mathematics

MDPI

Article

MambaSR: Arbitrary-Scale Super-Resolution Integrating Mamba with Fast Fourier Convolution Blocks

Jin Yan [1], Zongren Chen [1,2], Zhiyuan Pei [1], Xiaoping Lu [1,*] and Hua Zheng [3]

[1] School of Computer Science and Engineering, Macau University of Science and Technology, Macao 999078, China; 2109853gii30010@student.must.edu.mo (J.Y.); 2009853gii30015@student.must.edu.mo (Z.C.); 2109853eii300027@student.must.edu.mo (Z.P.)
[2] Computer Engineering Technical College (Artificial Intelligence College), Guangdong Polytechnic of Science and Technology, Zhuhai 519090, China
[3] School of Mathematics and Statistics, Shaoguan University, Shaoguan 512005, China; hzheng@sgu.edu.cn
* Correspondence: xplu@must.edu.mo

Abstract: Traditional single image super-resolution (SISR) methods, which focus on integer scale super-resolution, often require separate training for each scale factor, leading to increased computational resource consumption. In this paper, we propose MambaSR, a novel arbitrary-scale super-resolution approach integrating Mamba with Fast Fourier Convolution Blocks. MambaSR leverages the strengths of the Mamba state-space model to extract long-range dependencies. In addition, Fast Fourier Convolution Blocks are proposed to capture the global information in the frequency domain. The experimental results demonstrate that MambaSR achieves superior performance compared to different methods across various benchmark datasets. Specifically, on the Urban100 dataset, MambaSR outperforms MetaSR by 0.93 dB in PSNR and 0.0203 dB in SSIM, and on the Manga109 dataset, it achieves an average PSNR improvement of 1.00 dB and an SSIM improvement of 0.0093 dB. These results highlight the efficacy of MambaSR in enhancing image quality for arbitrary-scale super-resolution.

Keywords: super-resolution; fast Fourier transform; state-space model; Mamba

MSC: 68T07

Citation: Yan, J.; Chen, Z.; Pei, Z.; Lu, X.; Zheng, H. MambaSR: Arbitrary-Scale Super-Resolution Integrating Mamba with Fast Fourier Convolution Blocks. *Mathematics* **2024**, *12*, 2370. https://doi.org/10.3390/math12152370

Academic Editor: Jonathan Blackledge

Received: 25 June 2024
Revised: 25 July 2024
Accepted: 29 July 2024
Published: 30 July 2024

1. Introduction

Single image super-resolution (SISR) is a fundamental task in computer vision, aimed at reconstructing high-resolution (HR) images from low-resolution (LR) inputs [1]. This problem is inherently ill posed, as defined by Hadamard, due to the loss of information when an image is downsampled [2]. According to the Hadamard definition, a problem is well posed if a solution exists, the solution is unique, and the solution's behavior changes continuously with the initial conditions. Since SISR often does not meet these criteria, it is classified as an inverse ill-posed problem. Image super-resolution has a wide range of applications in various fields, such as face recognition [3], medical imaging [4,5], satellite imagery [6], and video surveillance [7]. According to Yang et al. [8], super-resolution methods can be roughly categorized into prediction methods [9], edge-based methods [10], statistical methods [11,12], patch-based methods [13–15], and deep learning methods [16]. Although traditional methods such as Kalman filters [12] have achieved some success in the field of image super-resolution, they have limitations when dealing with complex and large-scale data. These methods rely on manual feature extraction and predefined models, making it difficult to cope with diverse and complex image data. With the development of computational power and big data, deep learning methods are rapidly emerging in the field of image super-resolution. In this article, the methods that employ deep learning methods are the focus. Recent advancements in deep learning have significantly improved the

performance of SISR, primarily through the development of convolutional neural networks (CNNs) and various upsampling techniques [17]. However, traditional SISR deep learning methods often focus on integer scale (e.g., ×2, ×3, or ×4) super-resolution, which limits their performance in practical applications where arbitrary-scale upsampling is required. In addition, traditional SISR deep learning methods require separate training for each integer scale, resulting in each model being trained at least three times (×2, ×3, ×4). This consumes substantial computational resources and time.

Arbitrary-scale super-resolution (ASSR) addresses this limitation by allowing for flexible and continuous scaling factors, thus providing a more general solution for real-world applications. For example, ASSR is typically trained with a random input of images magnified one to four times, so it only needs to be trained once. Furthermore, in the context of display applications, the utilization of ASSR enables generating a HR image in any size input. Additionally, the ability to zoom in on an image freely renders ASSR a valuable tool for discerning details in tasks such as face recognition. Several approaches have been proposed to tackle ASSR, including Meta-SR by Hu et al. [18], which employed meta-learning to dynamically predict the weights of upscaling filters based on the input scale factor, and the Learning Implicit Image Function (LIIF) framework by Chen et al. [19], which represents images as continuous functions to allow for flexible upscaling.

However, there are still challenges in achieving high-quality ASSR. The most commonly employed ASSR methods are based on CNNs. Although CNNs can effectively extract local features, they have difficulty in capturing the global context and long-range dependencies [20]. The super-resolution image will be limited by their local receptive field.

Recently, structured state-space models (S4) [21,22] inspired by classical state-space models have gained significant interest for their outstanding ability to model long-range dependencies. Fundamentally, these models can be understood as a hybrid of CNNs and recurrent neural networks (RNNs). Moreover, Mamba [23], a state-of-the-art selective structured state-space model, can model better long-range dependencies in natural language processing (NLP). This implies that Mamba-based ASSR networks can inherently capture global context and long-range dependencies, thereby enhancing the reconstruction quality. However, the potential of state-space models (SSMs) in ASSR networks has not been fully studied. Given the impressive efficiency and powerful long-range dependency modeling capabilities of SSMs, we attempted to employ an SSM in ASSR networks to explore the potential of Mamba for achieving efficient long-range modeling. More specifically, we introduce MambaSR, a novel SSM-based framework for arbitrary-scale super-resolution that leverages Mamba, an innovative sequence modeling technique, in conjunction with Fast Fourier Convolution (FFC) blocks to capture frequency information. MambaSR is designed to efficiently handle arbitrary scaling factors while maintaining high-quality reconstruction. The core contributions of this work are as follows:

- To the best of our knowledge, this is the pioneering research effort that seeks to apply an SSM to arbitrary-scale super-resolution and demonstrates its effectiveness;
- We introduce the Residual Fast Fourier Transform State-Space Block (RFFTSSB), which combines the strengths of Vision State-Space Modules (VSSM) and Fast Fourier Transform Convolutional Blocks (FFTConv) to enhance features by leveraging both spatial and frequency domain information;
- We conduct extensive experiments to evaluate the performance of MambaSR, demonstrating its superiority over existing methods in terms of both visual comparisons and quantitative metrics.

2. Related Work

2.1. Arbitrary-Scale Super-Resolution

Arbitrary-scale single image super-resolution (SISR) improves flexibility by supporting both integer and non-integer scale factors, addressing the limitations of traditional SISR methods. The challenge of arbitrary-scale super-resolution has garnered attention due to its practical importance in various real-world applications, such as security surveillance,

medical imaging, and satellite imagery. Meta-SR, proposed by Hu et al. [18], introduced a novel approach to arbitrary-scale super-resolution by leveraging meta-learning principles. Unlike traditional SR methods that require separate models for different scaling factors, Meta-SR employs a single model capable of handling any scaling factor. This is achieved through a Meta-Upscale Module that dynamically predicts the weights of the upscaling filters based on the input scale factor. This method ensures efficient computation and practical scalability, as it eliminates the need for storing multiple models for different scales. The experimental results demonstrated the superiority of Meta-SR over traditional methods in both performance and computational efficiency. The Learning Implicit Image Function (LIIF) framework by Chen et al. [19] extended the concept of arbitrary-scale SR by representing images as a continuous function. This method allows for flexible and continuous upscaling, addressing the limitations of discrete scaling factors in traditional approaches. LIIF utilizes implicit neural representations to infer pixel values at arbitrary coordinates, providing high-quality SR outputs across various scales. The integration of continuous image representation techniques makes LIIF a robust solution for tasks requiring flexible zooming capabilities. LTE, or Learning Texture Encoding, introduced by Jin et al. [24], focuses on enhancing SR performance by explicitly learning texture information. The LTE framework incorporates a texture encoder–decoder structure that captures high-frequency details, which are crucial for high-quality SR. By learning texture priors, LTE effectively reconstructs fine details and textures, outperforming conventional SR methods that often struggle with texture synthesis. This approach highlights the importance of texture information in achieving superior SR results. The Super-Resolution Neural Operator (SRNO) by Li et al. [25] leverages the concept of neural operators to address the arbitrary-scale SR problem. SRNO introduces a neural operator framework that learns mappings between function spaces, allowing for scalable and efficient SR. This method utilizes a hierarchical structure to process images at multiple scales, ensuring that both global structure and local details are well preserved. The neural operator framework provides a flexible and powerful tool for SR, capable of adapting to various scaling requirements with high fidelity.

2.2. State-Space Models

In recent advancements, state-space models (SSMs) [21,22] have demonstrated significant potential in diverse applications. For instance, a state-space model for a continuous reheating furnace was developed using the finite volume method, optimizing energy efficiency and heating quality [26]. A semi-complete data augmentation algorithm was introduced to enhance state-space model fitting efficiency by combining data augmentation with numerical integration [27]. Similarly, a state-space model for a micro-high-temperature gas-cooled reactor (Mi-HTR) with a helium Brayton cycle was developed, demonstrating accuracy under various disturbances [28]. Furthermore, SSMs were applied to monitor multistage healthcare processes, integrating machine learning techniques with statistical control charts to detect anomalies early in surgical outcomes [29].

In addition, state-space models have been extensively studied in the context of sequence modeling due to their ability to capture long-range dependencies effectively. Gu et al. [21] proposed the structured state-space (S4) model, which addressed the computational inefficiencies of traditional SSMs. By introducing a novel parameterization for the SSM, S4 achieved significant improvements in handling long sequences, as demonstrated by its performance on the Long Range Arena (LRA) benchmark. Building on the foundations laid by S4, Goel et al. [30] introduced the Simplified State-Space Layer (S5), which further streamlined the computational process by utilizing a single multi-input, multi-output (MIMO) SSM and efficient parallel scans. This resulted in a model that maintained the theoretical strengths of S4 while being more efficient and easier to implement. The Gated State-Space (GSS) model, presented by Gu et al. [31], leverages the advantages of SSMs in language modeling. By recasting the model as a convolution with a large kernel, the GSS achieves significant performance gains in tasks requiring the integration of information from distant parts of the input. Finally, the Mamba model [23] explores

hardware-aware state expansion techniques to optimize the execution of SSMs. By introducing a selective state-space model (S6) that adapts to input-dependent dynamics, Mamba effectively balances computational efficiency and performance, making it suitable for deployment in resource-constrained environments. Following the success of SSMs in modeling long sequences, researchers began exploring their application in computer vision tasks. For example, VMamba [32] and Vim [33] incorporate an innovative vision backbone based on Mamba. As a result of the remarkable performance in visual tasks, researchers actively explored its applications across different fields including image classification [32,33], medical image segmentation [34–37], and others [38–40]. Therefore, this paper proposes a super-resolution Mamba model to explore the potential of Mamba for arbitrary-scale super-resolution.

3. Method

3.1. Preliminaries

Recent advancements in SSM-based frameworks, such as the structured state-space sequence model (S4) and Mamba, are founded on a traditional continuous system to map a unidimensional input function or sequence, designated as $a(u) \in \mathbb{R}$, through an implicit latent state $b(u) \in \mathbb{R}^M$ to an output $c(u) \in \mathbb{R}$. This framework can be characterized using a linear Ordinary Differential Equation (ODE) [23]:

$$\begin{aligned} b'(u) &= \boldsymbol{D}b(u) + \boldsymbol{E}a(u), \\ c(u) &= \boldsymbol{F}b(u). \end{aligned} \tag{1}$$

where $\boldsymbol{D} \in \mathbb{R}^{M\times M}$ is the state matrix, and $\boldsymbol{E} \in \mathbb{R}^{M\times 1}$ and $\boldsymbol{F} \in \mathbb{R}^{1\times M}$ are the projection parameters. More details can be found in the statements in Mamba [23].

Subsequently, the discretization procedure is applied for deep learning purposes by incorporating a timescale parameter $\boldsymbol{\Lambda}$ to transform $\boldsymbol{D}$ and $\boldsymbol{E}$ into their discrete forms $\overline{\boldsymbol{D}}$ and $\overline{\boldsymbol{E}}$ using a predetermined discretization rule. The zero-order hold (ZOH) technique is typically utilized for this discretization, and it can be formulated as follows:

$$\begin{aligned} \overline{\boldsymbol{D}} &= \exp(\boldsymbol{\Lambda D}), \\ \overline{\boldsymbol{E}} &= (\boldsymbol{\Lambda D})^{-1}(\exp(\boldsymbol{\Lambda D}) - \boldsymbol{I}) \cdot \boldsymbol{\Lambda E}. \end{aligned} \tag{2}$$

After the discretization process, a_k is applied instead of a continuous input signal $a(u)$. Equation (1) with a time interval Λ can be reformulated as:

$$\begin{aligned} b_k &= \overline{\boldsymbol{D}}b_{k-1} + \overline{\boldsymbol{E}}a_k, \\ c_k &= \boldsymbol{F}b_k. \end{aligned} \tag{3}$$

As a result, Equation (3) can be mathematically interpreted as a convolution operation:

$$\begin{aligned} \overline{\boldsymbol{H}} &= (\boldsymbol{F}\,\overline{\boldsymbol{E}}, \boldsymbol{F}\,\overline{\boldsymbol{D}}\,\overline{\boldsymbol{E}}, \ldots, \boldsymbol{F}\,(\overline{\boldsymbol{D}}^{\boldsymbol{L-1}})\,\overline{\boldsymbol{E}}), \\ c &= a \circledast \overline{\boldsymbol{H}}, \end{aligned} \tag{4}$$

where $\overline{\boldsymbol{H}} \in \mathbb{R}^L$ is a structured convolutional kernel, and L represents the length of the input sequence a. The symbol $\circledast$ denotes the convolution operation.

Recent enhancements to the Mamba state-space model have improved its ability to support dynamic feature representation, making $\overline{\boldsymbol{E}}$, $\boldsymbol{F}$, and $\boldsymbol{\Lambda}$ adaptive to input variations. Mamba's methodology for image super-resolution leverages the strengths of the S4 model. Mamba employs the same recursive structure as outlined in Equation (3), facilitating the retention of extremely long sequences and the activation of additional pixels for reconstruction. Furthermore, Mamba benefits from a parallel scan algorithm, as described in Equation (4), which facilitates efficient parallel processing and training.

3.2. MambaSR

The proposed MambaSR network mainly consists of three parts: a Feature Representor ($F_{represention}$), a Feature Enhancer ($F_{enhance}$), and a Feature Reconstructor ($F_{reconstruct}$).

These three parts are illustrated in Figure 1, which shows the process of scaling the input from (3,h,w) to (3,h*2,w*2). The Feature Representor is used to obtain the shallow features from the low-resolution (LR) input image, which can be deployed with a feature extraction network, such as the enhanced deep SR network (EDSR) [41], and residual dense network (RDN) [42]. The second part is designed for feature enhancement, which includes several Residual Fast Fourier Transform State-Space Units (RFFTSSUs) to enhance the features extracted from the first part and facilitate the integration of contextual information from various sources and perspectives. Finally, the Feature Reconstructor reconstructs the high-resolution (HR) image from the enhanced features.

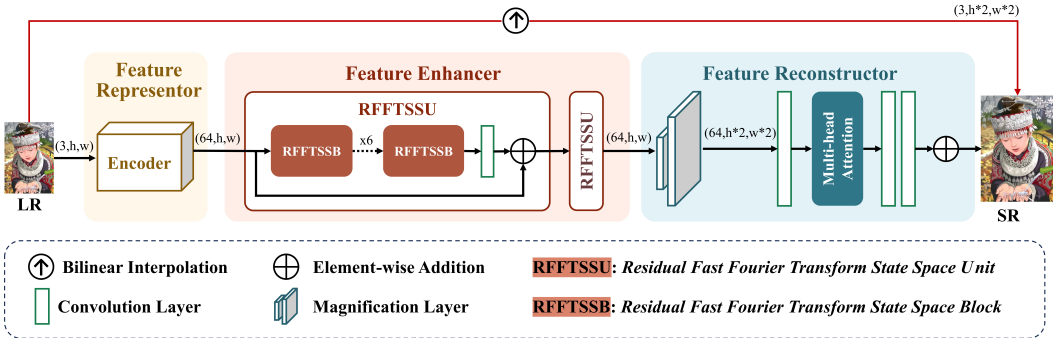

Figure 1. The framework of the proposed network: MambaSR.

The LR input image I_{LR} is first fed into an encoder to obtain the initial feature representation:

$$F_0 = F_{represention}(I_{LR}), \tag{5}$$

where F_0 represents the extracted shallow features and $F_{represention}$ corresponds to the feature extraction network, either EDSR or RDN.

The extracted features F_0 are then enhanced using the proposed RFFTSSU. Each RFFTSSU consists of several Residual Fast Fourier Transform State-Space Blocks (RFFTSSB). In the Feature Enhancer, there are two RFFTSSUs. The enhanced features are represented as:

$$F_{enhanced} = RFFTSSU(RFFTSSU(F_0)). \tag{6}$$

After passing through the i RFFTSSB, the features are processed by a 3×3 convolutional layer $Conv_{3\times3}$ and then added element-wise to the original features F_0:

$$RFFTSSU(F_0) = Conv_{3\times3}(RFFTSSB_i(\cdots RFFTSSB_2(RFFTSSB_1(F_0)))) + F_0, \tag{7}$$

where i is equal to six because there are six RFFTSSBs in the RFFTSSU.

The enhanced features $F_{enhanced}$ are then magnified [25] and passed through a series of convolutional operations and a multi-head attention mechanism to reconstruct the final HR image. First, the magnified features are passed through a 1×1 convolutional layer:

$$F_{magnified} = Magnification(F_{enhanced}), \tag{8}$$

$$F_{conv1} = Conv_{1\times1}(F_{magnified}), \tag{9}$$

followed by a multi-head attention mechanism and a convolutional block consisting of two 1×1 convolutions:

$$F_{attn} = MultiHeadAttention(F_{conv1}), \tag{10}$$

$$F_{conv2} = Conv_{1\times1}(F_{attn}), \tag{11}$$

$$F_{conv3} = Conv_{1\times1}(F_{conv2}). \tag{12}$$

The reconstructed features are then combined with the bilinearly interpolated LR input to produce the final output:

$$I_{SR} = F_{conv3} + I_{LR}^{up}, \tag{13}$$

where I_{LR}^{up} denotes the bilinearly interpolated LR input.

In summary, the MambaSR network effectively enhances and reconstructs high-resolution images through its three-part architecture, leveraging the innovative RFFTSSU to improve feature representation and contextual information integration.

3.3. Residual Fast Fourier Transform State-Space Block (RFFTSSB)

As shown in Figure 2, the Residual Fast Fourier Transform State-Space Block (RFFTSSB) is a critical component as a Feature Enhancer of our MambaSR network. It is designed to enhance features by leveraging both spatial and frequency domain information. The RFFTSSB consists of two main parts: the Vision State-Space Module (VSSM) and the Fast Fourier Transform Convolutional Block (FFTConv).

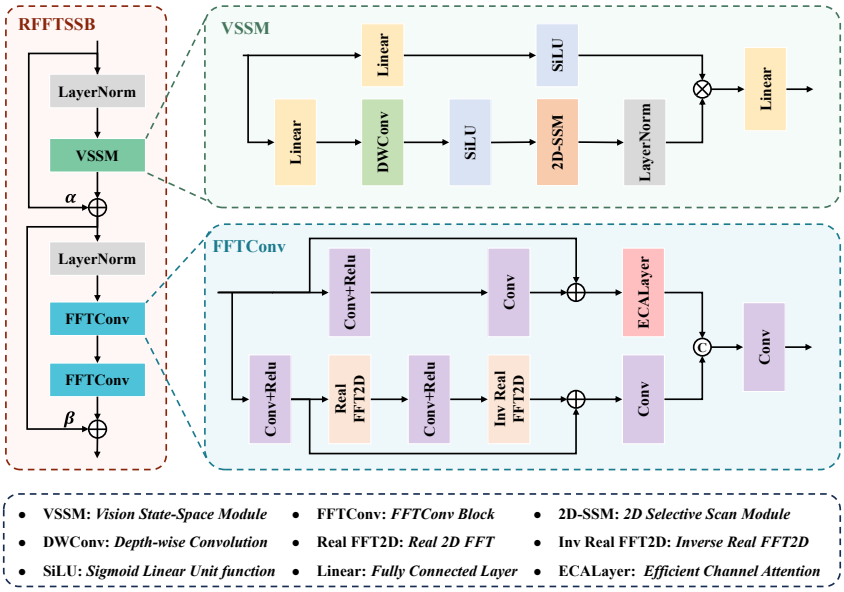

Figure 2. The architecture of the Residual Fast Fourier Transform State-Space Block.

The input features F_{input} first pass through a Layer Normalization layer, followed by the VSSM [32], which can extract the spatial long-term dependencies. The output of the VSSM is added element-wise to the input features scaled by a factor. The resulting features are then passed through another Layer Normalization layer and processed by the FFTConv. Finally, the output of the FFTConv is added element-wise to the original input features to form the residual connection. This process can be summarized as follows:

$$V^n = F_{vssm}(LayerNorm(F_R^n)) + \alpha F_R^n, \tag{14}$$

$$F_R^{n+1} = F_{fftconv}(F_{fftconv}(LayerNorm(V^n))) + \beta V^n, \tag{15}$$

where $V^n \in \mathbb{R}^{H \times W \times C}$ is the input feature at the n-th layer, F_{vssm} and $F_{fftconv}$ denote the functions representing the operations of the VSSM and the FFTConv, respectively. The terms $\alpha, \beta \in \mathbb{R}^C$ are learnable parameters that scale the features to modulate the importance of the residual connections. And, n is from 0 to 5.

The variables H, W, and C represent the height, width, and number of channels of the feature map, respectively. The LayerNorm function denotes the Layer Normalization operation, which normalizes the input features across the channel dimension to stabilize and accelerate the training process. The element-wise addition operations are crucial for

incorporating the residual connections that help in preserving the original features while enhancing them with the extracted spatial and frequency domain information.

3.4. Vision State-Space Module (VSSM)

The Vision State-Space Module (VSSM) captures spatial dependencies and consists of several layers. It is designed to extract and enhance spatial long-term dependencies within the input features through a series of transformations and operations.

The input features F_{input} are first passed through a Linear layer, transforming the feature dimensions. This is followed by a Depthwise Convolution (DWConv) layer, which performs spatial filtering to capture local spatial information. The output of the DWConv layer is then processed by a SiLU activation function, which introduces non-linearity and helps in better feature representation. Next, the features are passed through a 2D Selective Scan (2D-SSM) module, which selectively scans and aggregates spatial information from the feature maps.

After the 2D-SSM, the output is normalized using a Layer Normalization layer to ensure stable training and better convergence. The normalized output is added element-wise to a linearly transformed version of the original input features. Finally, the combined features are processed by another Linear layer to produce the output of the VSSM. This process can be summarized with the following equations:

$$F_{v1} = LayerNorm(2D\text{-}SSM(SiLU(DWConv(Linear1(F_{input}))))), \tag{16}$$

$$F_{v2} = SiLU(Linear2(F_{input})), \tag{17}$$

$$F_{output} = Linear3(F_{v1} \odot F_{v2}), \tag{18}$$

where $F_{input} \in \mathbb{R}^{H \times W \times C}$ represents the input feature map, $Linear1$, $Linear2$, and $Linear3$ denote the linear transformations applied at different stages of the module, $DWConv$ represents the Depthwise Convolution operation, and $\odot$ is the Hadamard product.

Equation (16) combines several transformations into a single operation that enhances the features by integrating spatial information. The addition operation in this equation is crucial for incorporating the residual connection, which helps in preserving the original features while enhancing them. Then, Equation (18) represents the final linear transformation that refines the enhanced features to produce the output of the VSSM, ready for further processing in the network.

3.5. 2D Selective Scan Module

Mamba [23] (the selective scan space-state sequence model (S6)) handles input data in a sequential manner, limiting their capability to extract information exclusively from the scanned data segment. Although this method suits natural language processing tasks due to their inherent sequential structure, it faces significant challenges when dealing with non-sequential data like images. To address this issue, we implement the 2D Selective Scan module (2D-SSM) as proposed in [32]. The 2D-SSM model is based on the selective scan space-state sequence model (S6), addressing the issue of direction sensitivity that was identified as a limitation of the S6 model.

As illustrated in Figure 3, this module transforms 2D image features into a 1D sequence by scanning in four different directions: from top-left to bottom-right, bottom-right to top-left, top-right to bottom-left, and bottom-left to top-right. The S6 block is employed to extract features from all sequences, facilitating comprehensive scanning of information from diverse directions. These sequences are eventually merged through summation and reshaped to reconstruct the original 2D structure, allowing a comprehensive representation of the spatial data.

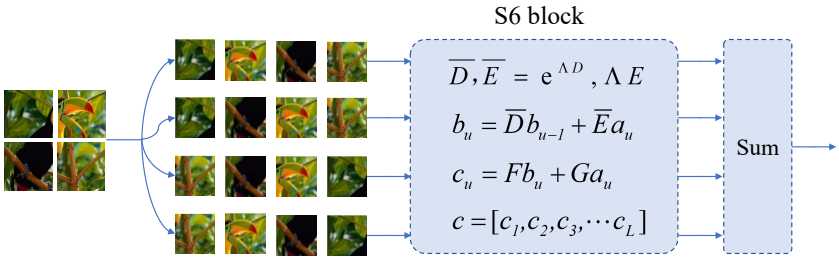

Figure 3. The architecture of the 2D Selective Scan module.

3.6. FFTConv Block

In Figure 2, the FFTConv block can be divided into the frequency branch and the spatial branch. The FFTConv block enhances the features by transforming them into the frequency domain, applying convolutions, and then transforming them back into the spatial domain. The process begins with the input features being passed through a convolutional layer followed by a ReLU activation function:

$$F_{conv\_relu1} = ReLU(Conv_{1\times1}(F_{input})). \tag{19}$$

The activated features are then transformed into the frequency domain using the Real FFT2D:

$$F_{fft} = RealFFT2D(F_{conv\_relu1}). \tag{20}$$

The Real 2D Fast Fourier Transform (Real FFT2D) converts spatial domain features into the frequency domain. For a 2D signal $x(m,n)$, the Real FFT2D is defined as:

$$X(k,l) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} x(m,n) e^{-j2\pi\left(\frac{km}{M}+\frac{ln}{N}\right)}, \tag{21}$$

where $X(k,l)$ represents the frequency domain representation of $x(m,n)$, and M and N are the dimensions of the input signal.

In the frequency domain, the features are processed through another convolution and then activated by a ReLU function:

$$F_{conv\_relu2} = ReLU(Conv_{1\times1}(F_{fft})). \tag{22}$$

The convolved features are then transformed back to the spatial domain using the Inverse Real FFT2D:

$$F_{ifft} = InvRealFFT2D(F_{conv\_relu2}). \tag{23}$$

The Inverse Real 2D Fast Fourier Transform (Inv Real FFT2D) converts frequency domain features back into the spatial domain. The inverse transform is defined as:

$$x(m,n) = \frac{1}{MN}\sum_{k=0}^{M-1}\sum_{l=0}^{N-1} X(k,l) e^{j2\pi\left(\frac{km}{M}+\frac{ln}{N}\right)}, \tag{24}$$

where $x(m,n)$ represents the reconstructed spatial domain signal, and $X(k,l)$ is the frequency domain representation.

The spatial features are further processed by a convolutional layer:

$$F_{conv2} = Conv_{1\times1}(F_{ifft}). \tag{25}$$

Therefore, the output of the frequency branch can be formulated as:

$$F_{fre} = Conv_{1\times1}(F_{conv2} + F_{conv\_relu2}). \tag{26}$$

At the same time, the input data are fed into the spatial branch, which can be expressed as follows:

$$F_{conv\_spa1} = Conv_{3\times3}(ReLU(Conv_{3\times3}(F_{input}))). \tag{27}$$

These features are then refined using an ECA (Efficient Channel Attention) layer [43], which implements local cross-channel interactions using one-dimensional convolution to extract inter-channel dependencies:

$$F_{spa} = ECALayer(F_{conv\_spa1}). \tag{28}$$

Eventually, the outputs of the frequency domain branch and the spatial domain branch are combined, and a convolutional layer is employed to integrate the outputs.:

$$F_{fftconv\_out} = Conv_{1\times1}(F_{spa} + F_{fre}). \tag{29}$$

4. Experiment

In this section, the performance of the model was evaluated through extensive experimentation on a continuous scale in four SR benchmark datasets with other arbitrary-scale SR methods. The details of the experimental setup, datasets, and evaluation metrics are introduced. In the end, ablation experiments were conducted to verify the effectiveness of our component.

4.1. Experimental Setup

Following the setup in previous works [19,24], a batch size of 64 and a low-resolution input size of 48×48 were employed in the training. To augment the training dataset, random rotation and horizontal flipping was adopted. The Adam optimizer [44] was used, with a learning rate of 4×10^{-5} while utilizing the L1 loss function. All model training was conducted for 1000 epochs, with the learning rate decaying according to the cosine annealing schedule, following a 50-epoch warm-up phase. Using the same input size, we replicated the SRNO with an input size of 48×48 to compare performance.

4.2. Dataset and Evaluation Metrics

The images used for training were obtained from the DIV2K dataset [45], which includes 1000 images at a 2K resolution, as is described in [25]. Additionally, the performance of the model on the validation sets Set5 [46], Set14 [47], Urban100 [48], and Manga109 [49] was evaluated using continuous scales, including peak signal-to-noise ratio (PSNR) and structural similarity index measurement (SSIM) values.

The PSNR was employed to quantify the quality between the super-resolution images and their original high-resolution images. It is commonly used in the field of image super-resolution and compression. PSNR is defined as:

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{\text{MAX}^2}{\text{MSE}}\right), \tag{30}$$

where MAX represents the maximum possible pixel value of the image. For example, in an 8-bit image, MAX is 255. The term MSE stands for Mean Squared Error, which is calculated as:

$$\text{MSE} = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2. \tag{31}$$

In this formula, m and n are the dimensions of the image, $I(i,j)$ is the pixel value at position (i,j) in the original image, and $K(i,j)$ is the pixel value at position (i,j) in the reconstructed image.

When the images are perfectly matched (i.e., MSE = 0), the reconstructed image is exactly the same as the original image. In this case, the PSNR tends to infinity because the logarithmic function tends to infinity as the input tends to zero. Therefore, theoretically, the PSNR can reach infinity.

When the MSE is large, the difference between images is significant, and the PSNR value becomes very low. When the image difference reaches its maximum (i.e., the reconstructed image and the original image are completely uncorrelated), the MSE reaches its

maximum value (for an 8-bit image, the maximum possible value is 255^2), and the PSNR value approaches 0.

Consequently, for the value of the MSE in Equation (31), the range is from 0 to 255^2. For the value of the PSNR in Equation (30), the range is from 0 to infinity. A higher PSNR value generally indicates better quality, as it implies that the reconstructed image is closer to the original.

The structural similarity index measurement (SSIM) is another important metric used to evaluate the quality of images. Unlike the PSNR, which primarily focuses on pixel differences, the SSIM considers changes in structural information and perceptual quality. The SSIM is computed by combining three comparison measurements between the images: luminance, contrast, and structure.

First, the luminance comparison function is defined as:

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}, \tag{32}$$

where μ_x and μ_y are the average pixel values of images x and y, respectively. The constant C_1 is introduced to avoid instability when the denominator is very close to zero. This ratio ranges between 0 and 1. When $\mu_x = \mu_y$, the luminance similarity reaches its maximum value of 1.

Next, the contrast comparison function is given by:

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}, \tag{33}$$

where σ_x and σ_y represent the standard deviations of x and y. Similar to C_1, the constant C_2 stabilizes the division. This ratio also ranges between 0 and 1. When $\sigma_x = \sigma_y$, the contrast similarity reaches its maximum value of 1.

Finally, the structure comparison function is expressed as:

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}, \tag{34}$$

where σ_{xy} denotes the covariance between x and y. The constant C_3 is typically chosen to be $C_2/2$ for simplicity. This ratio similarly ranges between 0 and 1. When $\sigma_{xy} = \sigma_x\sigma_y$, the structure similarity reaches its maximum value of 1.

Combining these three components, the overall SSIM index is calculated as:

$$\mathrm{SSIM}(x,y) = [l(x,y)]^\alpha \cdot [c(x,y)]^\beta \cdot [s(x,y)]^\gamma, \tag{35}$$

where α, β, and γ are parameters used to adjust the relative importance of each component. Commonly, $\alpha = \beta = \gamma = 1$, which simplifies the SSIM index to:

$$\mathrm{SSIM}(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}. \tag{36}$$

The constants C_1 and C_2 are defined as:

$$C_1 = (K_1 L)^2 \quad \text{and} \quad C_2 = (K_2 L)^2, \tag{37}$$

where L is the dynamic range of the pixel values (for an 8-bit image, $L = 255$), and K_1 and K_2 are small constants (typically, $K_1 = 0.01$ and $K_2 = 0.03$). The SSIM index ranges from 0 to 1, where a value of 1 indicates perfect structural similarity.

4.3. Results

In this section, the performance of the proposed MambaSR model is compared with advanced arbitrary-scale super-resolution (SR) methods, such as MetaSR, LIIF, LTE, and SRNO. Each method was evaluated on the Urban100, Manga109, Set5, and Set14 datasets. During the training process, the MambaSR model was trained on low-resolution (LR) datasets with various scale factors. As shown in Tables 1–4, the performance of each

method was quantitatively evaluated using PSNR and SSIM metrics with various scale factors. The experiments were conducted using two different encoders: EDSR and RDN.

Table 1. PSNR/SSIM values achieved by different methods with EDSR and RDN on Urban100 datasets. The best results are in bold. The leftmost column represents a scale of magnification ranging from 2.1 to 4.9.

Dataset			**Urban100**		
Method	EDSR-MetaSR	EDSR-LIIF	EDSR-LTE	EDSR-SRNO	EDSR-MambaSR
2.1	31.53/0.9213	31.58/0.9222	31.71/0.9234	31.87/0.9244	**32.58/0.9311**
2.2	31.02/0.9134	31.08/0.9145	31.22/0.9159	31.36/0.9169	**32.05/0.9242**
2.3	30.55/0.9057	30.61/0.9068	30.75/0.9083	30.89/0.9094	**31.55/0.9173**
2.4	30.14/0.8982	30.20/0.8993	30.33/0.9010	30.47/0.9022	**31.11/0.9104**
2.5	29.73/0.8906	29.80/0.8917	29.93/0.8935	30.07/0.8948	**30.71/0.9036**
2.6	29.36/0.8830	29.44/0.8843	29.56/0.8862	29.70/0.8875	**30.32/0.8966**
2.7	29.02/0.8756	29.11/0.8772	29.23/0.8790	29.37/0.8806	**29.97/0.8901**
2.8	28.69/0.8682	28.79/0.8699	28.90/0.8717	29.05/0.8735	**29.64/0.8833**
2.9	28.40/0.8609	28.50/0.8627	28.61/0.8647	28.76/0.8665	**29.34/0.8765**
3.1	27.86/0.8467	27.97/0.8490	28.07/0.8510	28.23/0.8531	**28.79/0.8638**
3.2	27.61/0.8397	27.73/0.8424	27.83/0.8444	27.99/0.8466	**28.54/0.8574**
3.3	27.37/0.8327	27.50/0.8355	27.60/0.8377	27.76/0.8400	**28.29/0.8510**
3.4	27.15/0.8258	27.28/0.8290	27.38/0.8313	27.54/0.8337	**28.06/0.8450**
3.5	26.93/0.8190	27.07/0.8224	27.17/0.8248	27.33/0.8275	**27.84/0.8390**
3.6	26.72/0.8122	26.86/0.8160	26.97/0.8183	27.13/0.8211	**27.64/0.8331**
3.7	26.54/0.8057	26.68/0.8097	26.78/0.8121	26.94/0.8150	**27.45/0.8272**
3.8	26.34/0.7988	26.50/0.8033	26.59/0.8057	26.75/0.8087	**27.25/0.8213**
3.9	26.16/0.7922	26.32/0.7970	26.41/0.7994	26.58/0.8027	**27.06/0.8154**
4.1	25.82/0.7793	25.99/0.7849	26.08/0.7874	26.24/0.7909	**26.71/0.8041**
4.2	25.65/0.7730	25.83/0.7791	25.92/0.7816	26.08/0.7852	**26.55/0.7987**
4.3	25.51/0.7669	25.69/0.7733	25.78/0.7759	25.94/0.7796	**26.41/0.7934**
4.4	25.35/0.7605	25.54/0.7674	25.63/0.7701	25.79/0.7738	**26.25/0.7879**
4.5	25.22/0.7546	25.40/0.7617	25.49/0.7644	25.65/0.7684	**26.10/0.7824**
4.6	25.08/0.7486	25.27/0.7561	25.36/0.7588	25.53/0.7631	**25.97/0.7773**
4.7	24.95/0.7427	25.13/0.7504	25.23/0.7533	25.38/0.7574	**25.83/0.7723**
4.8	24.83/0.7371	25.02/0.7452	25.10/0.7479	25.27/0.7522	**25.70/0.7671**
4.9	24.70/0.7315	24.89/0.7398	24.97/0.7425	25.14/0.7472	**25.57/0.7623**
Method	RDN-MetaSR	RDN-LIIF	RDN-LTE	RDN-SRNO	RDN-MambaSR
2.1	32.30/0.9293	32.30/0.9293	32.45/0.9306	32.42/0.9299	**32.71/0.9321**
2.2	31.80/0.9222	31.79/0.9223	31.95/0.9237	31.91/0.9230	**32.21/0.9255**
2.3	31.32/0.9152	31.30/0.9153	31.47/0.9168	31.43/0.9160	**31.72/0.9187**
2.4	30.89/0.9084	30.89/0.9086	31.05/0.9102	31.01/0.9092	**31.27/0.9119**
2.5	30.50/0.9015	30.49/0.9016	30.64/0.9034	30.61/0.9025	**30.87/0.9052**
2.6	30.11/0.8942	30.09/0.8945	30.26/0.8965	30.23/0.8954	**30.47/0.8982**
2.7	29.77/0.8876	29.75/0.8878	29.91/0.8899	29.89/0.8888	**30.12/0.8918**
2.8	29.44/0.8807	29.41/0.8809	29.57/0.8831	29.56/0.8821	**29.79/0.8851**
2.9	29.13/0.8739	29.12/0.8742	29.26/0.8764	29.25/0.8753	**29.48/0.8786**
3.1	28.57/0.8610	28.56/0.8613	28.71/0.8637	28.70/0.8625	**28.92/0.8659**
3.2	28.32/0.8546	28.32/0.8550	28.46/0.8575	28.46/0.8564	**28.67/0.8598**
3.3	28.06/0.8480	28.07/0.8484	28.22/0.8511	28.21/0.8500	**28.42/0.8535**
3.4	27.84/0.8419	27.85/0.8425	27.98/0.8450	27.98/0.8438	**28.19/0.8476**
3.5	27.61/0.8357	27.64/0.8364	27.77/0.8390	27.77/0.8380	**27.97/0.8416**
3.6	27.40/0.8294	27.42/0.8303	27.56/0.8331	27.56/0.8319	**27.76/0.8357**
3.7	27.19/0.8232	27.22/0.8242	27.37/0.8272	27.35/0.8259	**27.56/0.8300**
3.8	26.99/0.8168	27.04/0.8181	27.17/0.8211	27.18/0.8200	**27.37/0.8242**
3.9	26.80/0.8107	26.85/0.8120	26.99/0.8153	27.00/0.8142	**27.18/0.8182**
4.1	26.43/0.7984	26.50/0.8004	26.63/0.8036	26.65/0.8028	**26.84/0.8072**
4.2	26.26/0.7926	26.33/0.7948	26.48/0.7983	26.48/0.7973	**26.68/0.8019**
4.3	26.10/0.7867	26.18/0.7891	26.33/0.7928	26.34/0.7918	**26.53/0.7966**
4.4	25.95/0.7808	26.02/0.7836	26.17/0.7872	26.18/0.7865	**26.37/0.7913**
4.5	25.79/0.7750	25.87/0.7782	26.02/0.7819	26.03/0.7810	**26.22/0.7859**
4.6	25.66/0.7695	25.74/0.7729	25.89/0.7767	25.90/0.7759	**26.09/0.7810**
4.7	25.50/0.7636	25.60/0.7674	25.74/0.7713	25.75/0.7703	**25.94/0.7757**
4.8	25.38/0.7584	25.49/0.7625	25.63/0.7663	25.64/0.7657	**25.82/0.7708**
4.9	25.25/0.7527	25.35/0.7573	25.49/0.7611	25.50/0.7603	**25.69/0.7659**

Table 2. PSNR/SSIM values achieved by different methods with EDSR and RDN on Manga109 datasets. The best results are in bold. The leftmost column represents a scale of magnification ranging from 2.1 to 4.9.

Dataset			**Manga109**		
Method	EDSR-MetaSR	EDSR-LIIF	EDSR-LTE	EDSR-SRNO	EDSR-MambaSR
2.1	37.87/0.9745	37.98/0.9747	38.03/0.9748	38.28/0.9755	**38.84/0.9765**
2.2	37.25/0.9715	37.37/0.9717	37.41/0.9719	37.67/0.9726	**38.27/0.9738**
2.3	36.67/0.9684	36.78/0.9686	36.83/0.9689	37.11/0.9697	**37.72/0.9711**
2.4	36.13/0.9651	36.24/0.9655	36.30/0.9658	36.57/0.9666	**37.20/0.9683**
2.5	35.62/0.9618	35.72/0.9623	35.79/0.9626	36.06/0.9636	**36.71/0.9655**
2.6	35.14/0.9585	35.22/0.9590	35.30/0.9594	35.56/0.9605	**36.23/0.9626**
2.7	34.71/0.9551	34.76/0.9557	34.84/0.9562	35.10/0.9573	**35.77/0.9596**
2.8	34.30/0.9518	34.35/0.9525	34.42/0.9530	34.68/0.9541	**35.34/0.9567**
2.9	33.93/0.9484	33.96/0.9492	34.02/0.9497	34.28/0.9509	**34.94/0.9537**
3.1	33.20/0.9416	33.22/0.9426	33.29/0.9432	33.54/0.9446	**34.18/0.9477**
3.2	32.87/0.9383	32.89/0.9394	32.95/0.9400	33.20/0.9415	**33.83/0.9447**
3.3	32.52/0.9346	32.54/0.9359	32.60/0.9365	32.83/0.9381	**33.48/0.9417**
3.4	32.21/0.9310	32.24/0.9326	32.30/0.9332	32.52/0.9348	**33.17/0.9387**
3.5	31.88/0.9273	31.95/0.9292	32.00/0.9299	32.21/0.9315	**32.87/0.9357**
3.6	31.59/0.9237	31.64/0.9256	31.69/0.9264	31.89/0.9281	**32.54/0.9326**
3.7	31.30/0.9198	31.36/0.9221	31.39/0.9229	31.60/0.9246	**32.24/0.9295**
3.8	31.00/0.9159	31.08/0.9184	31.11/0.9193	31.31/0.9210	**31.95/0.9262**
3.9	30.70/0.9119	30.79/0.9147	30.84/0.9157	31.02/0.9174	**31.65/0.9230**
4.1	30.18/0.9039	30.29/0.9073	30.33/0.9086	30.48/0.9101	**31.09/0.9162**
4.2	29.88/0.8996	30.03/0.9035	30.10/0.9049	30.23/0.9063	**30.84/0.9127**
4.3	29.62/0.8955	29.78/0.8996	29.83/0.9012	29.97/0.9027	**30.59/0.9093**
4.4	29.36/0.8912	29.56/0.8960	29.60/0.8974	29.75/0.8992	**30.35/0.9059**
4.5	29.15/0.8872	29.37/0.8926	29.41/0.8939	29.54/0.8959	**30.17/0.9029**
4.6	28.90/0.8830	29.13/0.8888	29.17/0.8901	29.31/0.8924	**29.91/0.8997**
4.7	28.69/0.8786	28.92/0.8850	28.96/0.8863	29.09/0.8886	**29.70/0.8963**
4.8	28.47/0.8747	28.71/0.8816	28.77/0.8829	28.89/0.8854	**29.52/0.8935**
4.9	28.24/0.8701	28.51/0.8778	28.56/0.8790	28.66/0.8816	**29.32/0.8902**
Method	RDN-MetaSR	RDN-LIIF	RDN-LTE	RDN-SRNO	RDN-MambaSR
2.1	38.70/0.9762	38.63/0.9761	38.68/0.9761	38.70/0.9763	**38.92/0.9768**
2.2	38.13/0.9735	38.04/0.9733	38.11/0.9734	38.12/0.9735	**38.35/0.9741**
2.3	37.56/0.9707	37.48/0.9705	37.55/0.9706	37.56/0.9707	**37.81/0.9714**
2.4	37.06/0.9678	36.96/0.9676	37.04/0.9678	37.05/0.9679	**37.30/0.9687**
2.5	36.55/0.9649	36.45/0.9647	36.53/0.9650	36.54/0.9651	**36.81/0.9659**
2.6	36.07/0.9620	35.96/0.9618	36.05/0.9621	36.06/0.9622	**36.34/0.9632**
2.7	35.62/0.9590	35.49/0.9588	35.60/0.9592	35.61/0.9593	**35.89/0.9604**
2.8	35.18/0.9560	35.06/0.9558	35.17/0.9563	35.17/0.9563	**35.46/0.9575**
2.9	34.79/0.9529	34.65/0.9527	34.77/0.9533	34.77/0.9532	**35.06/0.9545**
3.1	34.02/0.9468	33.88/0.9466	34.02/0.9473	34.00/0.9472	**34.30/0.9487**
3.2	33.70/0.9438	33.54/0.9436	33.68/0.9444	33.68/0.9443	**33.96/0.9458**
3.3	33.35/0.9406	33.20/0.9404	33.33/0.9412	33.32/0.9411	**33.60/0.9427**
3.4	33.04/0.9375	32.89/0.9374	33.02/0.9382	33.01/0.9381	**33.29/0.9398**
3.5	32.70/0.9343	32.60/0.9344	32.71/0.9352	32.69/0.9351	**32.98/0.9369**
3.6	32.40/0.9312	32.28/0.9312	32.40/0.9321	32.38/0.9319	**32.66/0.9338**
3.7	32.13/0.9280	32.00/0.9280	32.12/0.9290	32.11/0.9289	**32.37/0.9308**
3.8	31.83/0.9246	31.71/0.9247	31.83/0.9257	31.80/0.9256	**32.09/0.9277**
3.9	31.56/0.9213	31.45/0.9215	31.56/0.9226	31.53/0.9225	**31.82/0.9246**
4.1	31.01/0.9144	30.91/0.9148	31.03/0.9158	31.00/0.9157	**31.26/0.9183**
4.2	30.75/0.9108	30.65/0.9113	30.76/0.9124	30.75/0.9121	**31.01/0.9151**
4.3	30.48/0.9070	30.41/0.9078	30.51/0.9090	30.50/0.9085	**30.76/0.9120**
4.4	30.20/0.9032	30.19/0.9045	30.29/0.9058	30.25/0.9051	**30.52/0.9087**
4.5	29.98/0.8995	29.97/0.9012	30.09/0.9026	30.04/0.9018	**30.30/0.9053**
4.6	29.75/0.8959	29.76/0.8980	29.86/0.8993	29.82/0.8986	**30.08/0.9022**
4.7	29.51/0.8919	29.53/0.8945	29.64/0.8960	29.61/0.8954	**29.88/0.8990**
4.8	29.29/0.8882	29.35/0.8914	29.45/0.8930	29.41/0.8924	**29.67/0.8959**
4.9	29.06/0.8842	29.13/0.8879	29.23/0.8895	29.18/0.8888	**29.44/0.8922**

Table 3. PSNR/SSIM values achieved by different methods with EDSR and RDN on Set5 datasets. The best results are in bold. The leftmost column represents a scale of magnification ranging from 2.1 to 4.9.

Dataset			**Set5**		
Method	EDSR-MetaSR	EDSR-LIIF	EDSR-LTE	EDSR-SRNO	EDSR-MambaSR
2.1	37.46/0.9587	37.47/0.9588	37.52/0.9589	37.56/0.9591	**37.70/0.9596**
2.2	36.96/0.9553	36.96/0.9554	37.01/0.9555	37.07/0.9559	**37.23/0.9564**
2.3	36.65/0.9523	36.67/0.9524	36.72/0.9525	36.78/0.9528	**36.93/0.9535**
2.4	36.25/0.9491	36.26/0.9492	36.30/0.9492	36.36/0.9497	**36.55/0.9505**
2.5	35.87/0.9454	35.91/0.9457	35.93/0.9457	36.02/0.9461	**36.20/0.9470**
2.6	35.60/0.9428	35.64/0.9431	35.64/0.9431	35.73/0.9435	**35.90/0.9443**
2.7	35.34/0.9396	35.36/0.9399	35.38/0.9399	35.44/0.9403	**35.63/0.9412**
2.8	35.02/0.9363	35.03/0.9366	35.06/0.9367	35.13/0.9371	**35.34/0.9383**
2.9	34.72/0.9332	34.74/0.9337	34.74/0.9338	34.84/0.9342	**35.08/0.9357**
3.1	34.19/0.9269	34.21/0.9274	34.29/0.9278	34.33/0.9284	**34.58/0.9300**
3.2	33.94/0.9239	33.97/0.9245	34.02/0.9247	34.08/0.9252	**34.37/0.9272**
3.3	33.64/0.9202	33.69/0.9210	33.72/0.9211	33.76/0.9217	**34.13/0.9242**
3.4	33.44/0.9170	33.52/0.9180	33.58/0.9183	33.64/0.9189	**33.96/0.9214**
3.5	33.20/0.9141	33.29/0.9152	33.36/0.9158	33.40/0.9162	**33.73/0.9186**
3.6	32.93/0.9107	33.02/0.9116	33.03/0.9121	33.06/0.9124	**33.43/0.9154**
3.7	32.75/0.9075	32.89/0.9090	32.85/0.9091	32.91/0.9097	**33.26/0.9128**
3.8	32.54/0.9041	32.67/0.9057	32.66/0.9058	32.70/0.9063	**33.01/0.9093**
3.9	32.32/0.9008	32.43/0.9024	32.44/0.9026	32.53/0.9039	**32.80/0.9064**
4.1	31.84/0.8934	32.02/0.8958	32.03/0.8961	32.06/0.8970	**32.42/0.9005**
4.2	31.73/0.8910	31.89/0.8940	31.90/0.8940	31.95/0.8952	**32.33/0.8987**
4.3	31.46/0.8861	31.65/0.8894	31.68/0.8897	31.75/0.8908	**32.10/0.8946**
4.4	31.23/0.8824	31.46/0.8860	31.48/0.8862	31.59/0.8877	**31.89/0.8913**
4.5	31.11/0.8798	31.29/0.8832	31.30/0.8833	31.36/0.8847	**31.76/0.8889**
4.6	30.91/0.8763	31.11/0.8796	31.12/0.8797	31.19/0.8812	**31.52/0.8853**
4.7	30.76/0.8731	30.95/0.8770	30.95/0.8770	31.06/0.8790	**31.41/0.8828**
4.8	30.55/0.8695	30.79/0.8746	30.79/0.8742	30.88/0.8761	**31.23/0.8804**
4.9	30.41/0.8654	30.65/0.8704	30.62/0.8699	30.74/0.8723	**31.09/0.8765**
Method	RDN-MetaSR	RDN-LIIF	RDN-LTE	RDN-SRNO	RDN-MambaSR
2.1	37.70/0.9596	37.66/0.9595	37.72/0.9597	37.73/0.9598	**37.74/0.9599**
2.2	37.19/0.9563	37.18/0.9563	37.23/0.9564	37.22/0.9564	**37.28/0.9566**
2.3	36.91/0.9534	36.88/0.9533	36.94/0.9536	36.94/0.9537	**36.98/0.9537**
2.4	36.48/0.9502	36.47/0.9502	36.50/0.9504	36.54/0.9505	**36.56/0.9506**
2.5	36.16/0.9469	36.15/0.9468	36.18/0.9470	36.18/0.9471	**36.24/0.9472**
2.6	35.81/0.9441	35.81/0.9441	35.86/0.9443	35.87/0.9444	**35.94/0.9447**
2.7	35.58/0.9411	35.56/0.9410	35.60/0.9412	35.61/0.9413	**35.65/0.9415**
2.8	35.28/0.9382	35.26/0.9379	35.29/0.9383	35.31/0.9383	**35.37/0.9386**
2.9	34.98/0.9353	34.98/0.9353	35.01/0.9356	34.99/0.9355	**35.10/0.9359**
3.1	34.46/0.9294	34.47/0.9294	34.54/0.9299	34.53/0.9299	**34.63/0.9306**
3.2	34.24/0.9266	34.24/0.9265	34.32/0.9271	34.32/0.9270	**34.43/0.9279**
3.3	33.92/0.9231	33.94/0.9232	34.03/0.9240	34.01/0.9237	**34.13/0.9244**
3.4	33.81/0.9206	33.84/0.9206	33.91/0.9213	33.92/0.9214	**33.98/0.9219**
3.5	33.56/0.9174	33.58/0.9177	33.66/0.9182	33.66/0.9183	**33.76/0.9193**
3.6	33.25/0.9140	33.27/0.9144	33.30/0.9148	33.29/0.9148	**33.47/0.9160**
3.7	33.06/0.9110	33.17/0.9121	33.21/0.9125	33.20/0.9127	**33.30/0.9136**
3.8	32.84/0.9076	32.93/0.9084	32.99/0.9090	32.94/0.9088	**33.10/0.9103**
3.9	32.61/0.9043	32.68/0.9052	32.76/0.9058	32.72/0.9056	**32.87/0.9073**
4.1	32.18/0.8977	32.21/0.8987	32.32/0.8996	32.24/0.8993	**32.44/0.9013**
4.2	32.04/0.8957	32.14/0.8970	32.23/0.8979	32.17/0.8976	**32.33/0.8992**
4.3	31.84/0.8913	31.95/0.8929	32.06/0.8941	31.99/0.8932	**32.16/0.8958**
4.4	31.64/0.8880	31.73/0.8895	31.84/0.8905	31.79/0.8899	**31.99/0.8928**
4.5	31.48/0.8853	31.61/0.8871	31.69/0.8882	31.64/0.8878	**31.84/0.8903**
4.6	31.31/0.8817	31.42/0.8842	31.51/0.8848	31.48/0.8848	**31.72/0.8881**
4.7	31.14/0.8789	31.22/0.8810	31.33/0.8819	31.29/0.8818	**31.54/0.8853**
4.8	30.95/0.8761	31.12/0.8793	31.16/0.8797	31.08/0.8792	**31.33/0.8824**
4.9	30.79/0.8717	30.95/0.8745	30.99/0.8751	30.94/0.8747	**31.16/0.8782**

Table 4. PSNR/SSIM values achieved by different methods with EDSR and RDN on Set14 datasets. The best results are in bold. The leftmost column represents a scale of magnification ranging from 2.1 to 4.9.

Dataset			**Set14**		
Method	EDSR-MetaSR	EDSR-LIIF	EDSR-LTE	EDSR-SRNO	EDSR-MambaSR
2.1	33.10/0.9126	33.17/0.9133	33.21/0.9138	33.27/0.9142	**33.56/0.9166**
2.2	32.65/0.9052	32.72/0.9059	32.76/0.9062	32.78/0.9066	**33.09/0.9093**
2.3	32.24/0.8968	32.31/0.8973	32.35/0.8978	32.37/0.8982	**32.70/0.9012**
2.4	31.89/0.8895	31.97/0.8902	32.01/0.8909	32.01/0.8911	**32.33/0.8939**
2.5	31.57/0.8817	31.62/0.8824	31.66/0.8830	31.69/0.8836	**31.99/0.8862**
2.6	31.34/0.8743	31.37/0.8754	31.41/0.8759	31.46/0.8764	**31.70/0.8787**
2.7	31.06/0.8672	31.10/0.8682	31.13/0.8685	31.19/0.8694	**31.41/0.8718**
2.8	30.83/0.8601	30.86/0.8612	30.90/0.8620	30.96/0.8627	**31.14/0.8650**
2.9	30.57/0.8527	30.60/0.8539	30.62/0.8546	30.71/0.8558	**30.87/0.8581**
3.1	30.13/0.8393	30.17/0.8409	30.21/0.8415	30.29/0.8427	**30.46/0.8459**
3.2	29.87/0.8325	29.92/0.8340	29.94/0.8345	30.02/0.8359	**30.21/0.8394**
3.3	29.70/0.8263	29.77/0.8282	29.76/0.8283	29.85/0.8297	**30.04/0.8336**
3.4	29.52/0.8201	29.56/0.8217	29.59/0.8223	29.67/0.8235	**29.86/0.8271**
3.5	29.35/0.8140	29.41/0.8159	29.42/0.8162	29.51/0.8175	**29.71/0.8212**
3.6	29.19/0.8085	29.25/0.8102	29.27/0.8109	29.36/0.8119	**29.52/0.8154**
3.7	29.03/0.8023	29.10/0.8046	29.13/0.8052	29.20/0.8061	**29.35/0.8091**
3.8	28.88/0.7977	28.95/0.7999	28.97/0.8006	29.05/0.8017	**29.20/0.8048**
3.9	28.70/0.7914	28.80/0.7939	28.82/0.7946	28.90/0.7956	**29.07/0.7994**
4.1	28.43/0.7812	28.50/0.7834	28.52/0.7841	28.60/0.7853	**28.78/0.7892**
4.2	28.28/0.7760	28.36/0.7783	28.39/0.7791	28.48/0.7806	**28.64/0.7843**
4.3	28.16/0.7708	28.24/0.7733	28.26/0.7740	28.36/0.7756	**28.55/0.7800**
4.4	28.04/0.7668	28.13/0.7693	28.14/0.7700	28.25/0.7715	**28.46/0.7762**
4.5	27.87/0.7615	27.96/0.7641	27.96/0.7645	28.07/0.7660	**28.30/0.7713**
4.6	27.75/0.7563	27.84/0.7588	27.87/0.7593	27.94/0.7607	**28.18/0.7667**
4.7	27.63/0.7514	27.72/0.7540	27.72/0.7543	27.81/0.7555	**28.05/0.7618**
4.8	27.54/0.7471	27.62/0.7494	27.63/0.7499	27.71/0.7511	**27.94/0.7575**
4.9	27.41/0.7430	27.49/0.7455	27.52/0.7461	27.61/0.7474	**27.83/0.7535**
Method	RDN-MetaSR	RDN-LIIF	RDN-LTE	RDN-SRNO	RDN-MambaSR
2.1	33.49/0.9168	33.47/0.9163	33.58/0.9168	33.55/0.9176	**33.67/0.9173**
2.2	33.03/0.9101	33.03/0.9094	33.10/0.9100	33.04/0.9099	**33.17/0.9099**
2.3	32.60/0.9013	32.66/0.9012	32.70/0.9016	32.59/0.9013	**32.74/0.9013**
2.4	32.26/0.8940	32.25/0.8936	32.31/0.8944	32.26/0.8941	**32.37/0.8940**
2.5	31.87/0.8861	31.87/0.8856	31.91/0.8863	31.89/0.8862	**32.01/0.8865**
2.6	31.60/0.8784	31.64/0.8783	31.66/0.8789	31.64/0.8789	**31.74/0.8793**
2.7	31.35/0.8715	31.36/0.8713	31.39/0.8720	31.39/0.8720	**31.42/0.8723**
2.8	31.08/0.8647	31.06/0.8645	31.12/0.8651	31.12/0.8652	**31.18/0.8656**
2.9	30.80/0.8577	30.80/0.8575	30.84/0.8582	30.84/0.8583	**30.94/0.8590**
3.1	30.39/0.8451	30.37/0.8450	30.40/0.8458	30.45/0.8459	**30.51/0.8466**
3.2	30.10/0.8381	30.09/0.8380	30.14/0.8388	30.15/0.8389	**30.26/0.8402**
3.3	29.95/0.8322	29.92/0.8318	29.97/0.8328	29.99/0.8330	**30.07/0.8342**
3.4	29.73/0.8258	29.73/0.8256	29.77/0.8265	29.79/0.8266	**29.92/0.8284**
3.5	29.58/0.8199	29.58/0.8198	29.63/0.8209	29.64/0.8206	**29.75/0.8224**
3.6	29.43/0.8141	29.38/0.8139	29.44/0.8150	29.46/0.8149	**29.58/0.8167**
3.7	29.26/0.8084	29.23/0.8080	29.29/0.8091	29.29/0.8089	**29.40/0.8107**
3.8	29.11/0.8039	29.09/0.8039	29.14/0.8050	29.13/0.8047	**29.25/0.8060**
3.9	28.96/0.7981	28.96/0.7983	29.00/0.7993	29.01/0.7991	**29.11/0.8005**
4.1	28.69/0.7881	28.69/0.7882	28.75/0.7894	28.75/0.7891	**28.84/0.7906**
4.2	28.55/0.7832	28.57/0.7837	28.62/0.7847	28.62/0.7844	**28.71/0.7859**
4.3	28.45/0.7786	28.45/0.7790	28.51/0.7802	28.51/0.7799	**28.62/0.7811**
4.4	28.35/0.7747	28.36/0.7750	28.43/0.7763	28.40/0.7758	**28.51/0.7776**
4.5	28.16/0.7694	28.18/0.7703	28.12/0.7697	28.22/0.7708	**28.34/0.7726**
4.6	28.04/0.7645	28.07/0.7654	27.99/0.7651	28.12/0.7660	**28.25/0.7683**
4.7	27.91/0.7599	27.94/0.7608	27.87/0.7602	28.01/0.7617	**28.11/0.7632**
4.8	27.80/0.7551	27.85/0.7566	27.78/0.7557	27.89/0.7569	**28.00/0.7588**
4.9	27.65/0.7510	27.71/0.7525	27.72/0.7528	27.75/0.7522	**27.89/0.7553**

To further illustrate the effectiveness of our proposed MambaSR model, the visual results of the super-resolution images reconstructed by different methods are presented in Figures 4–11. The red square in the figure represents the position of the details that have been cropped from the super-resolution image. The best results have been marked in red and the upward arrow in the figure indicates that higher values correspond to better quality. Figures 4 and 5 show the visual results of MambaSR and other models based on EDSR for the Urban100 dataset, where MambaSR demonstrates superior detail preservation and texture reconstruction in urban scenes, capturing intricate structural elements more effectively in images like "img004.png" and "img015.png". Figure 6 presents the visual results for the Manga109 dataset using EDSR, where MambaSR excels in maintaining line clarity and edge sharpness in manga illustrations, as seen in "img011.png", avoiding the blurring common with other methods.

Figure 7 displays results from the Set14 dataset using EDSR, with the "barbara.png" image highlighting MambaSR's robustness in handling natural images by effectively reconstructing fine textures and minimizing artifacts. In Figure 8, showing the results for the Set5 dataset using EDSR, the "woman.png" image illustrates MambaSR's ability to maintain high visual quality and sharpness across different visual contexts. Figure 9 provides visual results for the Urban100 dataset using RDN, where MambaSR continues to show its superiority in urban scenes, preserving fine details and structural elements better than other models in "img096.png". Figure 10 demonstrates the model's performance on the Manga109 dataset using RDN, with MambaSR maintaining line clarity and edge sharpness in manga illustrations like "img060.png", outperforming other models. Finally, Figure 11 presents the results for the Set14 dataset using RDN, where MambaSR delivers superior visual quality in the "zebra.png" image, providing clear textures and minimizing artifacts more effectively than other methods. The presented visual results demonstrate MambaSR's capacity to reconstruct high-quality super-resolution images across a range of datasets and contexts.

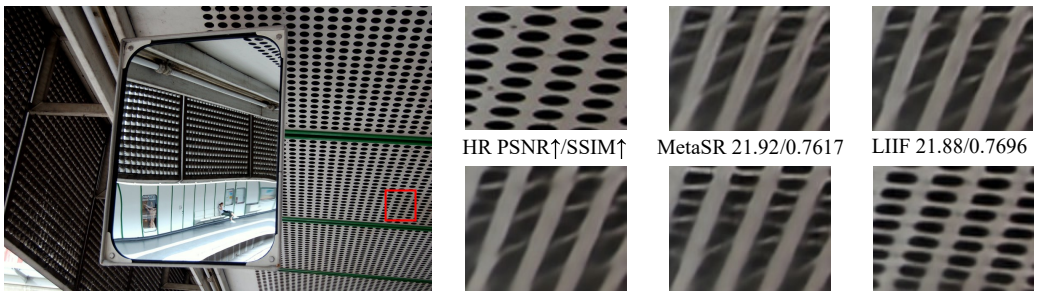

Figure 4. Comparisons with different methods on the Urban100 dataset using EDSR as the encoder.

Figure 5. Comparisons with different methods on the Urban100 dataset using EDSR as the encoder.

Figure 6. Comparisons with different methods on the Manga109 dataset using EDSR as the encoder.

Figure 7. Comparisons with different methods on the Set14 dataset using EDSR as the encoder.

Figure 8. Comparisons with different methods on the Set5 dataset using EDSR as the encoder.

Figure 9. Comparisons with different methods on the Urban100 dataset using RDN as the encoder.

Figure 10. Comparisons with different methods on the Manga109 dataset using RDN as the encoder.

Figure 11. Comparisons with different methods on the Set14 dataset using RDN as the encoder.

4.4. Ablation Study

To evaluate the impact of the FFTConv block in our proposed MambaSR model, we conducted ablation experiments, as shown in Table 5. The experiments were conducted on four benchmark datasets: Set5, Set14, Urban100, and Manga109. We compared the performance of the full MambaSR model with a variant where the FFTConv block was removed. The performance metrics used for evaluation were the PSNR and SSIM. The inclusion of the FFTConv block consistently improved the performance across all datasets. Specifically, the PSNR increased by 0.10 dB on Set5, 0.04 dB on Set14, 0.11 dB on Urban100, and 0.22 dB on Manga109. Similarly, the SSIM saw improvements of 0.0010, 0.0011, 0.0032, and 0.0019 respectively. These results underscore the importance of the FFTConv block in our architecture, validating its role in achieving superior super-resolution performance by effectively leveraging both spatial and frequency domain information

Table 5. Ablation experiments for FFTConv.

Method	Set5	Set14	Urban100	Manga109
MambaSR + w/o FFTConv	32.23/0.8977	28.60/0.7832	26.44/0.7955	30.62/0.9108
MambaSR	32.33/0.8987	28.64/0.7843	26.55/0.7987	30.84/0.9127

5. Discussion

The results presented in this paper highlight the significant progress that has been made by MambaSR in the field of arbitrary-scale super-resolution (ASSR). The Mamba state-space model and Fast Fourier Convolution (FFC) blocks are utilized in MambaSR to effectively address several inherent limitations of traditional SISR methods. The integration of Mamba facilitates the extraction of long-range dependencies, which are important for preserving intricate details and textures across varying scales. Moreover, the FFC blocks adeptly handle global frequency domain information, enhancing the overall reconstruction quality.

One of the standout features of MambaSR is its ability to perform well across different datasets, including Urban100 and Manga109, where it demonstrates a clear superiority

over existing methods such as MetaSR and LIIF. The performance gains, particularly the improvement in PSNR and SSIM values, highlight the model's robustness. These improvements can be attributed to the innovative combination of spatial and frequency domain processing, which allows MambaSR to maintain a high quality in image reconstruction regardless of the scale factor.

Moreover, the proposed Residual Fast Fourier Transform State-Space Block (RFFTSSB) plays a pivotal role in enhancing feature representation by seamlessly integrating spatial and frequency domain information. This dual-domain approach ensures that the enhanced features retain critical contextual information, leading to superior visual and quantitative results. The ablation studies further validate the effectiveness of the RFFTSSB, confirming its contribution to the overall performance of MambaSR.

Additionally, a comparative analysis of Normalized Cross Correlation (NCC) and Normalized Absolute Error (NAE) across four datasets (Set5, Set14, Urban100, and Manga109) at scaling factors of 2, 3, and 4 revealed the superior performance of MambaSR. As highlighted in Table 6, MambaSR consistently achieved the best performance metrics. NCC values ranged from -1 to 1, where higher values indicate better similarity between the super-resolved and original images. NAE values ranged from 0 to infinity, where lower values indicate smaller differences between the super-resolved and original images. Notably, MambaSR's ability to achieve higher NCC and lower NAE across all tested datasets and scaling factors underscores its robustness and efficacy. These results further substantiate the model's advantage in maintaining a high reconstruction quality and accurate feature representation across various conditions.

Table 6. Comparison of Normalized Cross Correlation (NCC) and Normalized Absolute Error (NAE) on four datasets: Set5, Set14, Urban100, and Manga109, at scaling factors of 2, 3, and 4. The best performance for each metric is highlighted in bold. In the table, the upward arrow indicates that a higher value corresponds to better performance, while the downward arrow indicates that a lower value corresponds to better performance.

Dataset	Scale	RDN-MetaSR		RDN-LIIF		RDN-LTE		RDN-SRNO		RDN-MambaSR	
		NCC↑	NAE↓	NCC↑	NAE↓	NCC↑	NAE↓	NCC↑	NAE↓	NCC↑	NAE↓
Set5	2	0.9981	0.0234	0.9981	0.0235	0.9981	0.0232	0.9981	0.0232	**0.9982**	**0.0230**
	3	0.9959	0.0332	0.9959	0.0333	0.9959	0.0329	0.9959	0.0330	**0.9960**	**0.0327**
	4	0.9929	0.0421	0.9931	0.0415	0.9933	0.0412	0.9932	0.0413	**0.9935**	**0.0406**
Set14	2	0.9900	0.0374	0.9898	0.0376	0.9898	0.0375	0.9901	0.0372	**0.9902**	**0.0371**
	3	0.9793	0.0536	0.9792	0.0537	0.9794	0.0534	0.9796	0.0532	**0.9798**	**0.0529**
	4	0.9705	0.0650	0.9704	0.0649	0.9708	0.0644	0.9707	0.0646	**0.9713**	**0.0639**
Urban100	2	0.9899	0.0407	0.9899	0.0407	0.9902	0.0400	0.9901	0.0401	**0.9906**	**0.0393**
	3	0.9768	0.0625	0.9766	0.0624	0.9773	0.0615	0.9772	0.0617	**0.9781**	**0.0604**
	4	0.9636	0.0804	0.9638	0.0792	0.9646	0.0780	0.9649	0.0780	**0.9661**	**0.0765**
Manga109	2	0.9981	0.0141	0.9981	0.0142	0.9981	0.0140	0.9981	0.0141	**0.9982**	**0.0137**
	3	0.9944	0.0228	0.9943	0.0231	0.9945	0.0227	0.9945	0.0227	**0.9948**	**0.0221**
	4	0.9890	0.0312	0.9890	0.0310	0.9892	0.0305	0.9892	0.0307	**0.9898**	**0.0298**

Furthermore, in a comprehensive performance evaluation among different super-resolution models on the Urban100 dataset using identical hardware configurations, as shown in Table 7, the EDSR-MambaSR model demonstrated superior efficacy. The experiments were conducted on a server equipped with an NVIDIA V100 GPU with 32 GB memory, 640 GB RAM, and 80 CPU cores. In particular, the EDSR-MambaSR method demonstrated the highest PSNR at 26.90 dB, indicating a markedly superior reconstruction quality in comparison to the other models. Although its runtime of 40.57 s was not the shortest, the model effectively balanced high-quality output with reasonable computational demands, surpassing other models like EDSR-LIIF and EDSR-LTE, which exhibited longer runtimes and lower PSNR values.

Table 7. Performance comparison of different super-resolution models on the Urban100 dataset at scaling factors of 4.

	EDSR-MetaSR	EDSR-LIIF	EDSR-LTE	EDSR-SRNO	EDSR-MambaSR
PSNR on Urban100 (dB)	25.95	26.15	26.24	26.41	26.90
Runtime (s)	13.15	37.87	48.30	29.73	40.57

In particular, the performance comparison of different super-resolution models on the Urban100 dataset, processed with Gaussian blur (kernel size 5×5, standard deviation 0.5), Gaussian noise (standard deviation 0.08), and bicubic downsampling at a scaling factor of 4, indicates that MambaSR consistently outperformed other models shown in Table 8. It achieved the highest PSNR and SSIM values across all scaling factors, demonstrating superior performance in image super-resolution under challenging conditions. This emphasizes the robustness and effectiveness of the MambaSR model in handling degraded image inputs, which more closely resemble real-world images.

Table 8. Performance comparison of different super-resolution models on the Urban100 dataset (processed with Gaussian blur, Gaussian noise, and bicubic downsampling) at scaling factors of 2, 3, and 4. The best performance for each metric is highlighted in bold. In the table, the upward arrow indicates that a higher value corresponds to better performance.

Dataset	Scale	EDSR-MetaSR		EDSR-LIIF		EDSR-LTE		EDSR-SRNO		EDSR-MambaSR	
		PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
Urban100	2	22.19	0.4540	22.24	0.4602	22.27	0.4587	22.13	0.4519	**22.39**	**0.4633**
	3	21.15	0.4046	21.04	0.3986	21.16	0.4048	21.00	0.3943	**21.27**	**0.4074**
	4	20.40	0.3763	20.44	0.3785	20.42	0.3771	20.38	0.3730	**20.52**	**0.3793**

6. Conclusions

In this paper, we introduce MambaSR, a pioneering approach to arbitrary-scale super-resolution leveraging the innovative Mamba state-space model combined with Fast Fourier Convolution Blocks (FFTConv). MambaSR addresses the challenges of traditional SISR methods by enabling flexible and continuous scaling factors, which provide a more versatile solution for real-world applications. The core innovation lies in Mamba's ability to dynamically represent features and capture long-range dependencies through efficient parallel processing.

Our extensive experiments on benchmark datasets such as Set5, Set14, Urban100, and Manga109 validated the superior performance of MambaSR over existing advanced methods. Specifically, MambaSR demonstrated a notable PSNR improvement of 0.93 dB and an SSIM enhancement of 0.0203 dB on the Urban100 dataset compared to MetaSR. On the Manga109 dataset, MambaSR achieved an average PSNR increase of 1.00 dB and an SSIM improvement of 0.0093 dB, underscoring its effectiveness in producing high-quality super-resolved images.

The integration of the FFTConv block further enhances MambaSR's capability by effectively combining spatial and frequency domain information, resulting in improved feature representation and image reconstruction quality. This study not only showcases the potential of Mamba in advancing the field of arbitrary-scale super-resolution but also sets the stage for future research to optimize and extend the application of the MambaSR architecture across diverse domains.

Future work will focus on refining the MambaSR architecture to further improve its efficiency and exploring its application in other areas, such as medical imaging and video surveillance, where high-quality image reconstruction is critical.

Author Contributions: Conceptualization, J.Y. and Z.C.; methodology, J.Y. and Z.P.; writing—original draft: J.Y.; formal analysis, Z.P.; software: J.Y.; writing—review and editing: J.Y., Z.C., Z.P., X.L. and H.Z.; supervision, X.L.; funding acquisition, X.L. and H.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Science and Technology Development Fund, Macau SAR (No. 0096/2022/A), Basic and Applied Basic Research Foundation of Guangdong (No. 2024A1515011822), Scientific Computing Research Innovation Team of Guangdong Province (No. 2021KCXTD052), Guangdong Key Construction Discipline Research Capacity Enhancement Project (No. 2022ZDJS049) and Technology Planning Project of Shaoguan (No. 230330108034184).

Data Availability Statement: The MambaSR model is available at https://github.com/ttys0001/mambasr.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Hijji, M.; Khan, A.; Alwakeel, M.M.; Harrabi, R.; Aradah, F.; Cheikh, F.A.; Sajjad, M.; Muhammad, K. Intelligent Image Super-Resolution for Vehicle License Plate in Surveillance Applications. *Mathematics* **2023**, *11*, 892. [CrossRef]
2. Kim, M.H.; Yoo, S.B. Memory-Efficient Discrete Cosine Transform Domain Weight Modulation Transformer for Arbitrary-Scale Super-Resolution. *Mathematics* **2023**, *11*, 3954. [CrossRef]
3. Singh, N.; Rathore, S.S.; Kumar, S. Towards a super-resolution based approach for improved face recognition in low resolution environment. *Multimed. Tools Appl.* **2022**, *81*, 38887–38919. [CrossRef] [PubMed]
4. Zhu, D.; Qiu, D. Residual dense network for medical magnetic resonance images super-resolution. *Comput. Methods Programs Biomed.* **2021**, *209*, 106330. [CrossRef] [PubMed]
5. Zhao, X.; Zhang, Y.; Zhang, T.; Zou, X. Channel splitting network for single MR image super-resolution. *IEEE Trans. Image Process.* **2019**, *28*, 5649–5662. [CrossRef]
6. Lu, T.; Wang, J.; Zhang, Y.; Wang, Z.; Jiang, J. Satellite image super-resolution via multi-scale residual deep neural network. *Remote Sens.* **2019**, *11*, 1588. [CrossRef]
7. Lucas, A.; Lopez-Tapia, S.; Molina, R.; Katsaggelos, A.K. Generative adversarial networks and perceptual losses for video super-resolution. *IEEE Trans. Image Process.* **2019**, *28*, 3312–3327. [CrossRef] [PubMed]
8. Yang, C.Y.; Ma, C.; Yang, M.H. Single-image super-resolution: A benchmark. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Proceedings, Part IV 13; Springer: Berlin/Heidelberg, Germany, 2014; pp. 372–386.
9. Irani, M.; Peleg, S. Improving resolution by image registration. *CVGIP Graph. Model. Image Process.* **1991**, *53*, 231–239. [CrossRef]
10. Fattal, R. Image upsampling via imposed edge statistics. In *ACM SIGGRAPH 2007 Papers*; Association for Computing Machinery: New York, NY, USA, 2007; pp. 95–es.
11. Huang, J.; Mumford, D. Statistics of natural images and models. In Proceedings of the 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149), IEEE, Fort Collins, CO, USA, 23–25 June 1999; Volume 1, pp. 541–547.
12. Sirota, A.; Ivankov, A. Block algorithms of image processing based on kalman filter for superresolution reconstruction. *Comput. Opt.* **2014**, *38*, 118–126. [CrossRef]
13. Freeman, W.T.; Jones, T.R.; Pasztor, E.C. Example-based super-resolution. *IEEE Comput. Graph. Appl.* **2002**, *22*, 56–65. [CrossRef]
14. Chang, H.; Yeung, D.Y.; Xiong, Y. Super-resolution through neighbor embedding. In Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2004, IEEE, Washington, DC, USA, 27 June–2 July 2004; Volume 1, p. 1.
15. Yang, J.; Lin, Z.; Cohen, S. Fast image super-resolution based on in-place example regression. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 1059–1066.
16. Dong, C.; Loy, C.C.; He, K.; Tang, X. Image super-resolution using deep convolutional networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *38*, 295–307. [CrossRef] [PubMed]
17. Zhang, Y.; Huang, Y.; Wang, K.; Qi, G.; Zhu, J. Single image super-resolution reconstruction with preservation of structure and texture details. *Mathematics* **2023**, *11*, 216. [CrossRef]
18. Hu, X.; Mu, H.; Zhang, X.; Wang, Z.; Tan, T.; Sun, J. Meta-SR: A magnification-arbitrary network for super-resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 1575–1584.
19. Chen, Y.; Liu, S.; Wang, X. Learning continuous image representation with local implicit image function. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 8628–8638.
20. Yue, Y.; Li, Z. Medmamba: Vision mamba for medical image classification. *arXiv* **2024**, arXiv:2403.03849.
21. Gu, A.; Goel, K.; Ré, C. Efficiently modeling long sequences with structured state spaces. *arXiv* **2021**, arXiv:2111.00396.
22. Gu, A.; Johnson, I.; Goel, K.; Saab, K.; Dao, T.; Rudra, A.; Ré, C. Combining recurrent, convolutional, and continuous-time models with linear state space layers. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2021; Volume 34, pp. 572–585.
23. Gu, A.; Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv* **2023**, arXiv:2312.00752.
24. Lee, J.; Jin, K.H. Local texture estimator for implicit representation function. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 1929–1938.

25. Wei, M.; Zhang, X. Super-resolution neural operator. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 18247–18256.
26. Skopec, P.; Vyhlídal, T.; Knobloch, J. Development of a continuous reheating furnace state-space model based on the finite volume method. *Appl. Therm. Eng.* **2024**, *246*, 122888. [CrossRef]
27. Borowska, A.; King, R. Semi-complete data augmentation for efficient state space model fitting. *J. Comput. Graph. Stat.* **2023**, *32*, 19–35. [CrossRef]
28. Qiu, L.; Fan, S.; Liao, S.; Sun, P.; Wei, X. State space modelling development of Micro-High-Temperature Gas-Cooled reactor with helium Brayton cycle. *Ann. Nucl. Energy* **2024**, *197*, 110284. [CrossRef]
29. Yeganeh, A.; Johannssen, A.; Chukhrova, N.; Rasouli, M. Monitoring multistage healthcare processes using state space models and a machine learning based framework. *Artif. Intell. Med.* **2024**, *151*, 102826. [CrossRef]
30. Smith, J.T.; Warrington, A.; Linderman, S.W. Simplified state space layers for sequence modeling. *arXiv* **2022**, arXiv:2208.04933.
31. Mehta, H.; Gupta, A.; Cutkosky, A.; Neyshabur, B. Long range language modeling via gated state spaces. *arXiv* **2022**, arXiv:2206.13947.
32. Liu, Y.; Tian, Y.; Zhao, Y.; Yu, H.; Xie, L.; Wang, Y.; Ye, Q.; Liu, Y. Vmamba: Visual state space model. *arXiv* **2024**, arXiv:2401.10166.
33. Zhu, L.; Liao, B.; Zhang, Q.; Wang, X.; Liu, W.; Wang, X. Vision mamba: Efficient visual representation learning with bidirectional state space model. *arXiv* **2024**, arXiv:2401.09417.
34. Ma, J.; Li, F.; Wang, B. U-mamba: Enhancing long-range dependency for biomedical image segmentation. *arXiv* **2024**, arXiv:2401.04722.
35. Xing, Z.; Ye, T.; Yang, Y.; Liu, G.; Zhu, L. Segmamba: Long-range sequential modeling mamba for 3d medical image segmentation. *arXiv* **2024**, arXiv:2401.13560.
36. Ruan, J.; Xiang, S. Vm-unet: Vision mamba unet for medical image segmentation. *arXiv* **2024**, arXiv:2402.02491.
37. Liu, J.; Yang, H.; Zhou, H.Y.; Xi, Y.; Yu, L.; Yu, Y.; Liang, Y.; Shi, G.; Zhang, S.; Zheng, H.; et al. Swin-umamba: Mamba-based unet with imagenet-based pretraining. *arXiv* **2024**, arXiv:2402.03302.
38. Islam, M.M.; Hasan, M.; Athrey, K.S.; Braskich, T.; Bertasius, G. Efficient movie scene detection using state-space transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 18749–18758.
39. Nguyen, E.; Goel, K.; Gu, A.; Downs, G.; Shah, P.; Dao, T.; Baccus, S.; Ré, C. S4nd: Modeling images and videos as multidimensional signals with state spaces. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2022; Volume 35, pp. 2846–2861.
40. Yamashita, S.; Ikehara, M. Image Deraining with Frequency-Enhanced State Space Model. *arXiv* **2024**, arXiv:2405.16470.
41. Lim, B.; Son, S.; Kim, H.; Nah, S.; Mu Lee, K. Enhanced deep residual networks for single image super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Honolulu, HI, USA, 21–26 July 2017; pp. 136–144.
42. Zhang, Y.; Tian, Y.; Kong, Y.; Zhong, B.; Fu, Y. Residual dense network for image super-resolution. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2472–2481.
43. Wang, Q.; Wu, B.; Zhu, P.; Li, P.; Zuo, W.; Hu, Q. ECA-Net: Efficient channel attention for deep convolutional neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11534–11542.
44. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
45. Agustsson, E.; Timofte, R. Ntire 2017 challenge on single image super-resolution: Dataset and study. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Honolulu, HI, USA, 21–26 July 2017; pp. 126–135.
46. Bevilacqua, M.; Roumy, A.; Guillemot, C.; Alberi-Morel, M.L. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In Proceedings of the 23rd British Machine Vision Conference (BMVC), London, UK, 3–7 September 2012.
47. Zeyde, R.; Elad, M.; Protter, M. On single image scale-up using sparse-representations. In Proceedings of the Curves and Surfaces: 7th International Conference, Avignon, France, 24–30 June 2010; Revised Selected Papers 7; Springer: Berlin/Heidelberg, Germany, 2012; pp. 711–730.
48. Huang, J.B.; Singh, A.; Ahuja, N. Single image super-resolution from transformed self-exemplars. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5197–5206.
49. Matsui, Y.; Ito, K.; Aramaki, Y.; Fujimoto, A.; Ogawa, T.; Yamasaki, T.; Aizawa, K. Sketch-based manga retrieval using manga109 dataset. *Multimed. Tools Appl.* **2017**, *76*, 21811–21838. [CrossRef]