-resolution. In: CVPR (2018)

[32] Tai, Y., Yang, J., Liu, X.: Image super-resolution via deep recursive residual network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)

[33] Tong, T., Li, G., Liu, X., Gao, Q.: Image super-resolution using dense skip connections. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 4809–4817 (2017). https://doi.org/10.1109/ICCV.2017.514

[34] Liu, J., Zhang, W., Tang, Y., Tang, J., Wu, G.: Residual feature aggregation network for image super-resolution. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2356–2365 (2020). https://doi.org/10.1109/CVPR42600.2020.00243

[35] Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: ECCV (2018)

[36] Cao, J., Liang, J., Zhang, K., Li, Y., Zhang, Y., Wang, W., Van Gool, L.: Reference-based image super-resolution with deformable attention transformer. In: European Conference on Computer Vision (2022)

[37] Yang, F., Yang, H., Fu, J., Lu, H., Guo, B.: Learning texture transformer network for image super-resolution. In: CVPR (2020)

[38] Gao, G., Wang, Z., Li, J., Li, W., Yu, Y., Zeng, T.: Lightweight bimodal network for single-image super-resolution via symmetric cnn and recursive transformer. ArXiv **abs/2204.13286** (2022)

[39] Zhang, A., Ren, W., Liu, Y., Cao, X.: Lightweight image super-resolution with superpixel token interaction. In: 2023 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 12682–12691 (2023). https://doi.org/10.1109/ICCV51070.2023.01169

[40] Li, H., Cai, D., Xu, J., Watanabe, T.: Residual learning of neural text generation with n-gram language model. In: Goldberg, Y., Kozareva, Z., Zhang, Y. (eds.) Findings of the Association for Computational Linguistics: EMNLP 2022,

pp. 1523–1533. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (2022). https://doi.org/10.18653/v1/2022.findings-emnlp.109 . https://aclanthology.org/2022.findings-emnlp.109

[41] Gu, A., Johnson, I., Goel, K., Saab, K.K., Dao, T., Rudra, A., R'e, C.: Combining recurrent, convolutional, and continuous-time models with linear state-space layers. In: Neural Information Processing Systems (2021). https://api.semanticscholar.org/CorpusID:239998472

[42] Voelker, A.R., Kajić, I., Eliasmith, C.: Legendre memory units: continuous-time representation in recurrent neural networks. Curran Associates Inc., Red Hook, NY, USA (2019)

[43] Gu, A., Dao, T., Ermon, S., Rudra, A., Ré, C.: Hippo: Recurrent memory with optimal polynomial projections. arXiv preprint arXiv:2008.07669 (2020)

[44] Fu, D.Y., Dao, T., Saab, K.K., Thomas, A.W., Rudra, A., Ré, C.: Hungry Hungry Hippos: Towards language modeling with state space models. In: International Conference on Learning Representations (2023)

[45] Gu, A., Goel, K., Ré, C.: Efficiently modeling long sequences with structured state spaces. In: The International Conference on Learning Representations (ICLR) (2022)

[46] Smith, J.T.H., Warrington, A., Linderman, S.: Simplified state space layers for sequence modeling. In: The Eleventh International Conference on Learning Representations (2023). https://openreview.net/forum?id=Ai8Hw3AXqks

[47] Zhu, L., Liao, B., Zhang, Q., Wang, X., Liu, W., Wang, X.: Vision mamba: Efficient visual representation learning with bidirectional state space model. arXiv preprint arXiv:2401.09417 (2024)

[48] Liu, Y., Tian, Y., Zhao, Y., Yu, H., Xie, L., Wang, Y., Ye, Q., Liu, Y.: Vmamba: Visual state space model. arXiv preprint arXiv:2401.10166 (2024)

[49] Ba, J., Kiros, J.R., Hinton, G.E.: Layer normalization. ArXiv **abs/1607.06450** (2016)

[50] Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: ICML 2010, pp. 807–814 (2010)

[51] Zhao, H., Gallo, O., Frosio, I., Kautz, J.: Loss functions for image restoration with neural networks. IEEE Transactions on Computational Imaging **3**(1), 47–57 (2017) https://doi.org/10.1109/TCI.2016.2644865

[52] Yu, W., Luo, M., Zhou, P., Si, C., Zhou, Y., Wang, X., Feng, J., Yan, S.: Metaformer is actually what you need for vision. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10809–10819 (2022). https://doi.org/10.1109/CVPR52688.2022.01055

[53] Huang, Z., Zhang, Z., Lan, C., Zha, Z.-J., Lu, Y., Guo, B.: Adaptive frequency filters as efficient global token mixers. In: ICCV (2023)

[54] Ding, X., Guo, Y., Ding, G., Han, J.: Acnet: Strengthening the kernel skeletons for powerful cnn via asymmetric convolution blocks. In: The IEEE International Conference on Computer Vision (ICCV) (2019)

[55] Ding, X., Zhang, X., Ma, N., Han, J., Ding, G., Sun, J.: Repvgg: Making vgg-style convnets great again. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13733–13742 (2021)

[56] Tolstikhin, I.O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., Lucic, M., Dosovitskiy, A.: Mlp-mixer: An all-mlp architecture for vision. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P.S., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems, vol. 34, pp. 24261–24272. Curran Associates, Inc., ??? (2021). https://proceedings.neurips.cc/paper_files/paper/2021/file/cba0a4ee5ccd02fda0fe3f9a3e7b89fe-Paper.pdf

[57] Shi, D.: Transnext: Robust foveal visual perception for vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 17773–17783 (2024)

[58] Hua, W., Dai, Z., Liu, H., Le, Q.: Transformer quality in linear time. In: Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., Sabato, S. (eds.) Proceedings of the 39th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 162, pp. 9099–9117. PMLR, ??? (2022). https://proceedings.mlr.press/v162/hua22a.html

[59] Ramachandran, P., Zoph, B., Le, Q.V.: Swish: a self-gated activation function. arXiv: Neural and Evolutionary Computing (2017)

[60] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)

[61] Hendrycks, D., Gimpel, K.: Gaussian error linear units (gelus). arXiv: Learning (2016)

[62] Wei, G., Zhang, Z., Lan, C., Lu, Y., Chen, Z.: Activemlp: An mlp-like architecture with active token mixer. arXiv preprint arXiv:2203.06108 (2022)

[63] Hui, Z., Wang, X., Gao, X.: Fast and accurate single image super-resolution via information distillation network. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 723–731 (2018). https://doi.org/10.1109/CVPR.2018.00082

[64] Zhang, X., Zeng, H., Guo, S., Zhang, L.: Efficient long-range attention network for image super-resolution. In: European Conference on Computer Vision (2022)

[65] Gu, J., Dong, C.: Interpreting super-resolution networks with local attribution maps. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9199–9208 (2021)

[66] Agustsson, E., Timofte, R.: Ntire 2017 challenge on single image super-resolution: Dataset and study. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (2017)

[67] Lim, B., Son, S., Kim, H., Nah, S., Lee, K.M.: Enhanced deep residual networks for single image super-resolution. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (2017)

[68] Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing **13**(4), 600–612 (2004) https://doi.org/10.1109/TIP.2003.819861

[69] Bevilacqua, M., Roumy, A., Guillemot, C., Morel, A.: Low-complexity single image super-resolution based on nonnegative neighbor embedding. In: British Machine Vision Conference (2012)

[70] Zeyde, R., Elad, M., Protter, M.: On single image scale-up using sparse-representations. In: International Conference on Curves and Surfaces (2010)

[71] Martin, D., Fowlkes, C., Tal, D., Malik, J.: 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, pp. 416–4232 (2001). https://doi.org/10.1109/ICCV.2001.937655

[72] Huang, J.B., Singh, A., Ahuja, N.: Single image super-resolution from transformed self-exemplars. In: IEEE (2015)

[73] Matsui, Y., Ito, K., Aramaki, Y., Fujimoto, A., Ogawa, T., Yamasaki, T., Aizawa, K.: Sketch-based manga retrieval using manga109 dataset. Multimedia Tools and Applications **76**, 21811–21838 (2015)