Supplementary Information for

The file contains:

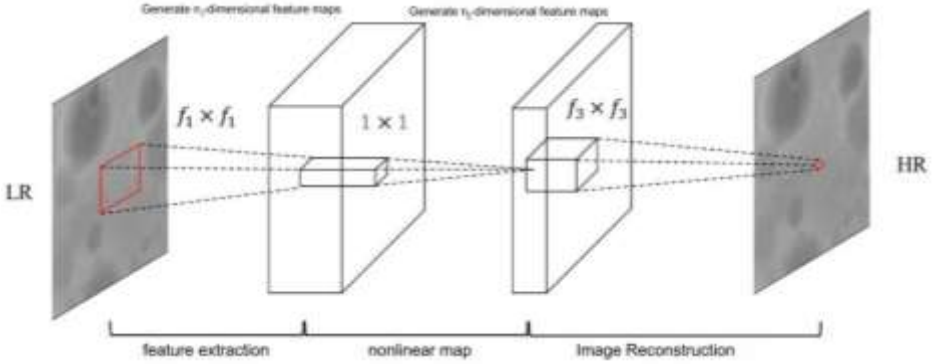
Supplementary Methods

Supplementary Tables

Supplementary Methods

A. Super-Resolution Convolutional Neural Network（SRCNN）

SRCNN is a pioneering deep learning-based super-resolution (SR) method that directly learns an end-to-end mapping between low-resolution (LR) and high-resolution (HR) images[1]. This approach employs a lightweight convolutional neural network (CNN) that captures feature relationships between LR-HR image pairs, optimizing all network parameters jointly to generate a model capable of rapid image restoration. SRCNN represents a significant advancement in applying deep learning to the SR domain, demonstrating profound impacts on image resolution enhancement. The network architecture is illustrated in Supplementary Fig. 1.



Supplementary Fig. 1 Network Architecture of SRCNN

SRCNN features a simple architecture with only three convolutional layers, employing kernel sizes of 9×99, 1×11, and 5×55, respectively. The input LR image undergoes bicubic downsampling and subsequent upsampling to obtain a preprocessed version before entering the network. The three convolutional layers are designed to learn a mapping function that reconstructs an output image closely resembling the HR counterpart.

Feature Extraction Layer:This layer extracts overlapping feature patches from the LR image and represents them as high-dimensional vectors, where the number of feature maps corresponds to the vector dimensions. The convolution operation at this stage is formulated as:

$$F_1(Y) = \max(0, w_1 * Y + b_1) \tag{1}$$

Where w_1 represents the parameters of the convolutional kernel, the convolutional layer contains n_1 kernels, each with a spatial size of f_1 and c channels, resulting in a kernel size of $c \times f_1 \times f_1$. $*$ denotes the convolution operation, and b_1 is a bias vector with a dimensionality of n_1.

Non-Linear Mapping Layer:This layer transforms the extracted high-dimensional feature vectors into another set of vectors through non-linear mappings. It plays a crucial role in feature abstraction and representation learning. The operation is defined as:

$$F_2(Y) = \max(0, w_2 * F_1(Y) + b_2) \tag{2}$$

Where w_2 represents the parameters of n_2 convolutional kernels, each with a size of f_2 and c channels, resulting in a kernel size of $c \times f_2 \times f_2$. b_2 is a bias vector with a dimensionality of n_2.

Reconstruction Layer:The final layer aggregates the transformed HR feature patches and reconstructs the final SR image. The operation follows:

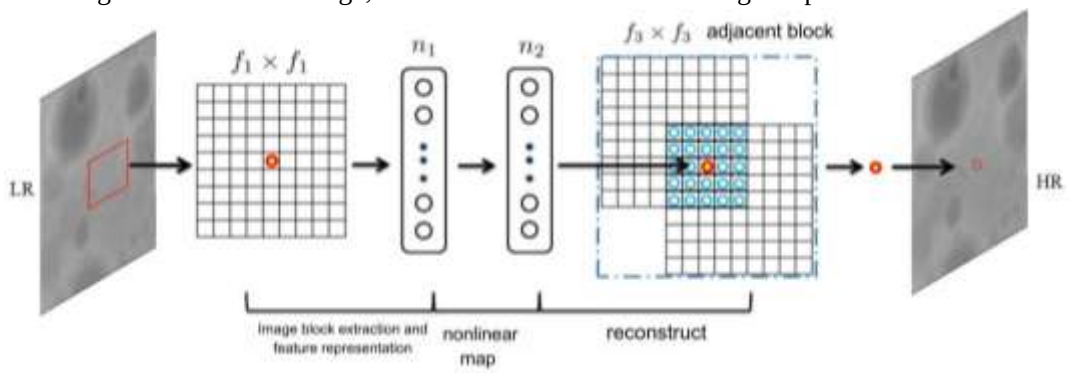
$$F(Y) = w_3 * F_2(Y) + b_3 \tag{3}$$

Where w_3 represents the parameters of c convolutional kernels, each with a size of $n_2 \times f_3 \times f_3$, and b_3 is a bias vector with a dimensionality of c.

To enhance the peak signal-to-noise ratio (PSNR), SRCNN employs a mean squared error (MSE) loss function:

$$l(\Theta) = \frac{1}{n}\sum_{1}^{n} \| F(Y_i; \Theta) - X_i \|^2 \tag{4}$$

where $\Theta = \{w_1, w_2, w_3, b_1, b_2, b_3\}$ denotes network parameters, $F(Y; \Theta)$ represents the SR image, X is the ground-truth HR image, and n is the number of training samples.

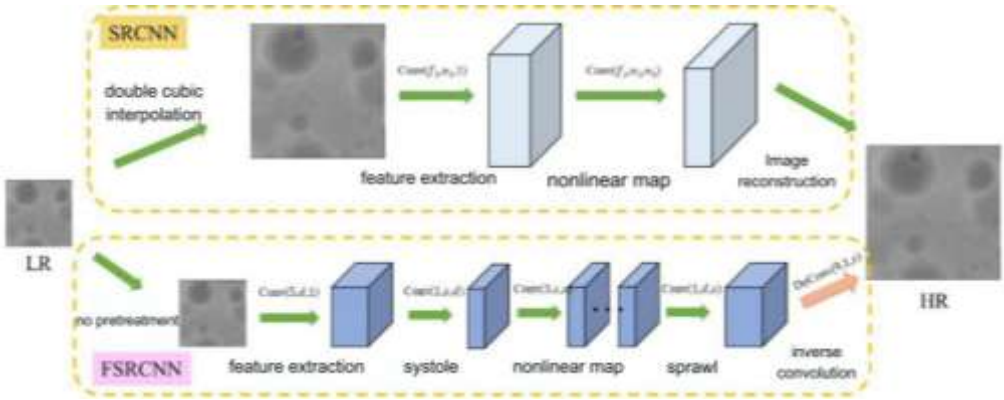


Supplementary Fig. 2 SRCNN Simulation of Sparse Coding-Based SR Method

SRCNN is conceptually akin to sparse-coding-based SR methods[2], as depicted in Supplementary Fig. 2. Unlike traditional sparse-coding-based approaches that require manual selection and optimization of parameters, SRCNN learns optimal parameters directly from training data, thereby improving model performance and adaptability.

B. Fast Super-Resolution Convolutional Neural Network（FSRCNN）

Although SRCNN achieves notable performance in SR tasks, its computational cost remains a bottleneck, restricting real-time applications. To address this limitation, FSRCNN was introduced by Dong et al. in 2016[3]. FSRCNN adopts a funnel-shaped CNN structure, significantly reducing model complexity while improving computational efficiency. The network architecture is shown in Supplementary Fig. 3.



Supplementary Fig. 3 Network Architecture of FSRCNN

Key Improvements Over SRCNN:

Elimination of the Bicubic Preprocessing Step: Unlike SRCNN, which requires an initial bicubic interpolation step, FSRCNN integrates a deconvolution layer at the network's end to directly learn the LR-to-HR mapping.

Efficient Feature Representation: FSRCNN consists of five key components: feature extraction, shrinking, mapping, expanding, and deconvolution. The shrinking and expanding layers effectively reduce and restore feature dimensionality, minimizing computational overhead while maintaining feature expressiveness.

Reduced Model Complexity: FSRCNN replaces the single-wide mapping layer in SRCNN with multiple narrower layers, further decreasing the number of parameters. Additionally, it employs smaller and fewer convolutional kernels to eliminate redundant parameters. The computational complexity of SRCNN and FSRCNN is given by:

$$C_{SRCNN} = O\{(n_1 f_1^2 + n_1 n_2 f_2^2 + n_2 f_3^2) Size_{HR}\} \quad (5)$$

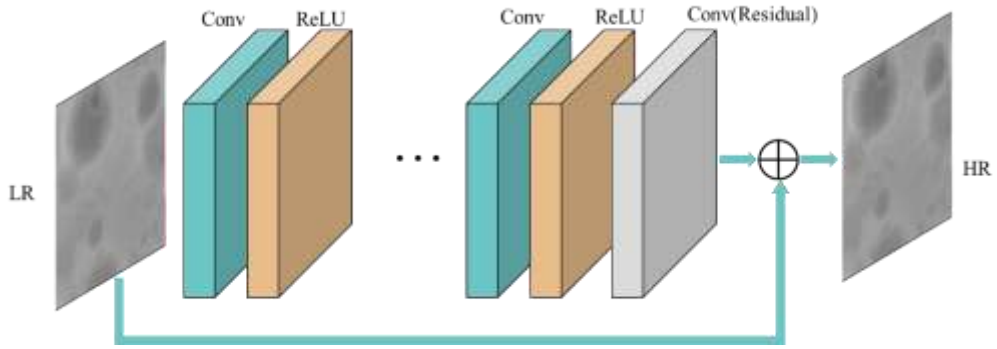
$$C_{FSRCNN} = O\{(f_1^2 dc + m f_2^2 dd + f_3^2 sd) Size_{HR}\} \quad (6)$$

Where $\{f_i\}_{i=1,2,3}$ and $\{n_i\}_{i=1,2,3}$ represent the kernel sizes and the number of kernels in the three layers of SRCNN, respectively, $Size_{HR}$ is the size of the HR image, m is the depth of the mapping layer, s is the number of kernels in the shrinking layer, and d is the dimensionality of the LR features.

To further enhance performance, FSRCNN can replace the final deconvolution layer with a pixel-shuffle operation (PixelShuffle), which reorganizes feature maps into spatial dimensions, significantly improving efficiency and mitigating checkerboard artifacts associated with standard deconvolutions.

C. Very Deep Super-Resolution Network (VDSR)

Inspired by the VGG-Net architecture developed by Oxford and DeepMind in 2014, VDSR network employs a deeper structure to enhance SR performance[4]. Unlike SRCNN, which is limited to extracting local contextual information, VDSR leverages a larger receptive field to capture global information while employing residual learning and gradient clipping to accelerate convergence. The network structure is illustrated in Supplementary Fig. 4.



Supplementary Fig. 4 Network Architecture of VDSR

In the VDSR architecture, all layers feature a uniform size of 3×3×64, except for the first and last layers, which each have a single filter. The first layer processes the bicubic-interpolated LR image, while the final layer reconstructs the HR image. The intermediate layers are responsible for predicting image details.

Convolution operations typically reduce the size of feature maps. For example, when a feature map of size $(n+1)\times(n+1)$ is passed through a convolutional layer with a receptive field of $n\times n$, the output size will be 1×1, where the operation uses surrounding pixels to infer the central pixel value. However, this process cannot utilize pixels beyond the boundary of the image, leading to the need to crop the resulting feature map. This cropping, however, reduces the final image size, resulting in diminished visual quality. This issue becomes more pronounced in deep networks that predict dense outputs. To address this, VDSR applies zero-padding before the convolution operation to ensure that all feature maps retain consistent sizes, thereby allowing accurate predictions even near image boundaries.

For training input image X, with output Y MSE loss function is formulated as:

$$L = \frac{1}{2} \| Y - F(X) \|^2 \quad (7)$$

where $F(X)$ denotes the network's mapping function. VDSR incorporates residual learning by defining the residual image $R = Y - X$, transforming the prediction task into predicting the residual image. Consequently, the loss function becomes:

$$L = \frac{1}{2} \| R - F(X) \|^2 \quad (8)$$

In the network structure, residual estimation, LR images, and HR images are input into the loss layer. To address the issue of gradient explosion, gradient clipping is applied during training, which is common in many neural network training processes[5]. For stochastic gradient clipping, the step size is adjusted by multiplying the gradient by the learning rate, reducing its size to

prevent explosive gradients due to high learning rates. VDSR clips the gradients to a value of $[-\theta/\gamma, \theta/\gamma]$, where γ represents the current learning rate. As a result, the 20-layer VDSR converges much faster—approximately ten times faster—than the 3-layer SRCNN.

Supplementary Tables

SUPPLEMENTARY TABLE 1

PSNR(DB)/SSIM VALUES FOR 20 TEST IMAGES

No.	Bicubic	SRCNN	FSRCNN	VDSR	EDSR
01	40.60/0.9675	46.13/0.9902	46.40/0.9905	46.78/0.9910	46.97/0.9912
02	40.62/0.9686	47.24/0.9921	47.56/0.9924	47.84/0.9927	47.99/0.9928
03	40.47/0.9674	46.85/0.9907	47.04/0.9912	47.55/0.9918	47.62/0.9920
04	40.01/0.9633	45.42/0.9881	45.52/0.9889	45.96/0.9894	46.14/0.9896
05	40.10/0.9640	45.61/0.9880	45.71/0.9889	46.10/0.9894	46.30/0.9898
06	40.69/0.9695	46.19/0.9899	46.43/0.9904	46.90/0.9910	46.94/0.9910
07	40.14/0.9657	45.40/0.9881	45.55/0.9889	45.91/0.9894	46.11/0.9897
08	40.29/0.9668	45.83/0.9890	46.04/0.9898	46.35/0.9901	46.53/0.9904
09	40.38/0.9663	45.44/0.9888	45.48/0.9894	46.09/0.9900	46.20/0.9902
10	40.27/0.9664	46.01/0.9896	46.18/0.9902	46.63/0.9908	46.75/0.9909
11	40.15/0.9649	45.74/0.9892	45.93/0.9898	46.37/0.9904	46.47/0.9906
12	40.34/0.9661	45.81/0.9894	45.97/0.9900	46.38/0.9904	46.50/0.9906
13	40.38/0.9655	45.84/0.9889	45.97/0.9897	46.37/0.9901	46.51/0.9903
14	40.04/0.9654	45.13/0.9877	45.32/0.9885	45.68/0.9890	45.80/0.9893
15	40.09/0.9672	45.46/0.9885	45.60/0.9892	46.00/0.9896	46.09/0.9897
16	40.11/0.9646	45.79/0.9889	45.95/0.9896	46.39/0.9901	46.54/0.9904
17	40.24/0.9642	45.20/0.9878	45.36/0.9886	45.74/0.9890	45.95/0.9894
18	40.32/0.9640	45.67/0.9883	45.78/0.9891	46.18/0.9895	46.37/0.9901
19	40.03/0.9627	45.14/0.9874	45.29/0.9882	45.71/0.9887	45.87/0.9890
20	40.53/0.9663	45.85/0.9886	45.97/0.9894	46.45/0.9899	46.55/0.9901
mean value	40.26/0.9658	45.79/0.9890	45.95/0.9896	46.37/0.9901	**46.51/0.9904**

Note: Bold text indicates the improved optimal metrics.

References

1. Dong C, Loy C, He K, et al. Image super-resolution using deep convolutional networks[J]. IEEE transactions on pattern analysis and machine intelligence, 2015, 38(2): 295-307.
2. Yang J, Wright J, Huang T, et al. Image super-resolution as sparse representation of raw image patches[C]. 2008 IEEE conference on computer vision and pattern recognition. IEEE, 2008: 1-8.
3. Dong C, Loy C, Tang X. Accelerating the super-resolution convolutional neural network[C]. Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14, 2016: 391-407.
4. Kim J, Lee J, Lee K. Accurate image super-resolution using very deep convolutional networks[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 1646-1654.

5. Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks[C]. International conference on machine learning. Pmlr, 2013: 1310-1318.